



Subversion ALM v6.0.8



Subversion ALM

Painless and powerful ALM integration of JIRA with Subversion



Contents

Introduction	4
Major Features.....	4
Installation & Configuration.....	5
Install the add-on jar file	5
Compatibility.....	5
Accessing to the configuration page.....	5
Registering Subversion repositories	7
The Indexing Process	9
Configuration parameters.....	10
Re-initializing the Subversion ALM configuration.....	10
Exporting and importing the registered repositories	10
The Subversion timeouts	11
The pool size	11
The Database Web Console	11
Compacting the database file size	13
Displaying Subversion commits on JIRA.....	14
Displaying Subversion commits on Agile (Greenhopper)	16
The Web Client for Subversion	17
JIRA integration.....	17
Subversion Web Client in <i>Standalone</i> mode	18
Major features	19
Repository browse	19
Revision List.....	19
Revision changes.....	20
Compare files	20
Commit graphs.....	21
Statistics	21
Reporting	22
JQL functions.....	22
Attribute JQL functions	23
Aggregation JQL functions	27



Commits Calendar Report.....	29
Extending the functionality with 3 rd party plugins.....	32
How to use the JDBC connections from a 3 rd party plug-in	32
Example.....	34
Security	35
Audit.....	36
Subscribing to Commit Events	37
Office and PDF documents.....	40



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



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:

The screenshot shows the JIRA administration interface. The top navigation bar includes 'Administration', 'Projects', 'Plugins', 'Users', 'Issues', and 'System'. The 'Plugins' menu is open, showing a list of installed plugins: 'Plugins', 'Application Links', 'Source Control', 'CVS Modules', 'FishEye Configuration', 'Perforce Job Integration', and 'Subversion Repositories'. The 'Subversion Repositories' item is highlighted with a blue selection bar. Below the plugin list, there are buttons for 'Manage Existing', 'Update', and 'Filter visible plugins'. A message at the bottom right says 'upgrade your installed plugins, as well as...' followed by 'Check', 'Audit Log', and 'OSGI' buttons. The overall title of the interface is 'The Universal Plugin Manager'.

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



	httpservice-bridge	Base POM for Atlassian projects
	OSGi R4 Compendium Bundle	OSGI Service Platform Release 4 Compendium Interfaces and Classes.
	plugin-data-editor	Plugin that provides an in-product editor utility for data stored with SAL's PluginSettings or ActiveObjects.
	Subversion Plus Painless and powerful Subversion integration with JIRA	
Plugin key:	com.kintosoft.jira.subversion-plus	Uninstall
Developer:	Kinto Soft Ltd	
Plugin version:	5.0.0	
Manage plugin modules - 115 of 115 modules enabled.		
Plugin Details Configure Disable		

System Plugins



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:

Registered Subversion Repositories

You have to register your existing Subversion repositories in order to see them from Subversion ALM.

[Add](#)

ID	Name	Status	Details	Operations
1	Apache	Scanning	Repository Root: https://svn.apache.org/repos/asf Index progress: 10923 / Unknown HEAD revision!	Edit Delete
3	DocMiner	Scanning	Repository Root: http://cagasego/svn/docminer Index progress: 355 / 2503	Edit Delete
2	Test	Active	Repository Root: http://cagasego/svn/test Index progress: 58 / 58	Edit Delete

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:



Most of the configuration is automatically populated 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.



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:

Indexing Process

The indexing process scans all the registered repositories within periodic time intervals. It is configured to scan upto 3 Subversion repositories in parallel.

Interval (secs.)	Latest scan	Operations
3600	2014-05-15 11:25:36	Re-schedule

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 screenshot shows the JIRA Subversion Plus indexing process re-schedule page. At the top, there's a navigation bar with links for Administration, Projects, Plugins, Users, Issues, System, and a search bar labeled "Administration Quick Search". The main content area has a header "Subversion Plus" with a gear icon. Below it is a section titled "Indexing Process Re-schedule" with the sub-section "Please, type the new interval in seconds". A form field labeled "Interval (secs.):" contains the value "3600", which is highlighted with a yellow background. A tooltip below the field says "Integer greater than 10". At the bottom of the form are "Save" and "Cancel" buttons.

The new interval value has to be in second units.

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

Registered Subversion Repositories

You have registered 3 Subversion repositories

The Indexing Process is configured to scan upto 3 repositories in parallel. Currently, there are 2 being indexed.

[Scan repositories now](#)

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



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

Subversion ALM Documentation

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.



