Informatics for Integrating Biology and the Bedside

Developer's Guide

i2b2 Workbench

Document Version: *1.7.00.1*

i2b2 Software Version: *1.7.00*

Table of Contents

DOCUMENT MANAGEMENT

Revision Number	Date	Author	Description of change
1.7.0	3/21/13	Janice Donahoe	Created 1.7 version of the document.
1.7.00.1	06/08/2015	Janice Donahoe	Updated section on editing the i2b2Workbench.properties file. It was referencing the old HarvardDemo information instead of i2b2demo.

ABOUT THIS GUIDE

Informatics for Integrating Biology and the Bedside (i2b2) is one of the sponsored initiatives of the NIH Roadmap National Centers for Biomedical Computing (http://www.bisti.nih.gov/ncbc/).

One of the goals of i2b2 is to provide clinical investigators broadly with the software tools necessary to collect and manage project-related clinical research data in the genomics age as a cohesive entity; a software suite to construct and manage the modern clinical research chart.

This guide is designed to provide an overview of installing and configuring the i2b2 Workbench within the Eclipse IDE. It also touches on the use of Eclipse Feature Projects in regards to the i2b2 Workbench.

The majority of the information in this document is the same for both ***Microsoft Windows*** ("Windows") and ***Apple's Macintosh*** ("Macintosh") operating systems. It is noted within the documentation when information is specific to either Windows or Macintosh.

1. PREREQUISITES

1.1. Third Party Software

1.1.1. Java JDK

JDK 6.0 is recommended and can be downloaded from the java website:

http://java.sun.com/products/archive/

1. Install the SDK into a directory of your choice.

   Example: */opt/java/jdk1.6.0* or *YOUR_JAVA_HOME_DIR*

1.1.2. Eclipse

You can download version 3.7 of the Eclipse SDK (Indigo) from the Eclipse website by going to:

http://www.eclipse.org/downloads

To install Eclipse, extract the downloaded zip file into a directory such as C:\

If you install a newer version of Eclipse, be sure to install it in an area separate from any previous Eclipse installations.

Further information regarding the eclipse product can be found on their website (http://www.eclipse.org/documentation/)

1.2. i2b2 Software

1.2.1. i2b2 Workbench Source Code

The latest version of the **source code** for the i2b2 Workbench needs to be downloaded.

1. Go to the i2b2 website (http://www.i2b2.org)
2. At the top of the page there is a navigation bar, click on "**Software**" to go to the *software page*.

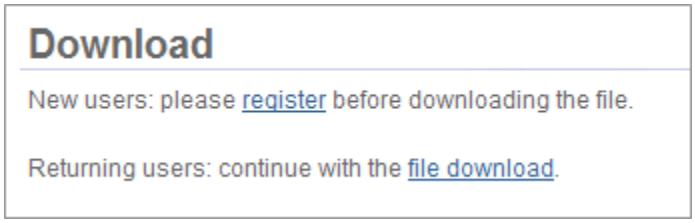
3. Once the software page loads, scroll down to the section called **Downloadables**.
4. Click on the **Source Code** box.

5. Information about the latest version and a list of zip files available for download will appear.
6. Select the appropriate file to download.

The name of the file shown in the above image may vary slightly depending on the version of the i2b2 software you are downloading. The file name will always start with "i2b2Workbench-src" and the last part of the name is dependent on the version of the software in the zip file.

Example: "1605" means the zip file contains version 1.6.05 of the i2b2 Workbench.

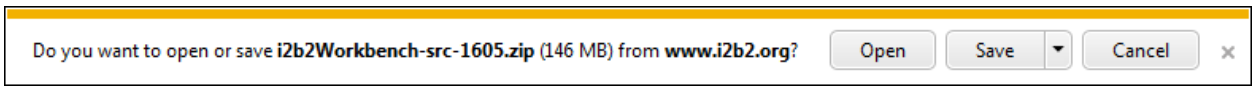
7. The **Download** page will open. At this point, if you are not already a registered user you will need to do so. If you are a returning user you can click on the "**file download**" link.



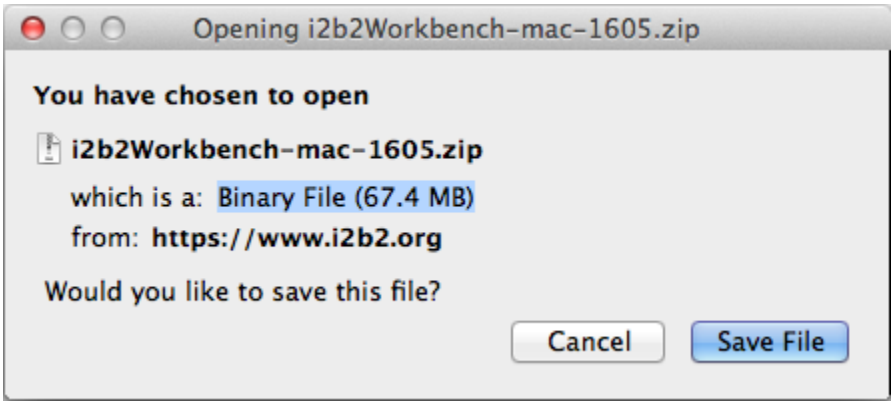
8. The next step is to review the i2b2 license and choose to either accept or decline.
    a. If you choose to **Decline** you will be brought back to the main i2b2 Software page.
    b. If you agree with the license click on **Accept** to continue with the download.

9. The **Save dialog box** will open.

Windows:



Macintosh:



10. Click on the **Save** button and the zip file will be saved to your *Download folder*.

- ***Depending on your browser and it's version you may need to select a target directory to save the zip file.***

2. INSTALL

2.1. i2b2 Workbench Project Files

The project software is contained in the Eclipse "*Archive*" file called **i2b2Workbench-src-xxx.zip**, which you downloaded in the previous section.

- ***The above file name has been modified slightly in that the numbers that would normally appear have been replaced with an 'x'. In actuality these 'x's would be the version of the software in the zip file. Example: i2b2Workbench-src-1605.zip means the zip file contains version 1.6.05 of the i2b2 Workbench.***

This zip file contains the following Eclipse projects:

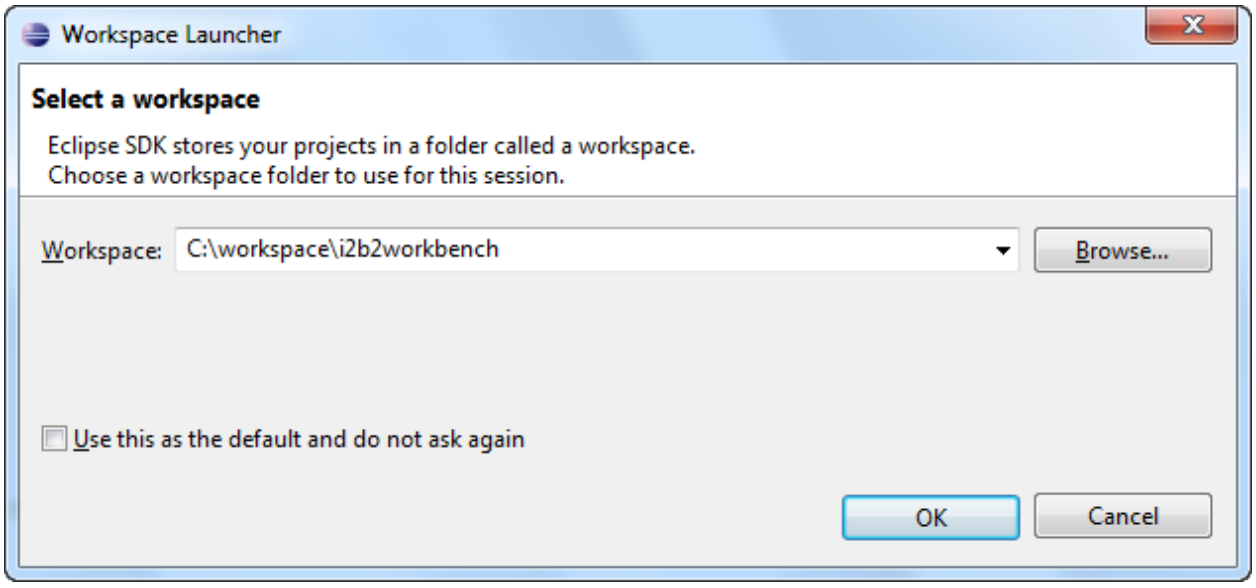
1. BuildAssist
2. edu.harvard.i2b2.common
3. edu.harvard.i2b2.eclipse
4. edu.harvard.i2b2.eclipse.features.core
5. edu.harvard.i2b2.eclipse.plugins.analysis
6. edu.harvard.i2b2.eclipse.plugins.commons_apache
7. edu.harvard.i2b2.eclipse.plugins.explorer
8. edu.harvard.i2b2.eclipse.plugins.jdnc_support
9. edu.harvard.i2b2.eclipse.plugins.log
10. edu.harvard.i2b2.eclipse.plugins.ontology
11. edu.harvard.i2b2.eclipse.plugins.previousQuery
12. edu.harvard.i2b2.eclipse.plugins.query
13. edu.harvard.i2b2.eclipse.plugins.webservicesAxis2
14. edu.harvard.i2b2.eclipse.plugins.workplace
15. edu.harvard.i2b2.eclipse.plugins.xml_support_jaxb
16. edu.harvard.i2b2.eclipse.plugins.xml_support_jdom
17. edu.harvard.i2b2.xml

- ***For additional information on the above projects please see the section called "Package Structure".***

2.2. Setting up the i2b2 Workbench within Eclipse

2.2.1. Starting up Eclipse

1. Go to the location where you installed Eclipse.
2. Run the Eclipse application by double clicking on **eclipse.exe**.
3. The **Workspace Launcher window** will open.

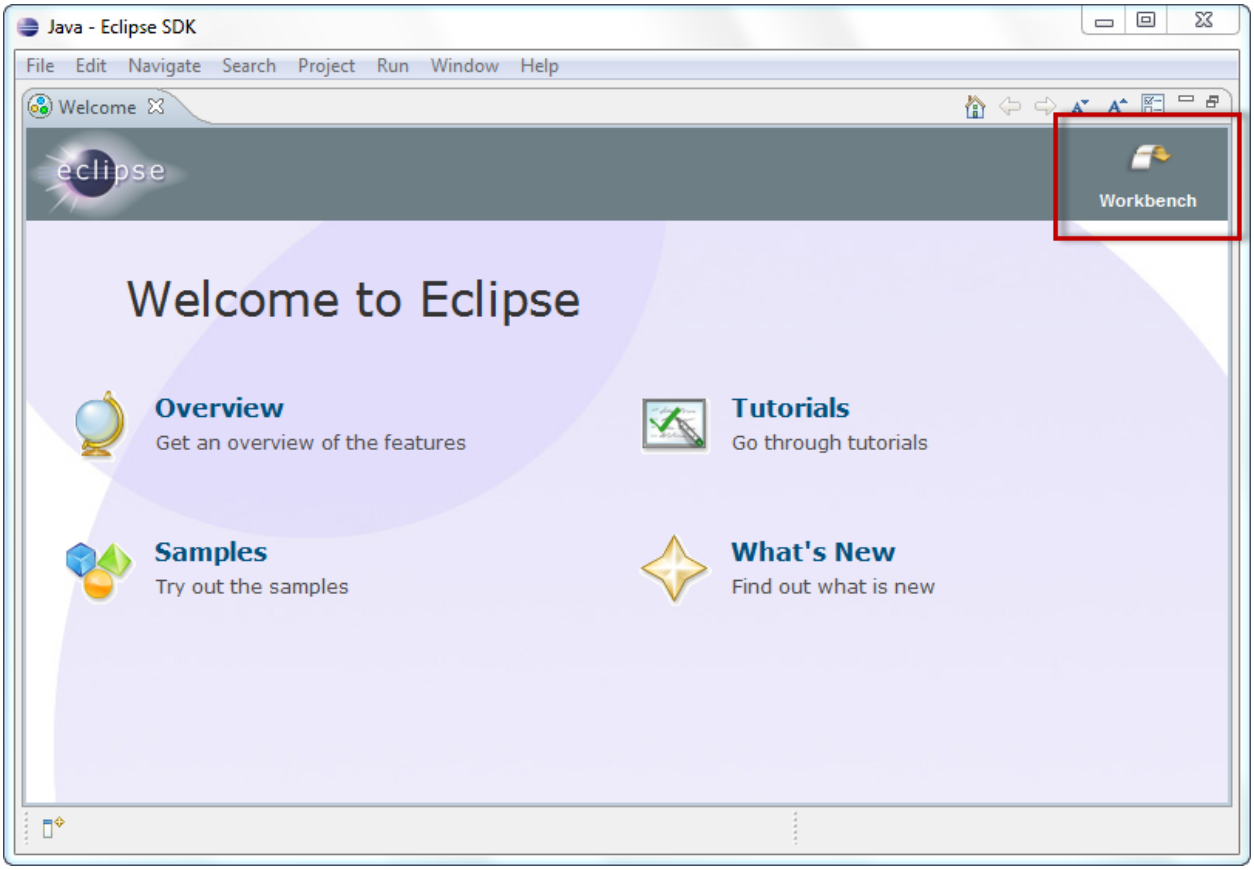


4. Select a workspace whose *pathname* does not contain spaces.

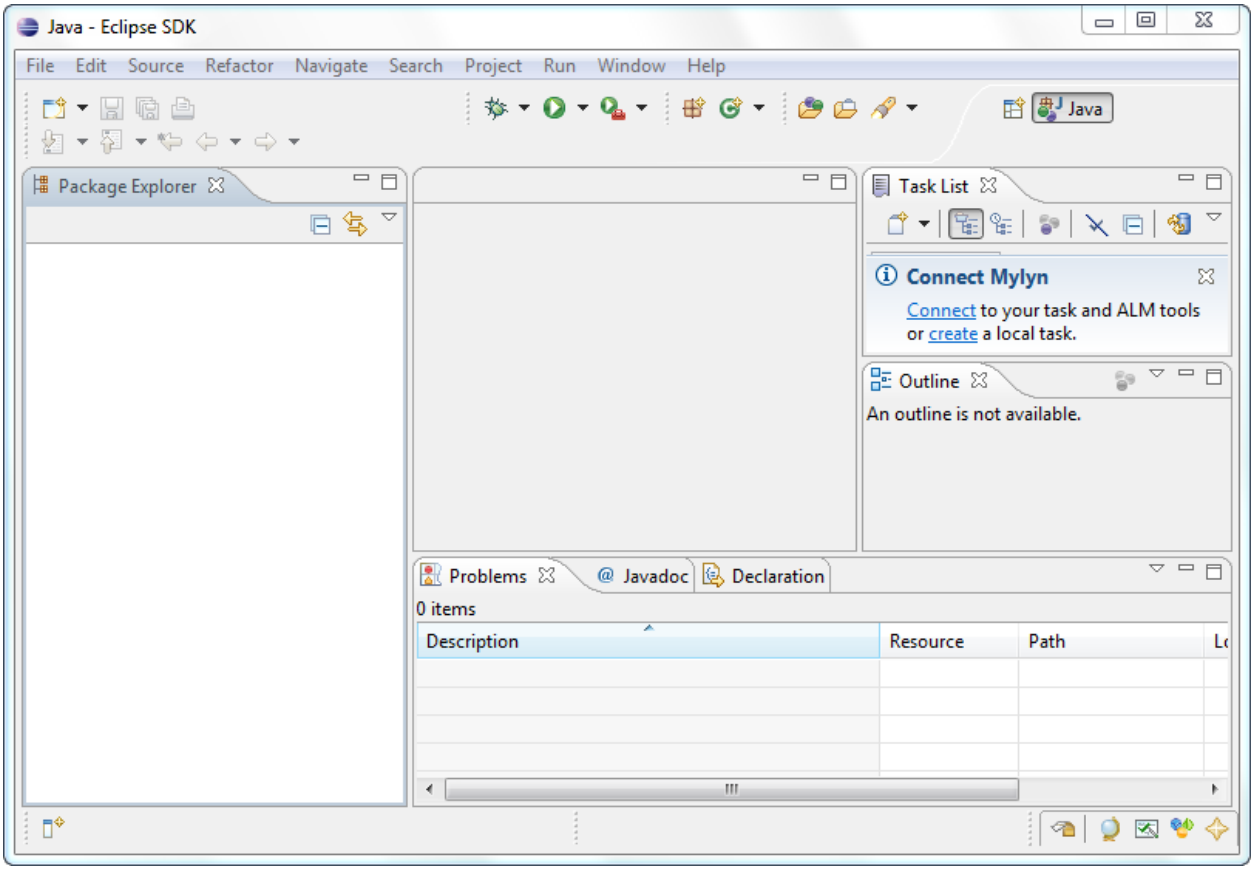
   Example (correct): c:\workspace\i2b2workbench

   Example (incorrect): c:\eclipse workspace\i2b2workbench

5. If launching Eclipse for the first time you will be greeted by a **Welcome page**.
6. In the right upper corner of the page there is an icon of an arrow that says **Workbench**. Click on this icon to startup the Eclipse SDK.

