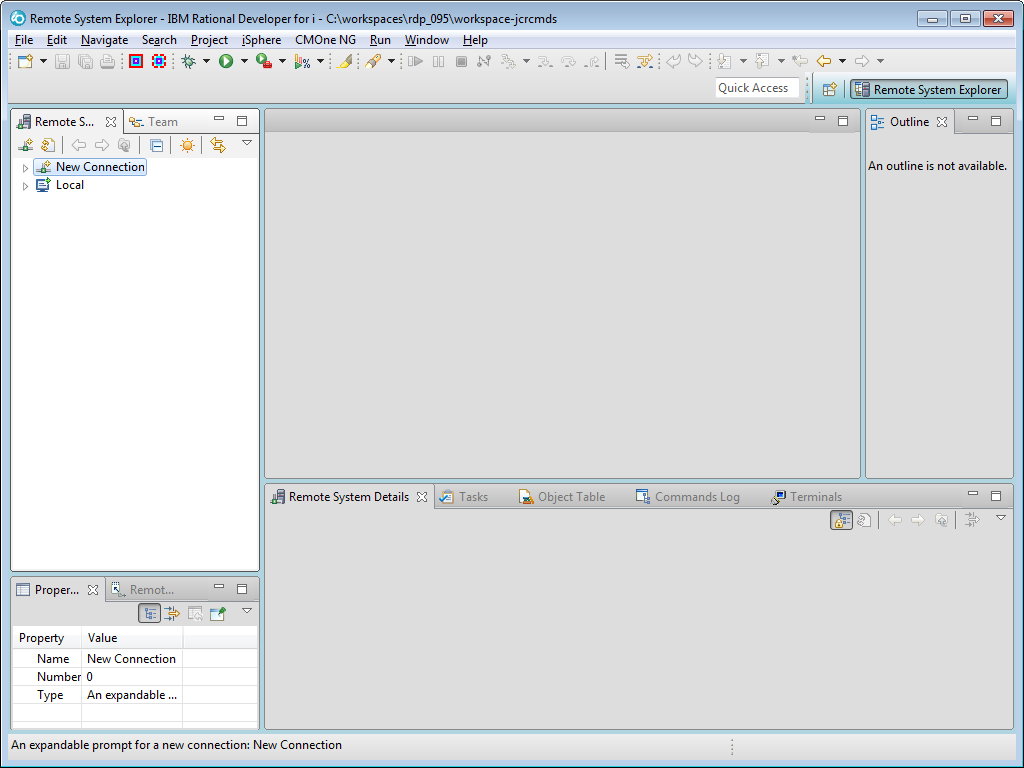
Working With JCRCMDS

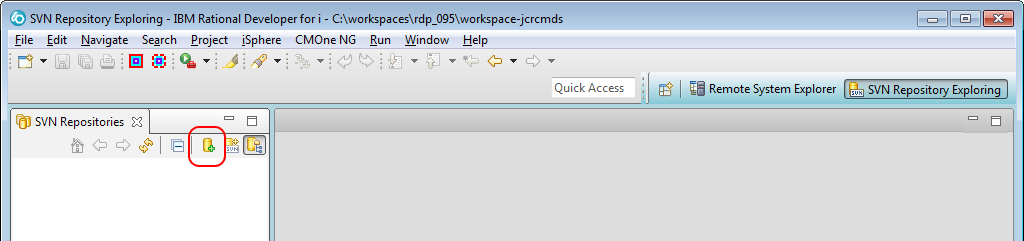
# Prerequisites

Installed IBM Rational Developer for i (RDi 9.5) with a dedicated and empty workspace for JCRCMDS. You should already have followed the instructions given in "Eclipse Setup (RDi 9.5).docx":



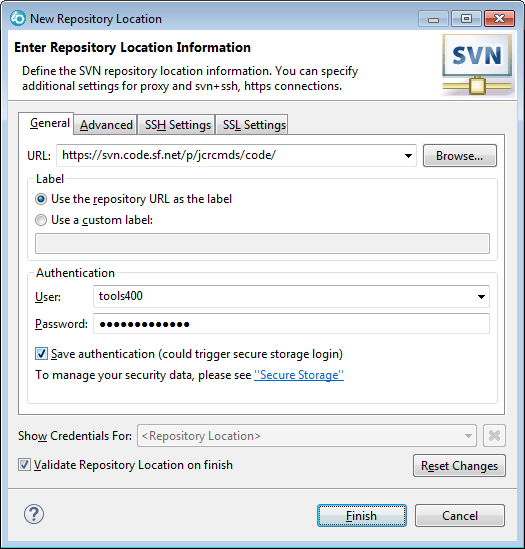
# Checking Out The Project From SourceForge

Switch to the "SVN Repository Exploring" perspective and click the "New Repository Location" button to add the JCRCMDS SVN repository, which is hosted at SourceForge:



Then enter the URL of the JCRCMDS repository and optionally your credentials at SourceForge:

https://svn.code.sf.net/p/jcrcmds/code/

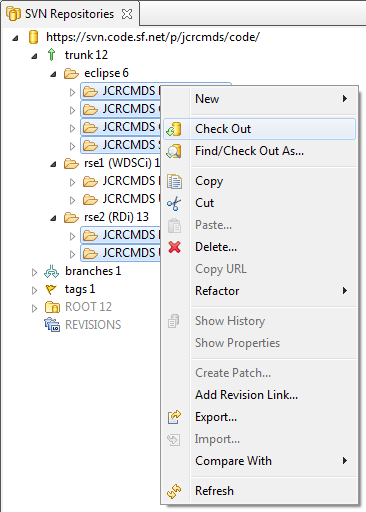


Click the [Finish] button to create the location.

