Documentation

Eclipse Environment Setup Guide

Jahia EE v6.1

Jahia delivers the first **Web Content Integration Software** by combining Enterprise Web Content Management with Document and Portal Management features.

Jahia

9 route des Jeunes,
CH-1227 Les acacias
Geneva, Switzerland

www.jahia.com
The Company website

www.jahia.org
The Community website

April 2010

Summary

1 Overview

1.1 Introduction

Jahia delivers the first Web Content Integration software by combining Enterprise Web Content Management with Document Management and Portal features.

By leveraging state of the art Open Source frameworks and libraries, Jahia offers a complete solution for developing, integrating, delivering, and managing content across intranets, extranets, and internets with a much lower total cost of ownership than proprietary systems.

1.2 What's in this documentation?

This document defines the steps that must be followed to install a development environment using the Eclipse IDE. The Eclipse IDE is very interesting because it is now a very strong solution for Java developers, and is at the same time entirely free.

You can download a Jahia EE v6.1 release and use Eclipse to develop, build and debug just your template projects. You can also check out Jahia from the SVN repository and use Eclipse to manage not only your template projects, but also the entire Jahia codebase. This may be interesting if you want to fork Jahia to create bugfixes or add-ons. In this guide we will help you set up both environments.

Please note that at the time of writing, these steps were all valid, but as software evolves these might not be as accurate.

Should you have questions, please do not hesitate to contact us as mentioned on our website (http://www.jahia.com) or Community website (http://www.jahia.org).

2 Prerequisites

This is a list of third party software this guide is based on:

- JDK 1.5 or later (http://www.jahia.org/cms/lang/en/home/download/system_requirements/JVM)
- Eclipse IDE for Java EE Developers 3.4.x (http://www.eclipse.org/downloads/)
- Maven 2 (http://maven.apache.org/download.html)

Set a M2_HOME environment variable pointing at the installation directory. Put M2_HOME/bin folder in path. Also increase the memory for Maven by setting the following environment variable: MAVEN_OPTS=-Xmx1024m

- Maven 2 Eclipse Plugin (http://m2eclipse.codehaus.org/ with installation requirements http://docs.codehaus.org/display/M2ECLIPSE/Installation+Requirements)

You have to install all the modules from Maven Integration, from the Maven Project Configurators you need Maven Integration for WTP and if you want to checkout Jahia's source, you need Maven SCM Integration and Maven SCM handler for Subclipse from the Maven Optional Components.

If you would like to checkout Jahia from SVN and create your own package you also need to install:

- Subversion client (for Windows http://www.collab.net/downloads/subversion/; for Mac OS X from http://www.finkproject.org/ or Xcode tools, under Linux install a subversion client from your package repository)
- Subclipse plugin for Eclipse (Eclipse update site URL: http://subclipse.tigris.org/update_1.4.x)
- Google Web Toolkit 1.5.3 (http://code.google.com/webtoolkit/download.html)
- Tomcat 6 (http://tomcat.apache.org/download-60.cgi) Or alternative J2EE application servers.

It may work with newer or older versions of the components and with different application servers, but perhaps with some variations to the procedures described in this guide.

3 Using Eclipse for Template development

3.1 Setup Procedure with a Jahia Runtime

1. Download and install a Jahia EE v6.1 release and setup the project development environment like described in Configure Jahia for Template development especially the Configure the Jahia Deployment Plugin.