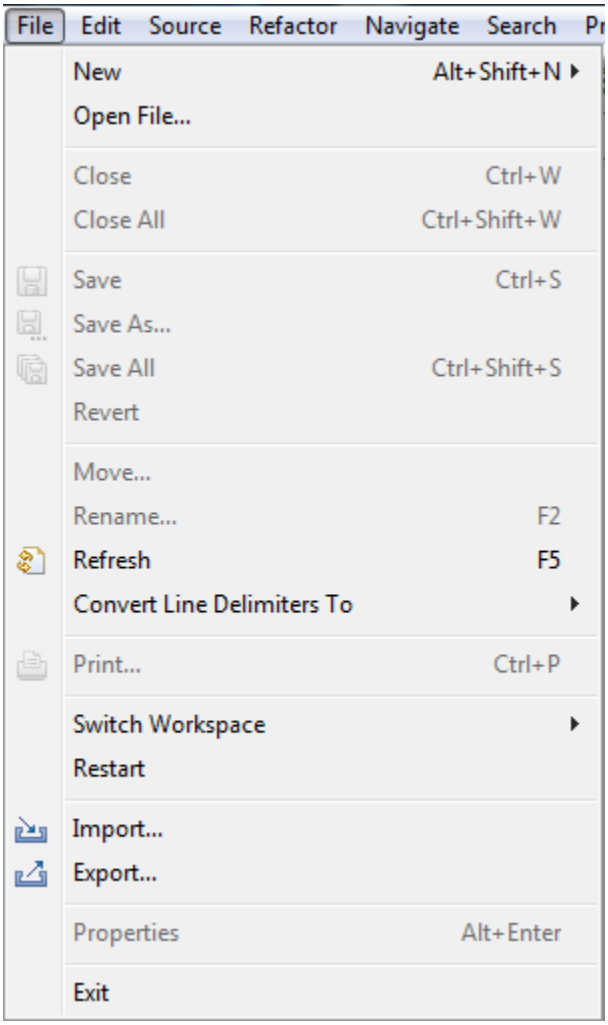


7. The Eclipse workbench will open.

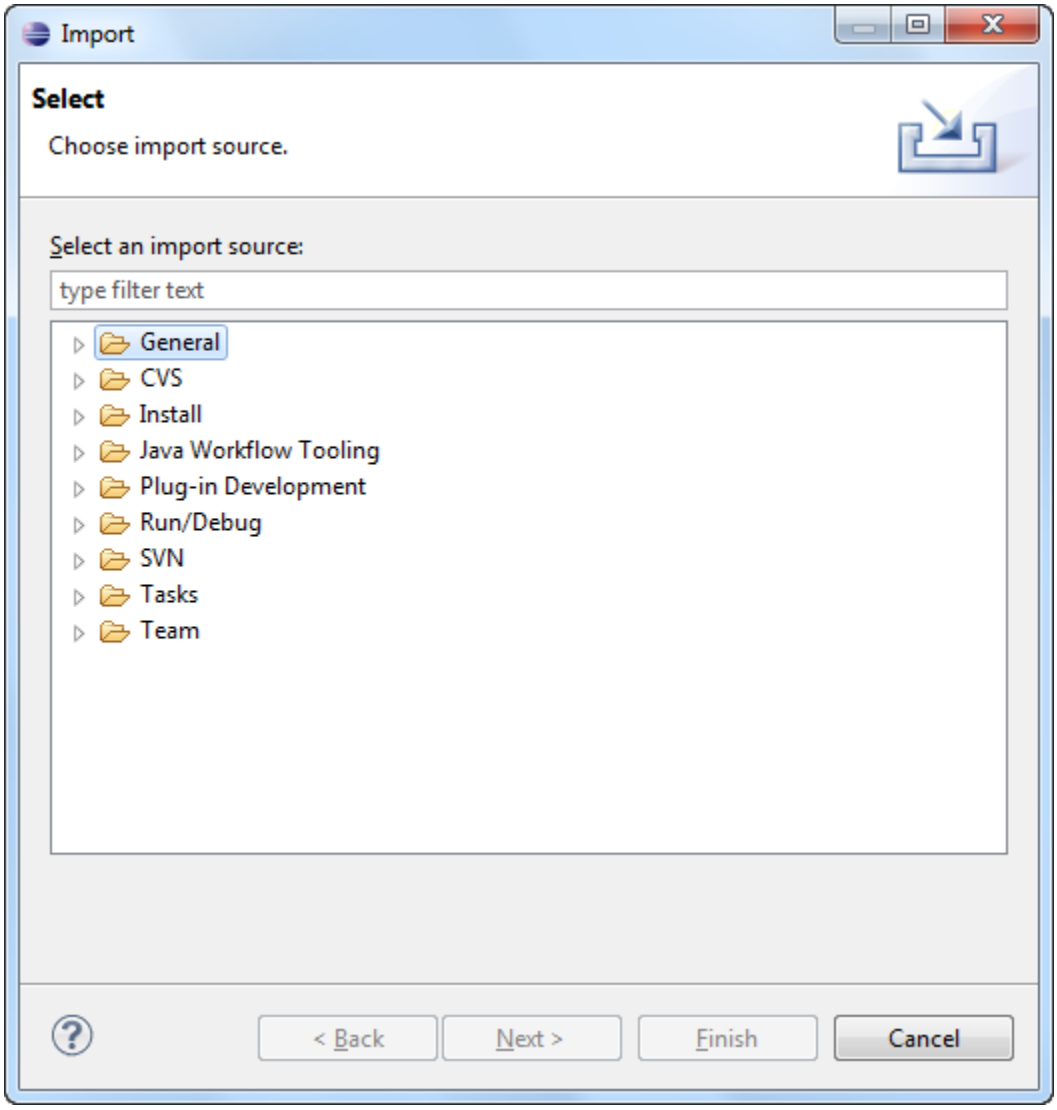


2.2.2. Importing the i2b2 Workbench projects

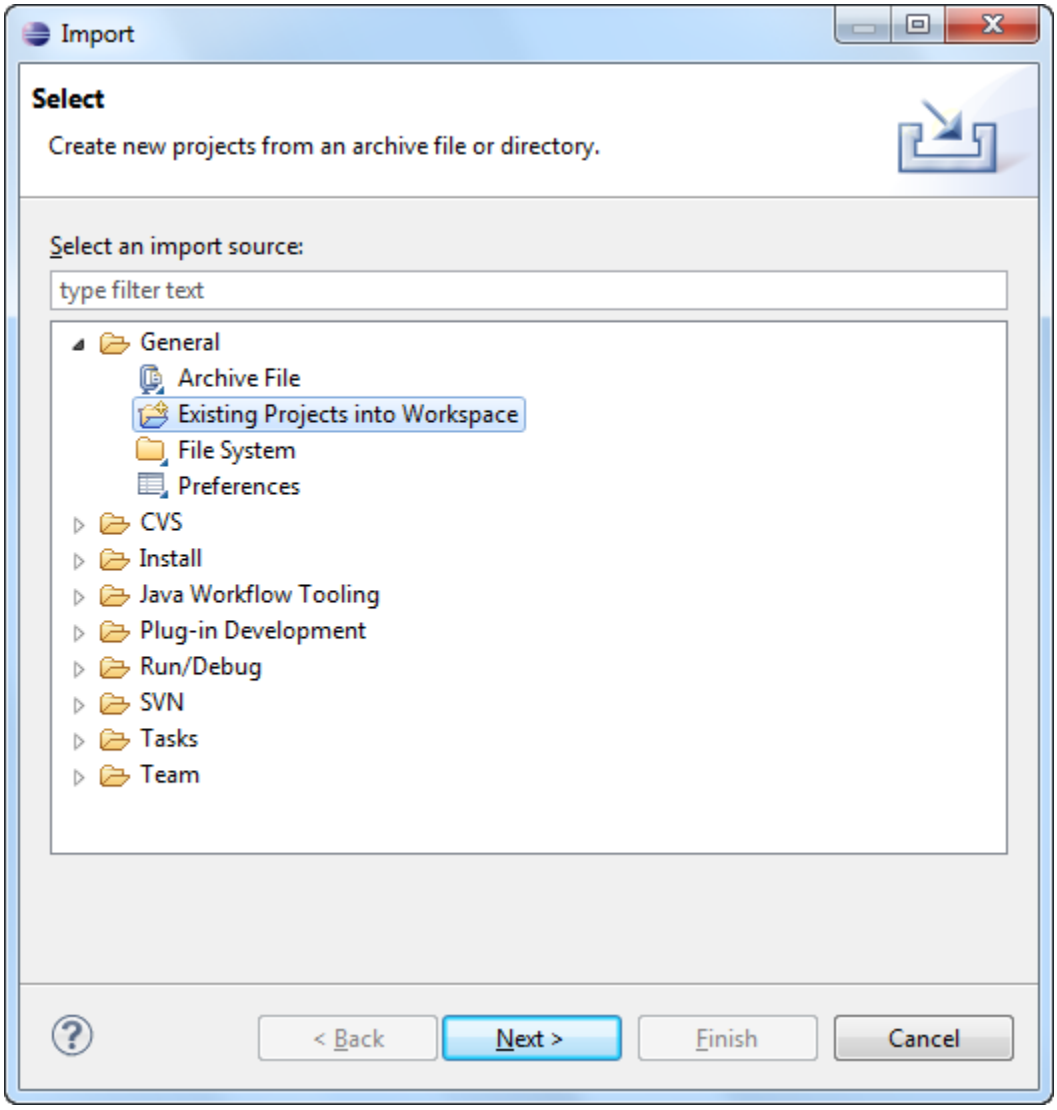
1. Select **File** from the toolbar in Eclipse.
2. A drop-down menu will display.



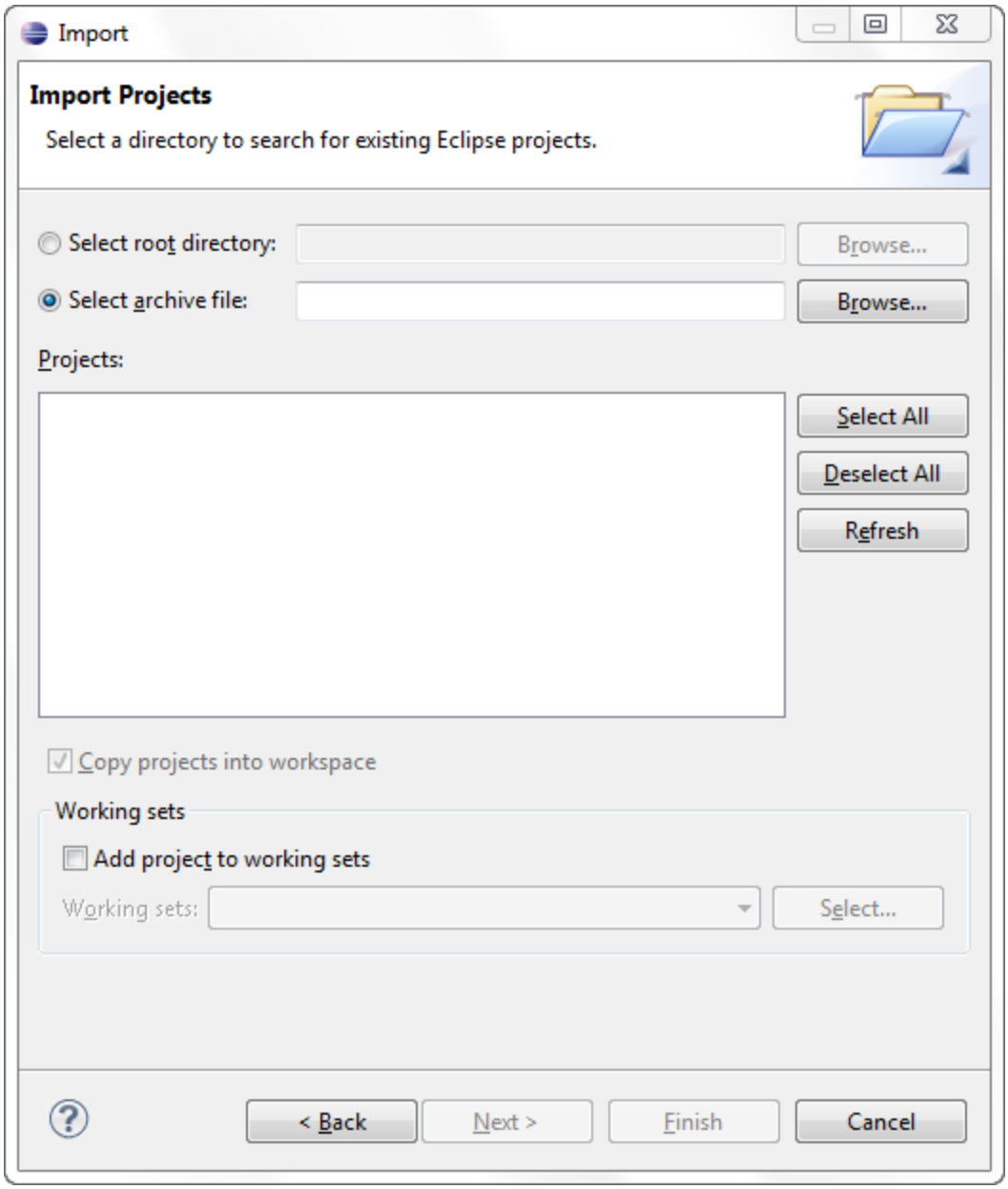
3. Select **Import**.
4. The **Import** wizard will open to the **Select** page.



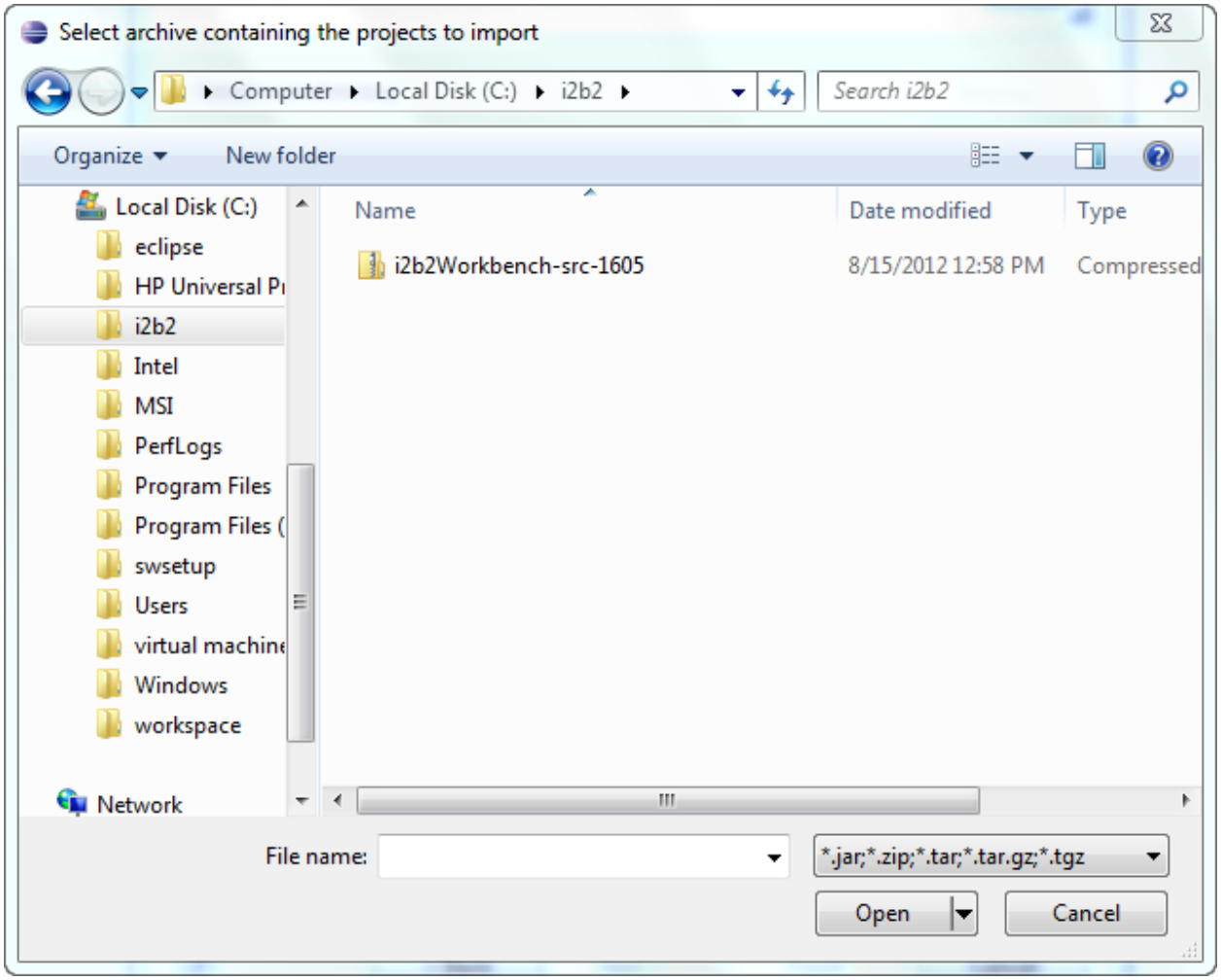
5. Double click on the folder called **General**.
6. Click on **Existing Projects into Workspace**.



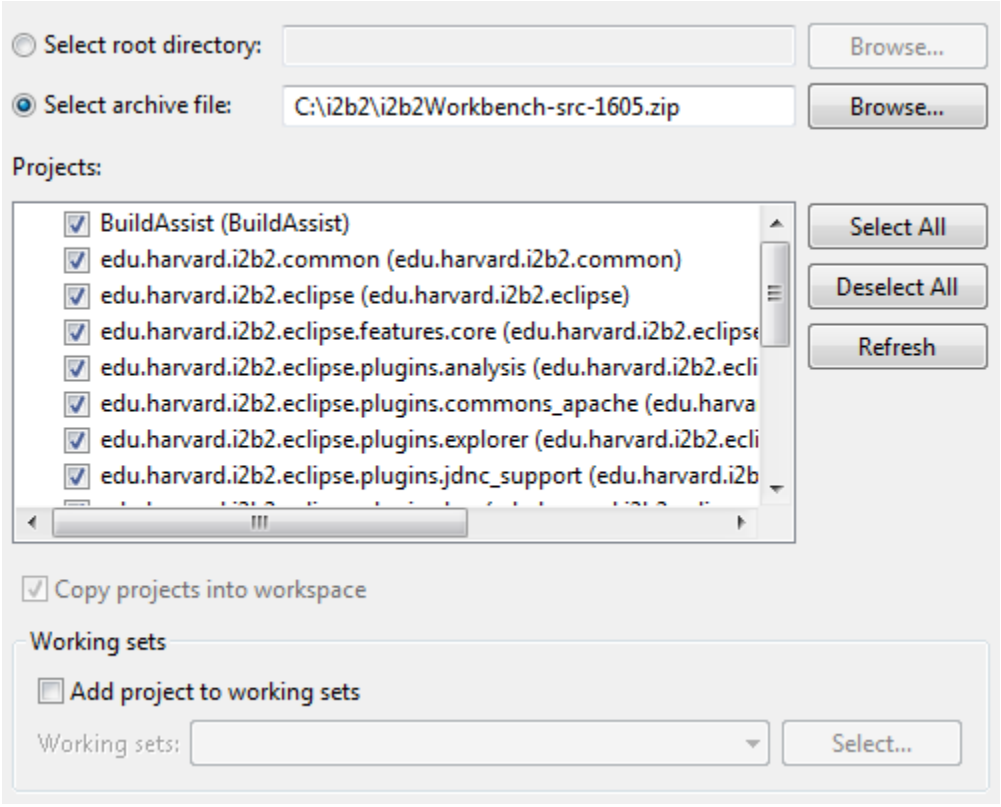
7. Click on the **Next** button.
8. The **Import Projects** page of the import wizard will open.
9. Click on the *radial dial* next to **Select archive file**.