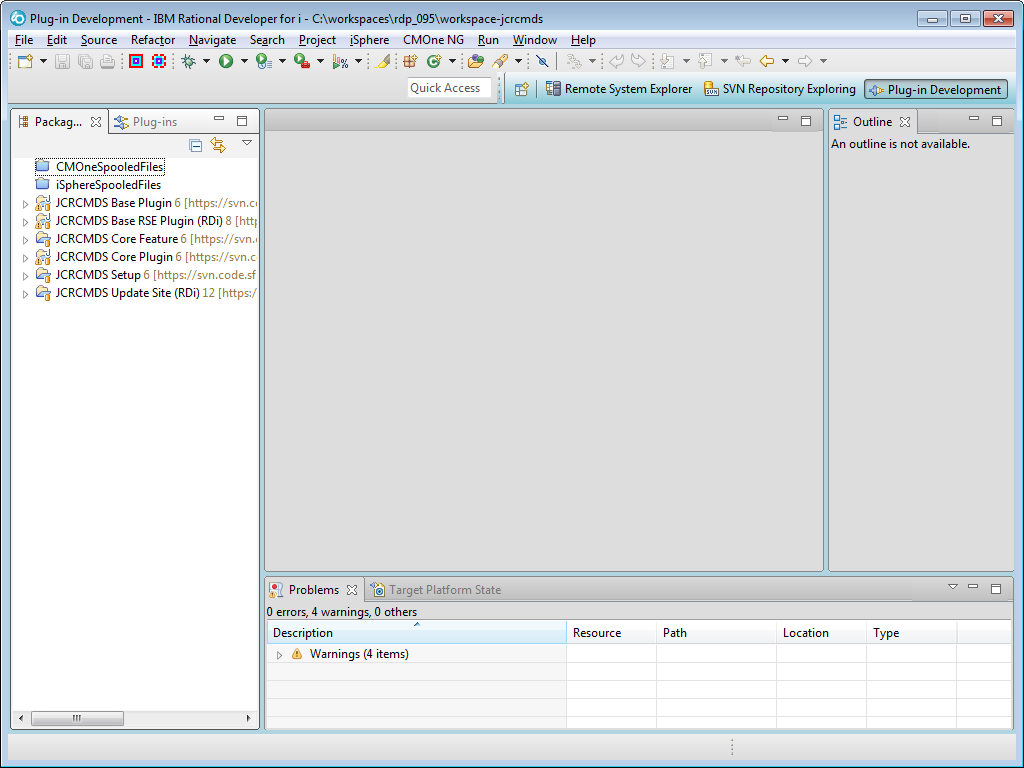
When you expand the location node, you should see the following items:

|  |  |
| --- | --- |
|  | The "trunk" folder always contains the latest development objects.  The "branches" folder is used for variants of the plug-in. It can also be used for temporary development streams.  The "tags" folder contains tags of the project. A tag is like a "freezed" status of the repository.  The "eclipseeclipse" folder contains the plug-ins that are shared between RDi and WDSCi.  The "rse1 (WDSCi)" folder contains the WDSCi specific plug-ins.  The "rse2 (RDi)" folder contains the WDSCi specific plug-ins. |

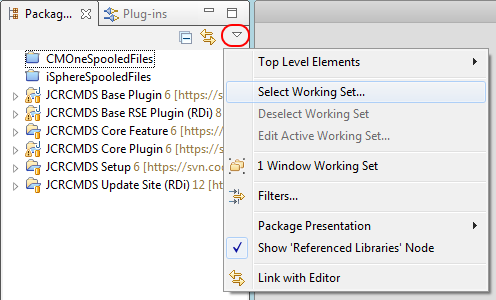
Go ahead and select the RDi specific plug-ins. Then select "Check Out" from the context menu:

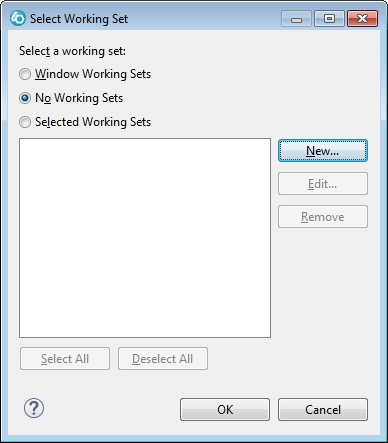


Now select the "Plug-in Development" perspective, which is used for developing plug-ins:

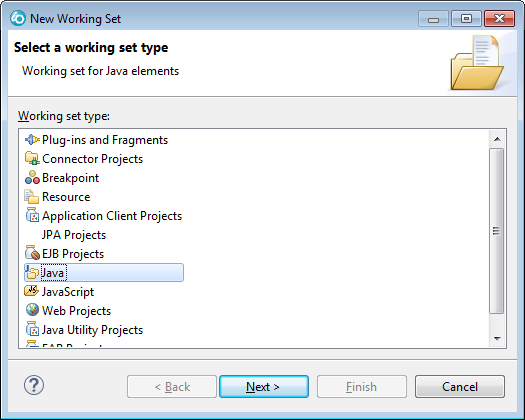


The items that are clearly do not belong to JCRCMDS, can be removed from the view, by creating a working set. Let us create a working set, so see how that works. Select "Select Working Set" from the view menu to create the JCRCMDS working set:

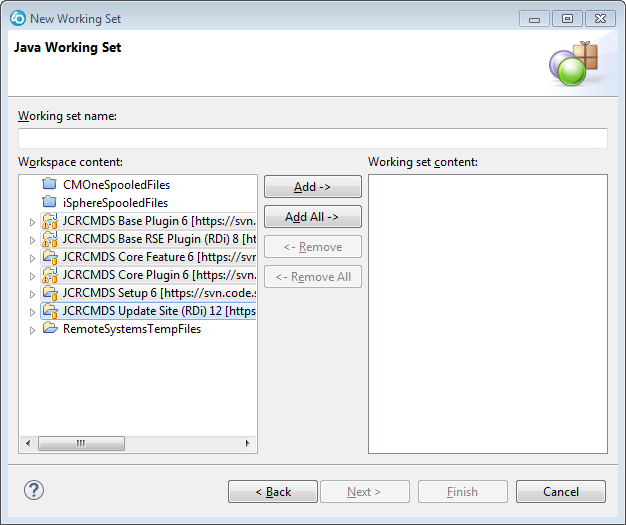




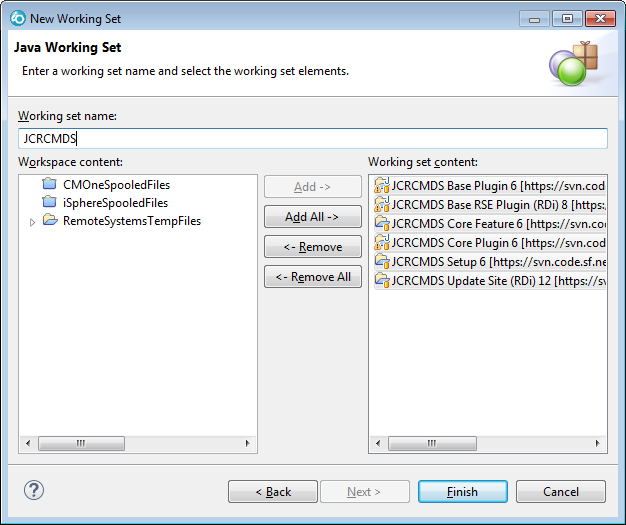
Click the [New] button to continue and the select the "Java" working set type:



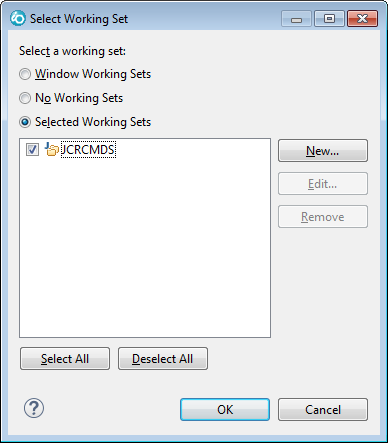
Click the [Next] button to continue:



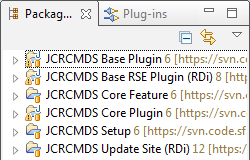
Now enter a name, e.g. "JCRCMDS", select the JCRCMDS projects and click the [Add] button:



Click the [Finish] button to create the working set. Then select the "JCRCMDS" working set and click the [OK] button:



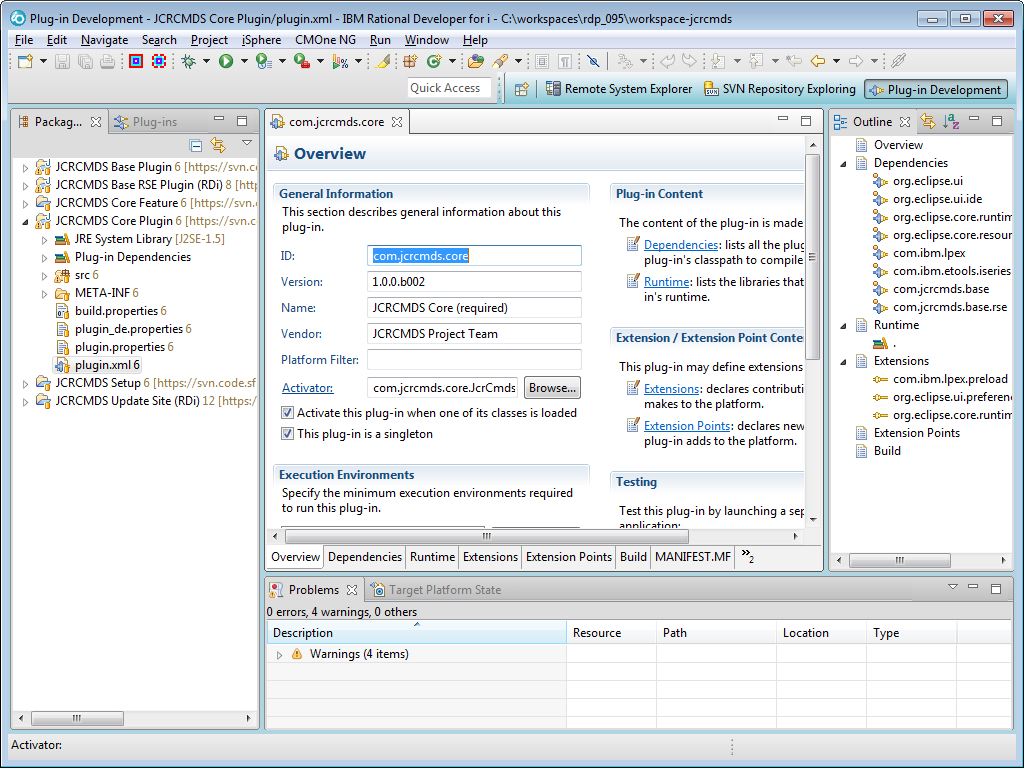
Now only the JCRCMDS projects are displayed in the Package Explorer view:



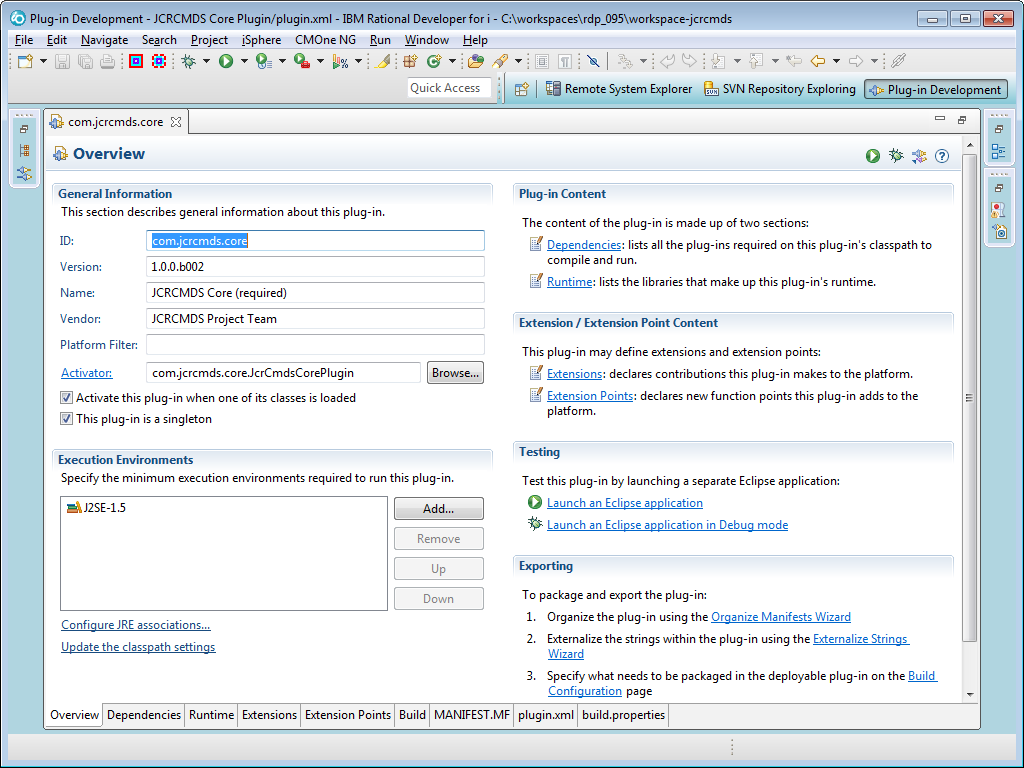
# Starting An Eclipse Test Application

As a developer you may want to start an Eclipse test application to test your Java code.

