JazzConnect-IntelliJ User Guide

A User's Guide to using JazzConnect-IntelliJ with Rational Team Concert

Table of Contents

Figures

Introduction

CM-Logic's JazzConnect-IntelliJ provides the ability for Java developers to use the highly popular **IntelliJ IDEA IDE** with **IBM Rational Team Concert SCM (RTC SCM).** JazzConnect-IntelliJ provides a rich set of functionality that allows the developer to perform most of the operations that is provided through the RTC Eclipse Client.

Note: *This document presumes the reader has already installed JazzConnect-IntelliJ and has a basic understanding of Rational Team Concert, specifically, the SCM component of Rational Team Concert. If more information is needed using RTC SCM, please refer to jazz.net article #41, which provides a good overview of RTC SCM.*

Prerequisites and Environment Set-up

Prerequisites

The following prerequisites must exist in order to use JazzConnect-IntelliJ:

- Rational Team Concert version 2.0.0.2 or later up to version 5.0.x
- IntelliJ IDEA version 9.0.1 or later up to version 14.x
- Rational Team Concert SCM Tools[1]

Rational Team Concert Set-up

The following must be set up before using JazzConnect-IntelliJ:

- A Project Area must be created with the appropriate Developer licenses, roles, and permissions given to the IntelliJ developers
- The source code must imported to Rational Team Concert using the Eclipse Client, with at least one Repository Workspace created

Note: For the purposes of this user guide, we will use the "`JUnit`" sample Project Area and the "`JUnit Workspace`" Repository Workspace.

[1]Rational Team Concert SCM tools are automatically downloaded with Rational Team Concert Eclipse Client.

Environment Variables

For the JazzConnect-IntelliJ plug-in to work properly, both the path to the JRE and SCM Tools (`lscm`) is required. The SCM Tools should be added to the PATH environment variable.

Example:

- `JDK_HOME= C:\Program Files (x86)\JetBrains\IntelliJ IDEA Community Edition 13.1.3\jre`
- `SCM_TOOLS=C:\IBM\SCMTools\4.0.6\scmtools\eclipse`
- `PATH=%SCM_TOOLS%;%PATH%`

Note: Read the following JetBrains article on Selecting the JDK for more information

In this example, SCM Tools was downloaded from jazz.net. Alternatively, you could also use the SCM Tools that comes with the Rational Team Concert Eclipse Client

Example:

- `SCM_TOOLS=C:\<path to Rational Team Concert client>\scmtools\eclipse`

*Note: The version of SCM Tools **must** match the version of Rational Team Concert where the IntelliJ IDE will be connected. So, for example, if you are using RTC version 4.0.5, then you will need to go to https://jazz.net/downloads/rational-team-concert/releases/4.0.5?p=allDownloads.*
Using the SCM Tools found within the Rational Team Concert client installation will ensure the correct SCM Tools will be used.

IntelliJ IDE Project Set-up using JazzConnect-IntelliJ

Configuring the VCS Default Settings

1. Start IntelliJ IDEA

Figure 1: IntelliJ IDEA Quick Start Window

2. Select "`Configure`" and then "`Settings`"
3. From the Default Settings window, select "`Version Control`">"`Rational Team Concert`".

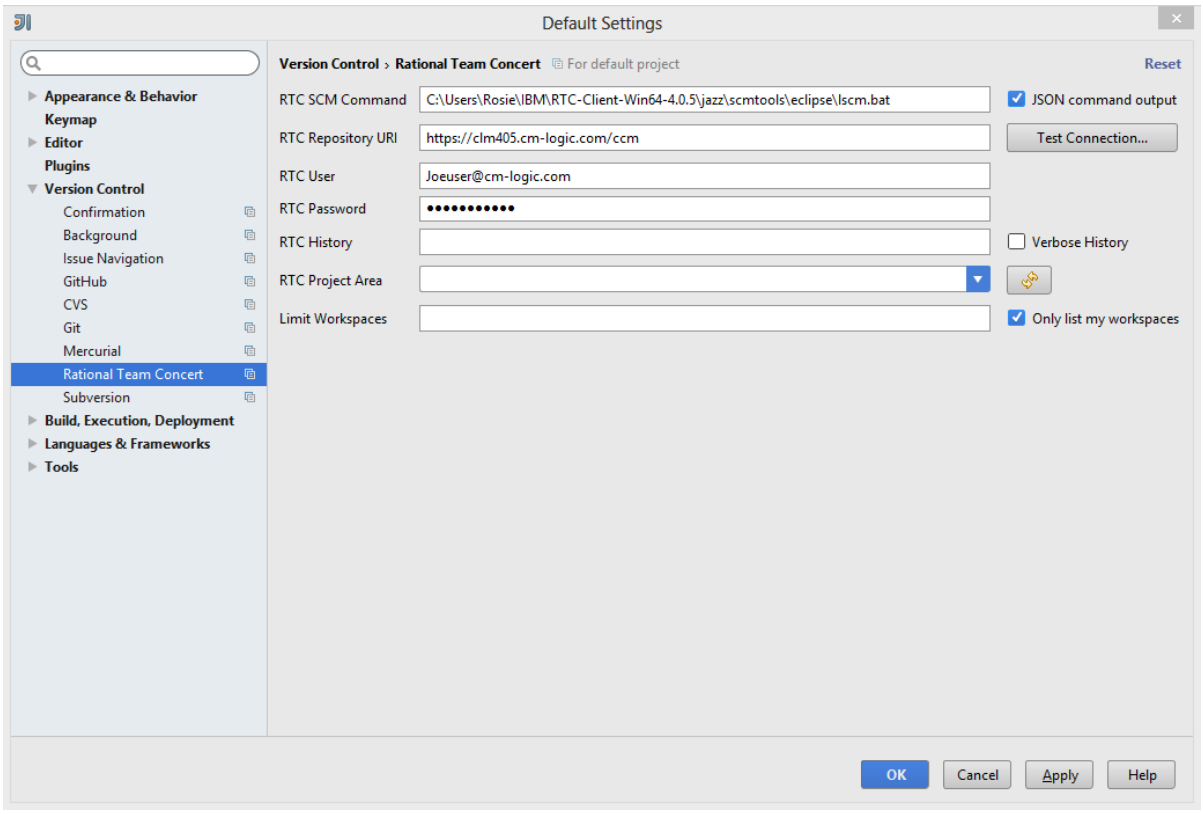

Figure 2: IntelliJ VCS RTC Default Settings

4. Complete the boxes as follows:
   - **RTC SCM Command:** Fill in the PATH to the SCM tools which will be used e.g. `C:\<path to Rational Team Concert client>\scmtools\eclipse\lscm.bat`
   - **RTC Repository URI:** Fill in the URI to the RTC Repository you wish to connect to e.g. `https://clm405.cm-logic.com/ccm`
   - **RTC User:** Fill in the username which you use to access the RTC Repository
   - **RTC Password:** Fill in the password which you use you access the RTC Repository
5. Select "OK"

Creating a new IntelliJ project from source already residing in RTC

1. From the IntelliJ Quick Start Window, select "`Check out from Version Control`", and then "`Rational Team Concert`"

Figure 3: IntelliJ IDEA Quick Start Window

*Note: The message below may appear and is a known issue. To resolve, complete Steps 2 to 3 of "*Configuring the VCS Default Settings*" and press "OK"; this will re-register the VCS adapter and you will then be able to proceed with the checkout.*

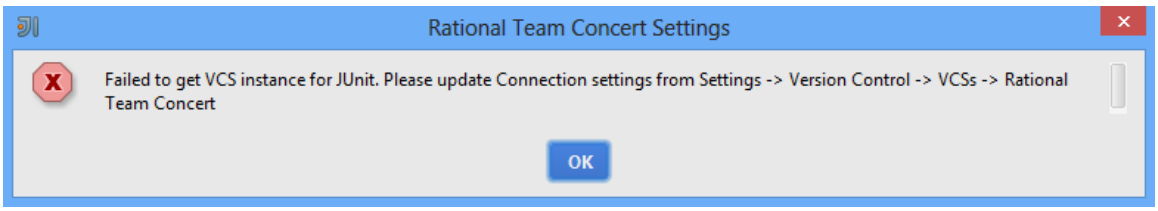

Figure 4: Failed to get VCS instance error

2. In the IntelliJ Checkout window, select:
    - **Repository Workspace:** The RTC Repository Workspace (e.g. "JUnit Work Space") you wish to checkout
    - **Project:** Select the folder where you wish to place the source code from RTC

Note: This should be a separate path from any RTC Eclipse workspaces

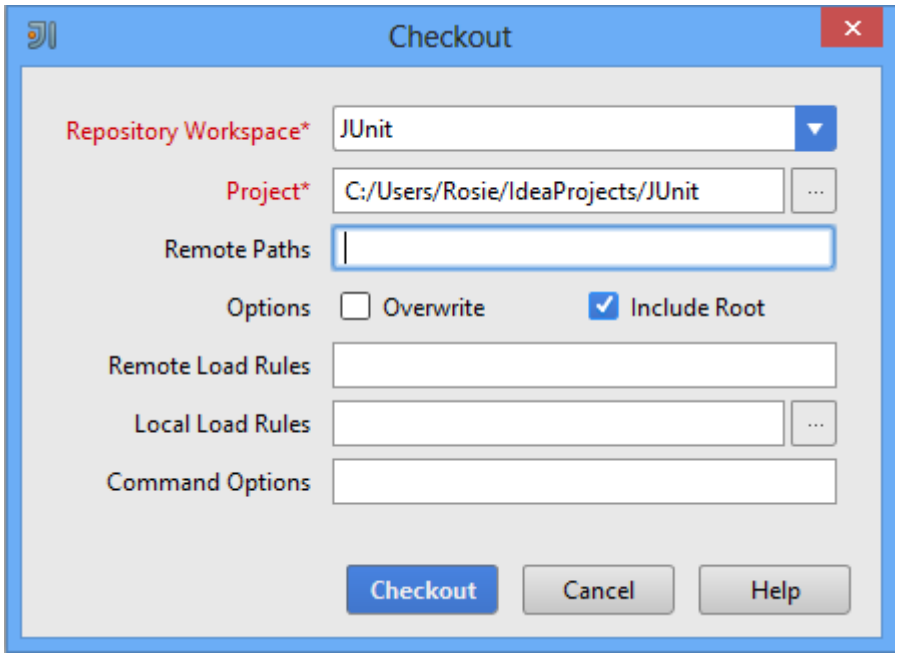

Figure 5: Checkout

3. Then select, "`Checkout`"

Aside: At this point, RTC has created the JUnit workspace. The .jazz5 folder is needed by RTC to keep track of the workspace.