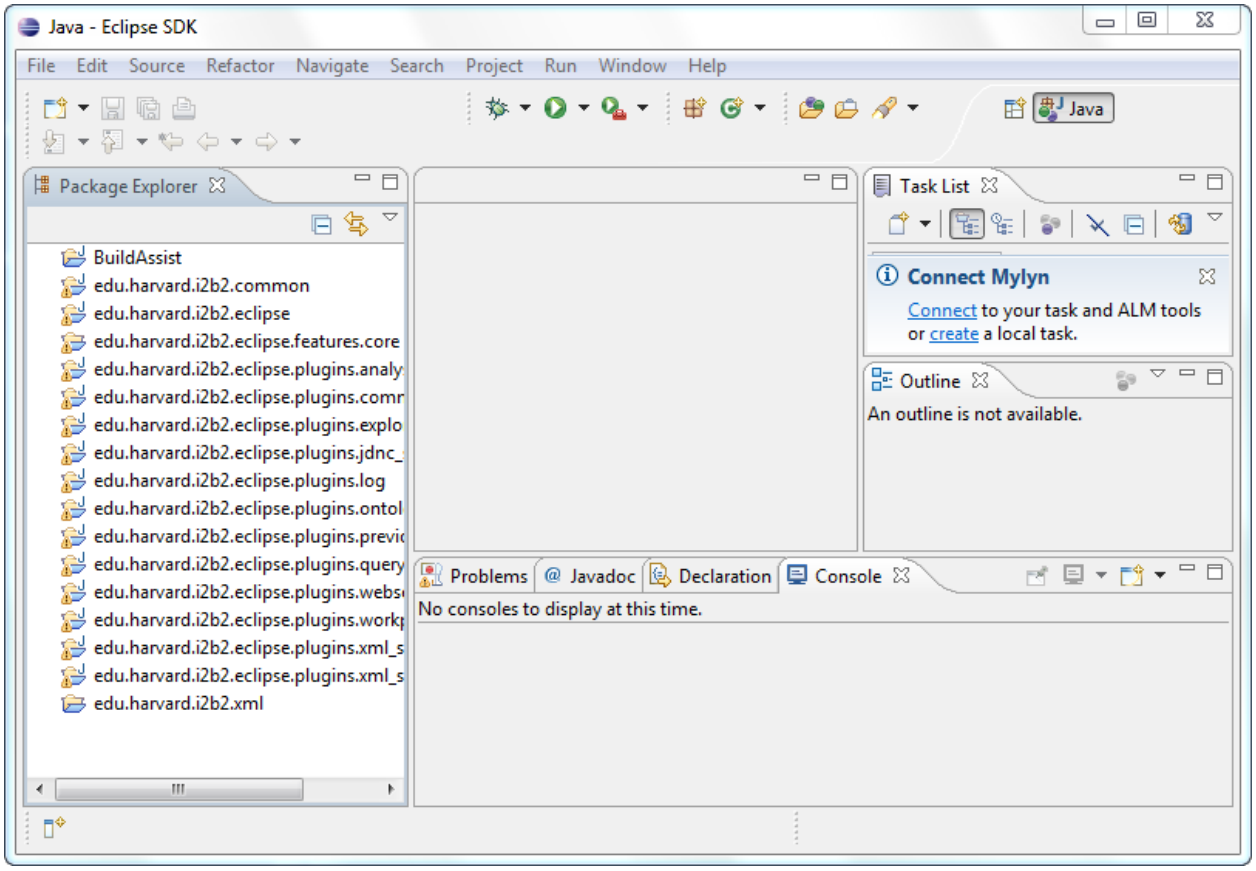
10. Enter the path of the i2b2 archive file or browse for the location where you downloaded the **i2b2Workbench-src-xxx.zip file**. (Step was done in the Prerequisite section called *i2b2 Workbench Source Code*)



11. Highlight the zip file by clicking on its name.
12. Click on the **Open** button.
13. All the projects in the zip file will now be listed in the **Projects** section of the *Import Projects* page.



14. Verify all the projects are checked off.
15. Click on the **Finish** button.
16. Once the projects have finished importing they will appear in your Eclipse workbench.

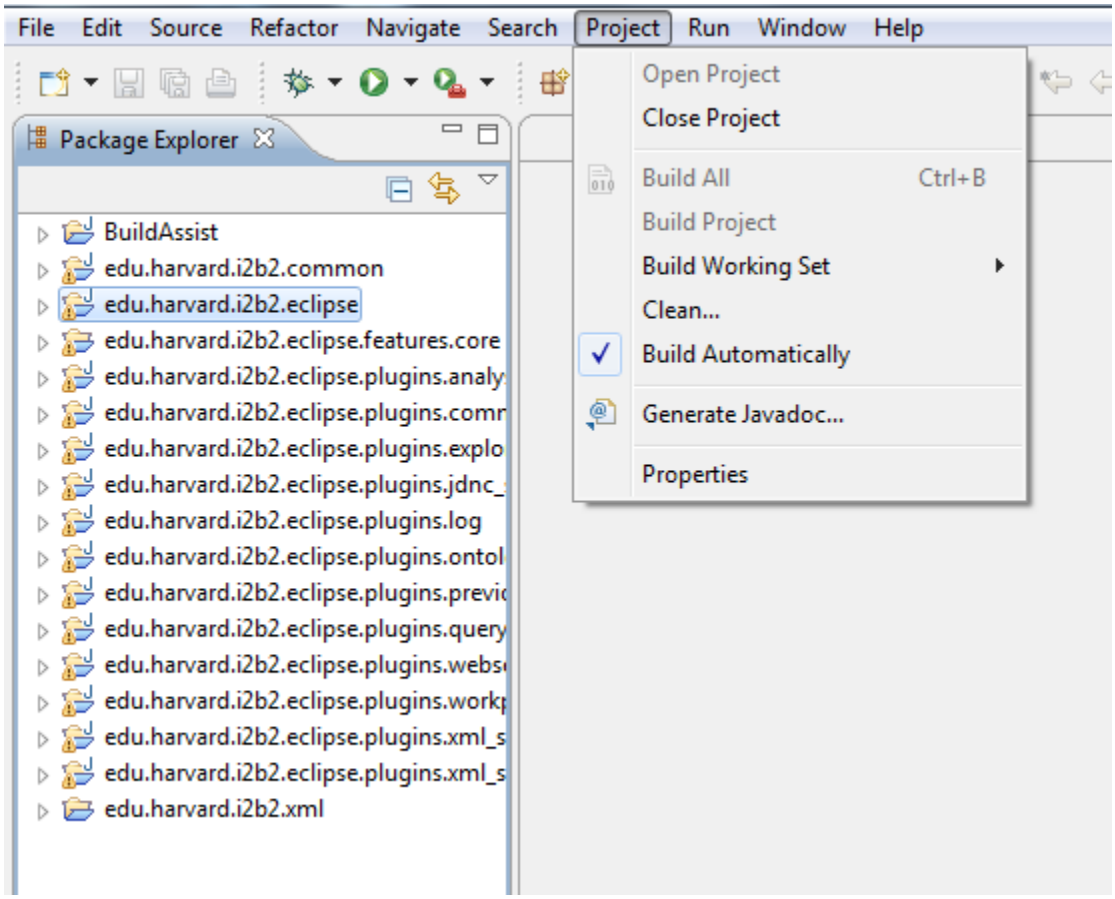


2.3. Building the i2b2 Workbench within the Eclipse IDE

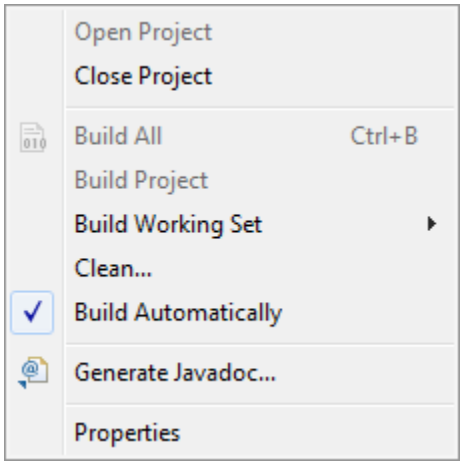
2.3.1. Java Compiler Settings

The following are the steps to check your Java Compiler settings.

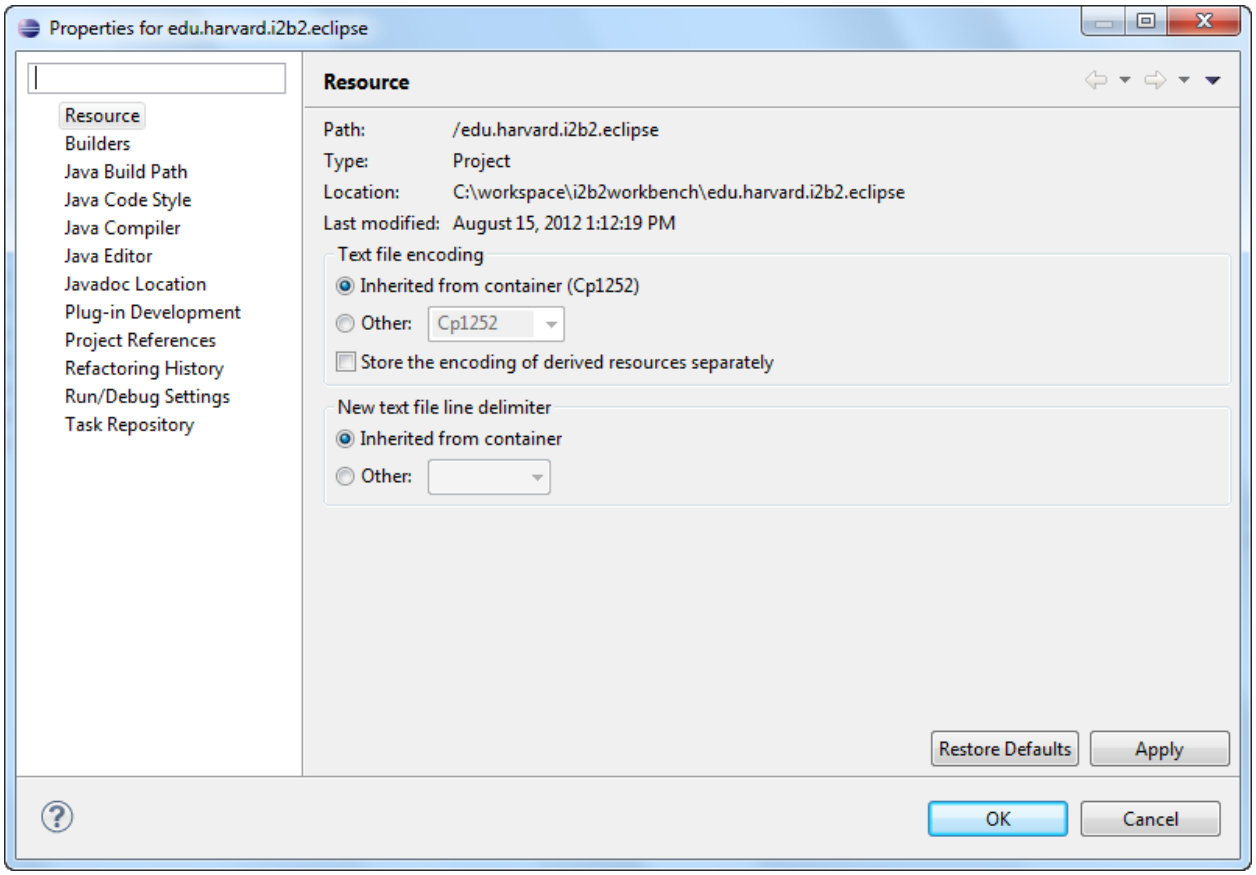
1. In the *Eclipse workbench* locate the project called **edu.harvard.i2b2.eclipse**.
2. Highlight the project by clicking on its name.
3. On the *Eclipse menu bar*, click on **Project**.



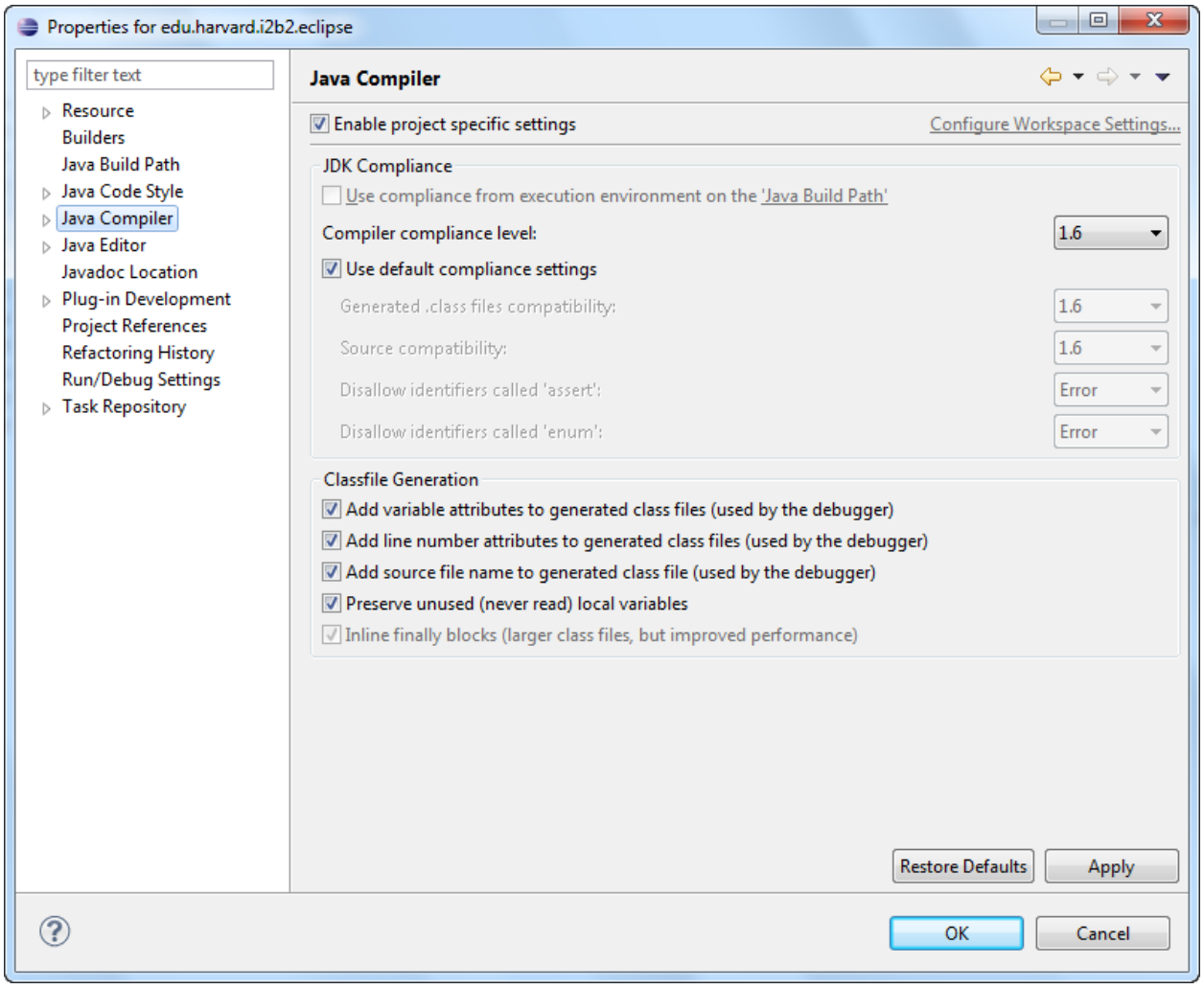
4. Select **Properties** from the drop-down menu.



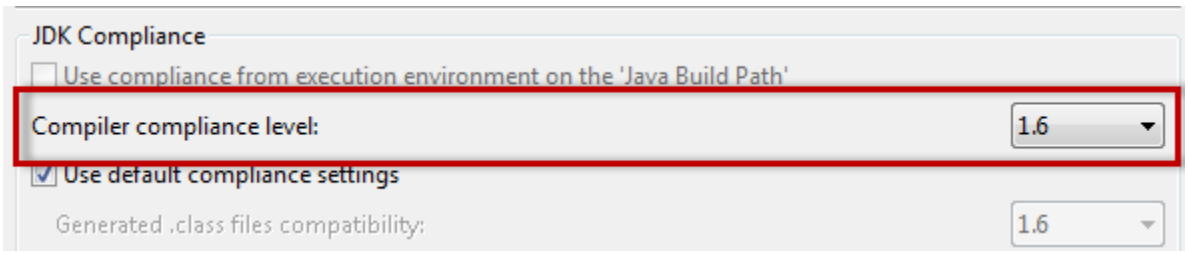
5. The **Properties** window will open.