2. Start Eclipse and select a (new) workspace for your template project.

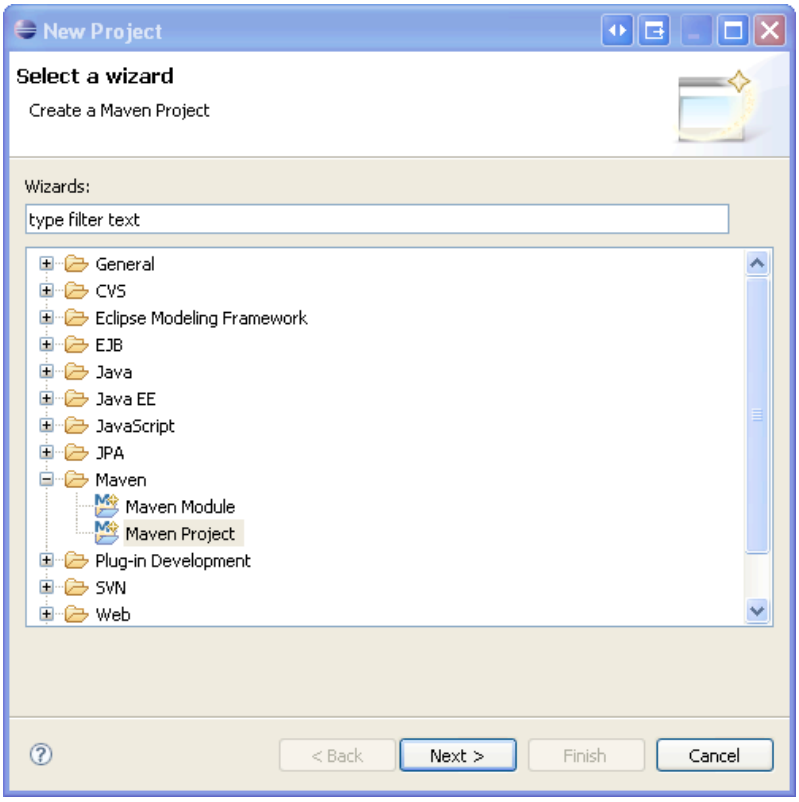
3. You should configure the Server runtime in Eclipse under Window → Show view → Other....

   Select Server → Servers in the tree and press OK.

   There will be a new Servers tab in the bottom right panel. Make a right button click with the mouse in the panel of this tab and select New → Server. Enter localhost in the server's host name and choose Apache → Tomcat v6.0 Server and click Next.

   For the Tomcat installation directory select the root directory of your downloaded Jahia runtime, which is also the installation directory of the embedded Tomcat server. Press Finish.

4. For creating a new Jahia template project, select File → New → Project... and choose Maven → Maven project from the tree. Push Next.