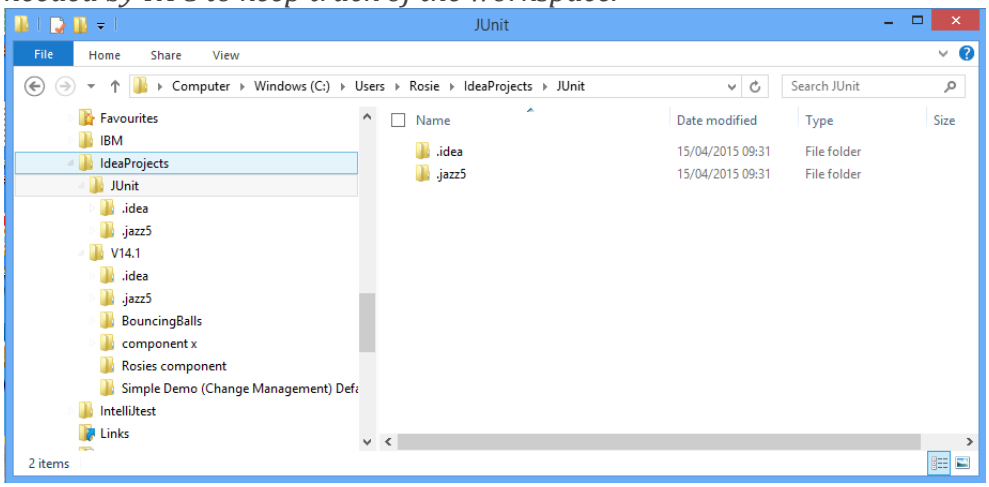

Figure 6: Select Workspace

4. In the Checkout from Version Control window, click "Yes" to create an IDEA project

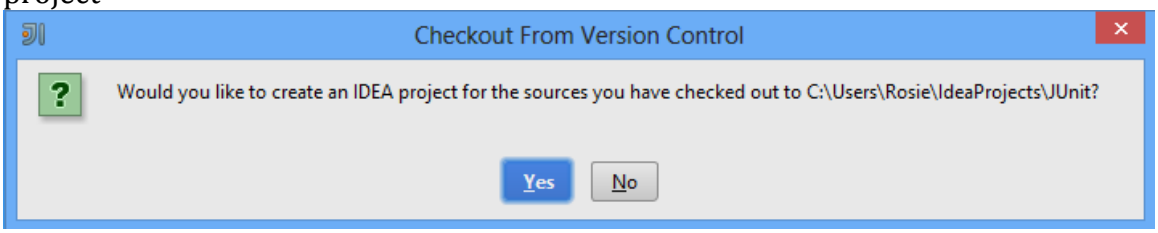

Figure 7: Checkout from Version Control

5. In the "Import Project" window click "Next"

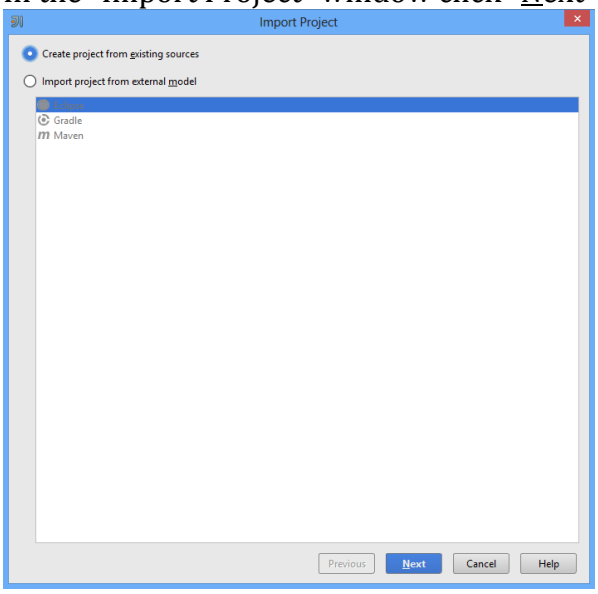

Figure 8: Import project - Create project from existing sources

6. Type in a Project Name to match the project root (e.g. `JUnit`) and click "Next"

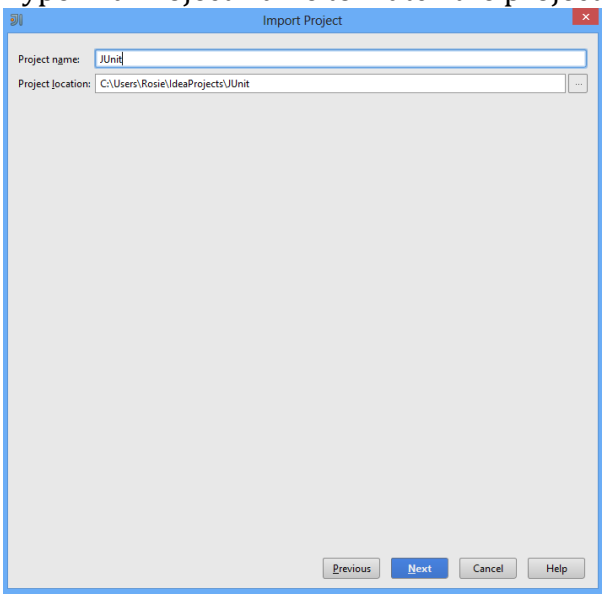

Figure 9: Import project - Project Name and Location

7. Click “Next”

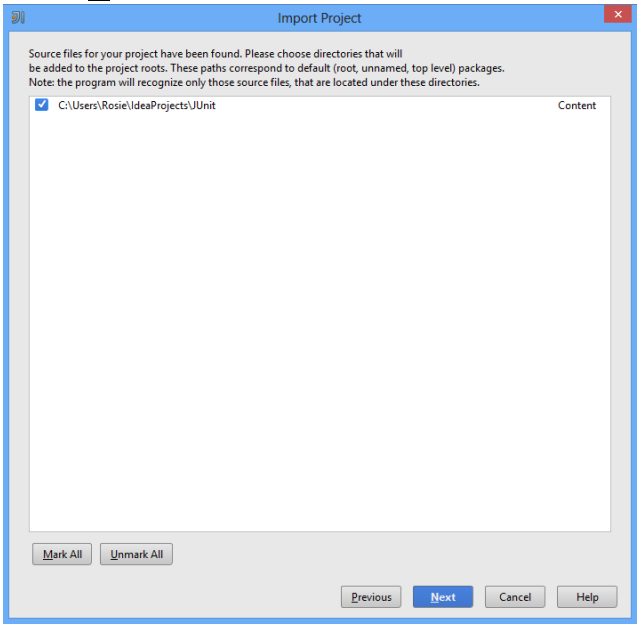

Figure 10: Import project - Source files for import

8. Click “Next”

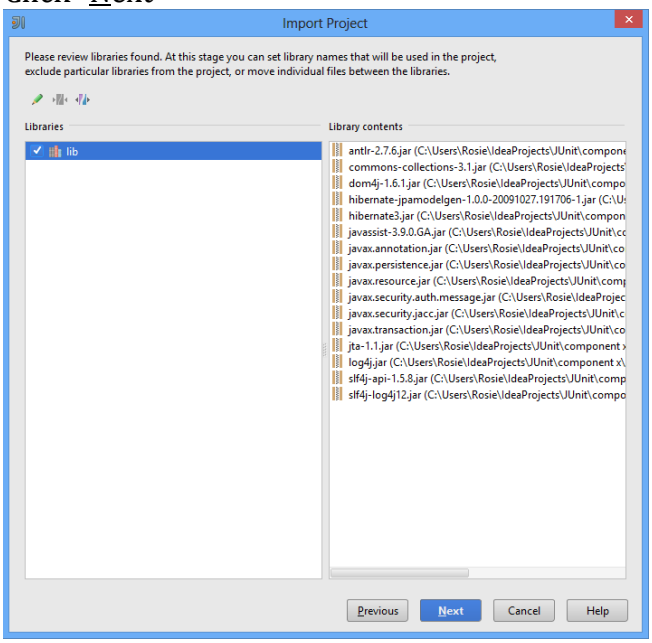

Figure 11: Import project - Review libraries

9. Click "Next"

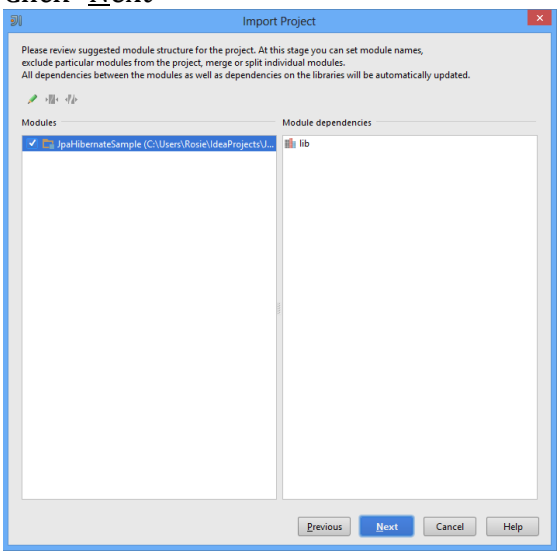

Figure 12: Import Project - Module Structure

10. Choose the appropriate option (If uncertain, click "Overwrite")

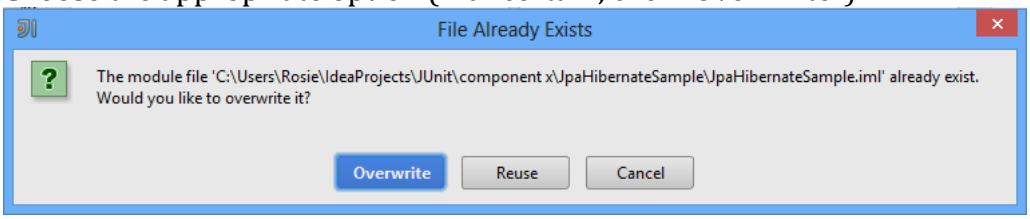

Figure 13: Import Project - File Already Exists

11. Select the appropriate SDK/JDK (or accept default) and click "Next"

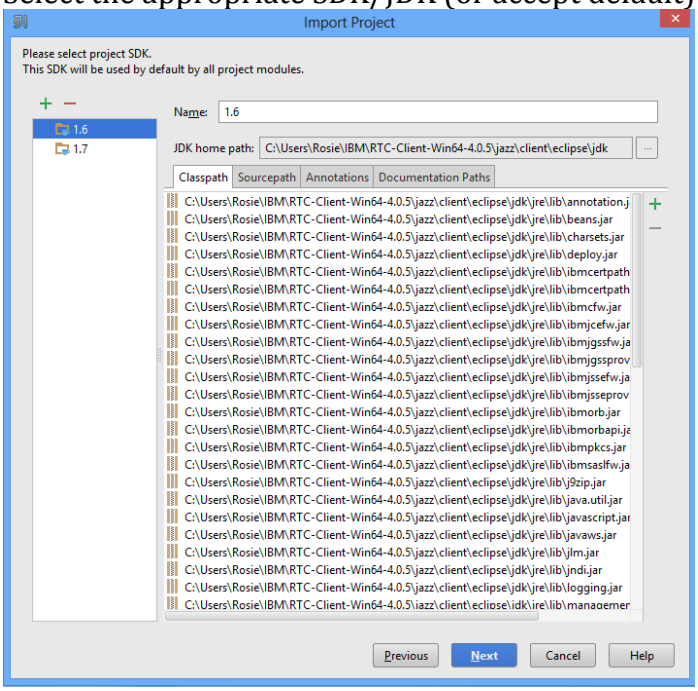

Figure 14: Import Project - Select SDK

Note: To develop applications, you need a Software Development Kit (SDK). A typical SDK example is the Java Development Kit (also known as Java SDK, or JDK). IntelliJ IDEA does not include an SDK. You can use the JDK that comes with the Eclipse Client, or download one from the internet.

12. Click "Finish"

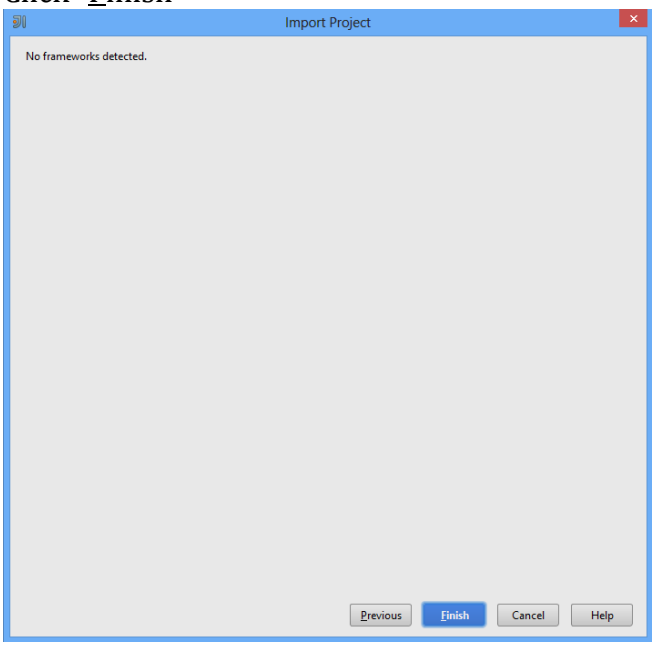

Figure 15: Import Project - Frameworks

13. If everything imported correctly, you should have something like Figure 16: Import Project - After ImportFigure 16: Import Project - After Import

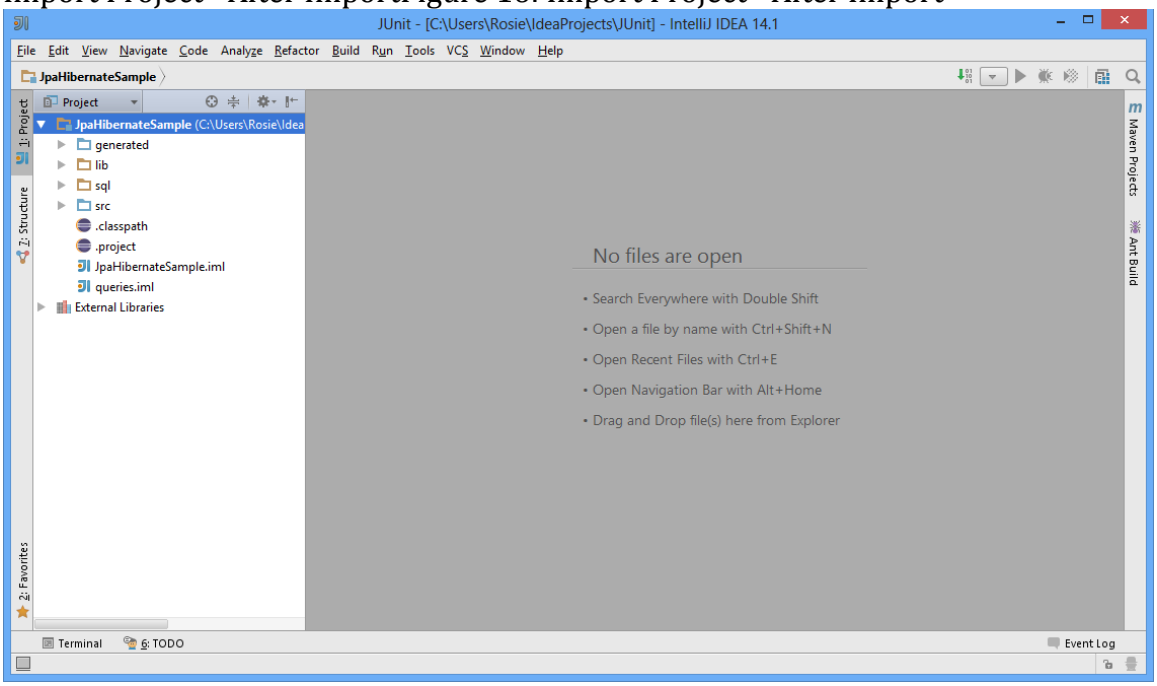

Figure 16: Import Project - After Import

Configure IDEA Project for Rational Team Concert

Each IDEA project must be configured to use a VCS (Version Control System); in this case, Rational Team Concert, as follows:

1. Navigate to "`File`" > "`Settings`"

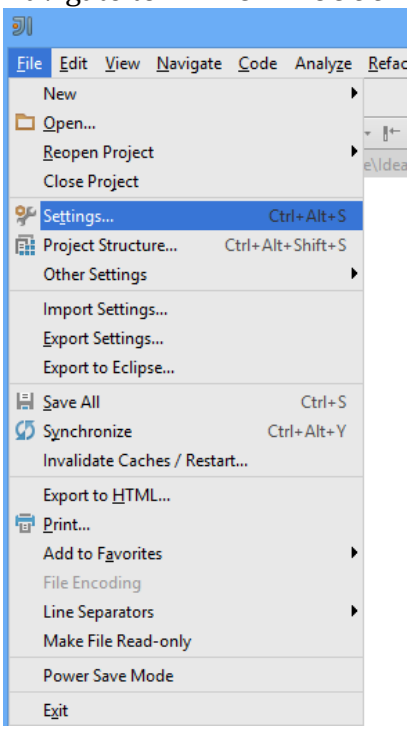

Figure 17: Project Settings Menu

2. Select "`Version Control`"
3. Select "`Rational Team Concert`" in the VCS dropdown for the <Project> content root in the right pane

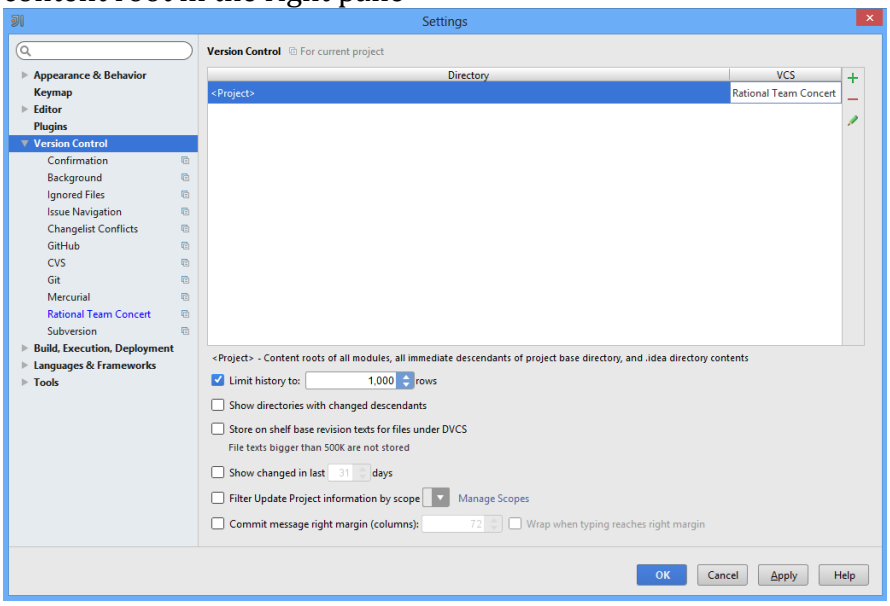

Figure 18: Project Settings Window

4. Click “OK”
5. Notice that you now have a set of options at the bottom pane of the IntelliJ window:

Figure 19: IntelliJ Tool Window Tabs

Connect to RTC Project Area

In order to be able to retrieve create Work Items from IntelliJ and run queries on work items, you must configure the RTC Project Area from which you want to retrieve work item information.

1. Navigate to “`File`”>”`Settings`”
2. Select “`Version Control`” and then “`Rational Team Concert`”
3. Select the Project Area you wish to connect to from the dropdown menu.

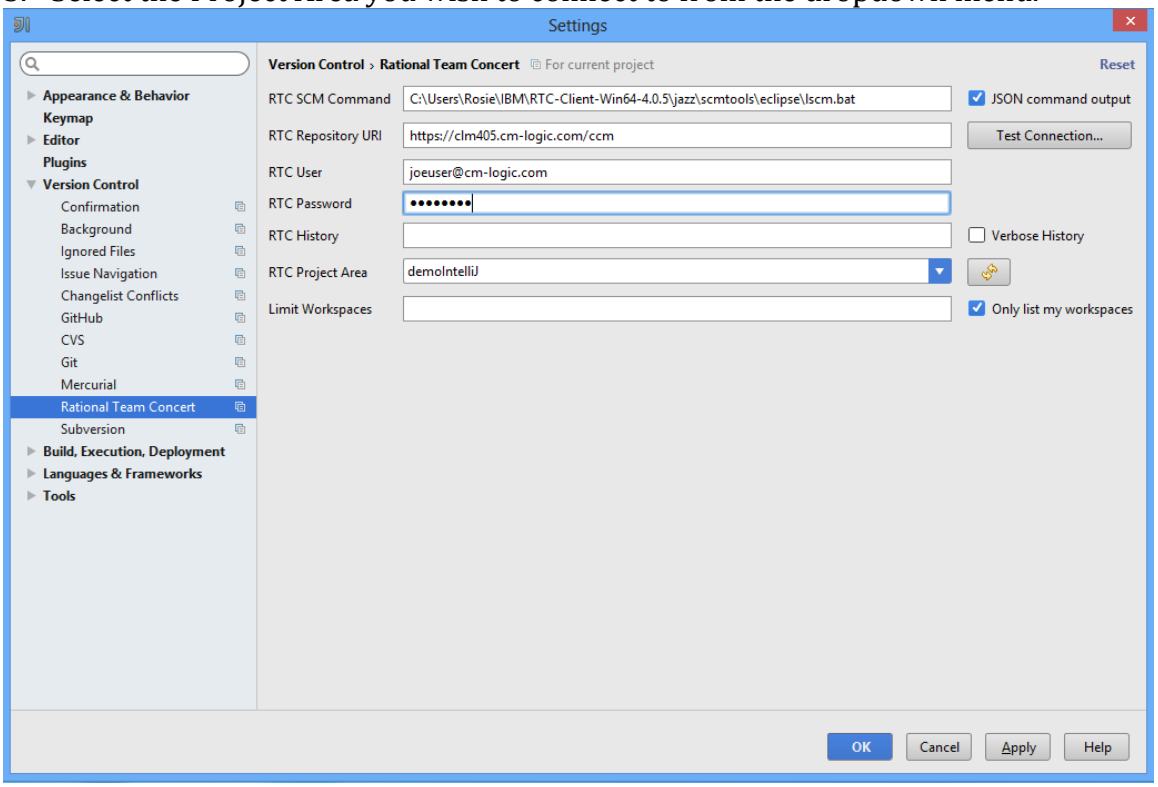

Figure 20: RTC Project Area

4. Click “OK”

Get Latest Pending Changes

The next step is to get the latest Pending Changes. These is done by the following steps:

1. Click on the "`Rtc.PendingChanges`" tab at the bottom of the main window

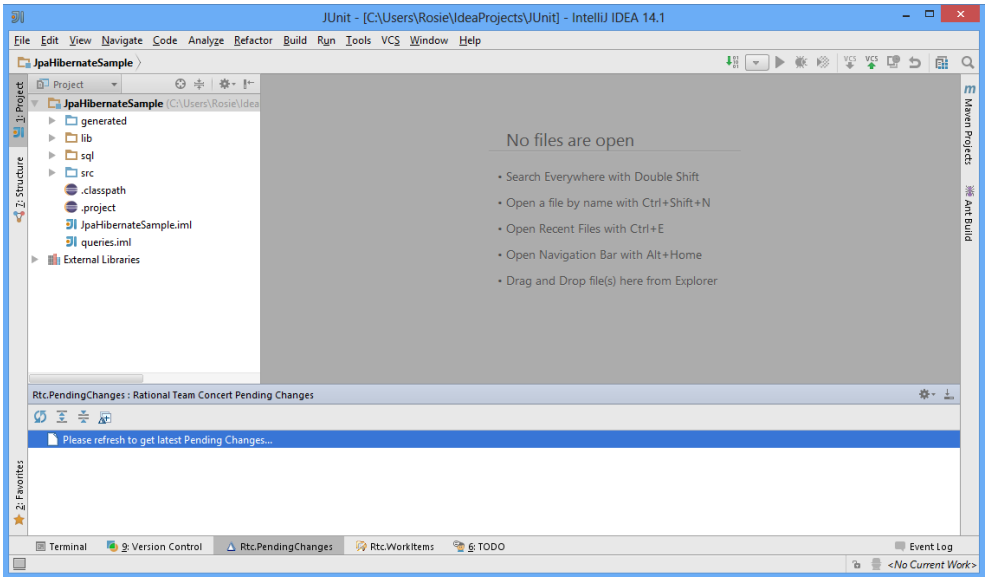

Figure 21: Pending Changes before

2. Click the "`Refresh`" button  :

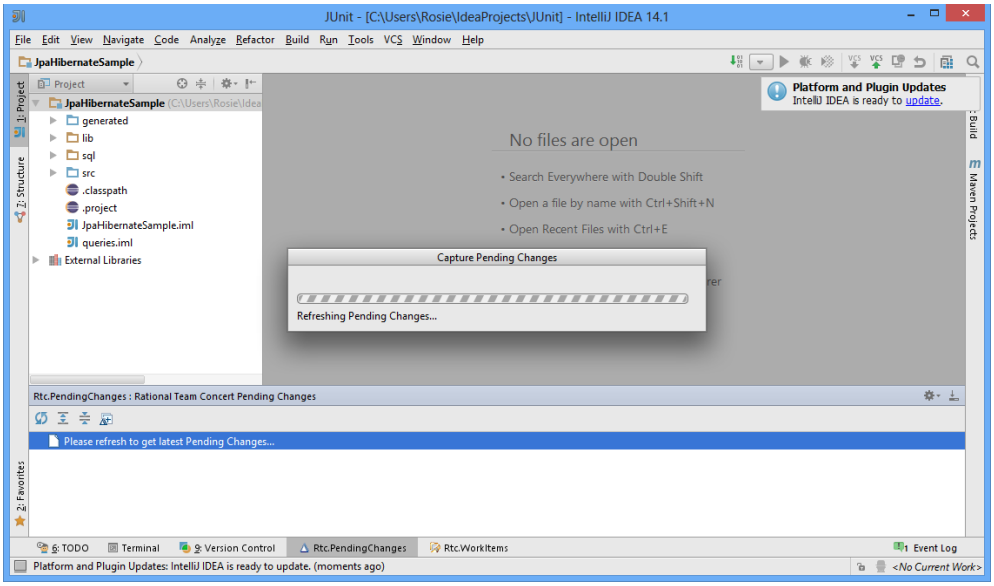

Figure 22: Capture Pending Changes

3. The Pending Changes will then be shown:

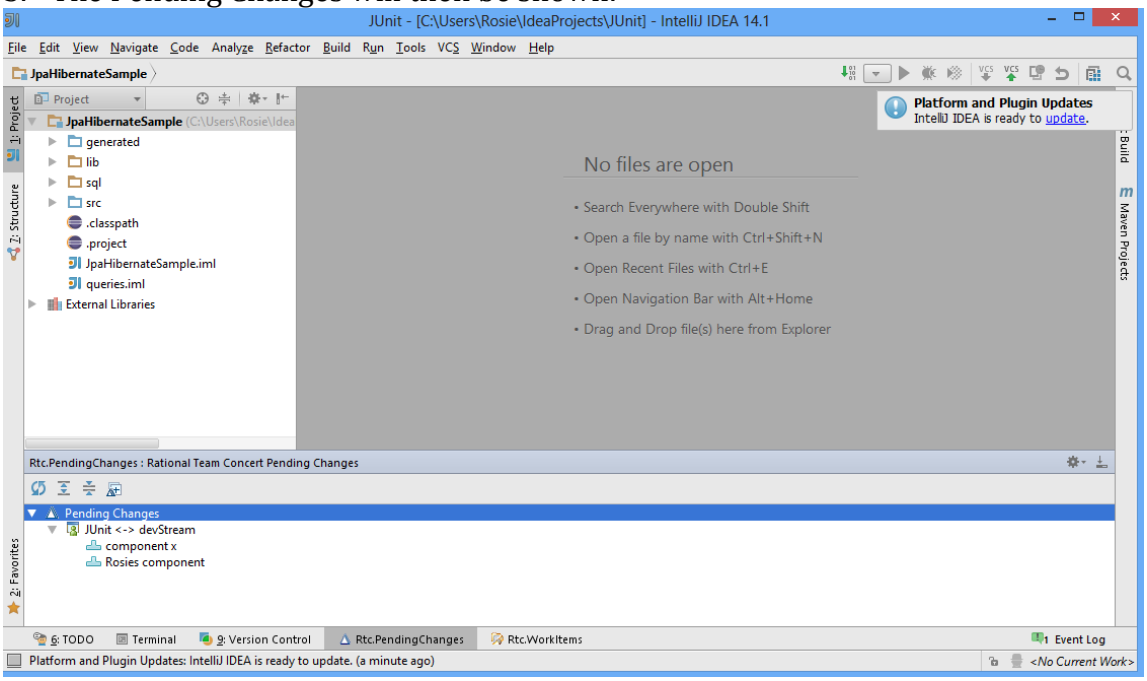

Figure 23: Pending Changes after

Note: There is a known issue where the Pending Changes may not refresh immediately after creating a new project. If this occurs, restart IntelliJ.