6. In the left navigation bar, click on **Java compiler** to display the Java compiler settings.



7. Verify the **Compiler compliance level** is set to 1.6.

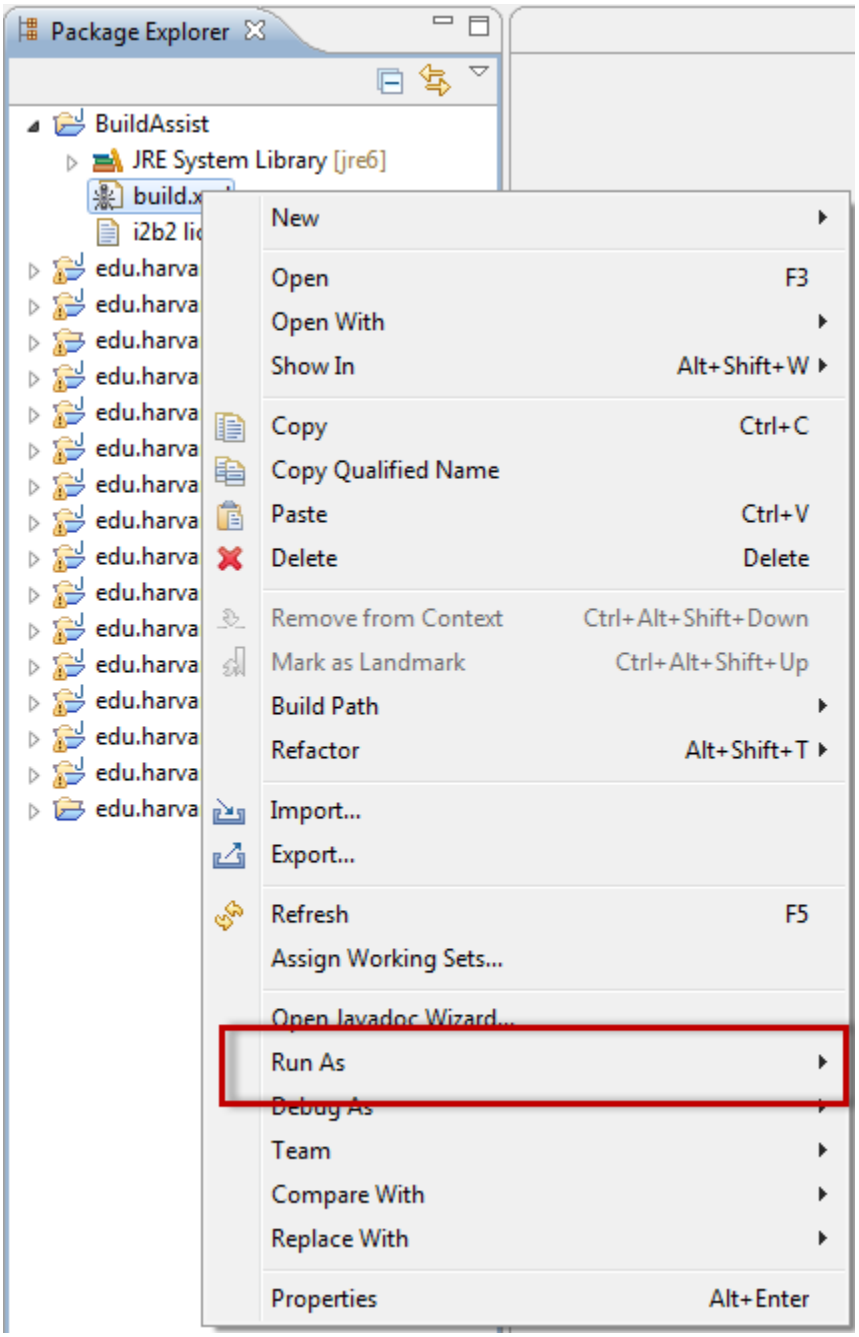


8. Click on the **OK** button.

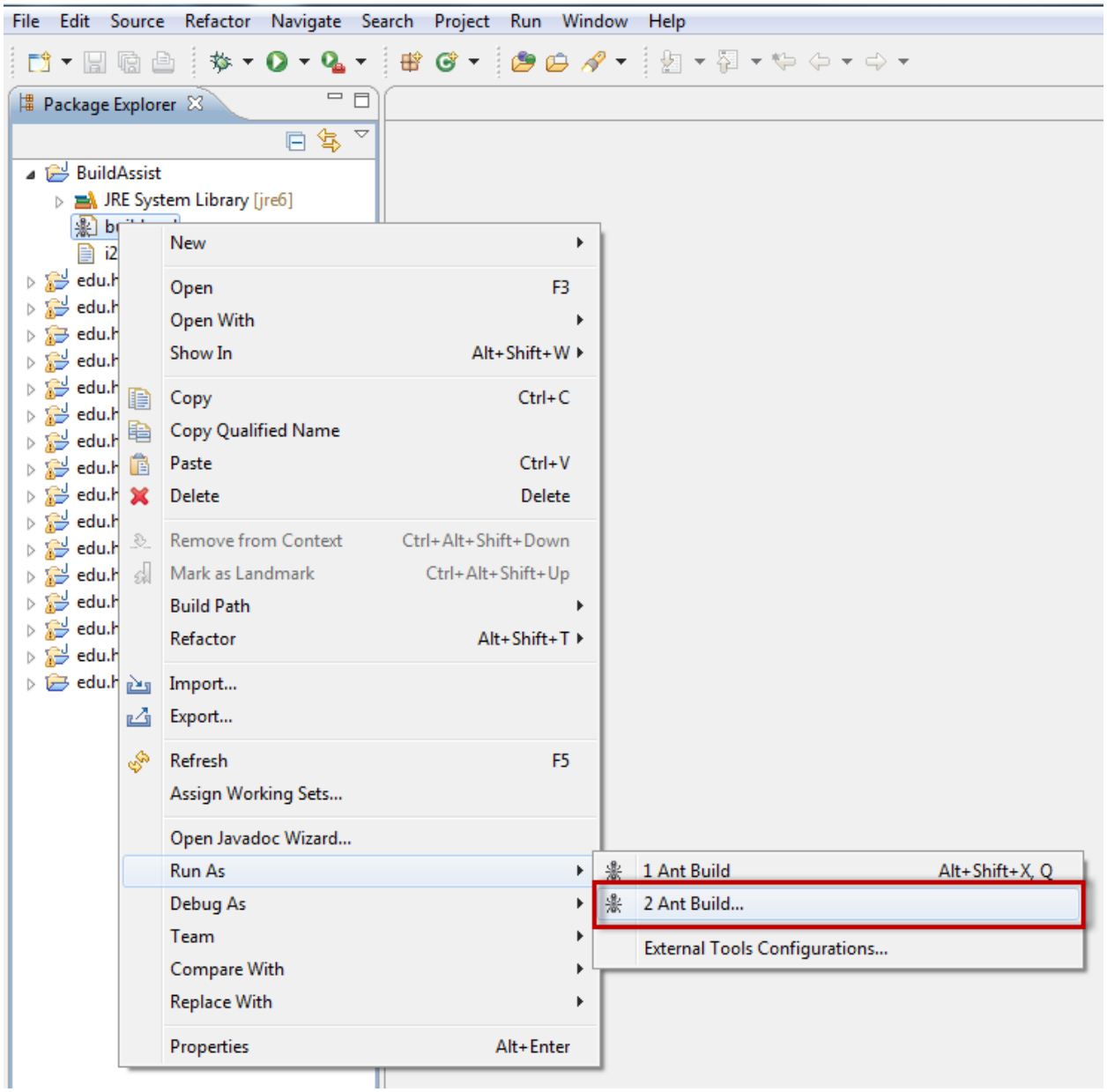
2.3.2. Cleaning and Building the Projects

The following steps outline the process of cleaning and building the projects.

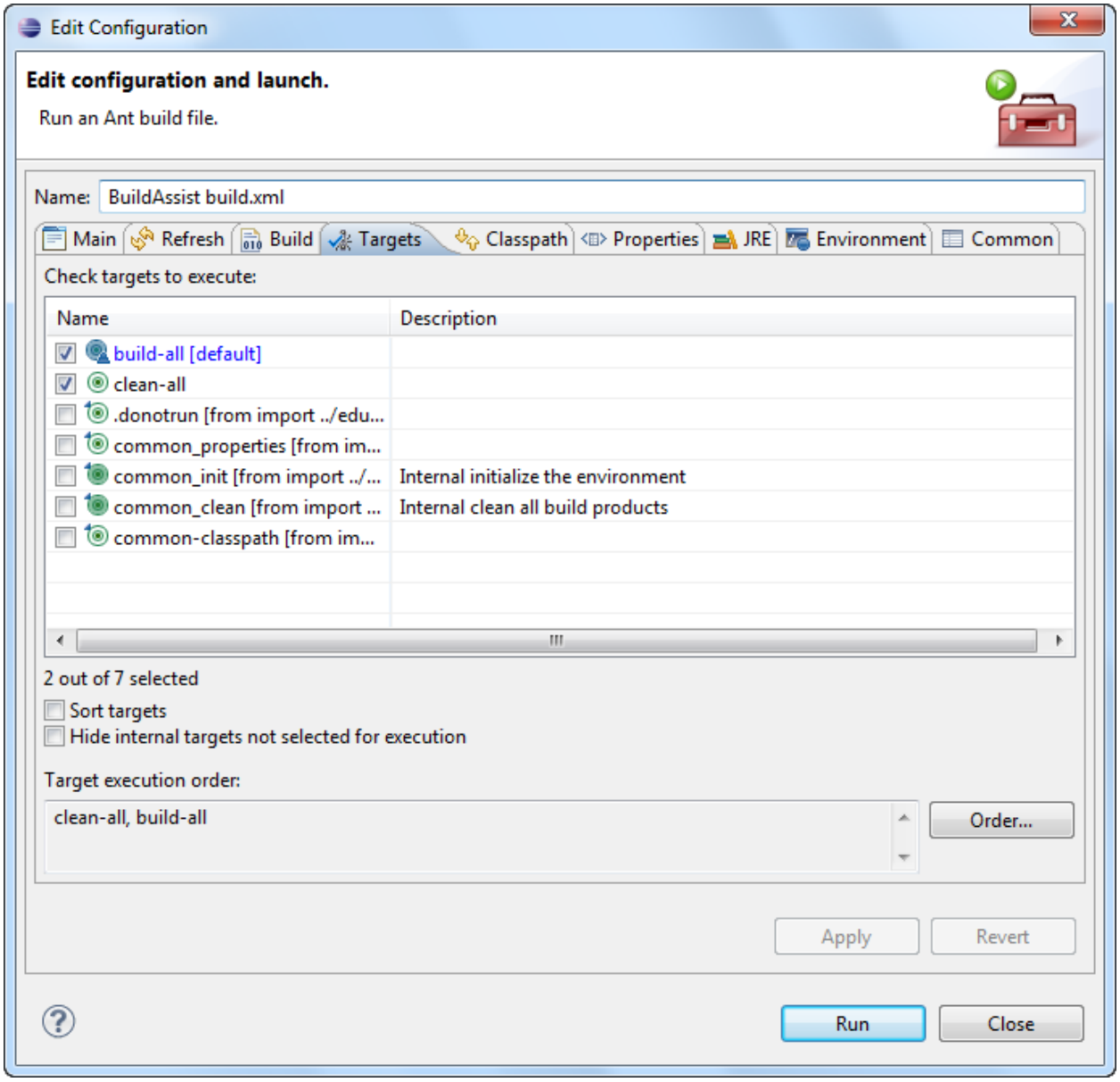
1. Double click on the **BuildAssist** project to open it.
2. Using the *right mouse button*, click on the **build.xml** file.
3. A pop-up menu will display with a list of options.
4. Using your *mouse*, hover over **Run As** to display a sub-menu.



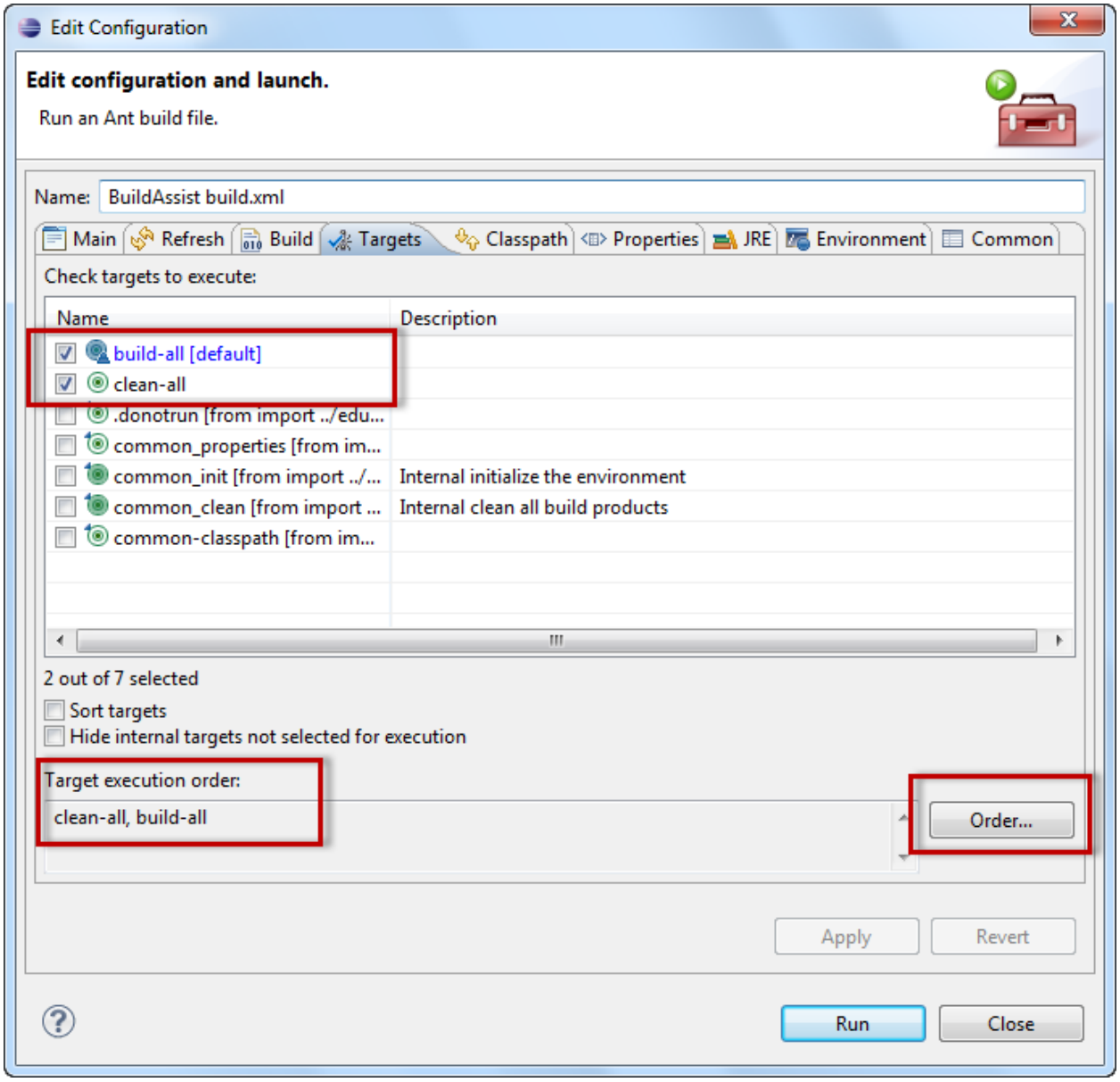
5. On the sub-menu there are two **Ant-Build** options, select the second occurrence.



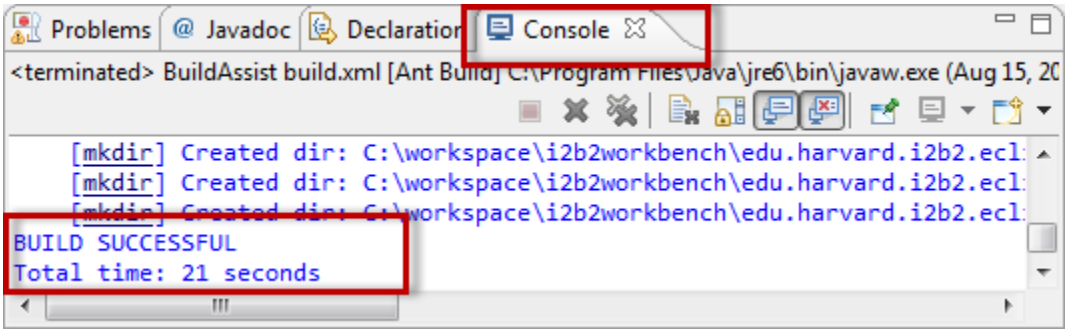
6. The build *Configuration* dialog box will open.



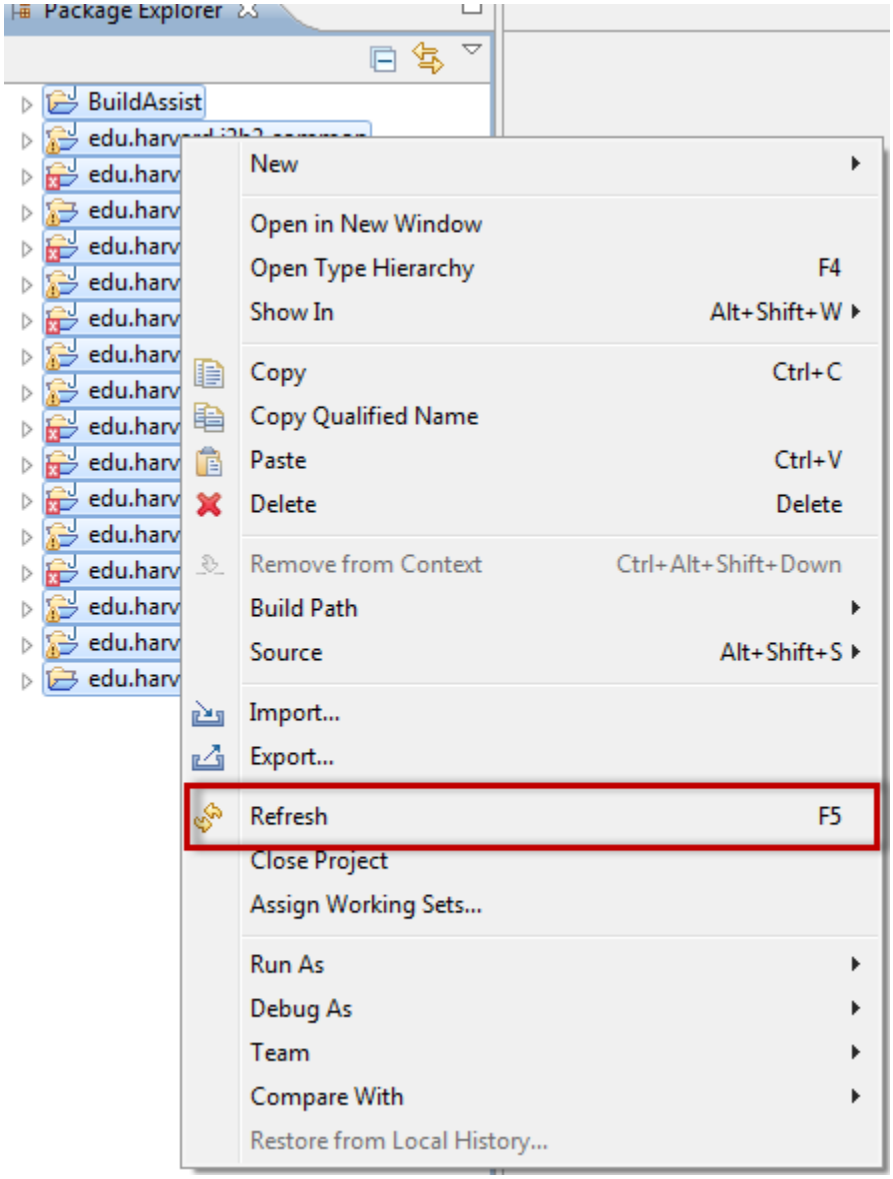
7. Verify the following settings:
    a. The **clean-all** and **build-all** boxes are checked.
    b. The **target execution order** is: “clean-all, build-all”
    c. If it is not in the correct order, simply click on the **Order** button and change the order.



8. Click on the **Run** button to execute.
9. Once the build has finished you should see **BUILD SUCCESSFUL i**n the console view.



10. The last step in the build process is to **refresh** all the projects.
    a. Select all the projects.
    b. Using the *right mouse button*, click on any one of the highlighted projects.
    c. A pop-up menu will display.
    d. Select **Refresh**.



e. A window will open showing the status of the refresh process.

f. Once the refresh has finished the status window will close.

Eclipse will automatically build your workbench once the refresh has finished.

11. Verify there are no errors once the process of building the workbench has completed. There are two ways to check for errors:

a. A red X appears on the project folder () indicating there is an error in that project.

b. Look at the **Problems** view to see if there are any errors listed.