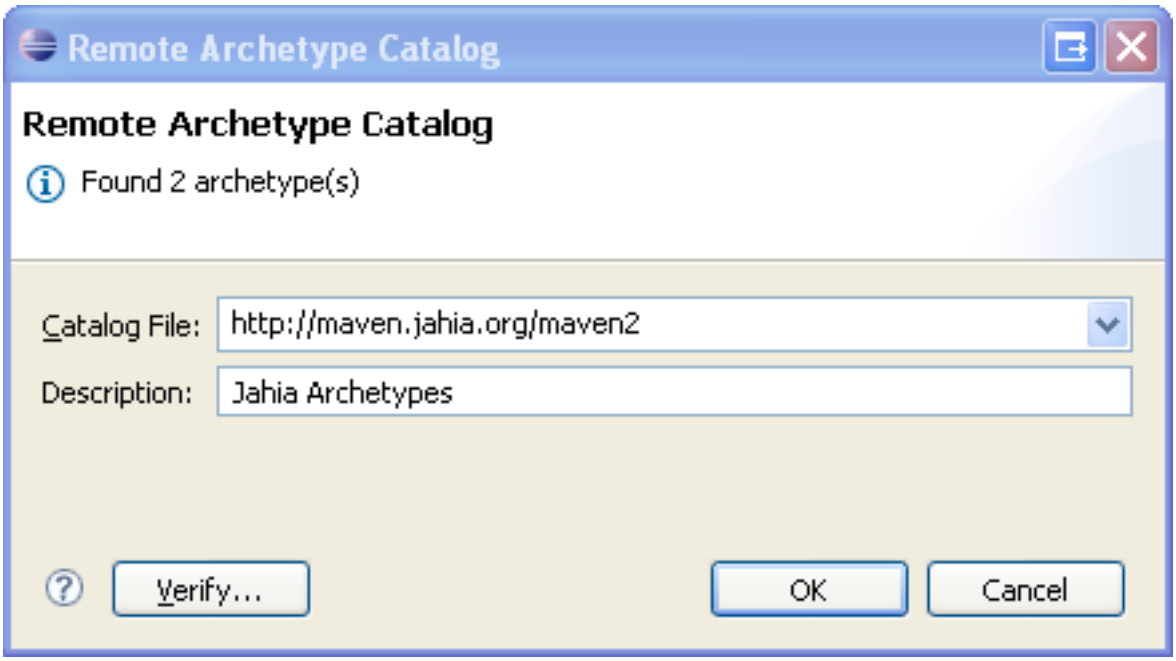


5. Leave the default settings on the next panel (the option with the Skip archetype selection should be unchecked) and push Next.

6. On the third panel you need to select Jahia Archetypes from the Catalog. If you do not have this option, you need to press Configure... and then Add Remote Catalog....

In the popup enter the Catalog file: `http://maven.jahia.org/maven2` and under Description enter `Jahia Archetypes`. With Verify... you can check the correctness of the URL. If successful, you should get a message with the number of found archetypes in the header of the popup.

Press OK for two times then.