Viewing the Change Notes for Rational Team Concert Integration

Details about the Rational Team Concert Integration plugin including current version and change notes can been seen in the settings window:

1. In IntelliJ, navigate to "`File`" then "`Settings`"
2. Select "`Plugins`"
3. Scroll down to find and select the "`Rational Team Concert Integration`" plugin
4. The changes for each new version are listed on the right-hand panel

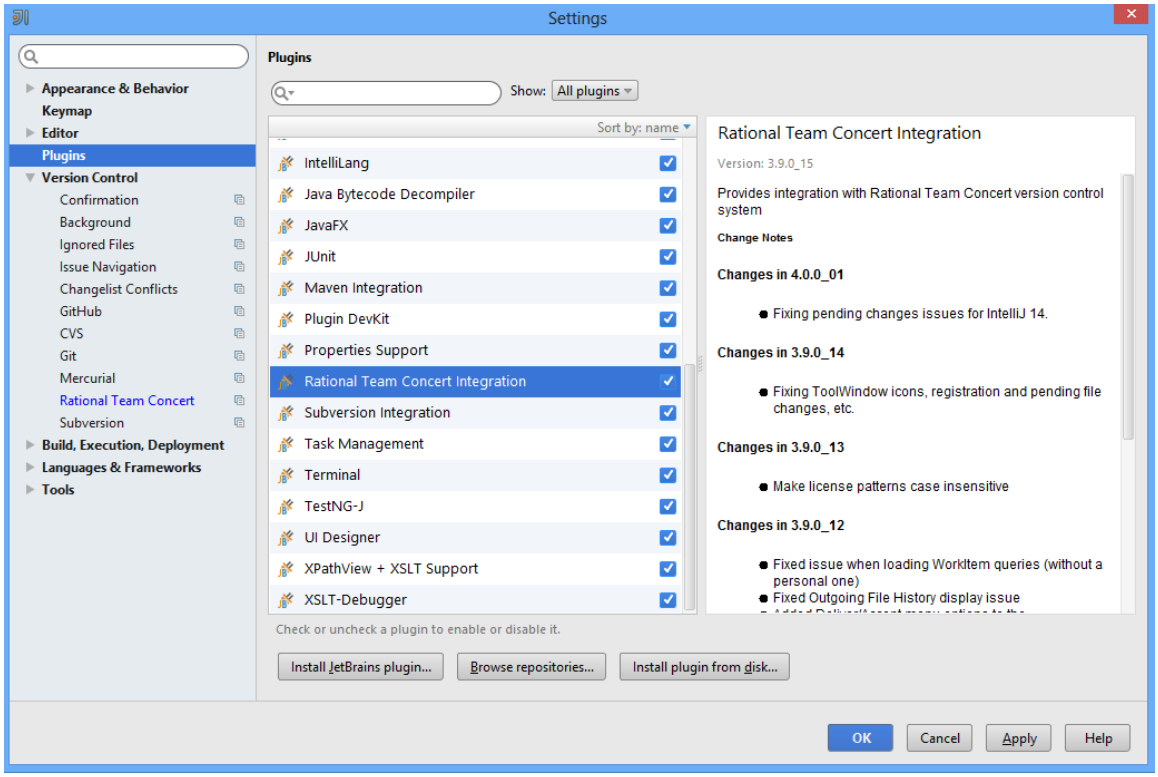

Figure 24: Viewing Changes to the RTC Integration Plugin

Rational Team Concert SCM Overview

This section will provide a high-level overview of the SCM system of Rational Team Concert. As always, a picture is worth a 1,000 words…

Let's take a look at what the source control model looks like in RTC:

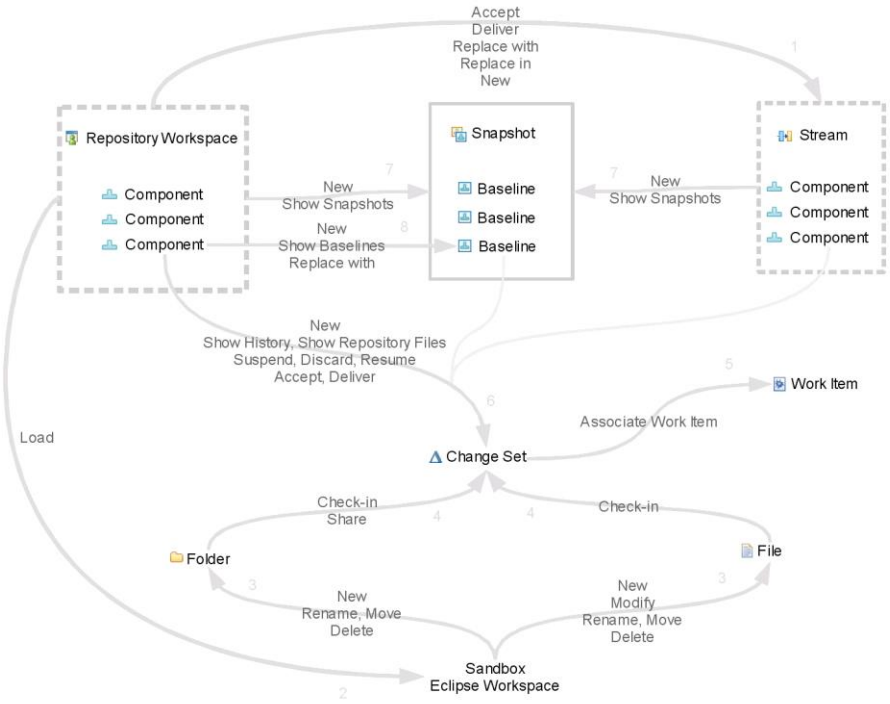

Figure 25: Source Control in Rational Team Concert

In the context of using RTC with JazzConnect-IntelliJ, replace the "Sandbox Eclipse Workspace" in the diagram above with JetBrains IntelliJ; i.e. think of IntelliJ as another client to RTC SCM. Most SCM operations you perform in Eclipse can be done with JazzConnect-IntelliJ.

Change set

A change set is a collection of file or directory changes that are grouped together to form a single unit. It is this complete unit that is committed to a single component and associated with work item(s). Change sets are created when you check-in changes from your local workspace into a repository workspace.

Component

This is the final destination for all change sets and baselines. When using Eclipse or IntelliJ, files and directories are shared into a component at the project level.

Stream

Use streams to create isolated configurations of change sets and baselines through which users can deliver their change sets to a component. Depending upon the process configuration of the IBM Rational Team Concert project, all users can create streams or a restricted subset.

As new change sets are delivered, the stream configuration updates to reflect its evolution. By comparing the differences between the stream and the repository workspace configuration rules, the system identifies change sets that users must accept into their repository workspace to maintain synchronization with the stream.

A stream is also a container for snapshot information.

Repository workspace

A repository workspace is a storage area on the Rational Team Concert server. The workspace contains a configuration consisting of components and baselines, storage for holding undelivered change sets and baselines (not yet stored within a component), and storage for snapshots not yet promoted to a stream. Change sets that are in a repository workspace can be delivered to a corresponding component.

Local workspace

This is a directory on the user's local workstation where they have loaded a repository workspace. Any changes that users make in their local workspaces are identified and highlighted as being candidates for being checked into the loaded repository workspace where they are grouped within a change set.

Baseline

When a repository workspace represents a configuration of file or directory versions that users want to share with others or mark for future use, then a baseline must be created. These markers are delivered to the component in the same manner that change sets are. A baseline can only be applied to change sets that reside within a single component.

Flow target

Both repository workspaces and streams can have a flow target relationship that is defined to another repository workspace or stream. Each repository workspace or stream can have many of these flow targets defined, but only one is active at any time. It is the active flow target that allows incoming and outgoing changes to be calculated.

A Day in the life of a developer using JazzConnect-IntelliJ

These sections will walk through using JazzConnect-IntelliJ with Rational Team Concert (RTC). We will use the `JUnit Project` in RTC for these examples.

The next section will walk through the following scenario:

1. Create a new Work Item and set it as the current work item
2. Rename a misspelled method
3. Compare, Check-in and create new Change set
4. Deliver Change set

Creating a new Work Item and setting it as the current work item

You can create a new Work Item directly through IntelliJ. Once the Work Item is created, you can set it as the Current Work Item. As with the Eclipse client, when you make changes, the changes are recorded to the current Work Item automatically. You can also set an existing Work Item as the Current Work Item.

Creating a Work Item

There are a few places where Work Items can be created, the most common from the Pending Changes or Work Items tabs. To create a new work item from the Pending Changes tab:

1. Right Click on "`Pending Changes`" in the Pending Changes tab.
2. Select "`Create new WorkItem`"
3. Select the appropriate work item type (In this example we will create a Task)

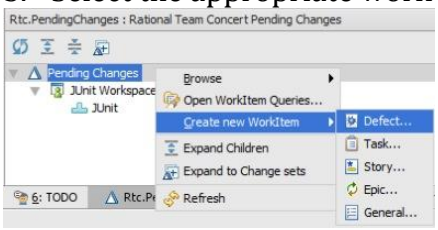

Figure 26: Create Work Item context menu

4. This will open up a new work item in RTC

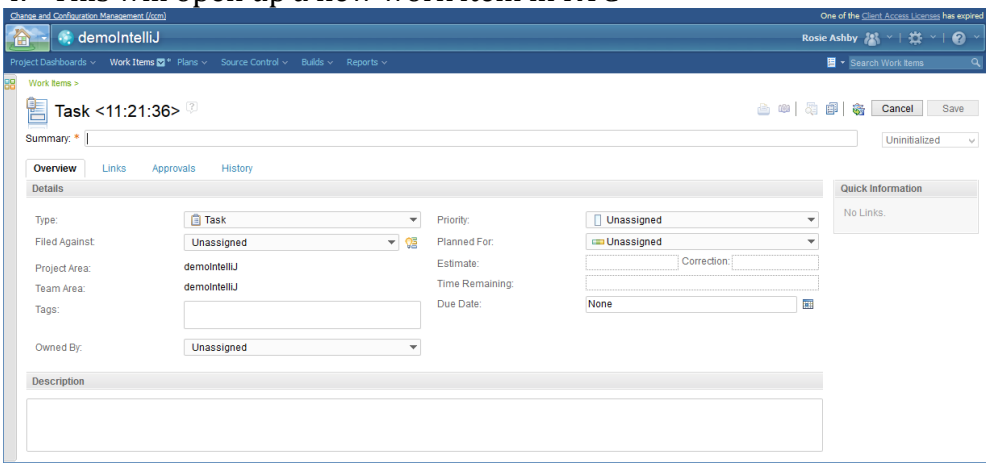

Figure 27: New Work Item

5. Complete the work item details and click "Save"

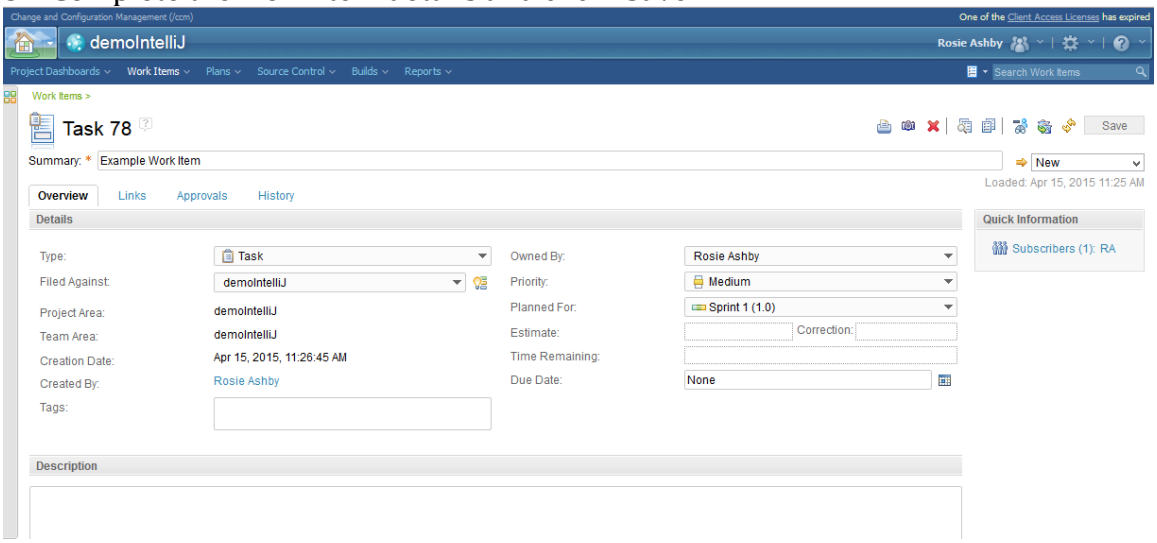

Figure 28: Saved Work Item

Setting a Work Item as the Current Work Item

Moving back to the IntelliJ window, notice at the bottom right of the IntelliJ main window that `No Current Work` has been set:

Figure 29: Current Work not set

In order to set the Work Item and the Current Work Item:

1. Navigate to the "`RTC.WorkItems`" tab
2. Click the "Refresh" button to refresh the work item queries
3. Right click the query you wish to run and select "`Run`"

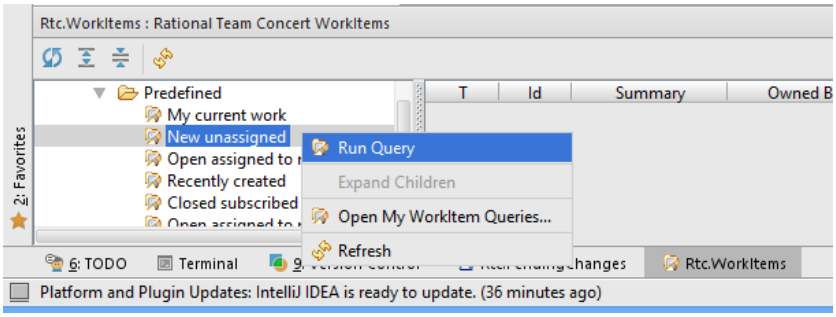

Figure 30: Run Query

4. A list of Work Items matching the query will appear:

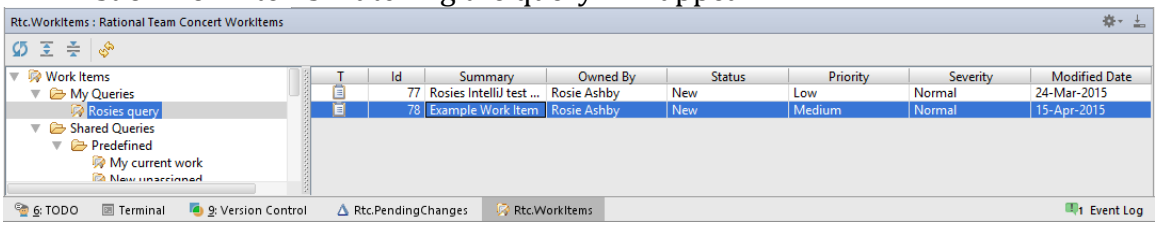

Figure 31: List of Work Items

5. Set the Work Item as the Current Work Item by right-clicking on the targeted Work Item and select "`Set as Current Work Item`"

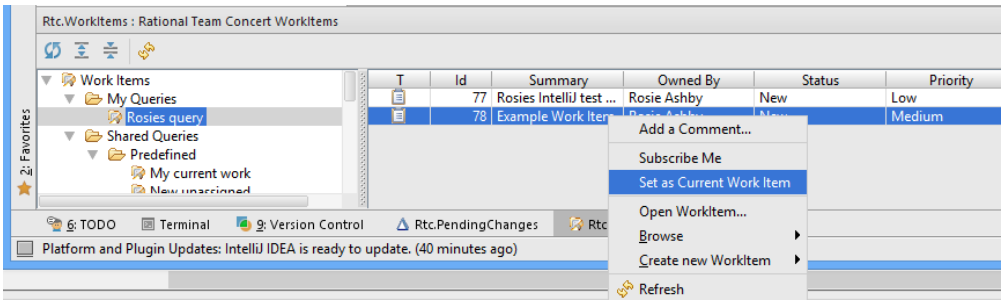

Figure 32: Set as Current Work Item

6. Notice at the bottom right of the IntelliJ main window that the Current Work Item has now been set:

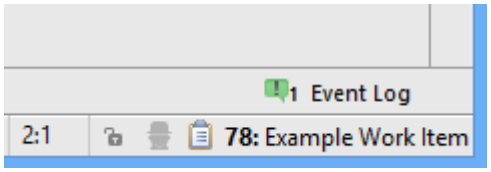

Figure 33: Current Work Item set

Making a Source Code change

The next step is to make a source code change, in this case, renaming a misspelled method. In this example, the "`testSubtraction`" method of the Math class is misspelled as "`testSubstraction`".

1. Right-click the misspelled method and select "`Spelling`"> "`Typo: Rename to…`"

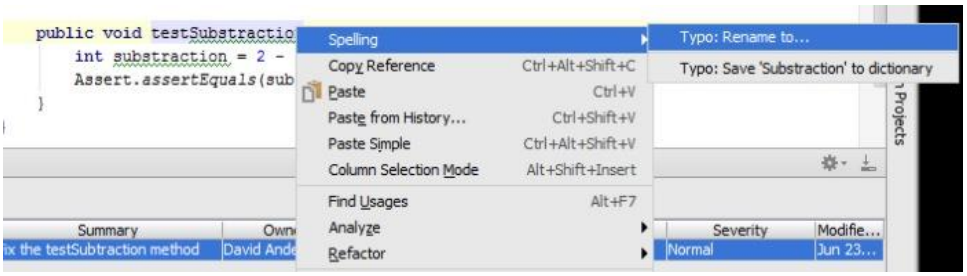

Figure 34: IntelliJ correct Spelling

2. A list of suggested spellings will appear. Double click the correct spelling to change

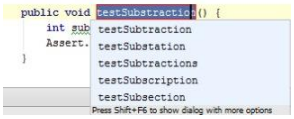

Figure 35: Inline Spelling

3. Repeat to fix the integer variable misspelling (`substraction`)

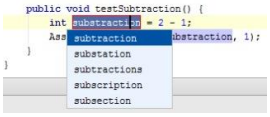

Figure 36: Inline Spelling (2)

Compare, Check-in and create a new Change set

Now that we have made the changes they will need to be checked-in so that others can see the changes that have been made. First we need to compare changes, then check-in the changes and create a new change set.

Compare Changes

1. Save the changes either via "`File`">"`Save All`" or by hitting "`ctrl`"+"`s`" on your keyboard. In this example, we save `Math.java`
2. Refresh Pending Changes by clicking the "`Refresh`" button in the Pending Changes tab. (Notice how RTC detects the change made to `Math.java`)

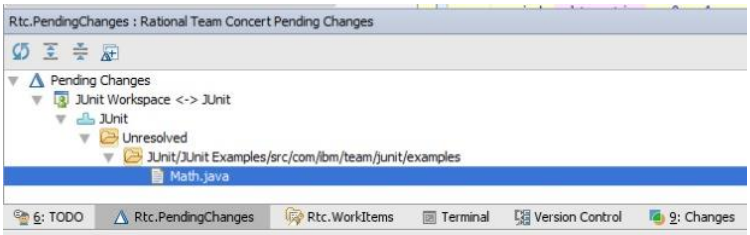

Figure 37: Refresh Pending Changes (Math.java)

3. Right-click on `Math.java` and Select "`Compare...`"

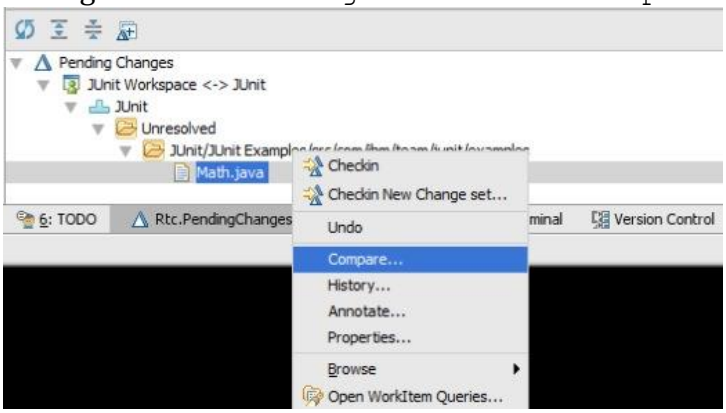

Figure 38: IntelliJ Compare...

4. This opens the "`Compare and Merge`" window in Eclipse

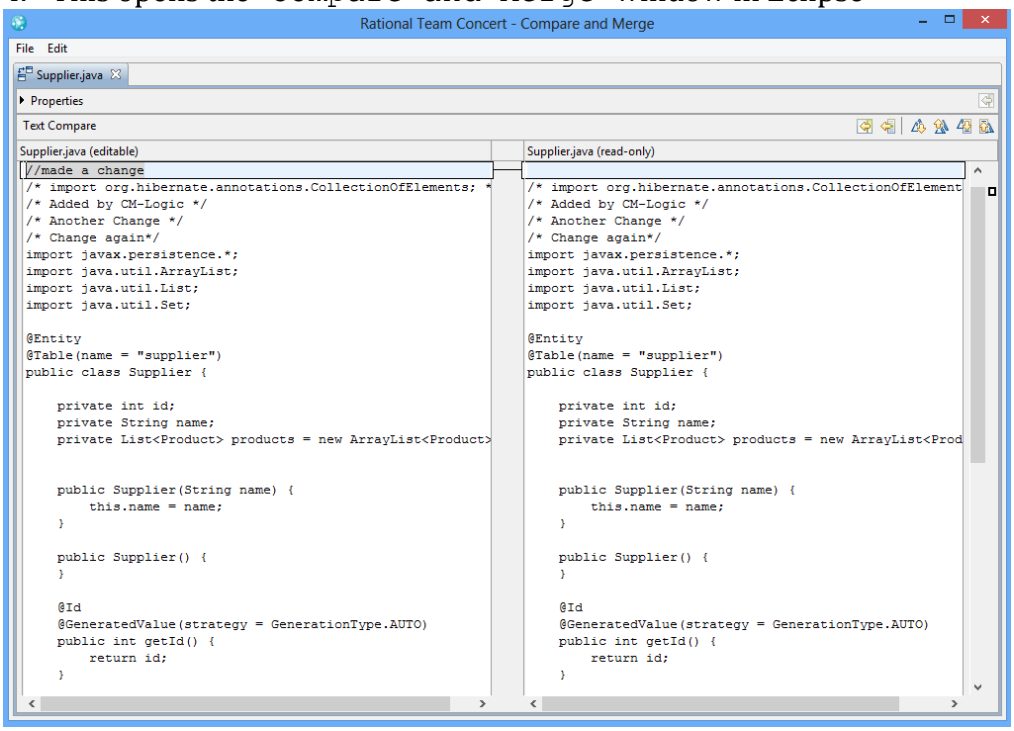

Figure 39: RTC Compare and Merge

So far, so good. We are happy with the changes and so decide to check-in our change and create a new change set.

Check-in and create a new Change set

1. Back in the IntelliJ window, create a new change set by right clicking on the "`Unresolved`" folder and selecting "`Checkin New Changeset…`"

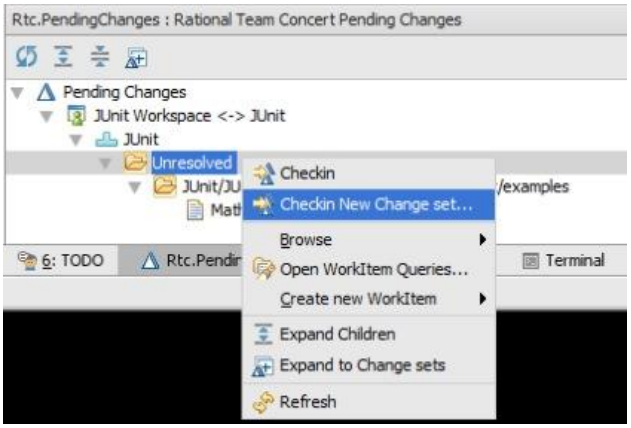

Figure 40: Check-in and create new Change set

2. Enter a description of the change made in the comment box

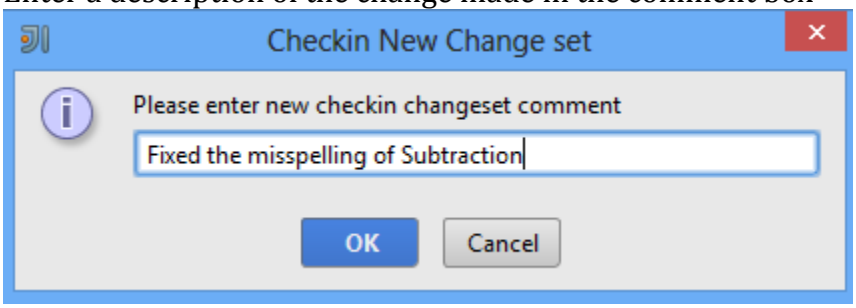

Figure 41: Change set comment

3. Notice the following:
   - The "`Unresolved`" folder is now titled "`Outgoing`", which holds the change sets ready to be "`Delivered`"
   - Under the "`Outgoing`" folder, the changes are arranged by Change set
   - Under each Change set, we have:
     - The folders and their files that were modified; in this case `Math.java` in the "examples" folder
     - The associated Work Item [Work Item 367]