If an error does occur you may want to try and repeat the steps shown above (build / refresh). Sometimes this will fix any errors that occur during the build process.

3. CONFIGURE THE I2B2 WORKBENCH

3.1. Define Hive Location

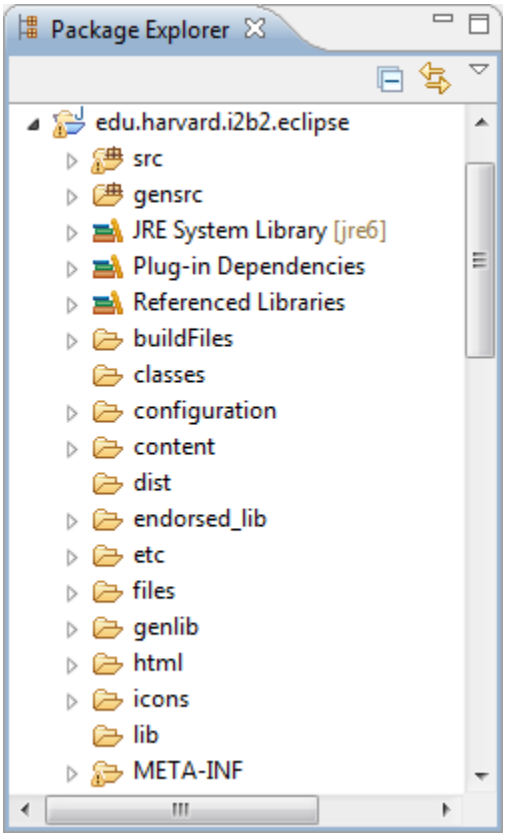
In the *edu.harvard.i2b2.eclipse* project there is a file called **i2b2workbench.properties**. When a user starts the logon process the i2b2 Workbench looks to this file to find out where the i2b2 hive is located.

By default the i2b2 Workbench is setup to work with the “i2b2demo” hive. In order for it to work with your hive you will need to update the properties file with your site information.

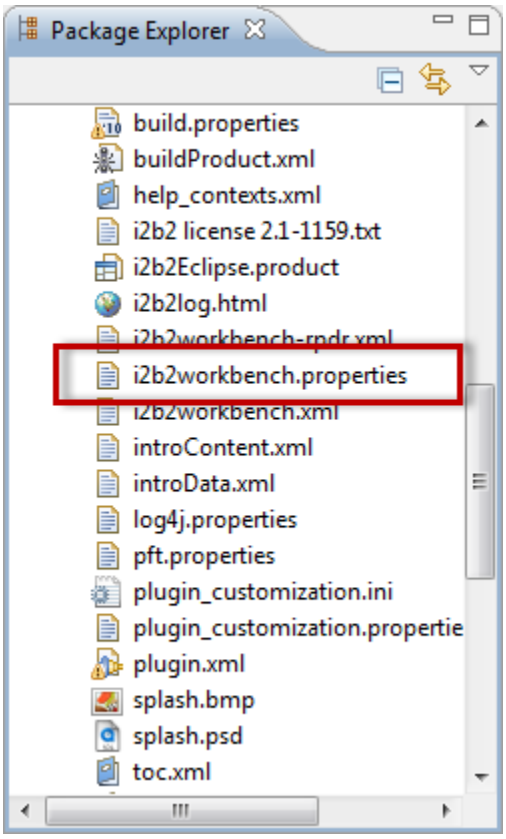
3.1.1. Access i2b2Workbench.properties file within Eclipse application

The method of accessing the i2b2Workbench.properties file from within the Eclipse applications is the same for both Windows and Macintosh.

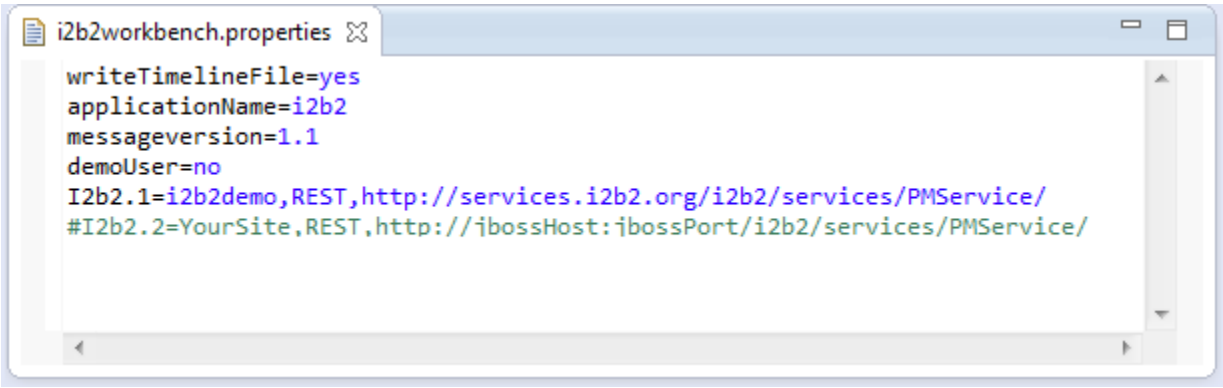
1. Double click on the **edu.harvard.i2b2.eclipse** project to open its folder.



2. Double click on the **i2b2workbench.properties** file.



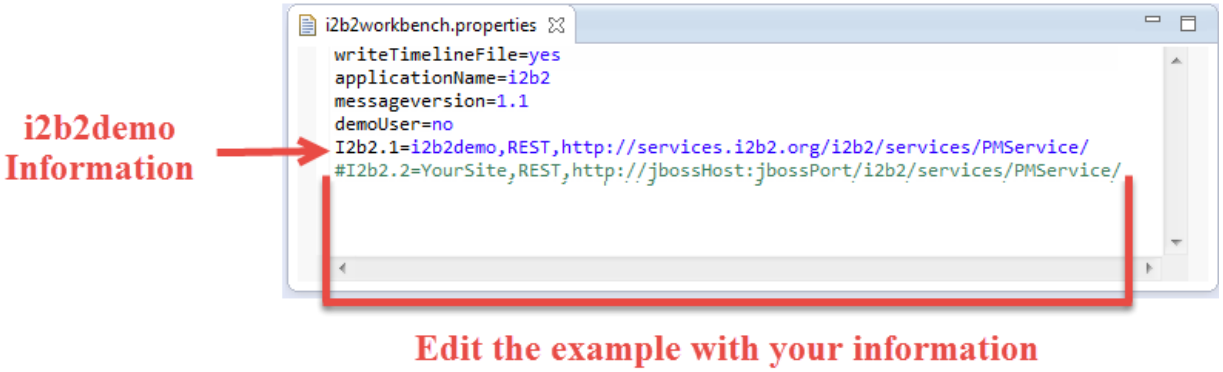
3. The properties file will open.



3.1.2. Editing the i2b2Workbench.properties file

The last two lines of the i2b2Workbench.properties file contain information about the location of the hive.

- The second to last line is the information for **i2b2demo**
- The last line is an example that can be edited with your information



1. Edit the following information:
    a. Remove the pound sign (#) at the beginning of the line as this denotes a comment. If it is not removed the i2b2 Workbench will ignore the string and you will not be able to log onto that domain.
    b. Update ***YourSite*** with the named of **your** domain. Keep in mind this is case sensitive.
    c. Update ***jbossHost:jbossPort*** with the name of **your** jboss host and its port.

If you do not want to see i2b2demo as an option when logging into the i2b2 Workbench you can either remove it or comment it out by placing a pound sign (#) in front of the string.

2. If you have more than one environment you can include all of them in this on properties file by doing the following:
    a. Copy the string you just edited to a new line.
    b. Update the domain and jboss information with the correct information for the additional hive.
    c. Make sure you advance the number at the beginning of the string (“I2b2”)

Example:

I2b2.1=i2b2demo,REST,http:// services.i2b2.org/i2b2/services/PMService/

I2b2.2=**TestDomain**,REST,http://**mytestJboss.org:jbossPort**/i2b2/services/PMService/

I2b2.**3**=**ProdDomain**,REST,http://**mytestJboss.org:jbossPort**/i2b2/services/PMService/

3. Click on the **Save** button  to save the changes to the file.

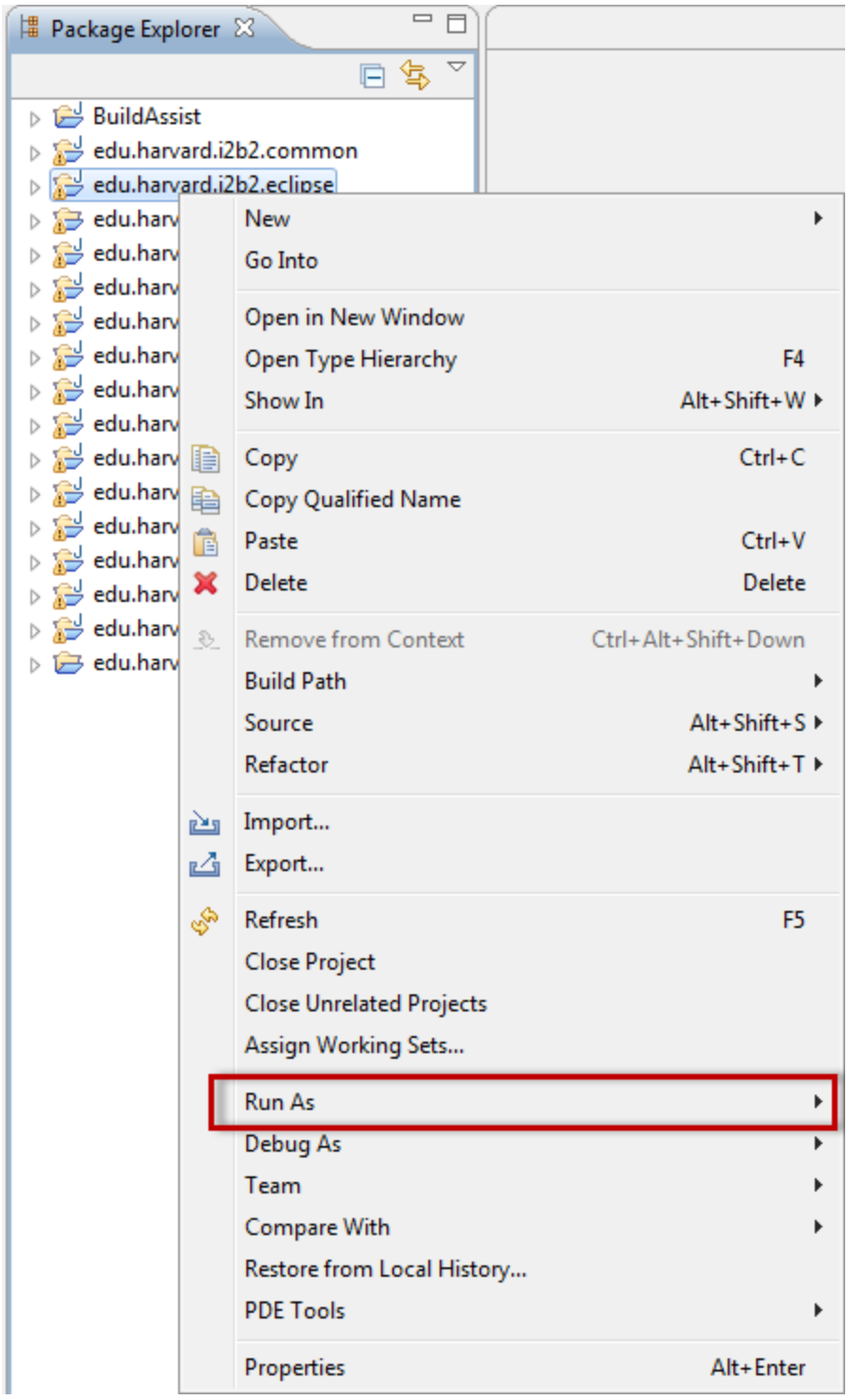
3.2. Define Run Configurations

In order to run the i2b2 Workbench from within Eclipse you need to define the run configurations.

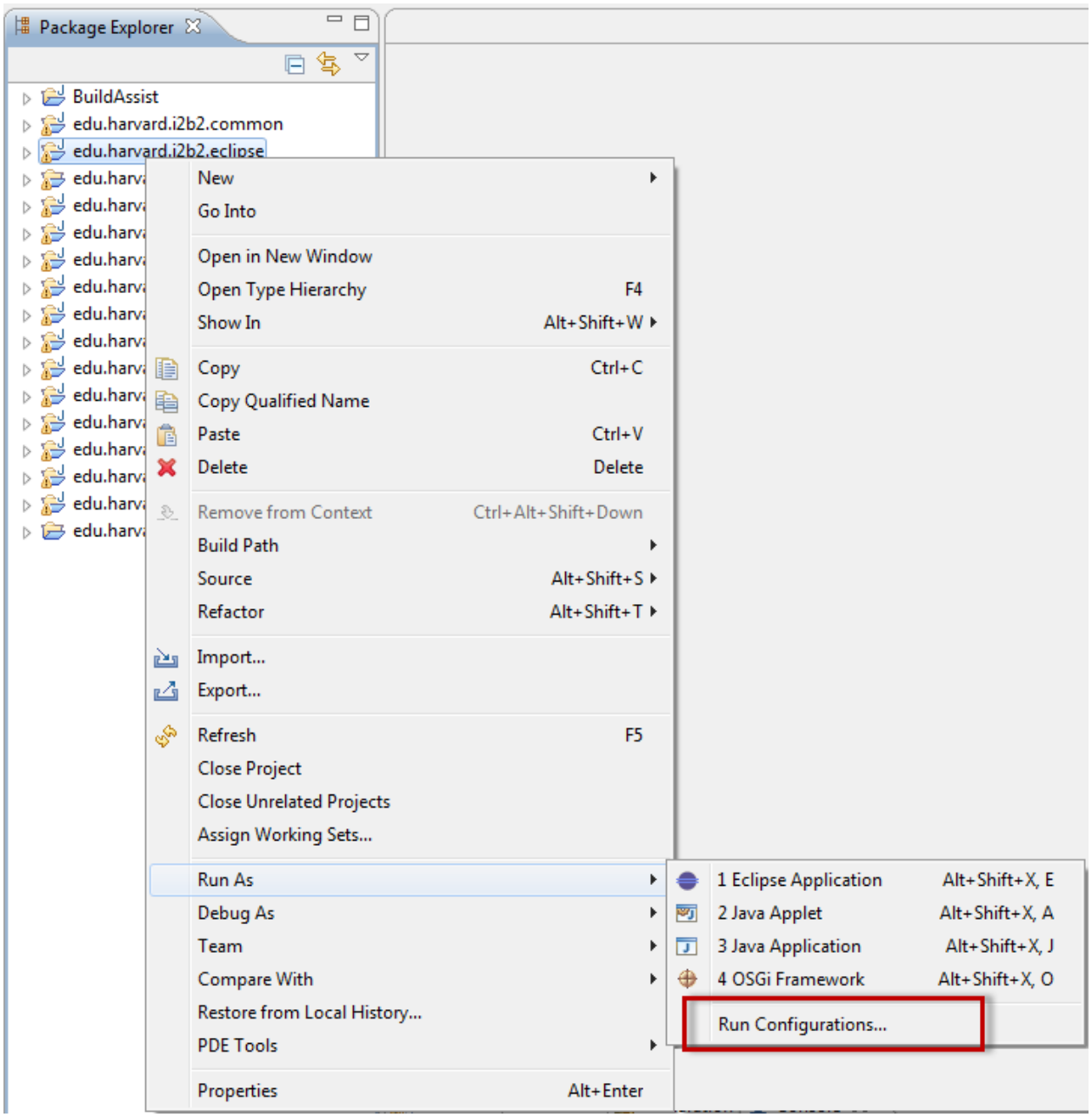
You only need to set up the run configuration once and it is only necessary when running from within Eclipse.

3.2.1. Accessing Run Configurations Window

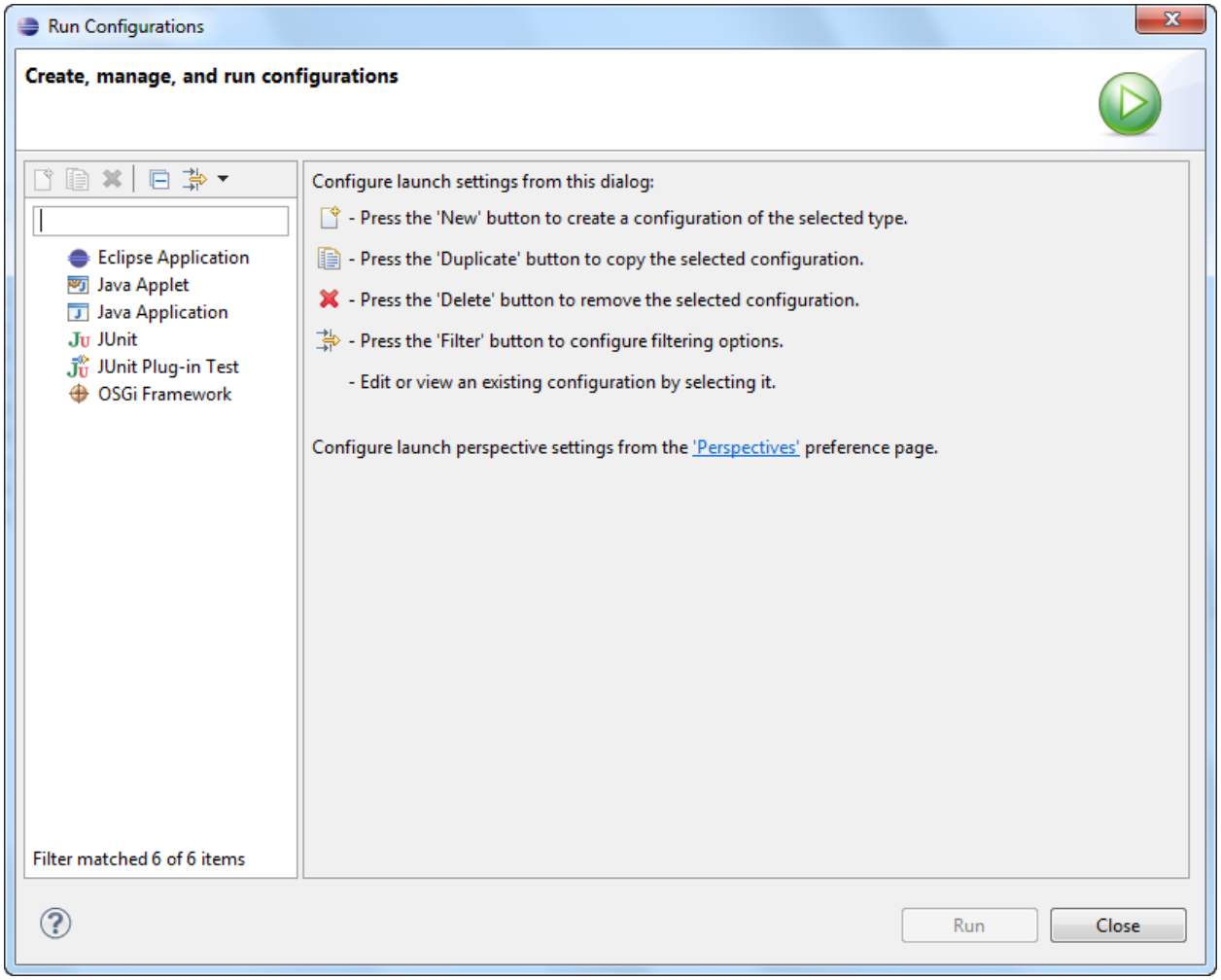
1. Using the *right mouse button*, click on the **edu.harvard.i2b2.eclipse** project.
2. A pop-up menu will display a list of options.
3. Using your *mouse*, hover over **Run As** to display a sub-menu.