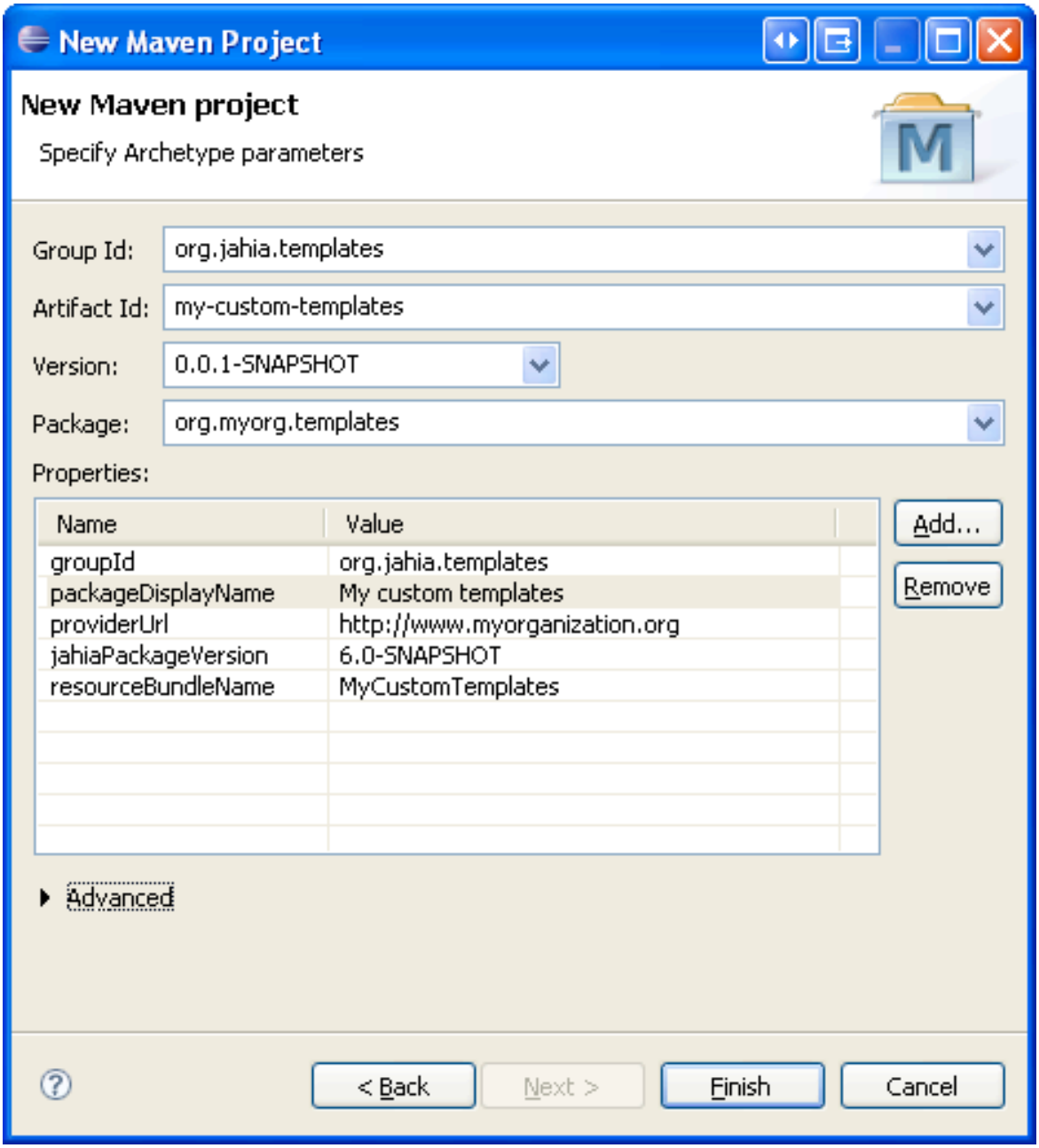


Now select the desired archetype for your template project, which depends on whether you want to start developing templates from scratch (`jahia-templates-archetype`) or inherit from existing templates (`jahia-web-templates-archetype`). Press Next.

7. On the next panel enter all the required information:

Group Id	This will only be asked if you do not confirm the settings. The group ID must be `org.jahia.templates`, so that the Jahia's maven deployment plugin will work correctly.
Artifact Id	This is your template project's artifact name (e.g. `my-custom-templates`) and will also be the name of the project's root folder. Don't use spaces, but either write all together or delimit with hyphens.
Version	This is the version of your template project (e.g. `0.1`)
Package	This is the default package name for your Java classes. If you have none you can leave this empty.
packageDisplayName	This is the display name of your template package (e.g. `My custom Jahia templates`)
providerUrl	This is the URL of the organization providing your templates: (e.g. `http://www.myorganization.org`)
jahiaPackageVersion	This is the Jahia version you are using (e.g. `6.0-CE`)
resourceBundleName	This is the name of the automatically created resource bundle holding all the labels in your templates (e.g. `MyCustomTemplates`). This term has to conform to Java naming conventions and must not use spaces, but simply the CamelCase conventio

After input press Finish.



8. As long as the bug http://jira.codehaus.org/browse/MNGECLIPSE-1054 has not been fixed, you need to rename the resource bundle under `src\main\resources\jahiatemplates\__resourceBundleName__.properties` to the name you have chosen in the popup, otherwise it would have been renamed already.

9. You can now start to create your JSP templates, add Java classes, edit the resource bundles in your project directly within Eclipse and will be able to make use of the several Content assist features Eclipse offers.

3.2 Deploying Templates to Jahia Server

To deploy the template project to Jahia, you could use Maven directly within Eclipse, by following the Maven as external tool advice at http://maven.apache.org/guides/mini/guide-ide-eclipse.html. Name the external tool configuration `Deploy WAR to Jahia` and enter as goal Arguments `install war:exploded jahia:deploy`. You can also make hot-deployment into a running server like this. Unless you made big structural changes in Java classes, the hot deployed changes will get active without having to restart the server. For changes in the resource bundle you will need to restart the server.

3.3 Debugging Templates in Eclipse

1. If you want to use Eclipse to debug your JSP and Java classes, you first have to add the following parameters to the startup job of your Tomcat server (`catalina.bat` or `catalina.sh` or a job calling these):

```
set JPDA_TRANSPORT=dt_socket
set JPDA_ADDRESS=17000
```

   Instead of `17000` you can take any free port number.

   You also need to add the parameter called `jpda start` when calling catalina.

2. In Eclipse you should then create a Remote Java Application Debug Configuration. Select Run → Debug Configurations... and then right click on Remote Java Application and chose New.

   Enter a name and on the Connect tab click the Browse... button and select your templates project. Make sure that the Connection Properties → Port is the same as used in `JPDA_ADDRESS` e.g. `17000`.