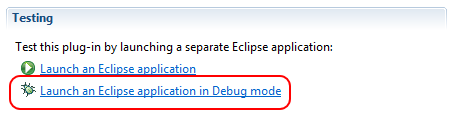
Expand the "JCRCMDS Core Plugin" and open file "plugin.xml":



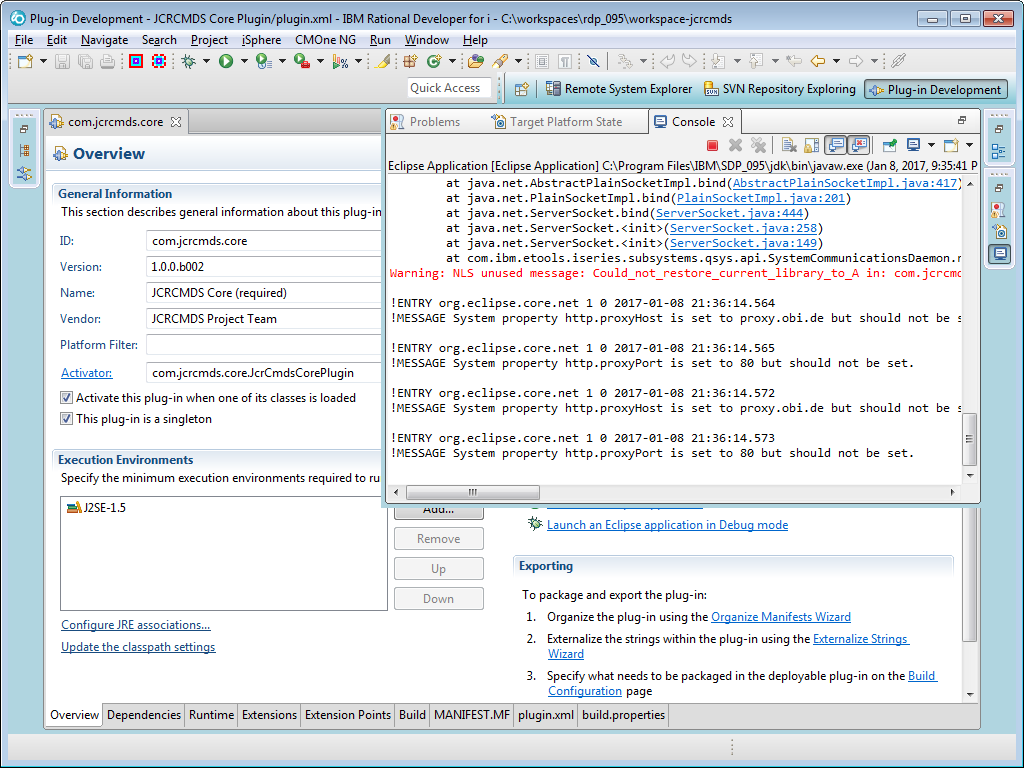
Double click the tab of the editor to maximize the editor, if you cannot see all options of the editor:



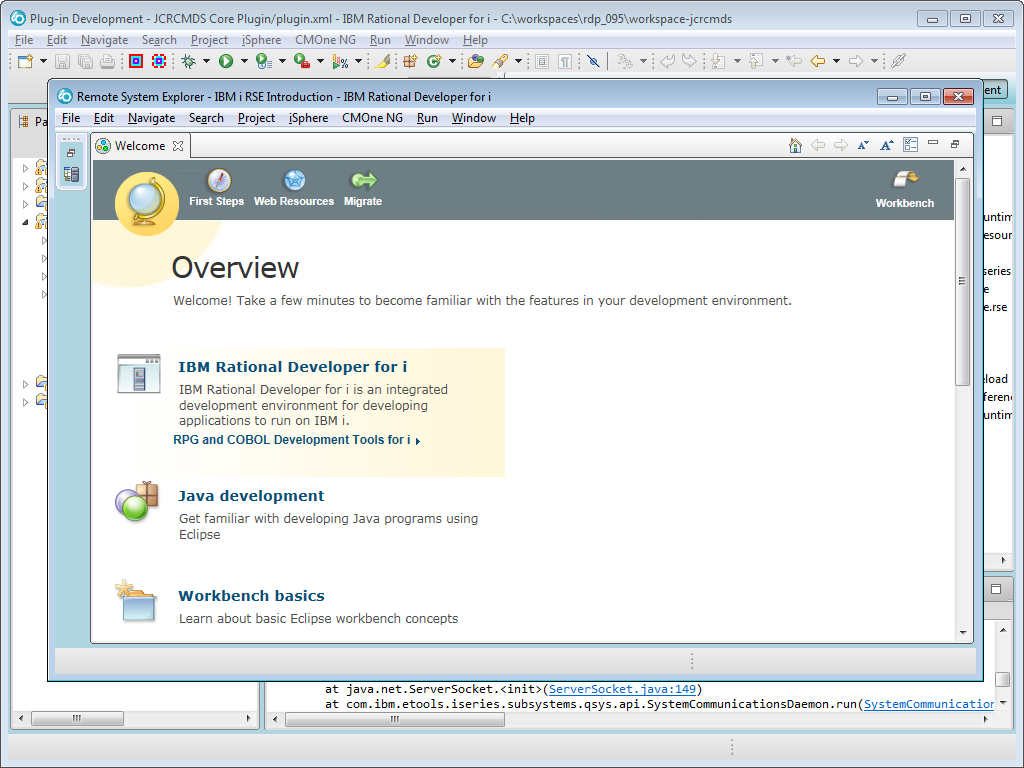
Click "Launch an Eclipse application in Debug mode":