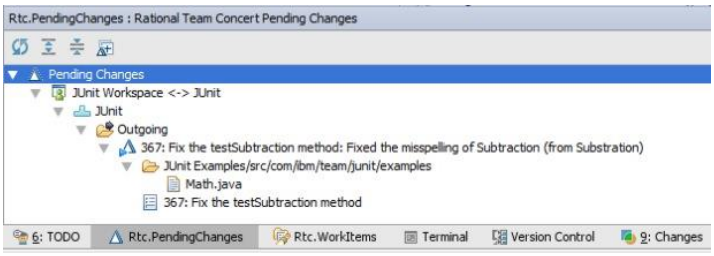

Figure 42: Outgoing Pending Changes

4. Finally, let's view the associated Work Item to see that the Change set has been linked to Work Item 367

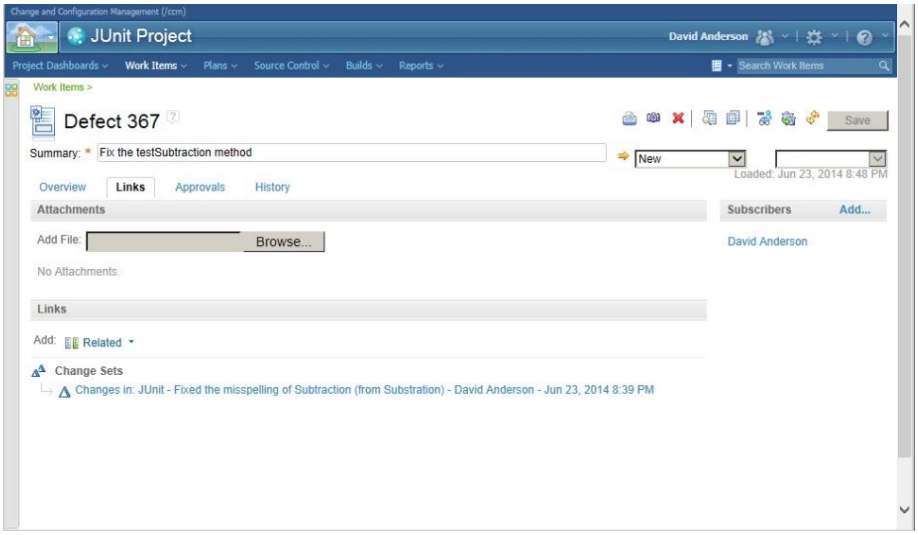

Figure 43: Work Item change set link

5. Clicking on the Change set link, will give you the following:

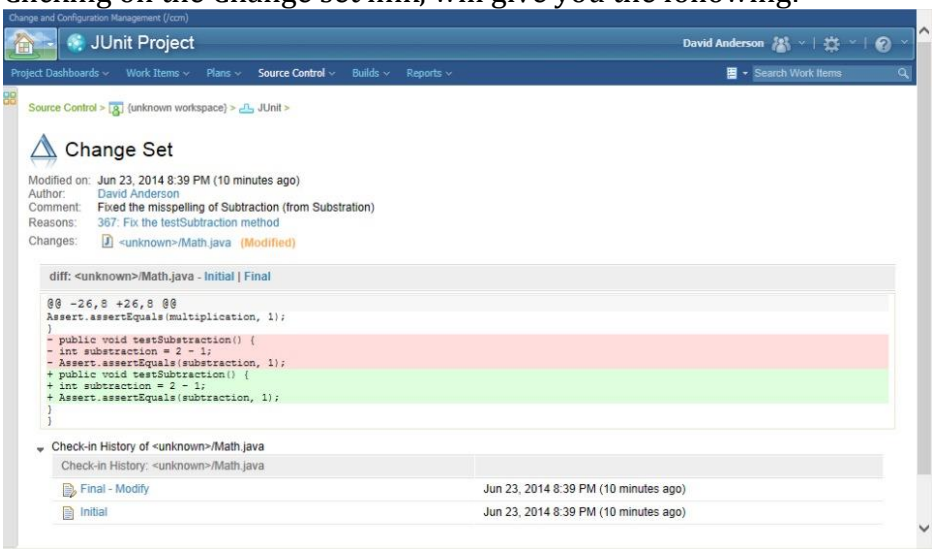

Figure 44: Work Item change set view

Reassigning a change set to a different Work Item

If you decide that the change set is associated to the incorrect Work Item, you can disassociate the change set from one Work Item and reassign it to another Work Item, as follows:

1. Right-click the change set and select "`Remove WorkItem`"

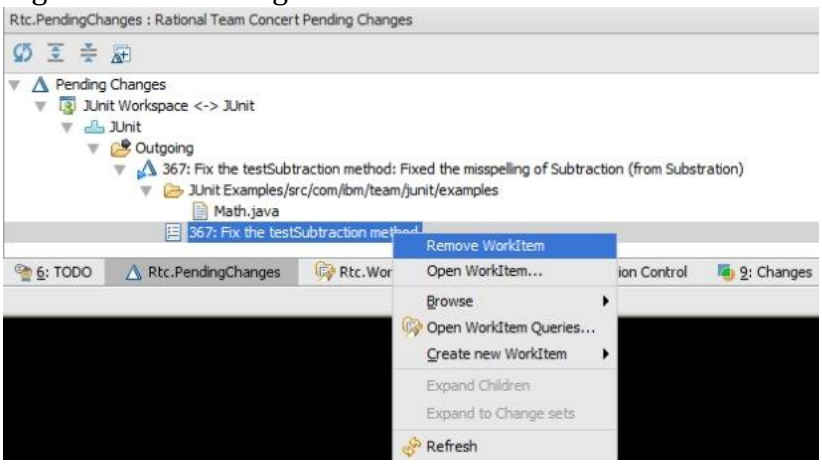

Figure 45: Remove Work Item from change set

2. Click "Yes" to disassociate the work item

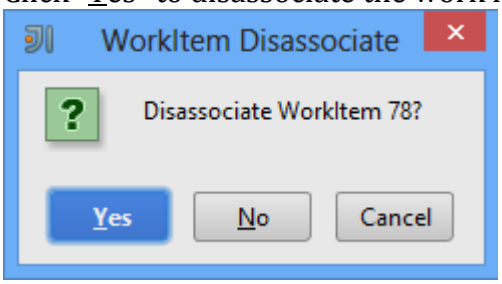

Figure 46: Confirmation Work Item disassociate

3. Assign a new work item to the change set by right-clicking on the change set and a selecting "`Associate Work Item`"

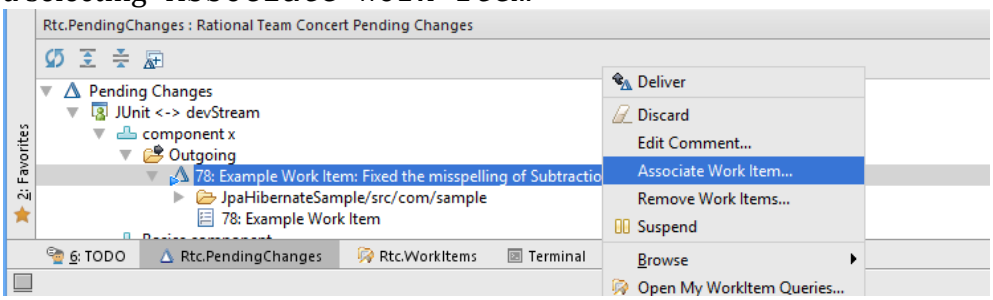

Figure 47: Associate Work Item

Delivering Change set(s)

When you are ready to deliver your changes back to the JUnit stream, simply right-click on either the JUnit Component to deliver all changes or Change set to deliver the specific change set, as follows:

1. Right-click the change set and select "`Deliver`"

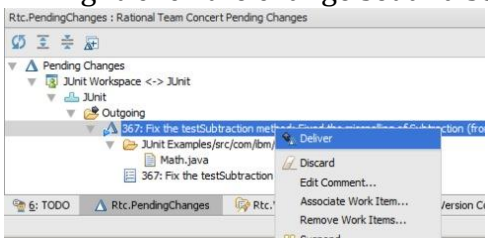

Figure 48: Deliver change set

2. Alternatively right-click the component to deliver all changes and then select "`Deliver`"

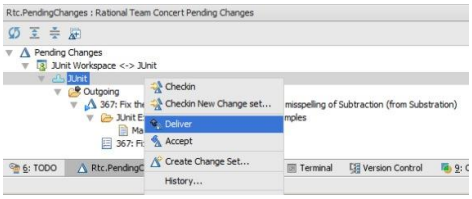

Figure 49: Deliver all change sets for component

Miscellaneous Operations

This section describes some of the other JazzConnect-IntelliJ options available.

Note: It is assumed that the reader already has familiarity with RTC SCM. If more information is needed using RTC SCM, please refer to jazz.net article #41, which provides a good overview of RTC SCM using the JUnit project.

Accept Changes from the Stream

When working with other developers, it is necessary to bring in changes from the team (`Accept`) and integrate them with your own changes before you decide to (`Deliver`) your own changes back to the stream.

To accept changes from the stream right-click on the Stream and Select "`Accept`":

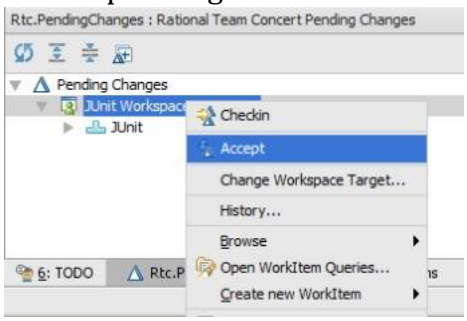

Figure 50: Accept changes

Change Flow Target

If you wish to deliver your changes to a different stream or other workspace (e.g. co-development with another developer), then you can change the target of the "`Deliver` or `Accept`" operation by right-clicking the Stream and Select "`Change Workspace Target…`"

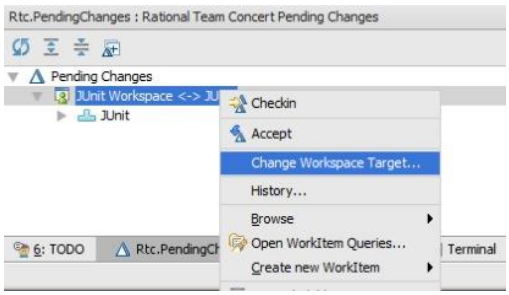

Figure 51: Change Workspace Target (Flow)

You can then select the new Project Area and Stream from the drop down boxes and select "`Change`"

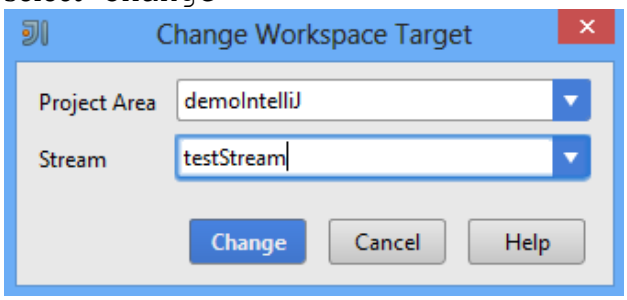

Figure 52: Change Workspace Target dialog

Note: Only streams within the same RTC server may be selected; i.e. the plug-in does not currently support Distributed SCM.

Rtc.WorkItems Tab

The Work Items tab enables the IntelliJ user to query and create Work Items directly with the IDE. You can also set the current work item as described in section "Setting a Work Item as the Current Work Item"

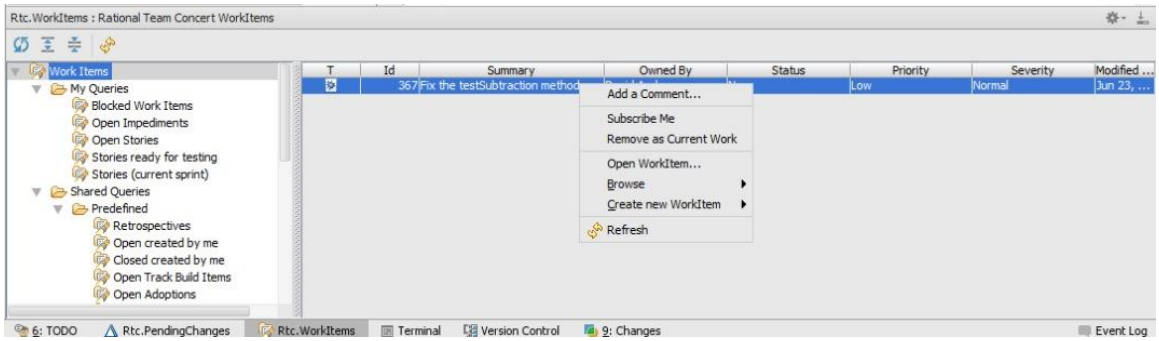

Figure 53: Rtc.WorkItems Tab

Right-clicking on a Work Item allows you to:

- **Add a Comment...** Add a comment to the existing Work Item
- **Subscribe Me**: Subscribe yourself to the Work Item so that you get notified when changes are made to the Work Item
- **Add/Remove as Current Work**: Set this Work Item to be the default Work Item where all changes associate
- **Open WorkItem**... Opens the selected Work Item in a browser
- **Browse →: Project...** Opens up the main RTC Project Area in a browser
- **Create new WorkItem →**: Creates a new Work Item in RTC
  - Defect...
  - Task...
  - Story...
  - Epic...
  - General...