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:

Subversion Timeouts (millis.)	Connection: 130000 Read: 140000	Edit

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:

Connection Pool Size

This configuration regards the internal H2 database instance used by the add-on and it is not related to the regular JIRA database instance

* Connection Pool Size: Integer greater than 10

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.



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

H2 Web Console Access Configuration

JIRA Administrators can access to the H2 database internal instance remotely by using a powerful web console.

You have to provide a the data below in order to access:

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`

Username: `h2admin`

Password: Please, type it below. It is saved in memory and reseted every time that the plugin is re-started

* Password:

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:

The screenshot shows the JIRA Login dialog box. The 'Saved Settings' dropdown is set to 'Generic H2 (Embedded)'. The 'Setting Name' field contains 'Generic H2 (Embedded)' with 'Save' and 'Remove' buttons. The 'Driver Class' field contains 'org.h2.Driver'. The 'JDBC URL' field contains the copied 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`. The 'User Name' field is filled with 'h2admin'. The 'Password' field is filled with a redacted value. At the bottom are 'Connect' and 'Test Connection' buttons.

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:



The screenshot shows the Subversion ALM database console interface. At the top, there's a toolbar with icons for back, forward, search, and other functions. Below the toolbar is a header bar with session information and settings like 'Auto commit' and 'Max rows: 1000'. The main area has a sidebar on the left listing database objects: dbch2:file:C:\Users\Pablo\workspace\CLIENT_B6NF_199H_89M5_045Z, followed by ACTION, APPLICATION_CONFIGURATIK, COMMENTS, COPIES, ITEMS, KEYS, REPOSITORIES, REPOSITORIES_CONFIGURAT, and REVIEWS. Below this is a section for 'Important Commands' with links to help pages, command history, executing statements, and disconnecting. A 'Sample SQL Script' section contains a code snippet for creating a table named TEST with columns ID (INT PRIMARY KEY) and NAME (VARCHAR(255)), and inserting two rows ('Hello' and 'World').

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):

Parameter Name	Value	Action
Compact database on plugin exit (it may take a long time and it should enabled only from time to time for maintenance)	false	Switch

ng Process

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.



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:

The screenshot shows the JIRA issue page for issue TEST-1. The Subversion tab is selected. The page displays commit details for three revisions:

Repository	Revision	Date	User	Message
Test	#20	Sun Jun 02 18:18:38 CEST 2013	admin	Delete elements remotely TEST-1 Files Changed DEL /acme/tests/dump.bat
Test	#3	Wed Apr 17 09:33:14 CEST 2013	admin	linked to TEST-1 and TEST-2 working? Files Changed MODIFY /svn/a.txt
Test	#2	Sun Apr 14 20:17:54 CEST 2013	admin	Testin TEST-1 integration Files Changed ADD /svn ADD /svn/a.txt

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



JIRA

Dashboards | Projects | Issues | Subversion | + Create Issue | Quick Search | admin | Administration

Test

Subversion

Summary
Issues
Popular Issues
Labels

Subversion
#disable_html_escaping()
Subversion Commits

Select version: All versions

Repository	Revision	Date	User	Message
Test	#20	Sun Jun 02 18:18:38 CEST 2013	admin	Delete elements remotely TEST-1 Files Changed DEL /acme/tests/dump.bat
Test	#13	Wed Apr 17 11:57:01 CEST 2013	admin	Message TEST-12 Files Changed MODIFY /svn/a.txt
Test	#12	Wed Apr 17 11:56:52 CEST 2013	admin	Message TEST-11 Files Changed MODIFY /svn/a.txt
Test	#11	Wed Apr 17 11:56:46 CEST 2013	admin	Message TEST-10 Files Changed MODIFY /svn/a.txt
Test	#10	Wed Apr 17 11:56:36 CEST 2013	admin	Message TEST-9 Files Changed MODIFY /svn/a.txt
Test	#9	Wed Apr 17 11:56:30 CEST 2013	admin	Message TEST-8 Files Changed MODIFY /svn/a.txt

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.