   Click on the Source tab and press the Add… button. Select File System Directory and then select the root directory of the Jahia web application (e.g. `C:\jahia_ANDROMEDA\webapps\ROOT`).

When you are done press the Apply button or Debug if the server is already started (externally with Tomcat's startup script in the bin folder). Once you have pressed the Debug button here, you will be able to choose the debug configuration quicker via Eclipse menu shortcuts.

3. Now you can set breakpoints in your JSPs or Java source files. When starting the Jahia server and connecting to it via the Eclipse remote debugging, processing will be stopped at the breakpoints you set. You will be able to view any variables and do step-by-step debugging.

   More information about Code assist and debugging can be found in the Eclipse documentation.

4 Using Eclipse to Work on Jahia's Codebase

If you are a Jahia contributor you may want to use Eclipse to work on Jahia's codebase. Or perhaps you may just want to delve into Jahia's code to investigate/debug and make some bugfixes or develop extensions locally.

Be aware that we are not recommending you to make a forked Jahia codebase if you are not contributing the extensions back to the product core, as it will make it more difficult for you to install Jahia's hotfixes or upgrade to newer releases. For extensions you should rather consider subclassing and/or configuring your own services, valves etc. Jahia has many possibilities to plug-in own implementations. We will talk about that in a separate article.

4.1 Setup Procedure for Working on Jahia's Codebase

1. If not done already install the downloaded JDK, Maven and Eclipse IDE for Java EE Developers, additionally also the Subversion client and Subclipse plugin (see Prerequisites above for download links).

   Please also install the M2Eclipse plugin, as Jahia uses a hierarchical Maven multiple project structure, which is best handled with this plugin.

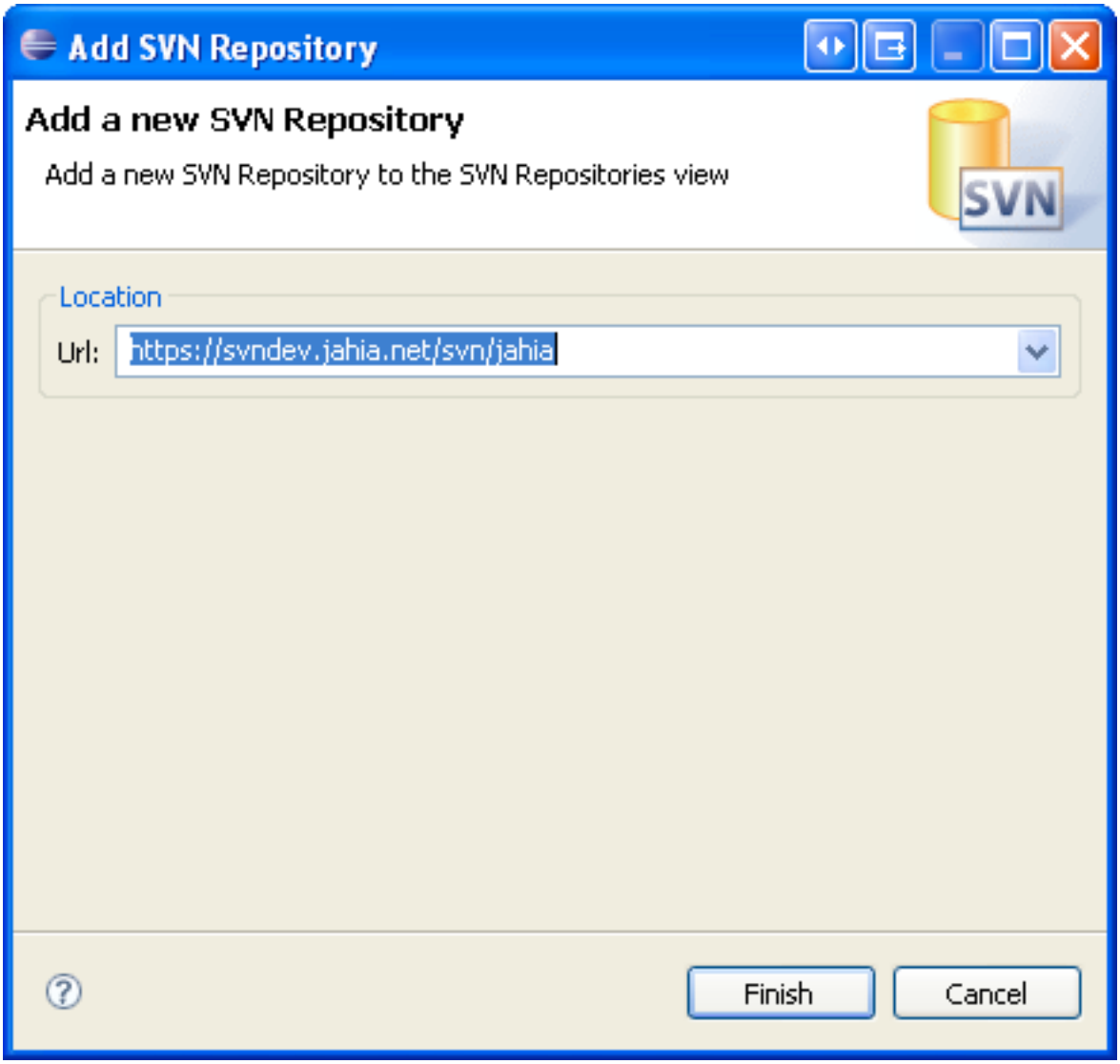
2. You need to also install a local Tomcat 6.x application server (or an alternative J2EE application server). Please also add this new server to the Server runtime configuration in Eclipse similarly as you have done when using a Runtime Jahia version (see step 5 in above section Setup procedure with a Jahia Runtime), just that you now use a distinct Tomcat or alternative application server installation and not the embedded Tomcat available with the Jahia runtime