Version Control Tab and Changes Tab

In IntelliJ v14 and later, the Version Control Tab shows:

- **Local Changes:** Shows the changes made locally

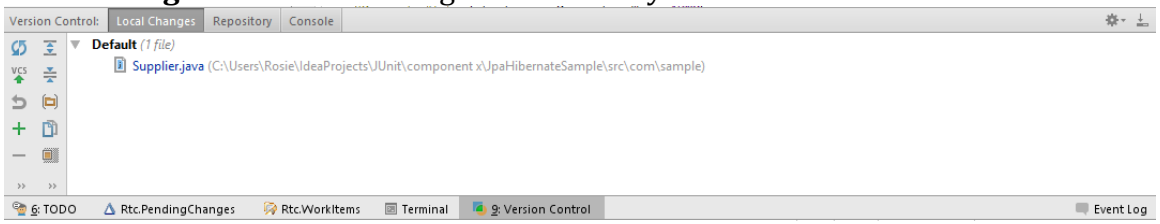

Figure 54: IntelliJ v14 Local changes

- **Repository:** Shows the repository changes

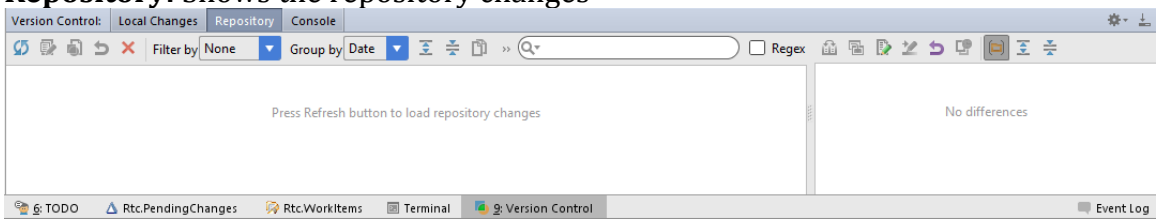

Figure 55: IntelliJ v14 Repository changes

- **Console:** Appears when output is to be displayed from the VCS system

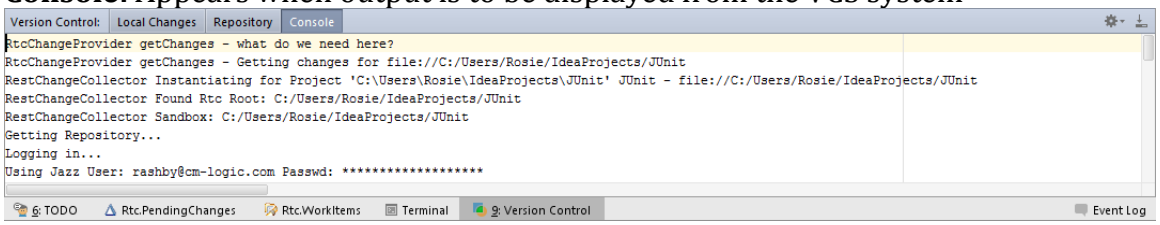

Figure 56: IntelliJ v14 Console

Note that for IntelliJ versions before version 14, the console is shown under the "`Version Control`" tab:

Figure 57: IntelliJ previous versions Version Control Tab

And the local changes appear in a separate tab:
The **Changes** tool window enables you to manage change lists and files within them, perform VCS-specific actions, or view modifications introduced by other team members. The actions can be performed using the toolbar buttons, or context menu commands. For more information on the Changes Tab, read Changes Tool Window.

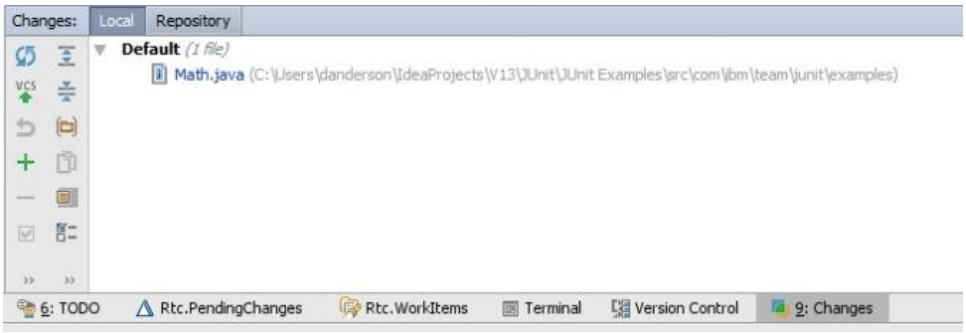

Figure 58: IntelliJ previous versions Changes Tab

Terminal Tab

This tab provides command-line terminal access. This is very useful for checking environment variable values, running scripts, builds, etc.

Figure 59: Terminal Tab

TODO Tab

Working on a large project, you often need to create the lists of tasks, and keep your teammates informed about the issues that requiring their attention. For more information on how to use the **TODO** feature of IntelliJ, read Using TODO.

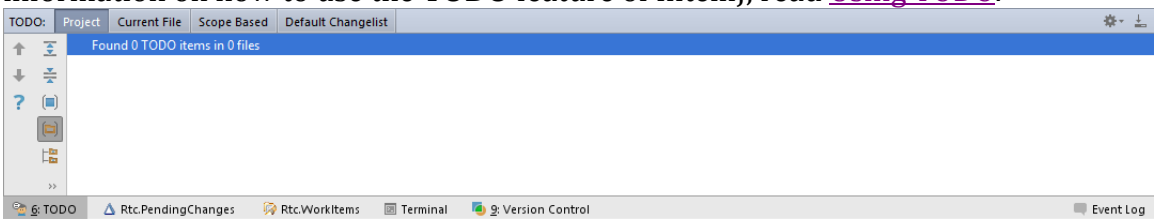

Figure 60: TODO Tab

IntelliJ Refactoring

JetBrains IntelliJ currently does not provide a way for SCM providers to track the refactoring process. The refactoring capability is one of the main features why IntelliJ is so popular among Java developers. CM-Logic has provided a *workaround* to in JazzConnect-IntelliJ, at least until a more permanent solution is made available.

Refactor using JUnit

The refactor using JazzConnect-IntelliJ is a 2-step process:

1. Refactor using the Rational Team Concert "`Move/Rename…`" context menu
2. Refactor using the standard IntelliJ option

Rename a class using the Rational Team Concert context menu

1. Rename/Refactor "`AllTests`" to "`AllMathTests`", using the Rational Team Concert context menu:

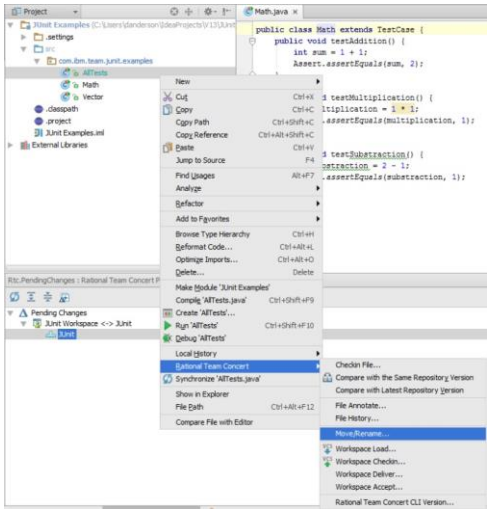

Figure 61: Refactor (RTC Move/Rename)

2. Select the Target

Figure 62: Move/Rename Resource Dialog

3. As you will see from the screenshot below, the class got renamed in the IDE and in the workspace; however, we still need to "refactor" the class files themselves…

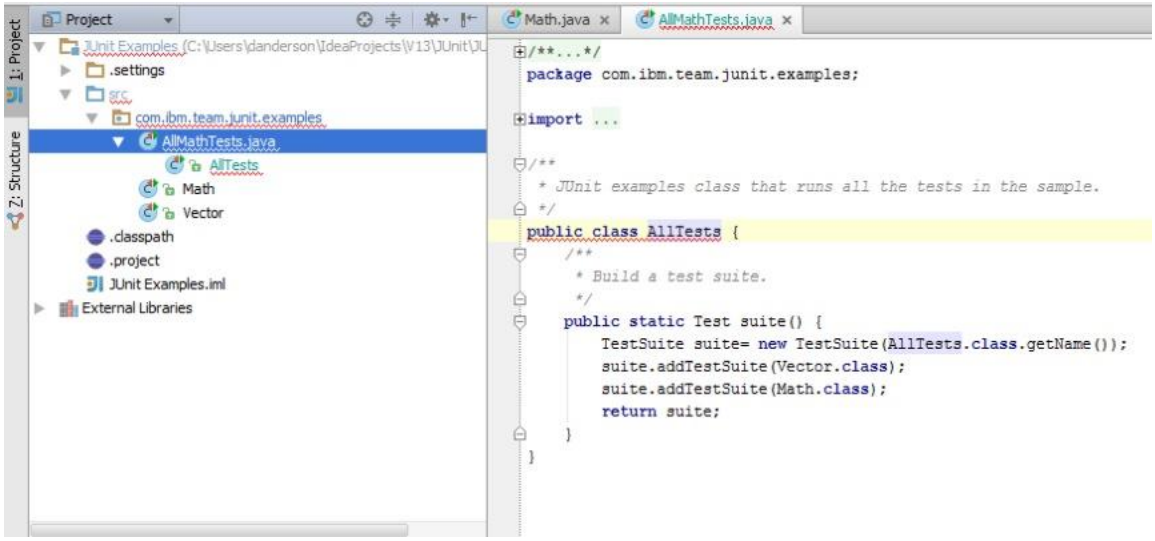

Figure 63: Refactor - Step 1 Class Rename

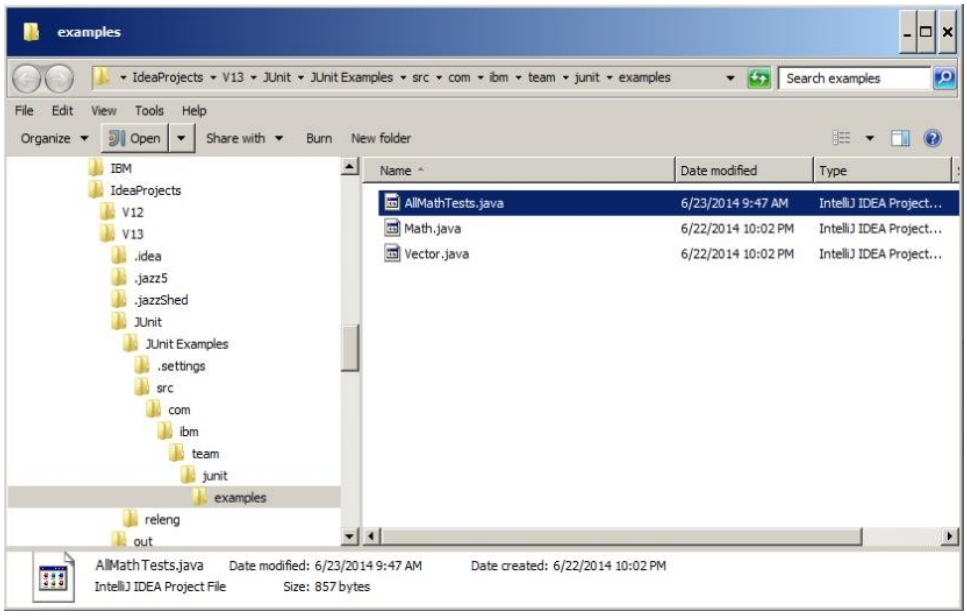

Figure 64: Refactor Work area

Refactor using IntelliJ

1. Refactor the class using IntelliJ Refactor:

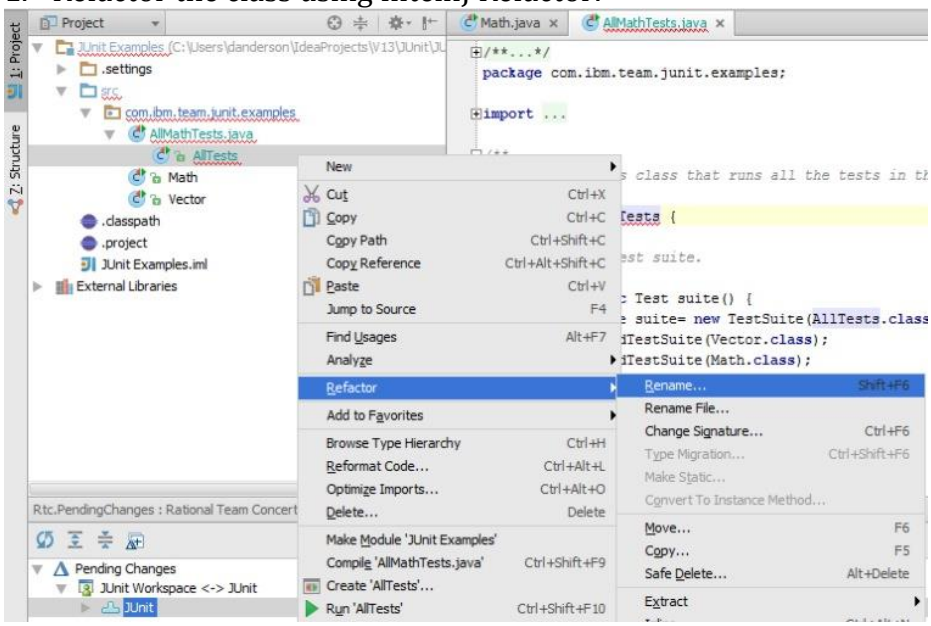

Figure 65: Refactor IntelliJ (Rename...)