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:

The screenshot shows the Subversion ALM integration with JIRA. On the left, the JIRA Agile interface displays a backlog of issues under the 'Backlog' section, with 5 issues listed: DEMO-2, DEMO-1, DEMO-4, DEMO-5, and DEMO-6. On the right, the Subversion commit history is integrated into the board, showing three recent commits by 'Test' user:

Repository Revision	Date	User	Message	Files Changed
#38	Wed Oct 30 12:49:12 CET 2013	admin	DEMO-1 Testing multi line DEMO-3 DEMO-4 commnets	ADD /proof_2/my_dir/dump.bat
#34	Sun Sep 29 14:16:51 CEST 2013	admin	Testing jql multi project DEMO-2 and TEST-2	ADD /proof_2/my_dir/crossproject
#15	Wed Apr 17 22:42:05 CEST 2013	admin	DEMO-2	MODIFY /svn/a.txt



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 screenshot shows the JIRA web interface. At the top, there is a navigation bar with links for Dashboards, Projects, Issues, Subversion, and Create Issue. Below the navigation bar, the Subversion tab is highlighted. A "Subversion Plus" logo is visible on the left side of the page. In the center, there is a "Login" dialog box with fields for Repository (set to "Test [1]"), Username ("admin"), and Password, along with a "Login" button.

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

The screenshot shows the JIRA web interface. At the top, there is a navigation bar with links for Dashboards, Projects, Issues, Subversion, and Create issue. The Subversion tab is highlighted. A "Subversion Plus" logo is visible on the left side of the page. In the center, there is a "Commit Calendar View" section. A dropdown menu for the Subversion tab is open, showing options: SVN Web Client (which is selected and highlighted in blue), and SVN Filter Report.

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:



The screenshot shows the login interface of Subversion ALM. The 'Repository' dropdown is set to 'Test [1]'. The 'Username' field has 'DocMiner [2]' and 'Test [1]' listed as suggestions. The 'Password' field contains masked text. A 'Login' button is at the bottom.

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



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 (SVNMWEB-300)

Revision List

Info			
Head revision: 2104 Displayed revisions: 20			Actions
Revision	Age	Author	Comment
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 (SVNMWEB-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 comparison 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



Revision changes

Info

Revision: 1529 Age: 1 months Author: Vinnykovl	Revision: 2087 Age: 1 weeks Author: Vinnykovl
Comment: Improve compare view	
Changed resources	
<ul style="list-style-type: none"> WEB-INF WEB-INF/web.xml directoryCompare.jsp directoryContent.jsp fileCompare.jsp fileCompareData.jsp fileContent.jsp images images/download.gif images/reclist.gif include include/actionPanel.jsp include/dialog include/dialog.jsp include/dialog/changeConfirmationContent.jsp include/dialog/deleteContent.jsp include/dialog/directoryAddContent.jsp include/dialog/fileAddContent.jsp 	