4. Click on **Run Configurations**.



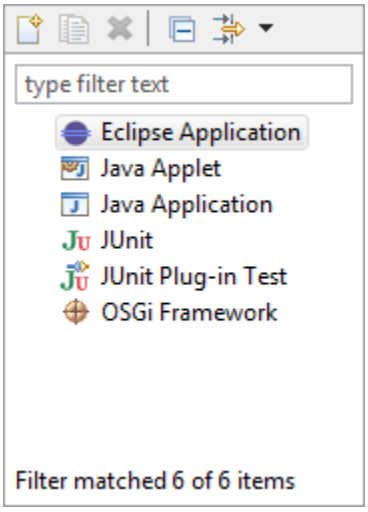
5. The **Run Configurations** window will open.



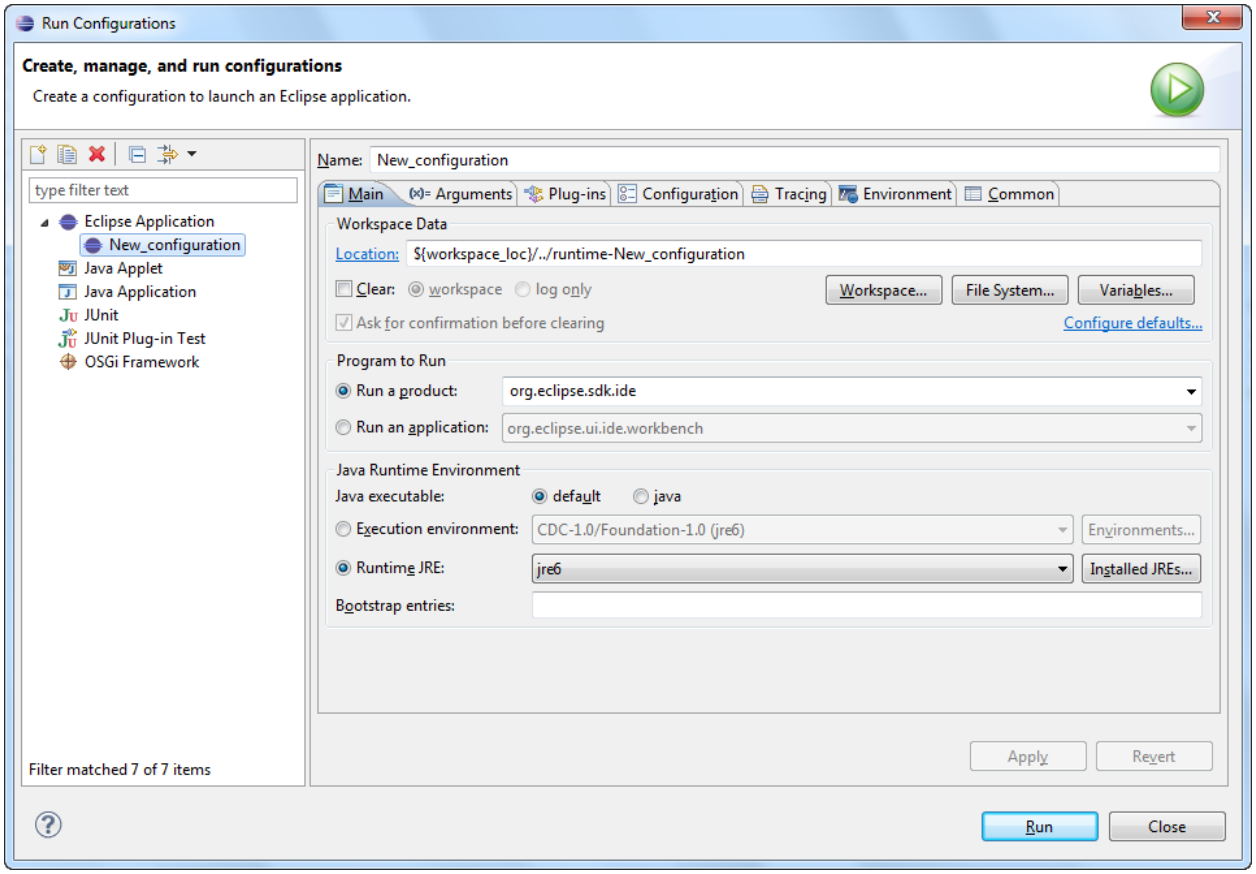
The next three sections outline the process of setting up the configuration information on the *Main*, *Arguments*, and *Plug-ins* tabs.

3.2.2. Main Tab

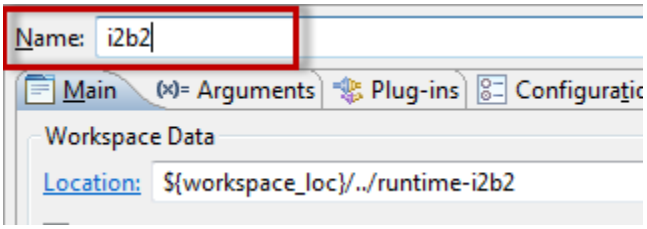
1. On the left side of the *Run Configuration window* is a navigation bar listing the different types of configurations.



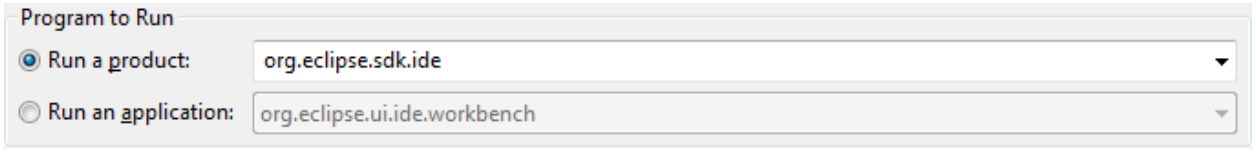
2. Highlight **Eclipse Application** by clicking on its name.
3. Click on the **New** button located above the navigation bar.
4. The following will occur:
    a. A new configuration called **New_configuration** will appear under *Eclipse Application*.
    b. On the right side of the Run Configurations window, the **Create, manage, and run configurations** wizard will open to the Main tab.



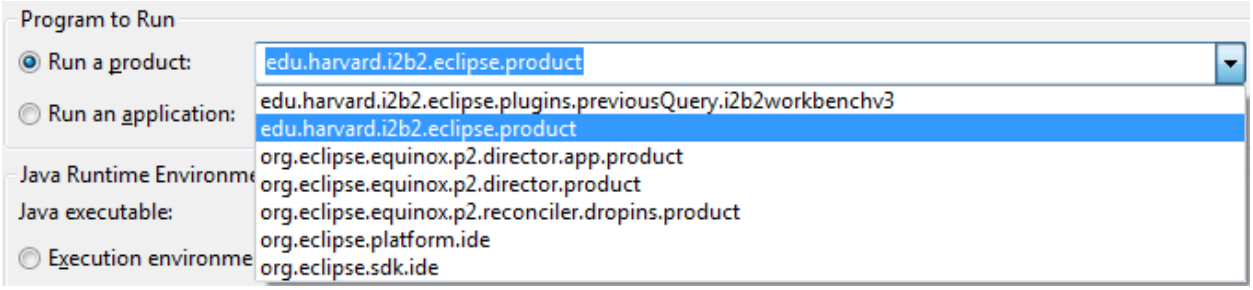
5. Enter a meaningful name at the **Name** field.



6. In the **Program to Run** section, verify the *radial dial* next to **Run a product** is selected.

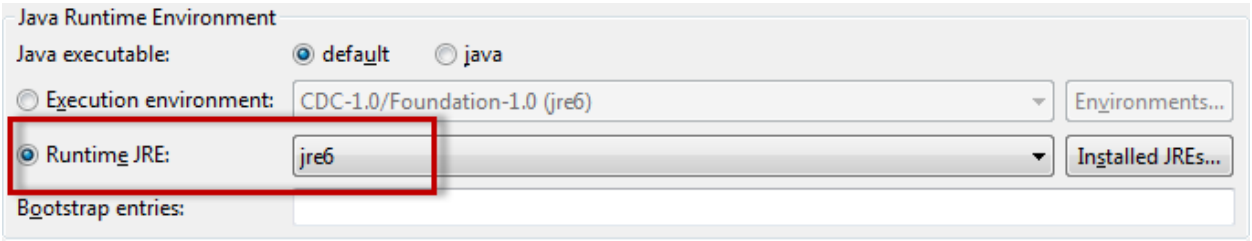


7. Click on the **down arrow** to display the *drop-down list*.
8. Select **edu.harvard.i2b2.eclipse.product** from the drop down list.

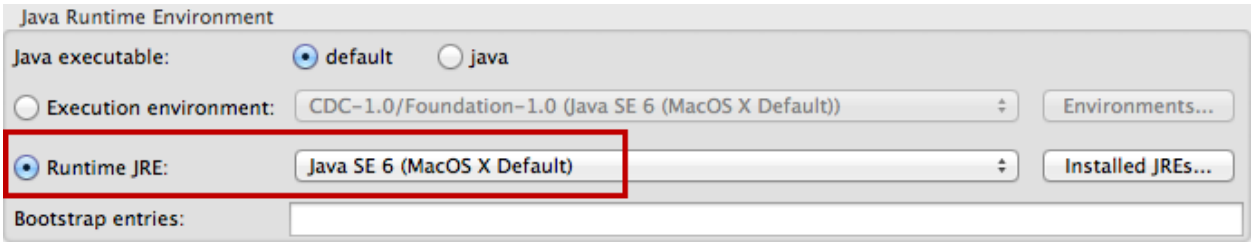


9. Update the **Runtime JRE** in the *Java Runtime Environment*.

Windows:

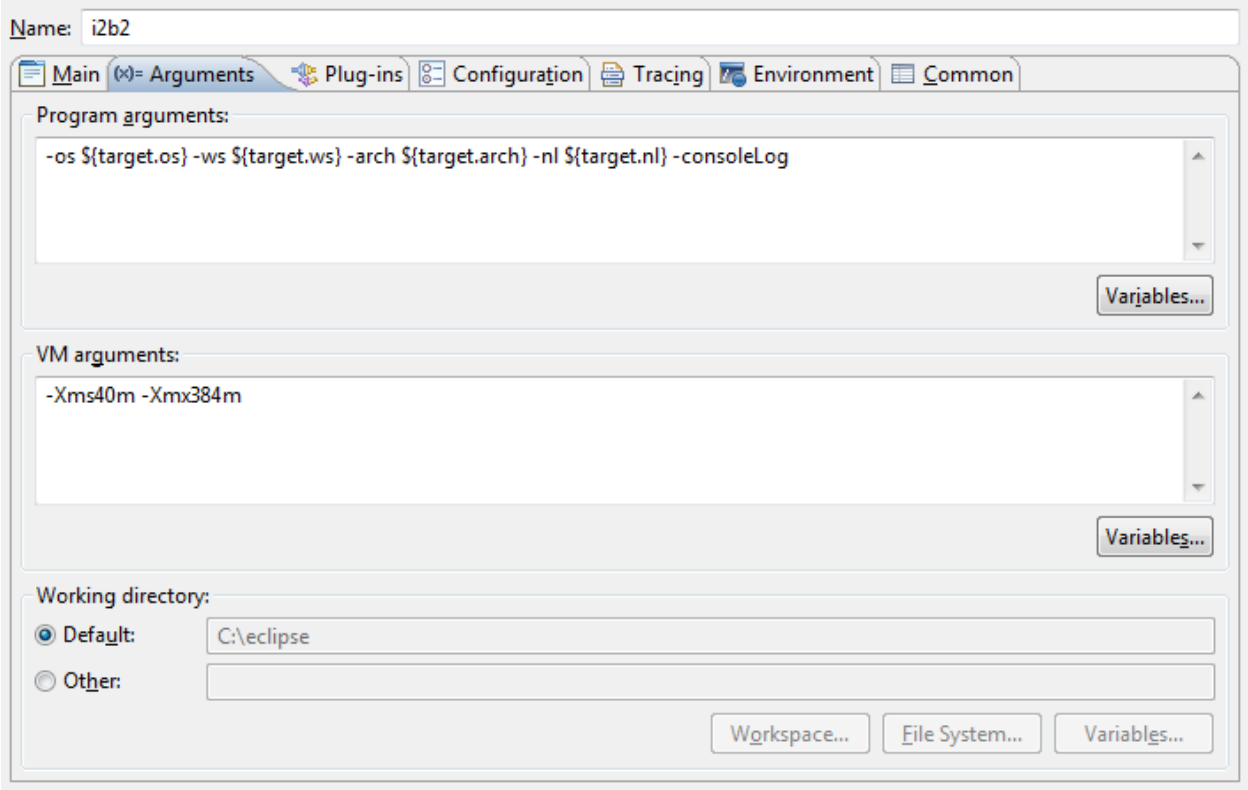


Macintosh:



3.2.3. Arguments Tab

1. Click on the **Arguments** tab.
2. The Arguments page will display.



3. Edit the **VM arguments** section with the following information.

Windows:

Keep the following arguments that Eclipse will default automatically. If not then they need to be added.

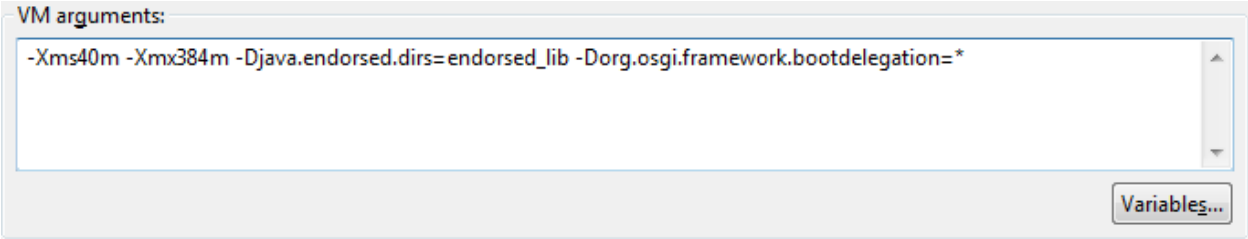
-Xms40m

-Xmx384m

Add the following arguments:

-Djava.endorsed.dirs=endorsed_lib

-Dorg.osgi.framework.bootdelegation=*



The value defined at Xmx is equal to the amount of physical memory on your computer. If needed you can increase this value to equal the physical memory on your computer.

Macintosh:

Keep the following arguments that Eclipse will default automatically. If not then they need to be added.

-Xms40m

-Xmx512m

-Dorg.eclipse.swt.internal.carbon.smallFonts

Remove the following arguments that Eclipse will default automatically.

-Xdock:icon=../Resources/Eclipse.icns

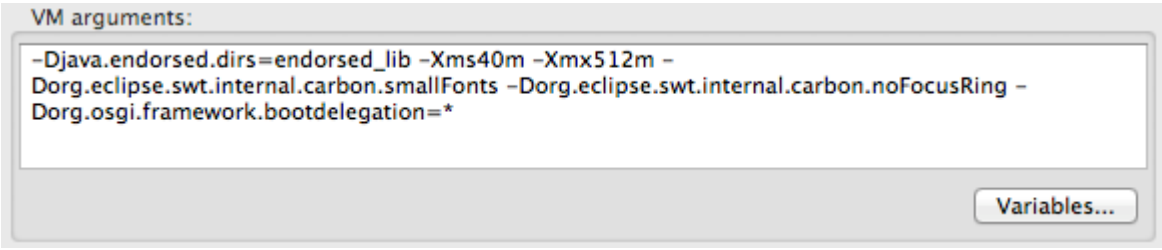
-XstartOnFirstThread

Add the following arguments:

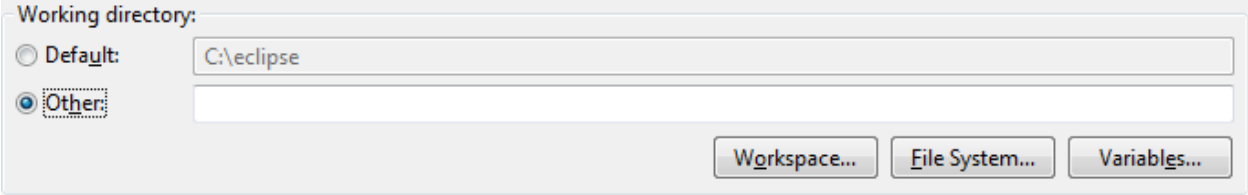
-Djava.endorsed.dirs=endorsed_lib

-Dorg.eclipse.swt.internal.carbon.noFocusRing

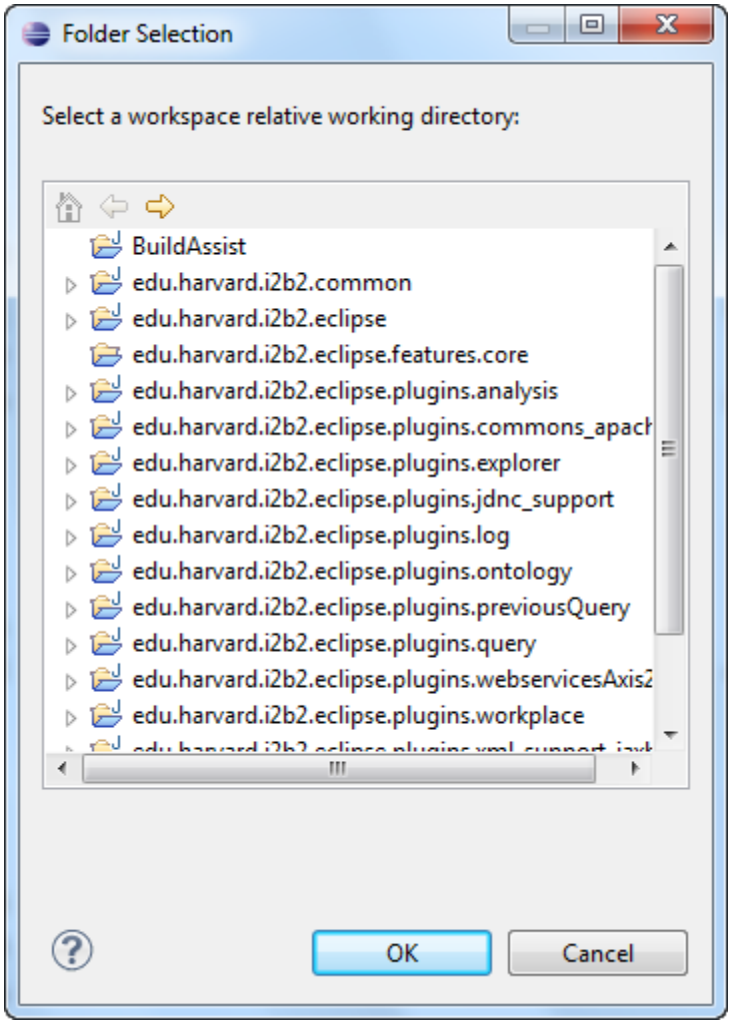
-Dorg.osgi.framework.bootdelegation=*



4. In the **Working directory** click on the *radial dial* next to **Other**.

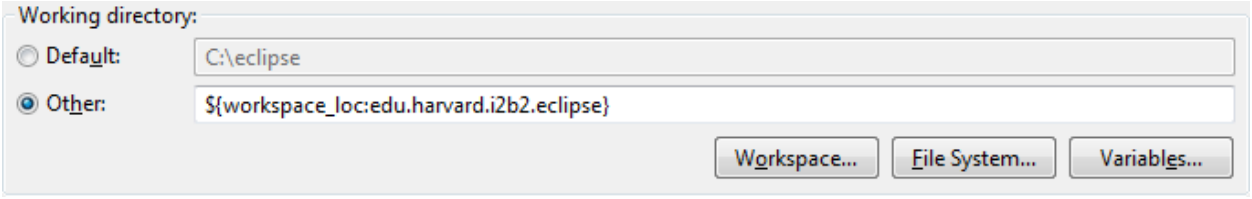


5. Click on the **Workspace** button.
6. The *Folder Selection* window will open.



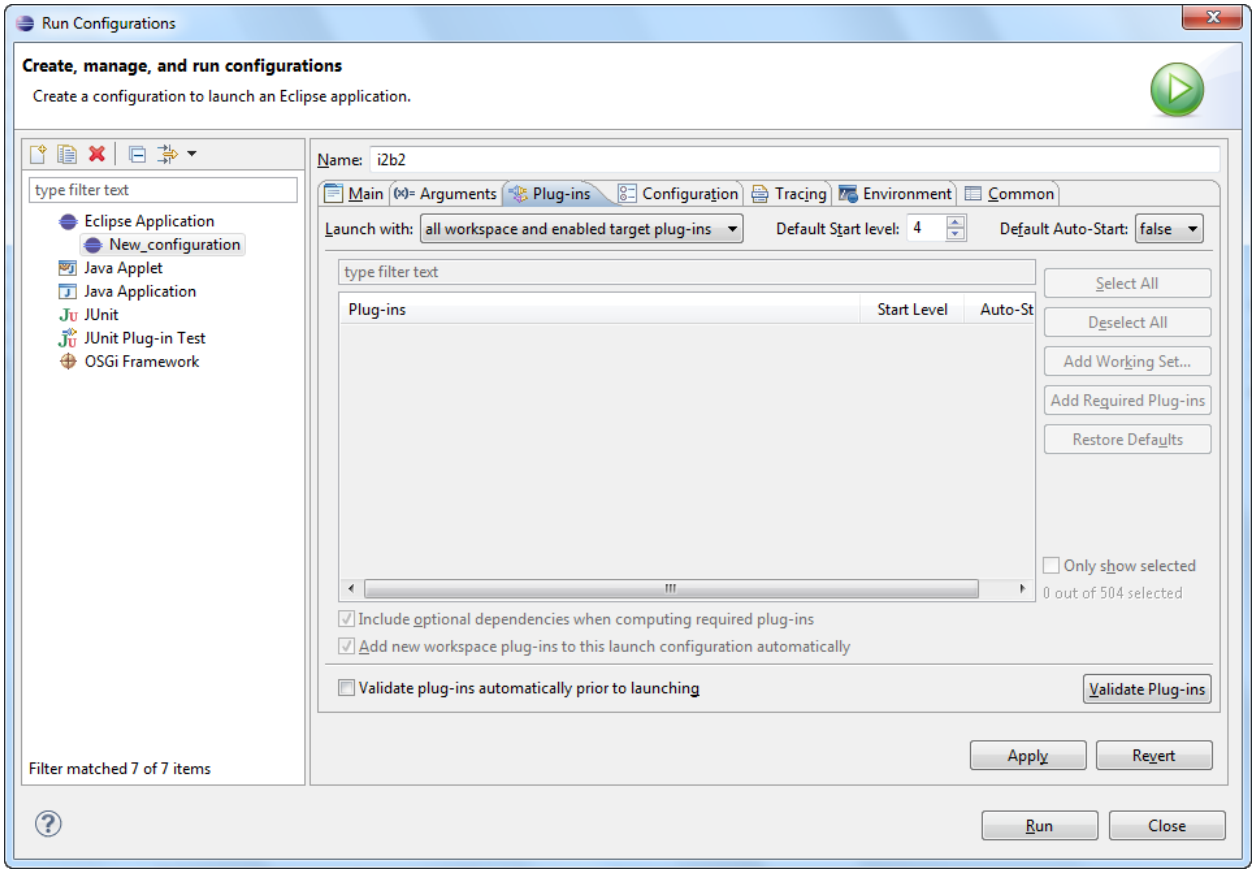
7. Select **edu.harvard.i2b2.eclipse**.

8. Click on the **OK** button.
9. The *Other* field will be populated with the workspace information.



3.2.4. Plug-Ins Tab

You can configure any combination of workspace plug-ins and target platform plug-ins as your IDE configuration. The default is that all **workspace and enabled target plug-ins** are launched.



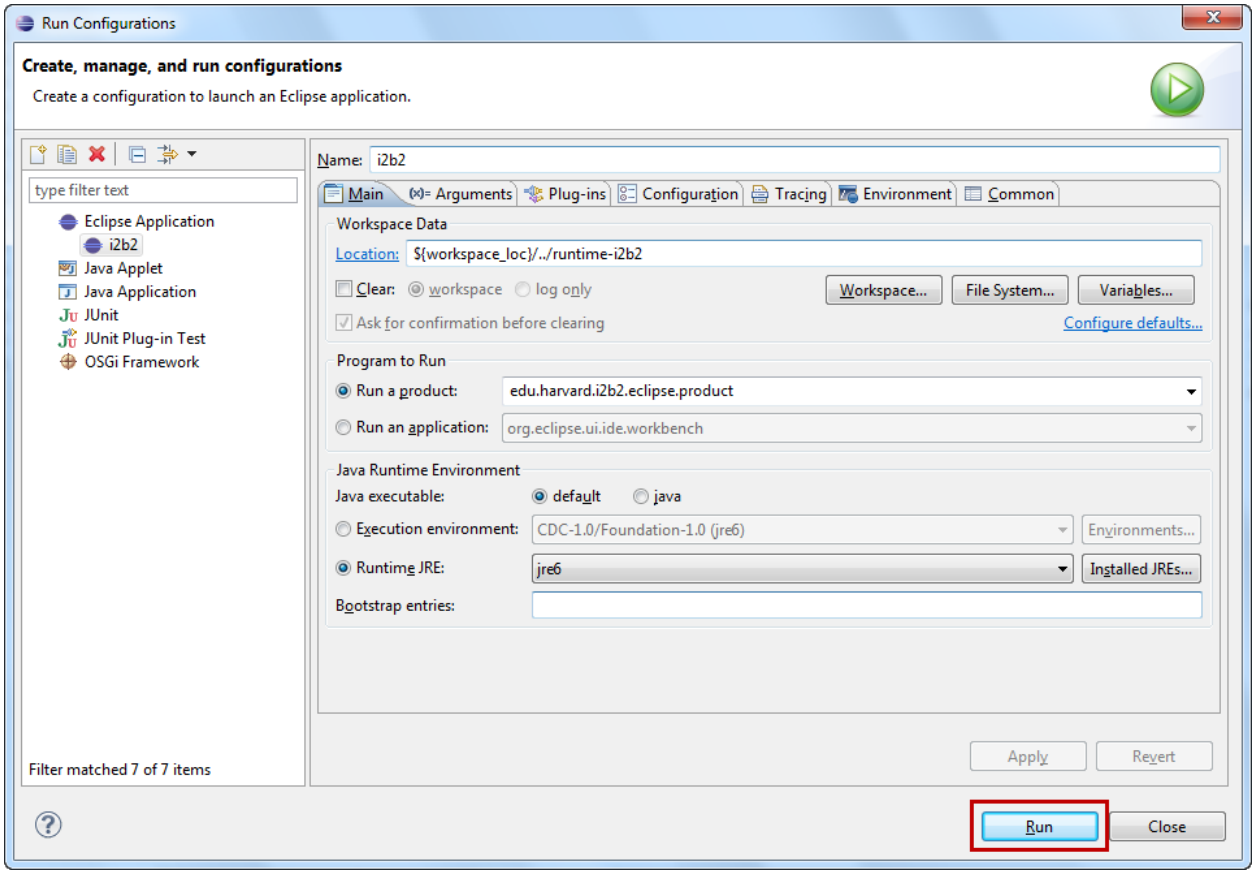
At this time we recommend that you click on the Apply button to save your settings.

4. EXECUTE

4.1. Different Methods to Run an Eclipse Application

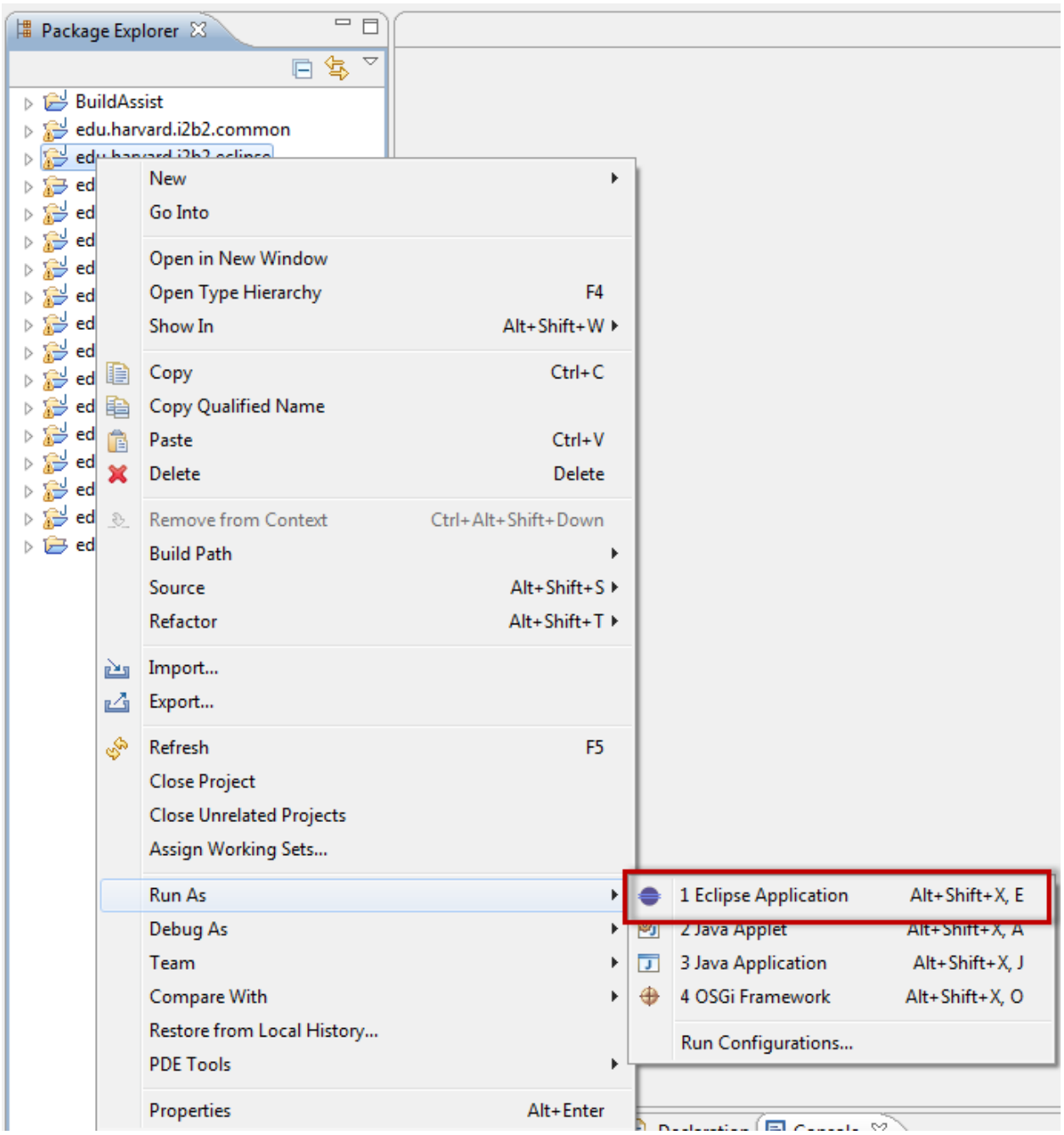
There are three different ways in which you can run an Eclipse Application like the i2b2 Workbench.

METHOD 1: Click on the **Run** button from within the *Run Configuration window*.



See section called "Accessing Run Configurations Window" if you need to know how to access this window.

METHOD 2: Using the *right mouse button* to display the pop-up menu and selecting **Run As** and then **Eclipse Application**.



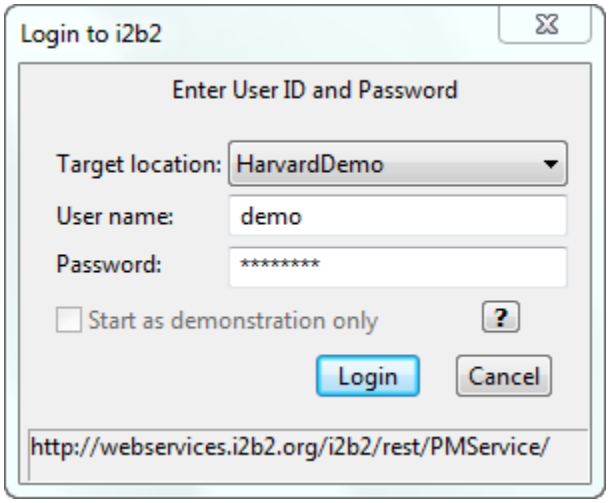
METHOD 3: Click on the **Run** button on the Eclipse toolbar .

4.2. Running the i2b2 Workbench

The previous section defines the different methods you can use to launch the i2b2 Application. This section assumes you have already selected to run the application using one of those methods.

1. The splash screen for the i2b2 Workbench will appear.

2. The **Login** window will open.



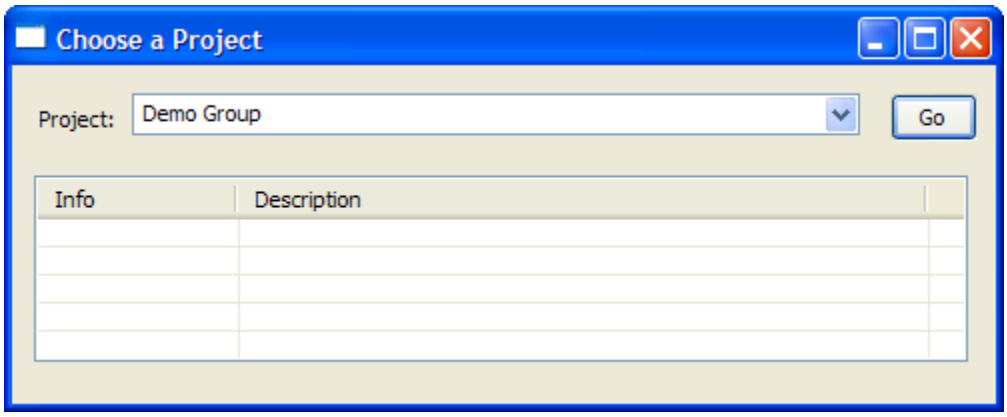
3. Enter your user name and password. For the i2b2 demo site you can enter the following user credentials.