3. In order to successfully build Jahia, you also need to install all the prerequisites mentioned in the Jahia template development Environment and Tools Guide.

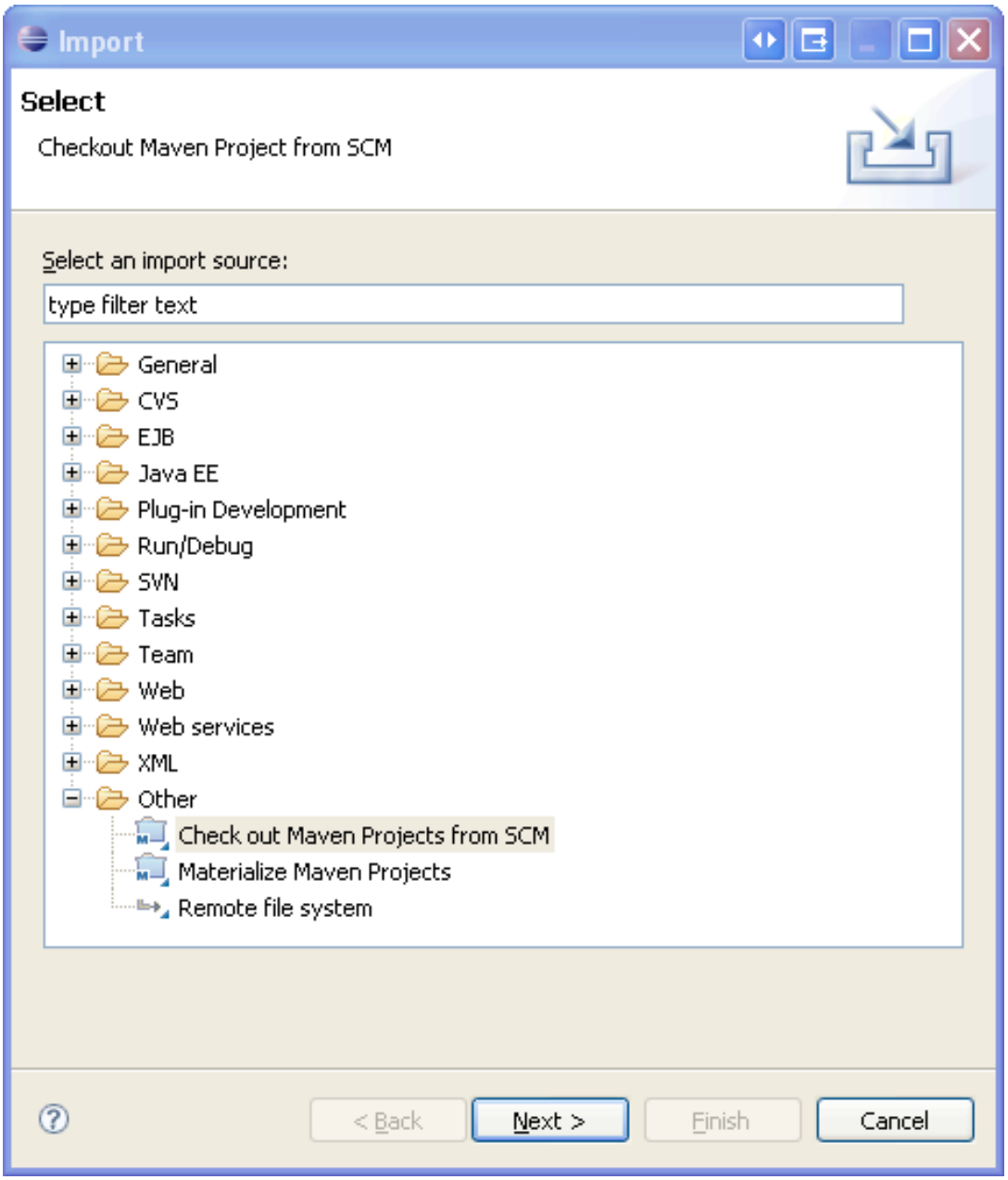
4. Once all is configured, start Eclipse and go into the SVN Repository Exploring view (can be done by opening in the Window menu the Open perspective option)