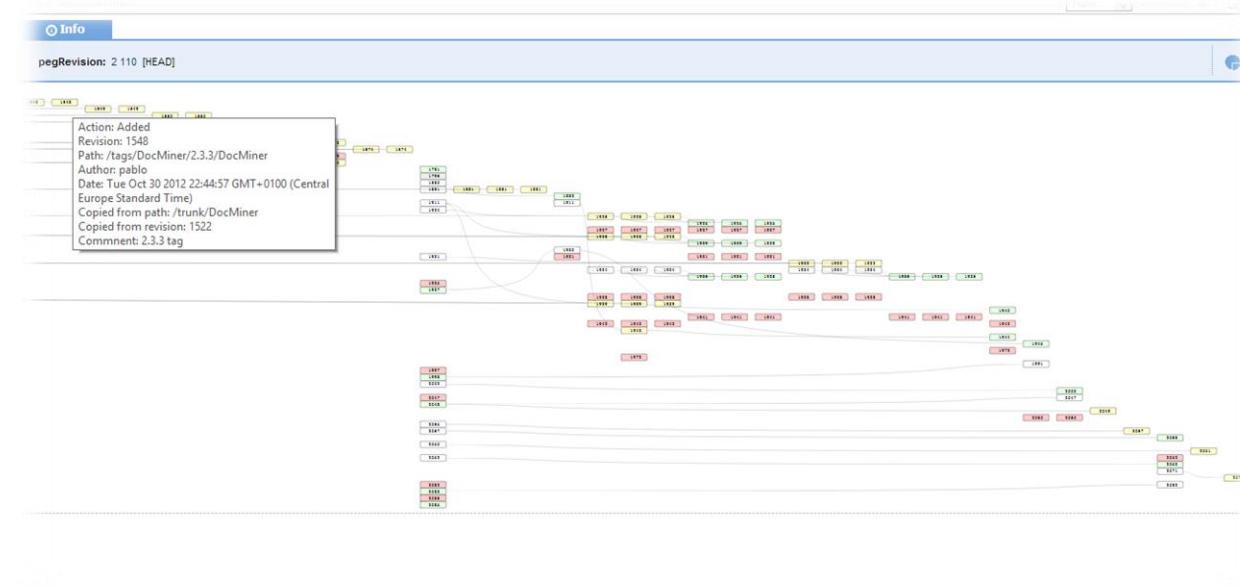
Compare files

Info

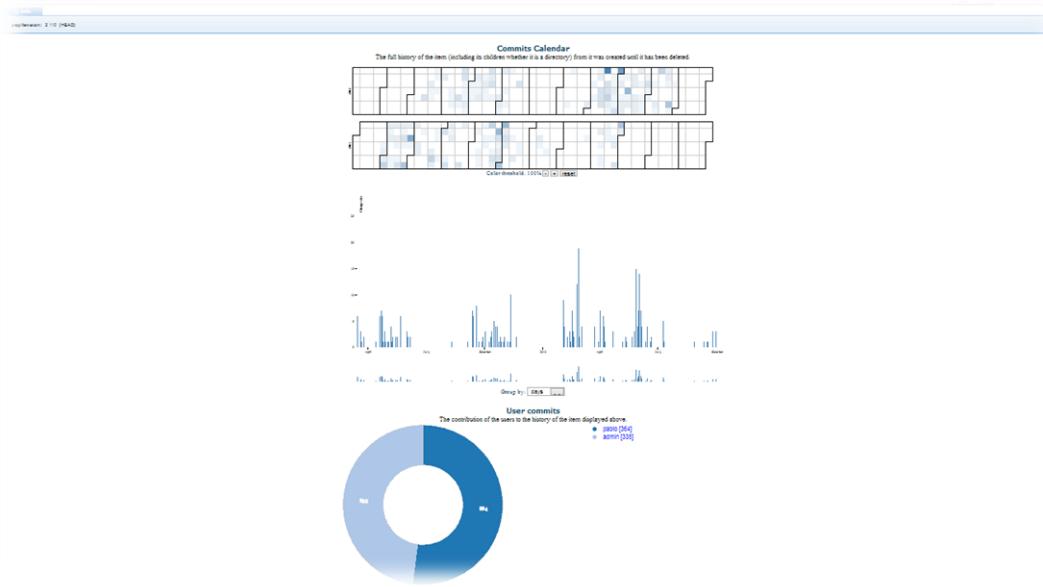
Revision: 543 Age: 2 months Author: dobisekm	Revision: 569 Age: 2 months Author: dobisekm
Comment: - importing the SVNWebClient project content	
<pre> 1 <?xml version="1.0" encoding="UTF-8"?> 2 <project> 3 <field id="active">true</field> 4 <field id="finish">2005-09-16</field> 5 <field id="start">2005-05-25</field> 6 <field id="description"> 7 8 9 10 </description> 11 12 13 14 15]></field> 16 <field id="trackerPrefix">SVN\WEBCLIENT</field> 17 <field id="name">SVN Web Client</field> 18 <field id="lead">Vinnykovl</field> 19 <field id="project">SVN\WebClient</field> 20 </project></pre>	
Comment: Polarion commit Thu Nov 03 16:55:43 CET 2005	



Commit graphs



Statistics





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.

	Assignee	Reporter	P	Status	Resolution	Created	Updated	Due
	admin	admin	↑	Open	Unresolved	02/Oct/13	02/Oct/13	

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:



- A) *repold*: 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

<i>svnCommitNumberRange</i>					
It returns the issues related to commits between a revision range on an specific repository					
#	Name	Type	Required	Values	Description
1	repold	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



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.

<i>svnCommitDateRange</i>					
It returns the issues related to commits between a date range on all the repositories					
#	Name	Type	Required	Values	Description
1	repold	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.

<i>svnAuthor</i>					
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	repold	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.



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	repold	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 “merged” 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	repold	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.



- 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.



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



Example:

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

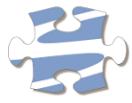
What are all the issues having commits on 2 or more repositories? (Cross repository issues are not allowed in this company).

<i>svnMultiProject</i>					
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 or more JIRA projects?



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:

The screenshot shows the JIRA Issues screen with a sidebar containing filters like 'My Open Issues', 'Reported by Me', 'Recently Viewed', and 'All Issues'. A 'FILTERS' section on the left shows a saved filter named 'issue in svnAuthor('admin')'. The main table lists 10 issues from TEST-1 to TEST-10, all created and updated on 06/Oct/13, assigned and reported by 'admin', and in an 'Unresolved' state. A 'Save as' button is visible at the top right of the search bar.