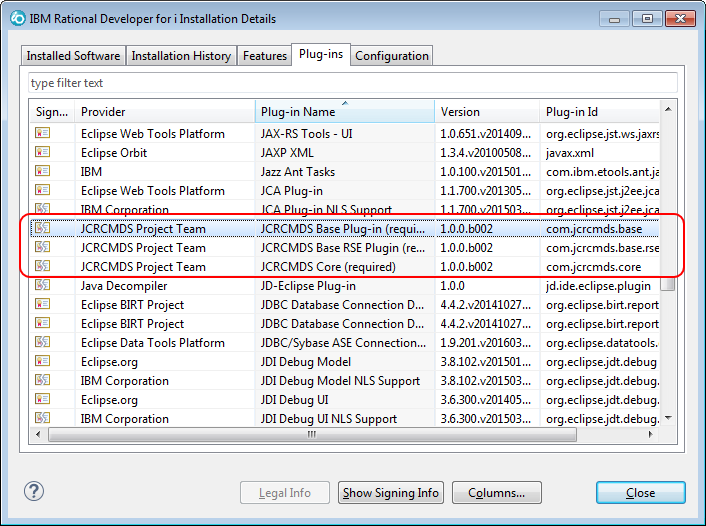
Double click the tab of the "plugin.xml" editor again, to restore the editor, because the Console view look a bit ugly at the moment:



Now you have launched a new Eclipse application in Debug mode. The new application automatically contains the plug-ins that are in the workspace on the main Eclipse application:



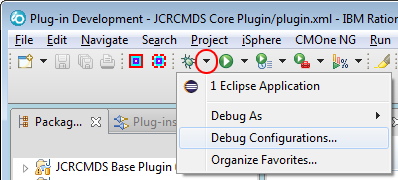
Let us verify, that the JCRCMDS plug-ins have been loaded by the test application. Select "Help -> About IBM Rational Developer for i". The click the [Installation Details] button to check the plug-ins:



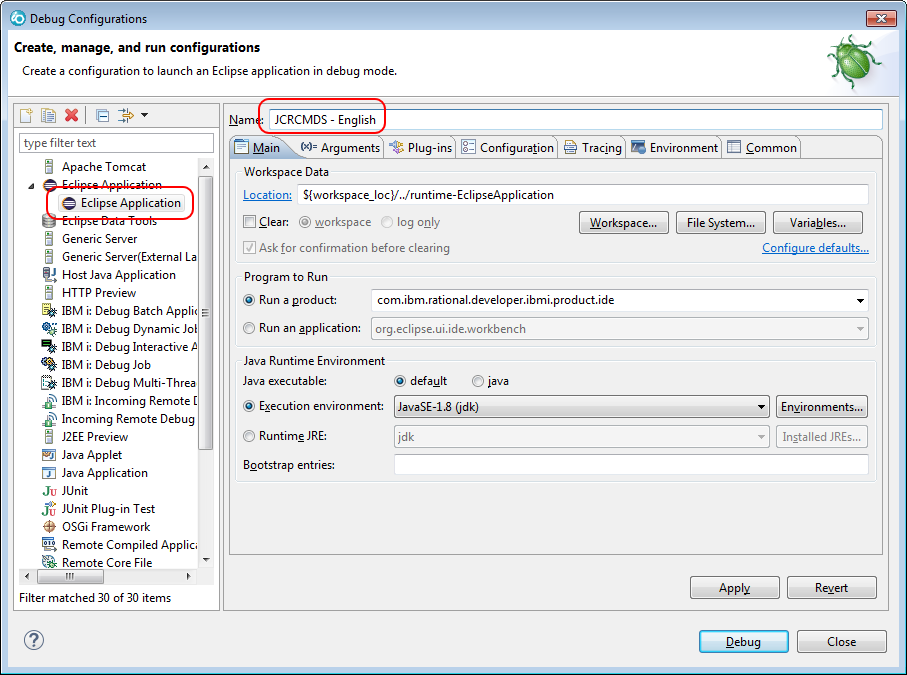
Nice, the plug-ins are there.

Now let is tweak the run/debug configuration of the Eclipse test application. We want to enable Java assertions and we want to specify the language, the test application is started with. Specifying the language is important, when you want to see, whether or not labels have been properly translated, e.g. to German.

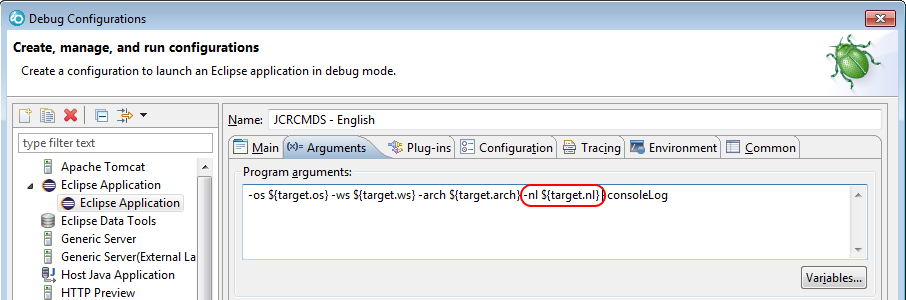
Open the debug configuration:



First let us rename the application to a more descriptive name:



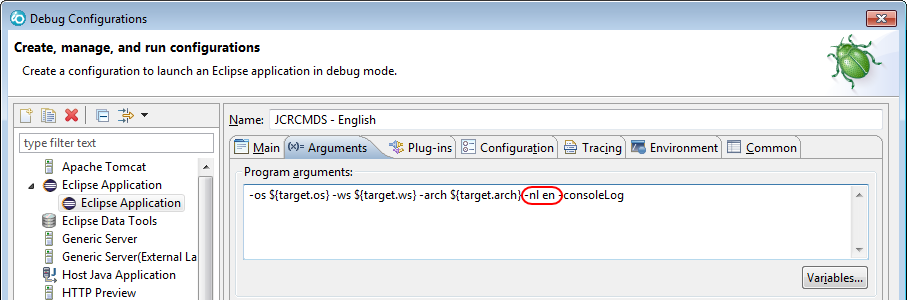
Then let us specify the language, the application is started with:



The "-nl" (national language) parameter specifies the language Eclipse is started with. The default value "${target.nl}" picks the language of the main application. You can change it to the following values, for example:

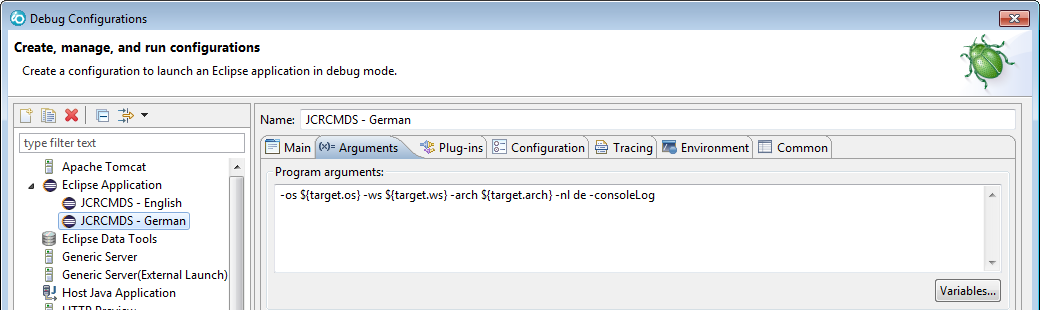
* en - English
* de - German
* it - Italian
* nl - Dutch

Since we renamed the application to "JCRCMDS - English", we change the "-nl" parameter to "en":

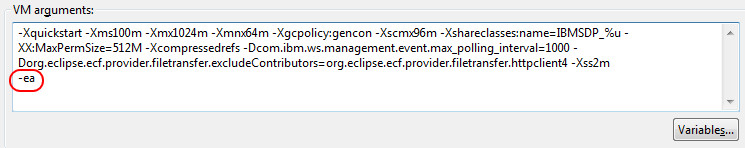


Currently JCRCMDS is developed in English and German. Therefore you may want to copy the debug configuration and change the language parameter to "de" for German:

Now you can start a test application in English and German:



Last but not least we should enable Java assertions, which sometimes are used by developers to verify conditions at certain statements. Add the "-ea" parameter to the "VM arguments" of your English and German debug configuration:

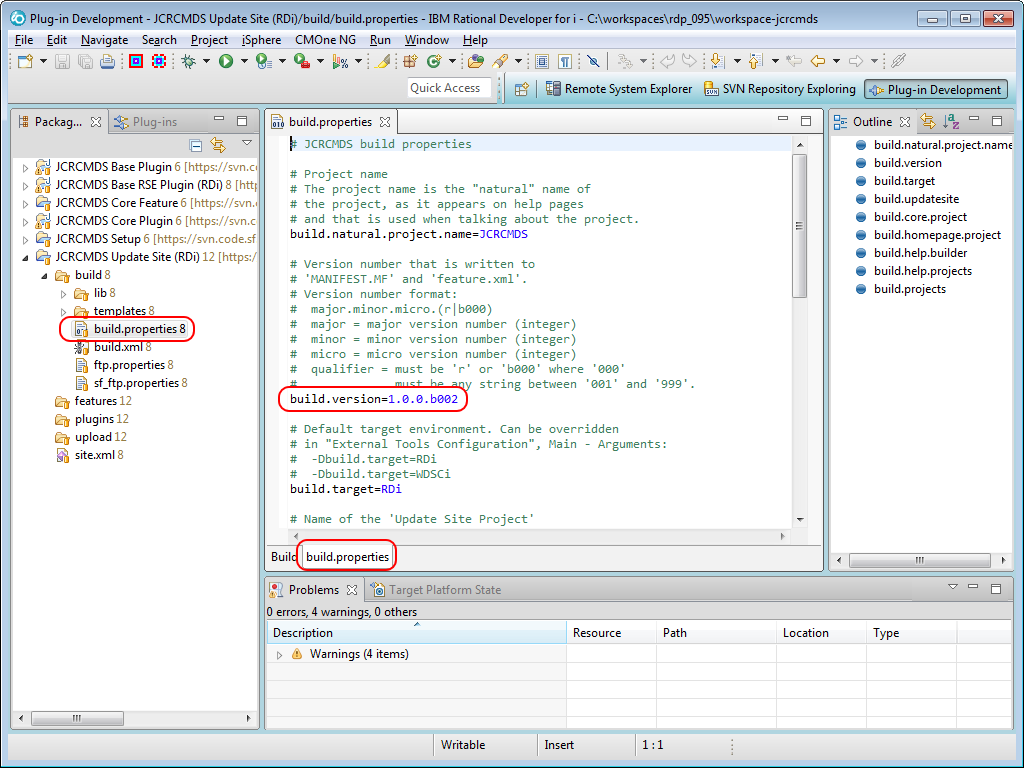


# Creating A New Release

When you finished another feature you may want to create a new beta or final release of the JCRCMDS plug-in. Creating a new release is not that difficult. It is nothing more than a few clicks.

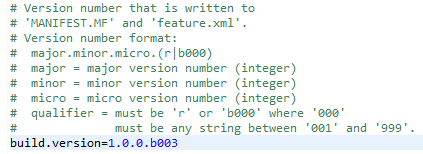
## Setting The Release Number

Expand the "JCRCMDS UpdateSite" project and open the "build.properties" file:

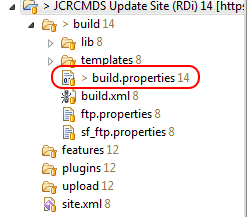


The version number is specified next to the "build.version" key.

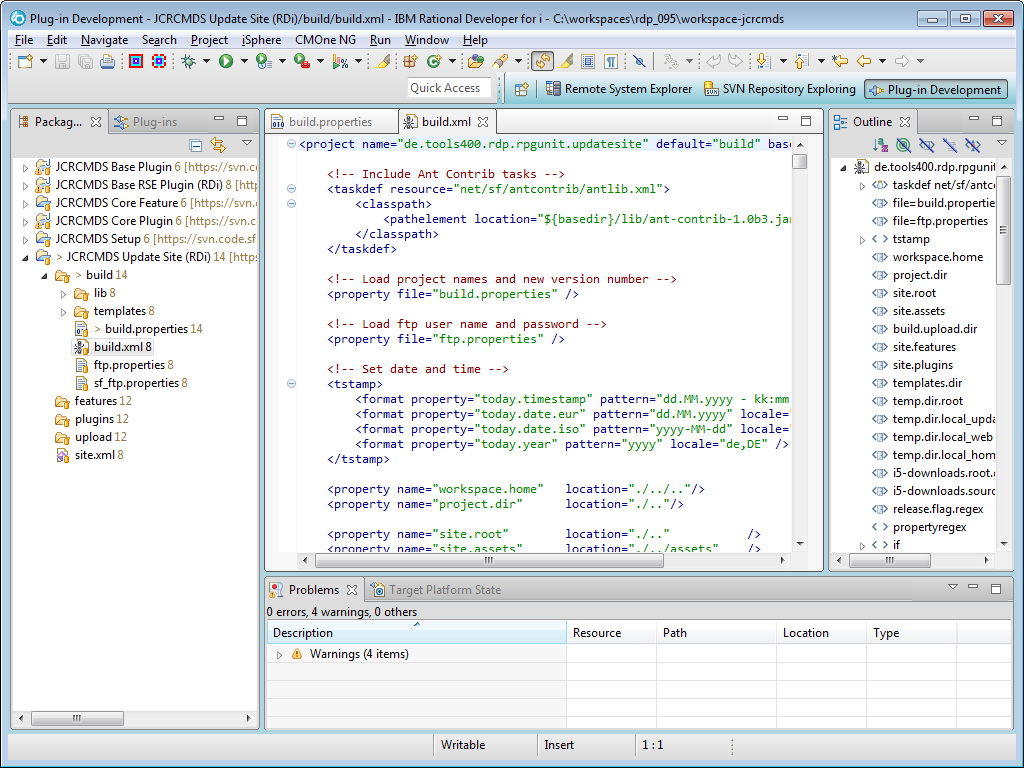
For now let us create a new beta version. For that we increment the sequential beta version number from 2 to 3:



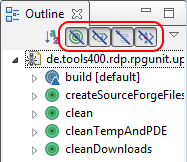
Notice the little greater sign next to "build.properties". It appeared after you have saved the file and it indicates that the file has been changed but not yet committed to the SourceForge repository:



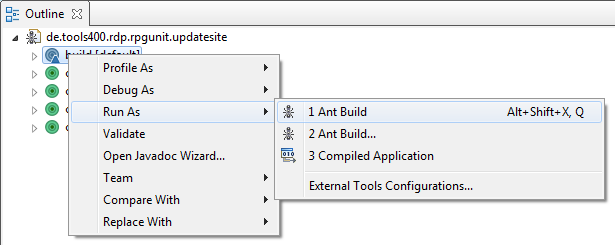
Now open the "build.xml" file:



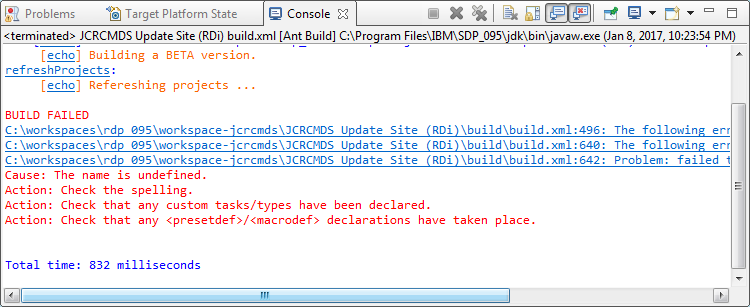
Hide everything but the exported Ant tasks from the Outline view:



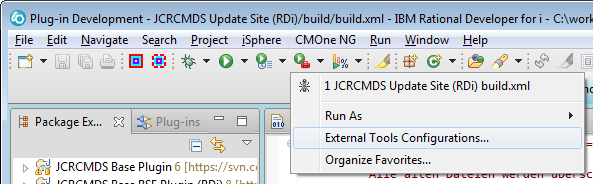
Right-click the "build" item and select "Run As -> Ant Build" from the menu:



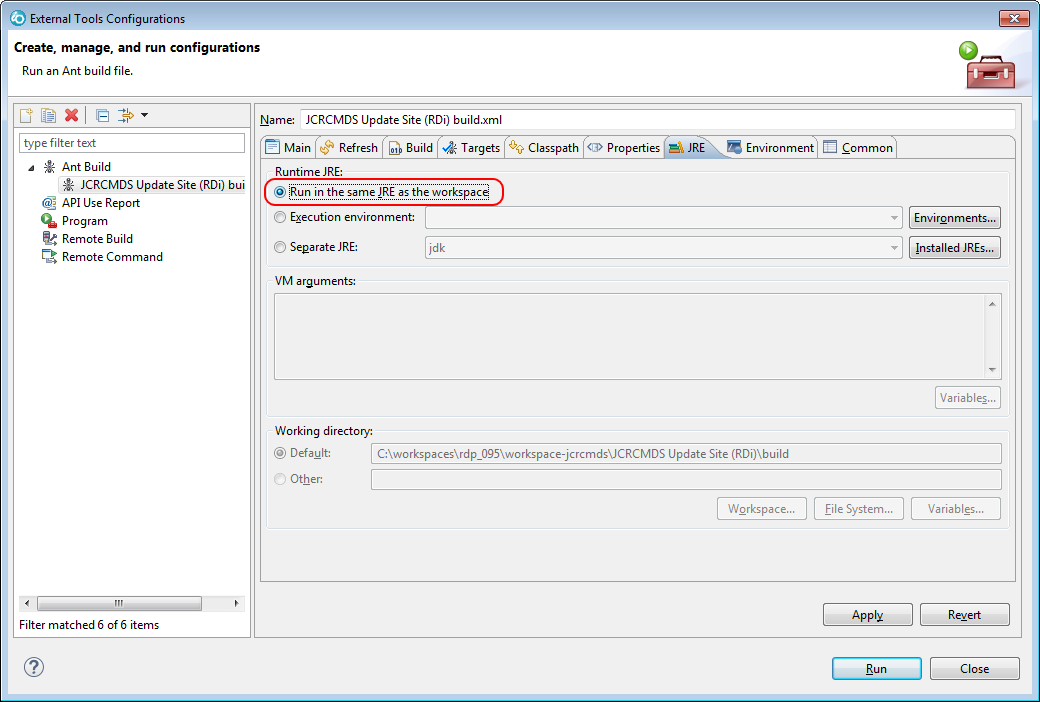
You will receive the following error:



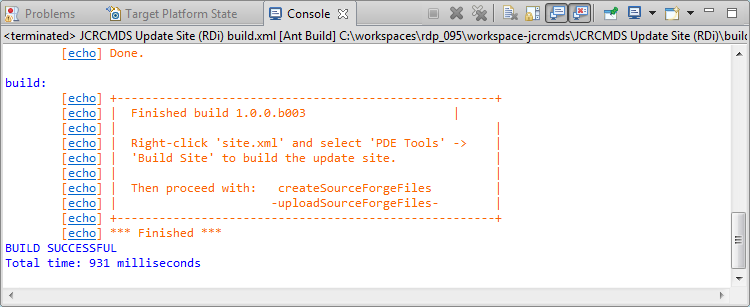
Open the Ant configuration to fix the problem:



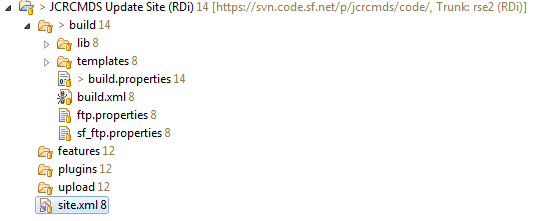
On the "JRE" tab check the "Run in the same JRE as the workspace" radio button:

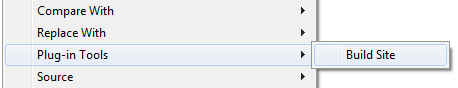


Click the [Apply] and then the [Close] button. Then start the "build" task again. Now the task should finish successfully:

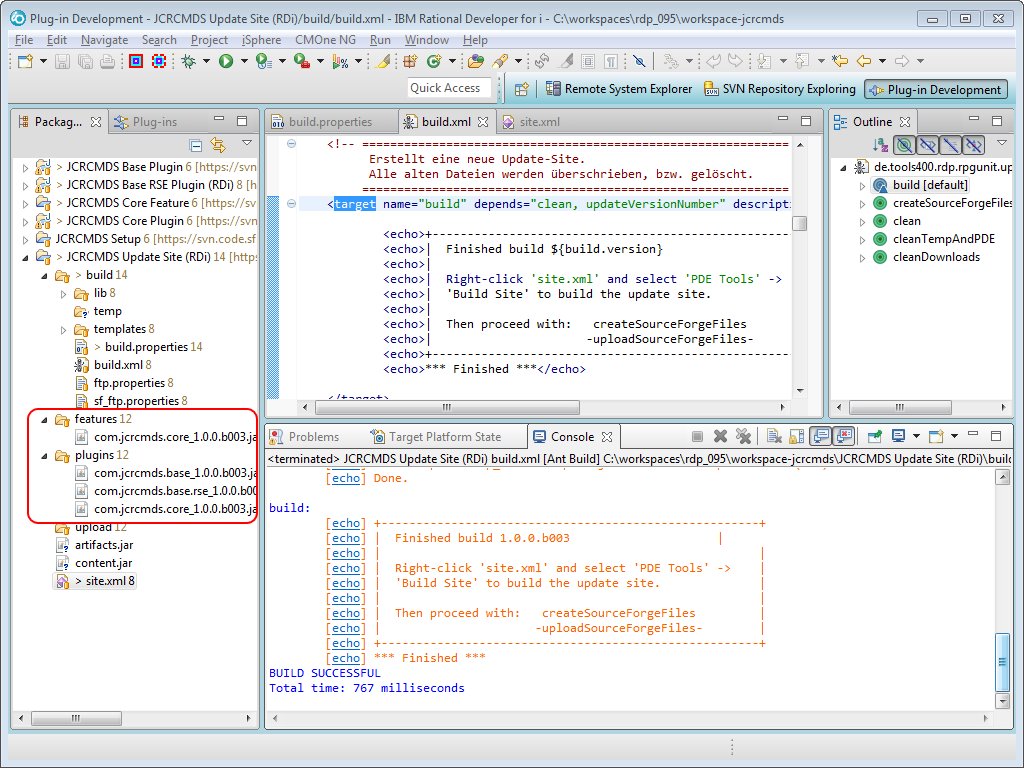


Go ahead and compile the application. Right-click "site.xml" and select "Plug-in Tools -> Build Site":



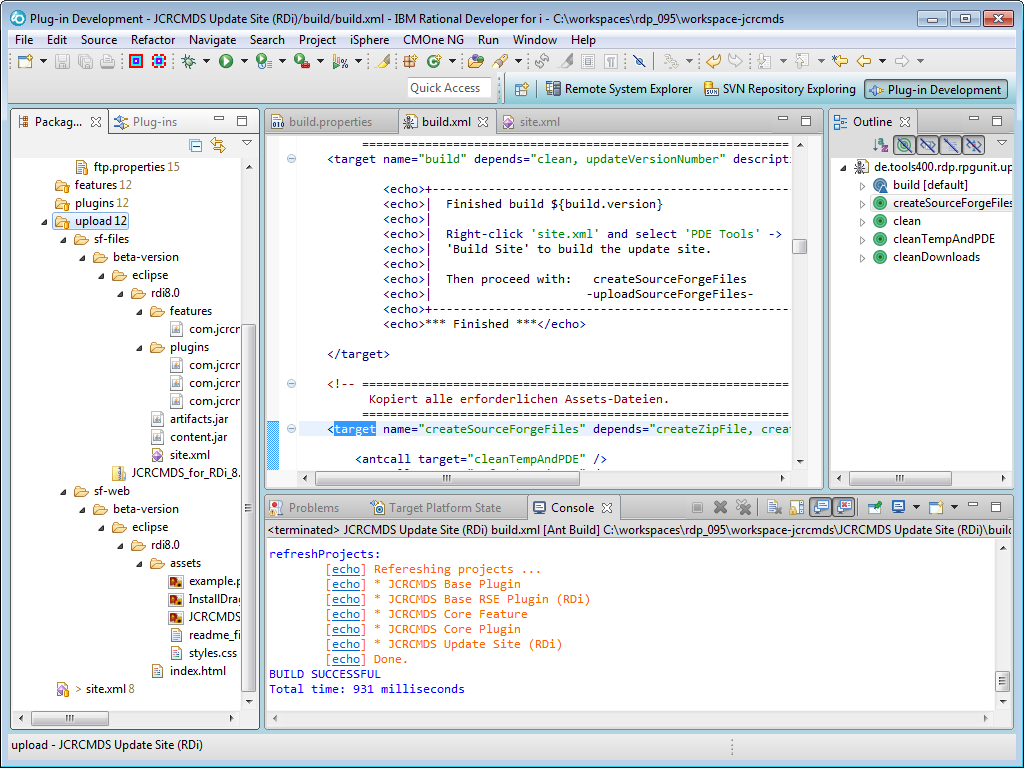


Notice the new files that have been created in the "features" and "plugins" folders:



Also notice all the greater signs next to the project names. There are lots of uncommitted changes.

Now launch the "createSourceForgeFiles" task of "build.xml". Al whole bunch of new files have been created in the "upload" folder:



Before you can upload the beta version, you need to copy "default\_ftp.properties" to "ftp.properties" and add your user name to key ""