5. Click on the Add SVN Repository button in the upper right corner of the repository view window: 

6. In the window, enter the URL `http://subversion.jahia.org/svn/jahia` and press Finish.



7. Now check out Jahia in the following way: Select File → Import... and select Other → Check out Maven Projects from SCM and push the Next button.



8. Select svn in the first combobox and then click on the button Browse. Navigate to http://subversion.jahia.org/svn/jahia/branches/JAHIA-6-0-COMMUNITY-BRANCH, select this entry and click OK.

You may also use other branches, but notice that the following instructions only apply to Jahia EE v6 (Andromeda) or newer versions.

9. You can let the other default entries and click Next → Finish or you can also already click Finish straight away.
10. This will check out and create all the multiple Jahia projects.

4.2 Deploying Jahia to Local Tomcat

1. Please follow the instructions in Configure the Jahia Deployment Plugin
2. Similar to the instruction in Deploying Templates to Jahia server, you should configure Maven as external tool in Eclipse. To deploy the entire Jahia project, you have to click on the jahia-root project and issue a Maven launch configuration (via Run → External Tools or the menu tab icon) with the argument `install`.
3. Once the server is deployed, you can always selectively deploy just a module.

   For web modules the Maven job arguments are `install war:exploded jahia:deploy` (like mentioned above for the template project).

For a utility project, which creates a JAR the Maven arguments are `install jahia:deploy`.

If the changes are just minor, even hotcode replacement may work, otherwise you will have to restart the server in order to pick up the changed classes.

4.3 Debugging Jahia in Eclipse

Follow the instructions under Debugging Templates in Eclipse. The same applies to debugging all the Jahia Java classes or JSPs.

If the debugger does not stop on JSPs, it may be that the M2Eclipse did not set the correct project natures. Please then right-click the project jahia-war in Eclipse's Package Explorer or Navigator and select Maven → Update project Configuration. This may have to be done also for any template projects.

9 route des Jeunes,
CH-1227 Les acacias
Geneva, Switzerland

www.jahia.com
The Company website

www.jahia.org
The Community website