The screenshot shows the JIRA Issues screen with a 'Save Filter' dialog box overlaid. The dialog box has a 'Filter Name' field containing 'Commits made by the admin user' and a 'Submit' button. The background table shows the same 10 issues as the previous screenshot.

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

The screenshot shows the Subversion ALM navigation bar with options like 'Dashboards', 'Projects', 'Issues', 'Subversion', and 'Create issue'. The 'Subversion' dropdown is open, showing 'SVN Web Client' and 'SVN Filter Report'. The 'SVN Filter Report' option is highlighted with a blue background.



XJIRA Dashboards ▾ Projects ▾ Issues ▾ Subversion ▾ Create issue

Configure - Subversion Plus Filter Report

Report: Commit Calendar View

Description:

USAGE: Build a query with the JQL svn functions, save it and display the results on a commit calendar view by using this report.

Filter Select Filter...

[Next](#) [Cancel](#)

Click on the *Select Filter...* link:

Filter Picker - Your Company JIRA - Google Chrome
cagasego:2990/jira/secure/FilterPickerPopup.jspa?field=filterId

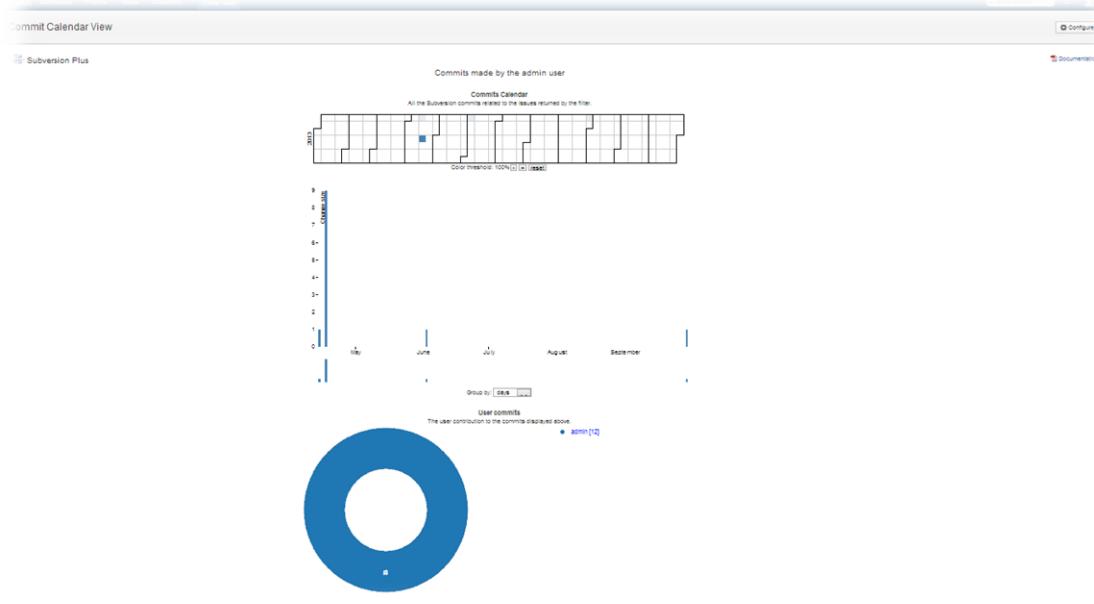
Filter Picker

Select a filter from the filter directory.

Favorites Popular Search

Name	Owner
Commits made by the admin user	Pablo (pablo)
Working in comment	Pablo (pablo)

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





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:

The commits made in 2013-04-17 included in the 'Commits made by the admin user' filter

repository	Revision	Date	User	Message
est	#11	Wed Apr 17 11:56:46 CEST 2013		TEST-10 Files Changed MODIFY /svn/a.txt
est	#10	Wed Apr 17 11:56:36 CEST 2013		TEST-9 Files Changed MODIFY /svn/a.txt
est	#9	Wed Apr 17 11:56:30 CEST 2013		TEST-8 Files Changed MODIFY /svn/a.txt
est	#8	Wed Apr 17 11:56:24 CEST 2013		TEST-7 Files Changed MODIFY /svn/a.txt
est	#7	Wed Apr 17 11:56:17 CEST 2013		TEST-6 Files Changed MODIFY /svn/a.txt
est	#6	Wed Apr 17 11:56:10 CEST 2013		TEST-5 Files Changed MODIFY /svn/a.txt
est	#5	Wed Apr 17 11:56:04 CEST 2013		TEST-4 Files Changed MODIFY /svn/a.txt