2. Rename the class

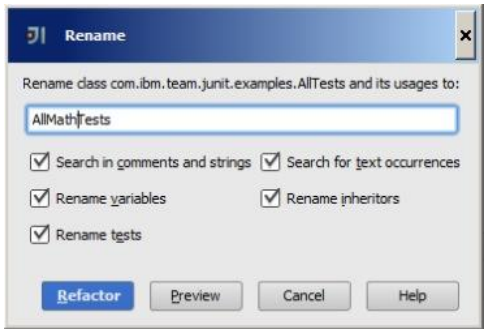

Figure 66: IntelliJ Rename Dialog

3. The class is now refactored:

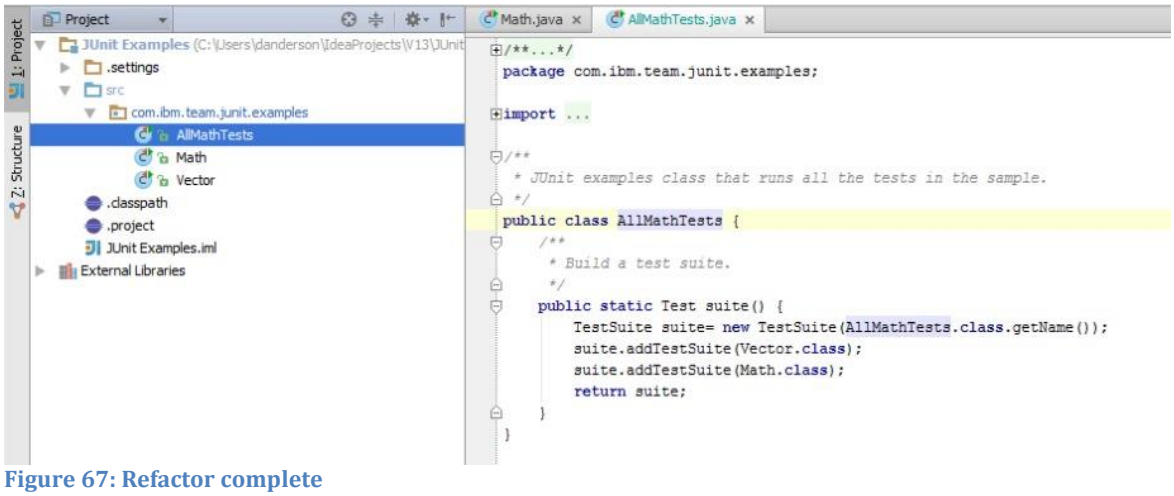

Figure 67: Refactor complete

Troubleshooting

Common error messages along with their resolutions can be found below.
In case of difficulties in resolving errors not listed here, please contact CM-Logic for assistance.

Failed to get VCS instance error

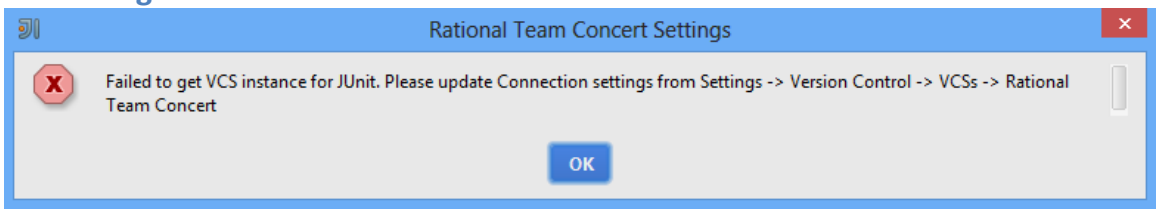

Figure 4: Failed to get VCS instance error

This error occurs because no VCS has been configured. Please see the section "Configuring the VCS Default Settings" to configure the VCS.

Project SDK is not defined error

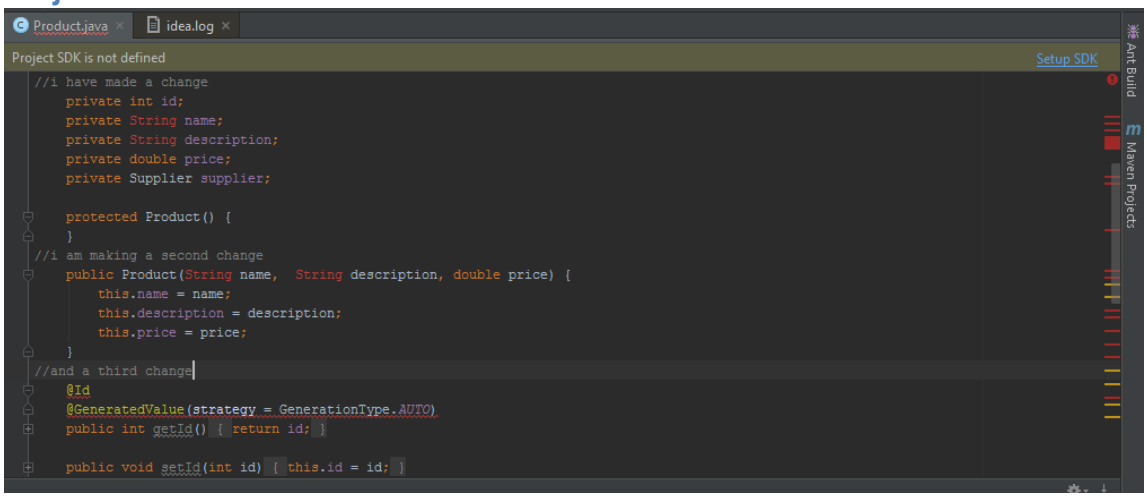

Figure 68: Project SDK is not defined error

1. Go to "`File`">"`Project Structure`"
2. Navigate to "`Project`">"`Project SDK`"

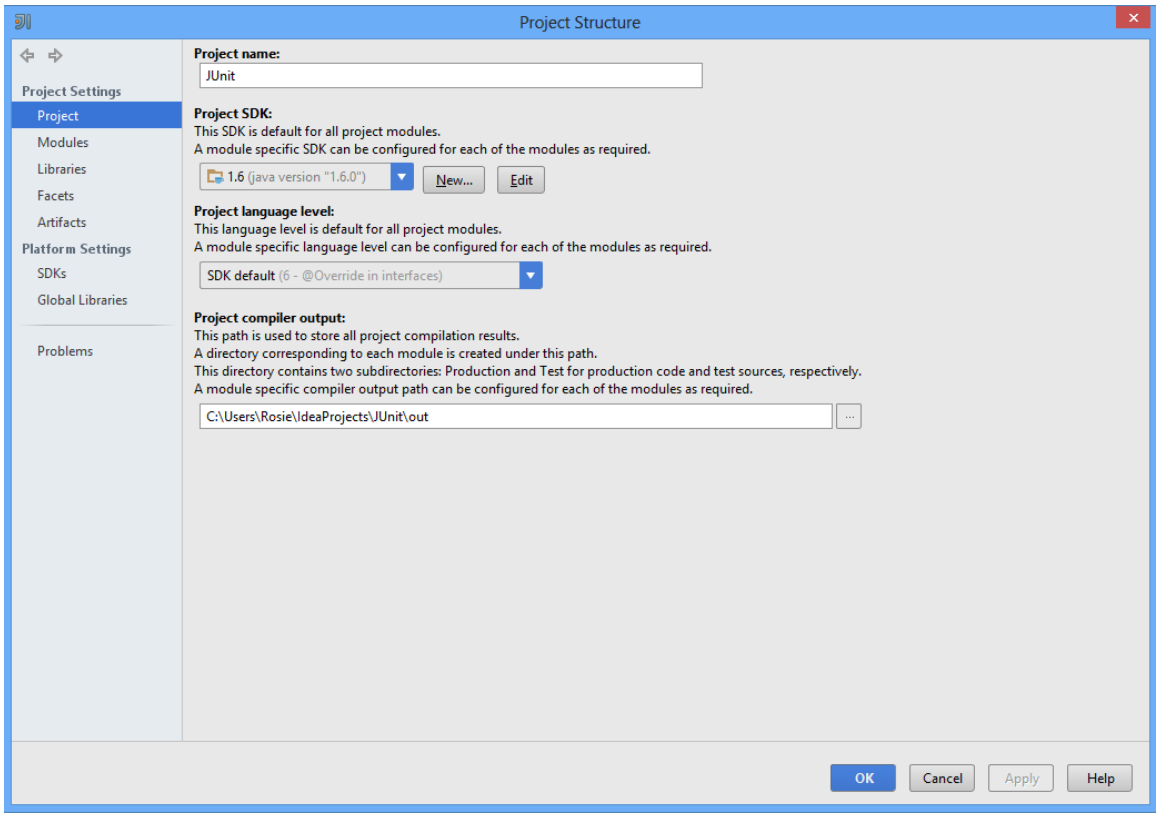

Figure 69: Project Structure > Project SDK

3. If the SDK you wish to connect to is listed, select it here and click "OK"
4. If not, click "New" and find the appropriate SDK from your computer and click "OK"
5. Back in the main IntelliJ window, select the link on the right-hand side of the error message "`Setup SDK`"
6. Select the SDK to use from the drop down menu

No Project name configured in RTC Settings error

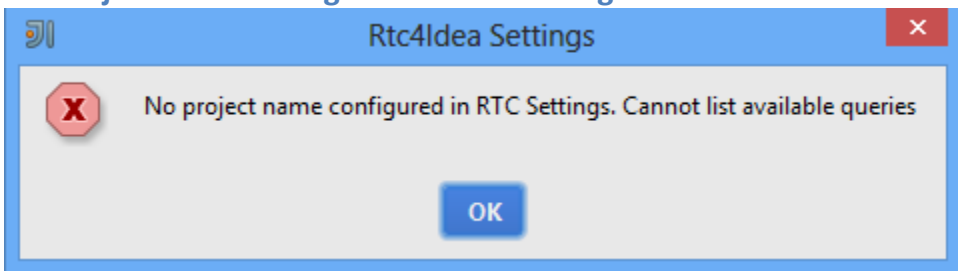

Figure 70: No Project name configured

Please see the section on "Connect to RTC Project Area"

Error occurred during 'scm logout' error

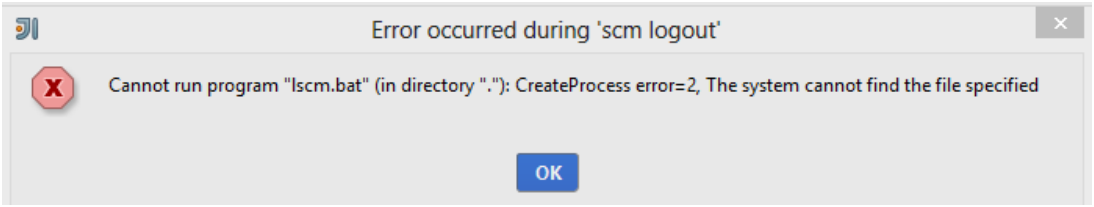

Figure 71: Error occured during 'scm logout'

If the above error occurs when setting up configure the default VCS settings:

1. Ensure that the PATH to the lscm tools is correct

> *Note: The version of SCM Tools **must** match the version of Rational Team Concert where the IntelliJ IDE will be connected. So, for example, if you are using RTC version 4.0.5, then you will need to go to* https://jazz.net/downloads/rational-team-concert/releases/4.0.5?p=allDownloads.
> *Using the SCM Tools found within the Rational Team Concert client installation will ensure the correct SCM Tools will be used.*

2. Ensure that you have selected "OK" to save the configuration before trying the test connection

No Queries appear when Rtc.WorkItems tab is refreshed

If no queries are appearing when the Rtc.WorkItems tab is refreshed, this may be due to having no project configured for the IntelliJ Workspace. Please see the section on "Connect to RTC Project Area" to connect an RTC project area to view queries

Unable to create a new Work Item

If when the trying to create a new work item by right-clicking "Pending Changes">"Create New WorkItem" all the work item types are greyed-out, this may be due to having no project configured for the IntelliJ Workspace. Please see the section on "Connect to RTC Project Area" to connect an RTC project area. When this has been connected, the work item types will be selectable.

Unable to view Rtc.PendingChanges, Rtc.WorkItems, Terminal or Version control Tabs

If none of the tabs above are appearing in your IntelliJ window:

1. Ensure you have completed section "Configure IDEA Project for Rational Team Concert"
2. If this does not resolve the problem, go to "`View`" > "`Tool Windows`" to manually bringing up each of the tool windows