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>
```



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

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.close();
      ps.close();
    }
  }
}
```



```
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:



Subversion ALM v6.0.8



JIRA Dashboards ▾ Projects ▾ Issues ▾ Subversion ▾ Create Issue

Demonstration Project
Key: DEMO Lead: admin

Overview Administration

Summary Subversion ALM Demo
Subversion ALM Demo

Issues Reports Popular Issues Subversion Labels

This project has 11 commits!

Bug tracking and project tracking for software development powered by Atlassian JIRA (v6.1.7#6163-sha1:94d557d) · About JIRA · Report a problem

This JIRA site is for non-production use only.

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:



JIRA Dashboards Projects Issues Subversion Create Issue

Administration Search JIRA admin

Projects Add-ons User Management Issues System

ATLASSIAN MARKETPLACE Find new add-ons Manage add-ons Purchased add-ons OSGI

APPLICATION LINKS Application Links

SOURCE CONTROL DVCS Accounts FishEye Configuration Perforce Job Integration

Subversion Repositories

BUILDS Bamboo Configuration

ISSUE COLLECTORS Issue Collectors

MONITORING Database Monitoring

Subversion ALM

Application Data

Subversion ALM uses a H2 Java database instance to store all the add-on application data

Attribute	Value	Operations
Directory	C:\Users\Pablo\workspace_SVN\svn_plus\target\jira\home\caches\indexes\plugins\kintosoft_subversion_h2	
Connection Pool Size	100	Change
Share JDBC connections	true	Stop

The H2 database instance is automatically started and stopped by the add-on. Optionally, a daemon can be started to listen on a configurable port number allowing to access to the application data through a remote web console.

URL	Username	Operations
The server daemon is stopped	h2admin	Start

Indexing Process

The indexing process scans all the registered repositories within periodic time intervals. It can scan upto ten Subversion repositories in parallel.

Interval (secs.)	Latest scan	Operations
3600	2014-02-10 21:00:13	Re-schedule Run it now

Registered Subversion Repositories

The table below shows the Subversion repositories currently configured for this server.

Add

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:



The screenshot shows the JIRA Administration interface under the 'Add-ons' tab. On the left, there's a sidebar with various links like 'ATLASSIAN MARKETPLACE', 'OSGI' (which is highlighted with a red box), and 'MONITORING'. The main content area is titled 'OSGi' and contains a sub-section 'Debugging information about the JIRA OSGi container, useful for add-on developers and system administrators.' Below this is a search bar labeled 'Filter visible bundles' with a magnifying glass icon. A search result for 'com.kintosoft.svnwebclient.jira.public_inter' is shown, with the entire search input field also highlighted with a red box. The list of bundles includes '146 - db-demo' and '145 - Subversion ALM', both of which are also highlighted with red boxes.

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>
```



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 plugin. 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 committed 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.startListening(commitListener);  
  
    log.info("MyPluginComponentImpl commit listener registered");  
  
}
```



Subversion ALM v6.0.8



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



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:

The screenshot shows a JIRA interface displaying an ODT document. At the top, there's a navigation bar with links for JIRA, Dashboards, Projects, Issues, Agile, Subversion, and a 'Create issue' button. Below the navigation is a search bar labeled 'Quick Search'. The main content area shows the document's content. The 'Info' tab is selected, displaying details like Revision: 55, Date: 2014-04-24 10:42, Author: admin, Size: 849 382, and Comment: Other ODT test. The document itself has two sections: 'Acknowledgments' and 'Modifications and Updates'. The 'Modifications and Updates' section contains a table with the following data:

Version	Date	Description of Change
[0.0]	[2005-02-28]	[gw: Initial English Language translation]
[0.1]	[2005-03-31]	[vtm: Initial English Language edition issued for comment]
[0.2]	[2005-02-28]	[grs: Initial edit and proofing]
[0.3]	[2005-02-28]	[vtm: Final edit and proofing]
[0.4]	[2008-05-12]	[ppi: Edit all text for grammar, reformat layout, reorganize content, update all content and screenshots for OpenOffice 2.4, expand existing content.]
[0.5]	[2009-07-20]	[sm: Update the content and screenshots for OpenOffice.org 3. Modify title.]
[0.6]	[2010-02-11]	[cc: Added Sun/Oracle logo and changed the contact email address]

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