   User name: demo

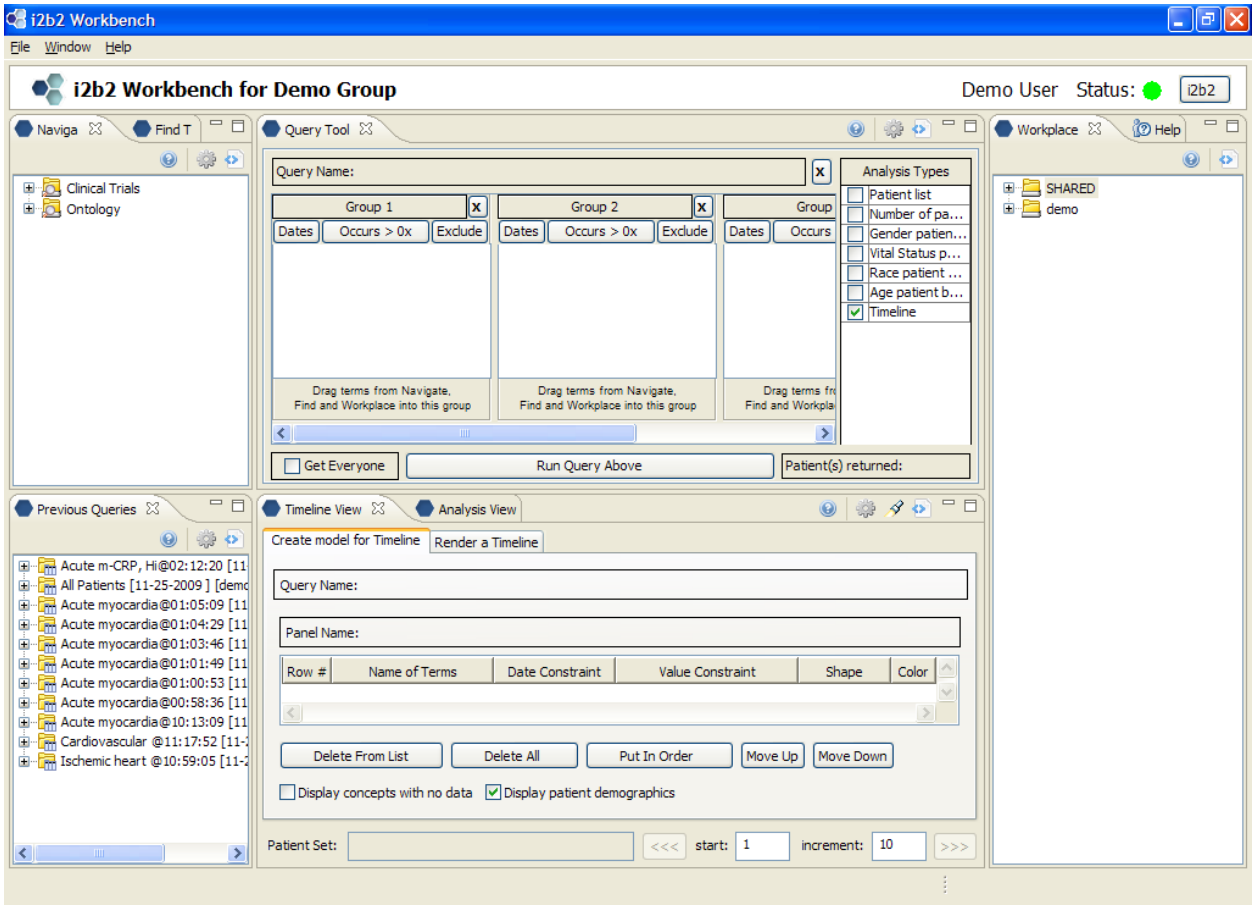
   Password: demouser

4. Click on the **Login** button.
5. The **Choose a Project** dialog box will open.

The project dialog box will only appear if you have access to multiple projects. If you only have access to one project you will be brought to the Welcome page.



6. Accept the default project by simply clicking on the **Go** button.
7. The **desktop** of the i2b2 Workbench will open.



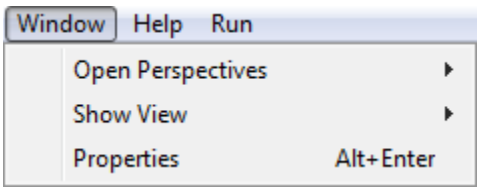
4.3. Reconfiguring the Default Layout

The default layout can be reconfigured to the user's needs.

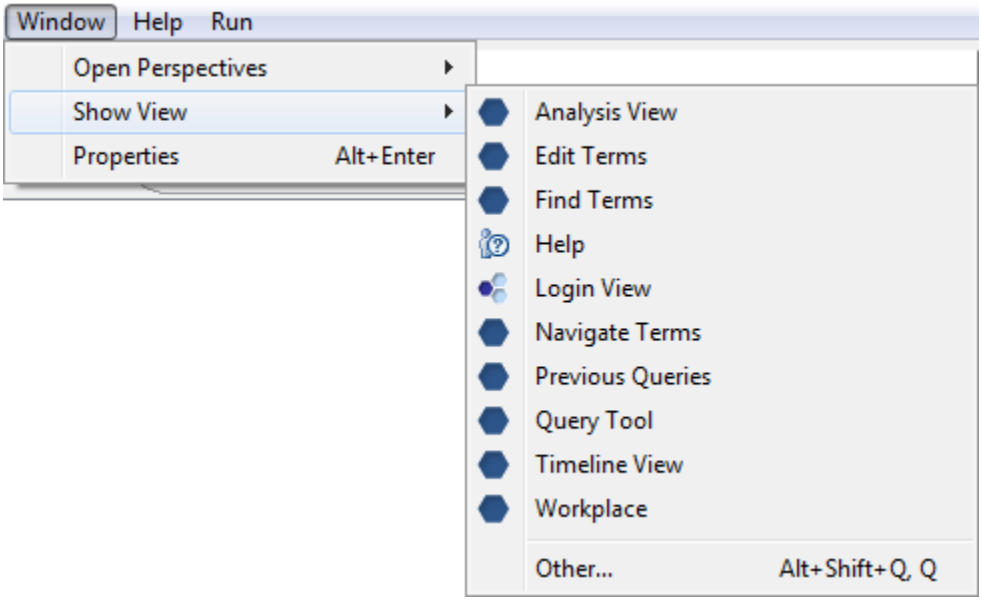
4.3.1. Adding a View

The following steps explain how to add a view to the desktop of the i2b2 Workbench.

1. On the toolbar, click on **Window**.
2. A drop-down menu will display.



3. Using your *mouse,* hover over **Show View** to display the sub-menu listing all the core views.



4. Click on the name of the view you want to add.
5. The view will be added to the desktop area of the i2b2 Workbench.

"Core views" are those views that are associated with the i2b2 cells that are part of the core hive. Select "other" to see a list of all views including the optional ones.

View	Description
Analysis View	Provides a visualization of the analysis results for queries that users run in the i2b2 Workbench.
Find Terms	Allows users to search for concepts and modifiers.
Help	Displays the help documentation for the active view.
Navigate Terms	Displays the i2b2 vocabulary concepts and modifiers that are maintained in the **Ontology Management Cell** of the **i2b2 Hive**.
Previous Queries	Displays the most recent queries run by a user. The results associated to the

	query are also available for viewing.
Query Tool	The Query Tool view is designed to simplify the process of retrieving information from the database associated to the i2b2 Workbench.
Timeline View	Displays the information in a graphical representation of a chronological sequence of events
Workplace	Information in the Workplace view is related to the most common concepts and queries that an individual uses and in essence becomes their personal workspace. The main objectives of this view are: ▪ Save and organize user specific items. ▪ Share information with other project users. ▪ Project managers can view the workplace for individual team members.

4.3.2. Moving a View

Move the position of any view by clicking on the view's tab and dragging it to another area of the i2b2 Workbench (desktop area).

4.3.3. Removing a view

At the top of each view is a tab that contains the name of the view and a close button. Simply click on the close button ⊠ to remove a view from the desktop.

5. PACKAGE STRUCTURE

The ***archive file*** contains the projects listed below. The project / directory **edu.harvard.i2b2.eclipse** is considered the *base plug-in* for the i2b2 Workbench and contains the property and configuration files.

Directory	Description
BuildAssist	Files to build the i2b2 Workbench.
edu.harvard.i2b2.common	Common core exception, jaxb, and utilities used by i2b2 hive.
edu.harvard.i2b2.eclipse	The base plug-in for our development (namely the user interface for the top panel)
edu.harvard.i2b2.features.core	The base features plug-in for the i2b2 core views.
edu.harvard.i2b2.plugins.analysis	User interface for the *Analysis view*.
edu.harvard.i2b2.plugins.commons_apache	Third party jars needed for common functions.
edu.harvard.i2b2.plugins.exporer	User interface for the *Timeline view*.
edu.harvard.i2b2.plugins.jdnc_support	Third party jars needed for the user interface support
edu.harvard.i2b2.plugins.log	Third party jars needed for logging
edu.harvard.i2b2.plugins.ontology	User interface for the *Ontology views*
edu.harvard.i2b2.plugins.previousQuery	User interface for the *Previous Query view*
edu.harvard.i2b2.plugins.query	User interface for the *Query view*
edu.harvard.i2b2.plugins.webservicesAxis2	Third party jars needed for web service support
edu.harvard.i2b2.plugins.workplace	User interface for the *Workplace view*
edu.harvard.i2b2.plugins.xml_support_jaxb	Third party jars needed for jaxb / message processing
edu.harvard.i2b2.plugins.xml_support_jdom	Third party jars needed for jdom / message processing
edu.harvard.i2b2.plugins.xml	XSD for the hive

6. ECLIPSE (I2B2) FEATURE

In Eclipse the plug-ins can be packaged together as an Eclipse Feature. The **edu.harvard.i2b2.eclipse.features.core** plug-in is the baseline Eclipse Feature for the i2b2 Workbench. Included in this feature are the core i2b2 Views and required plug-ins to run the i2b2 Workbench. As optional views are developed they will be packaged up into a feature and added to an update site where users can easily install the new view from within the i2b2 Workbench.

The following sections provide a general overview. For more detailed information about Eclipse Features please go to the Eclipse website.

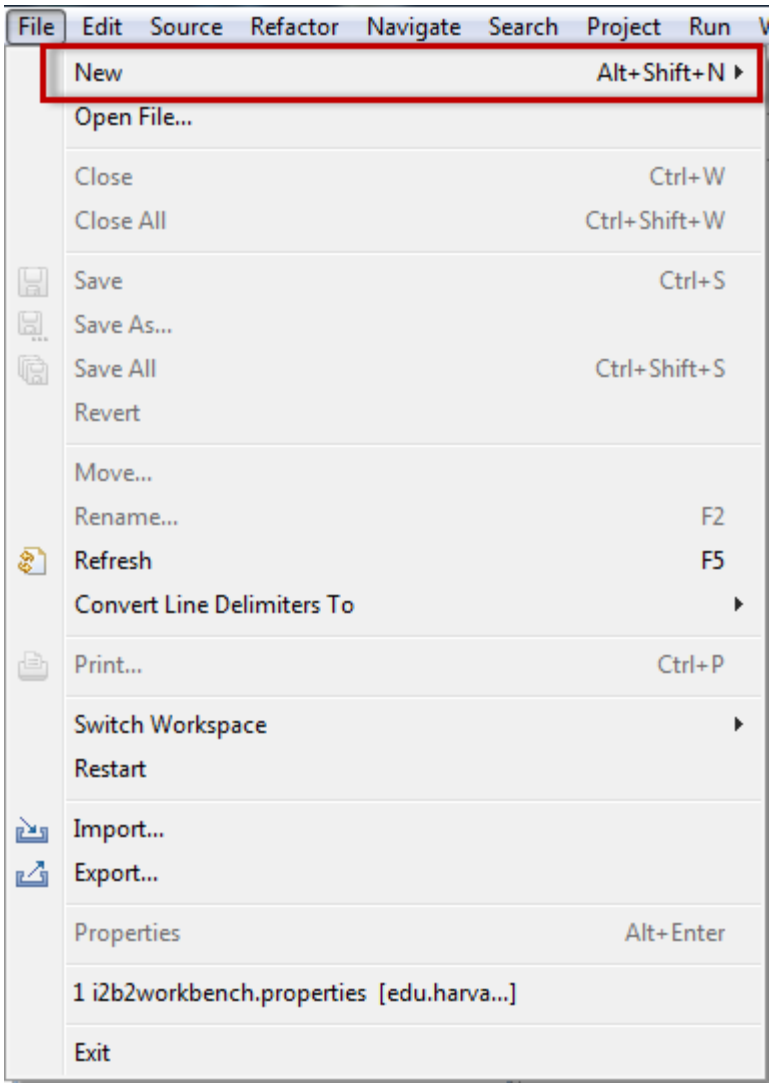
The following sections outline the process of creating a Feature plug-in for the i2b2 Workbench. For information on how to install the feature from within the i2b2 Workbench please see the **Installation Guide** for the i2b2 Workbench.

6.1. Creating a New "Feature" Project

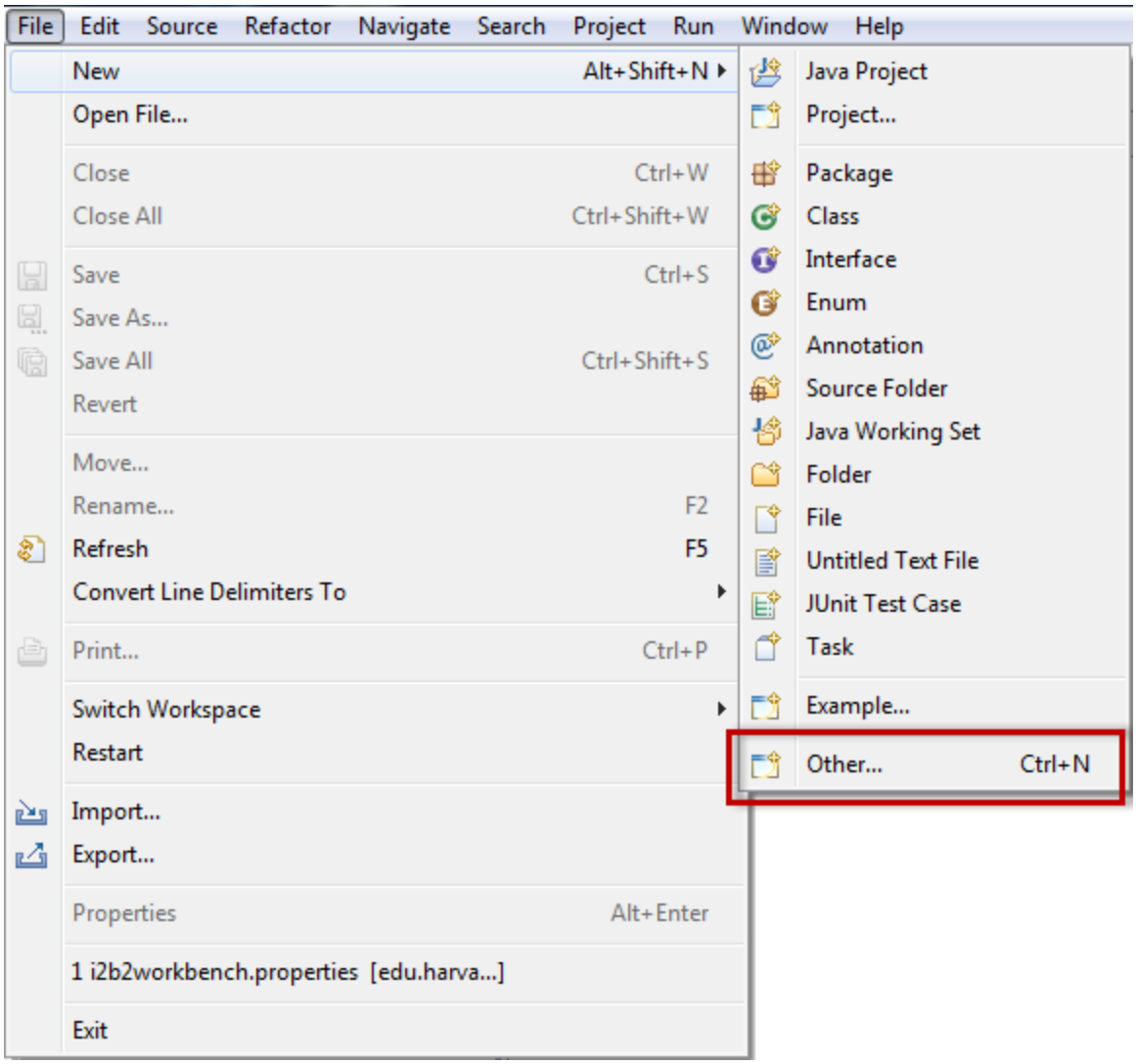
The following steps outline how to create a new Feature Project for a custom plug-in developed by the user.

This section assumes the user has already created their custom plug-in (project). For more detailed information about creating plug-ins please see the documentation provided by the Eclipse application.

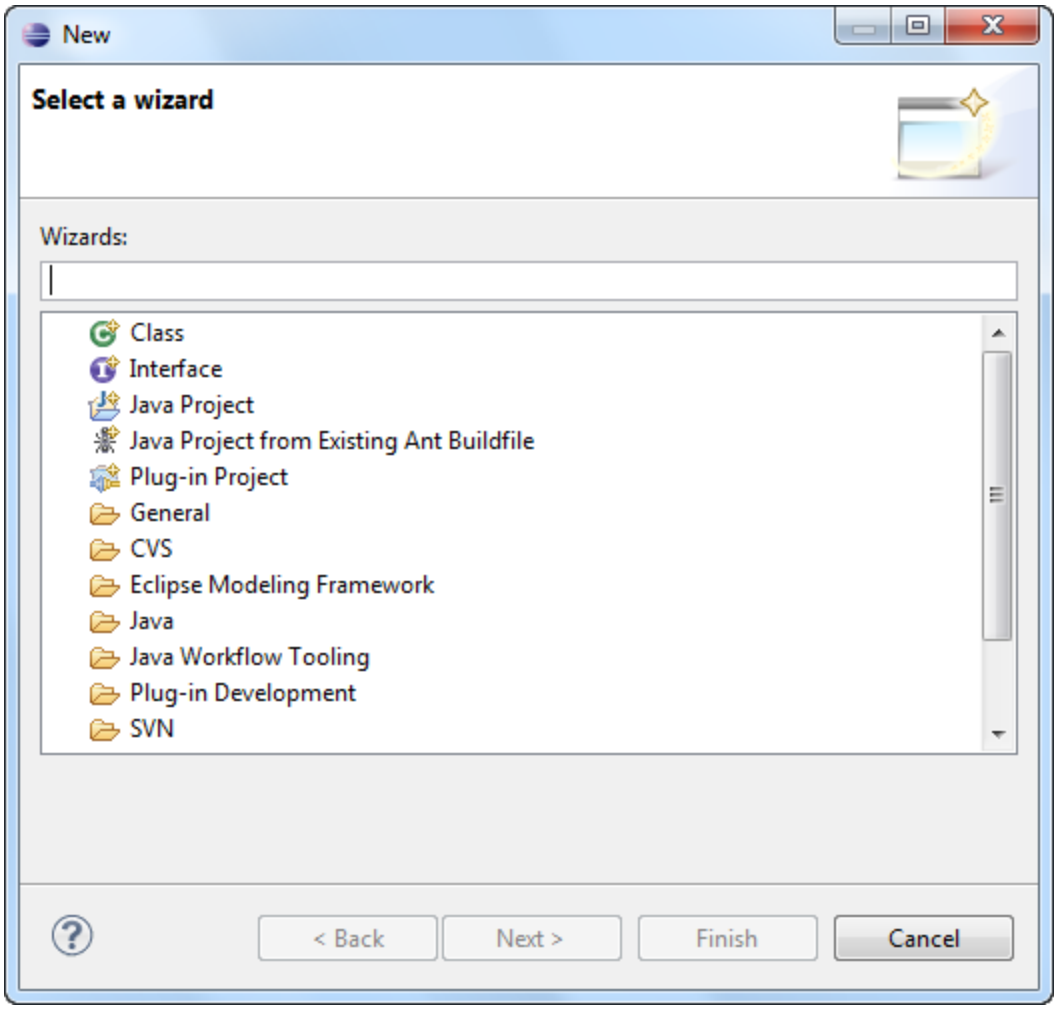
1. On the Eclipse menu bar click on **File**.
2. A drop-down menu will display.
3. Using your *mouse*, hover over **New** to display a sub-menu.



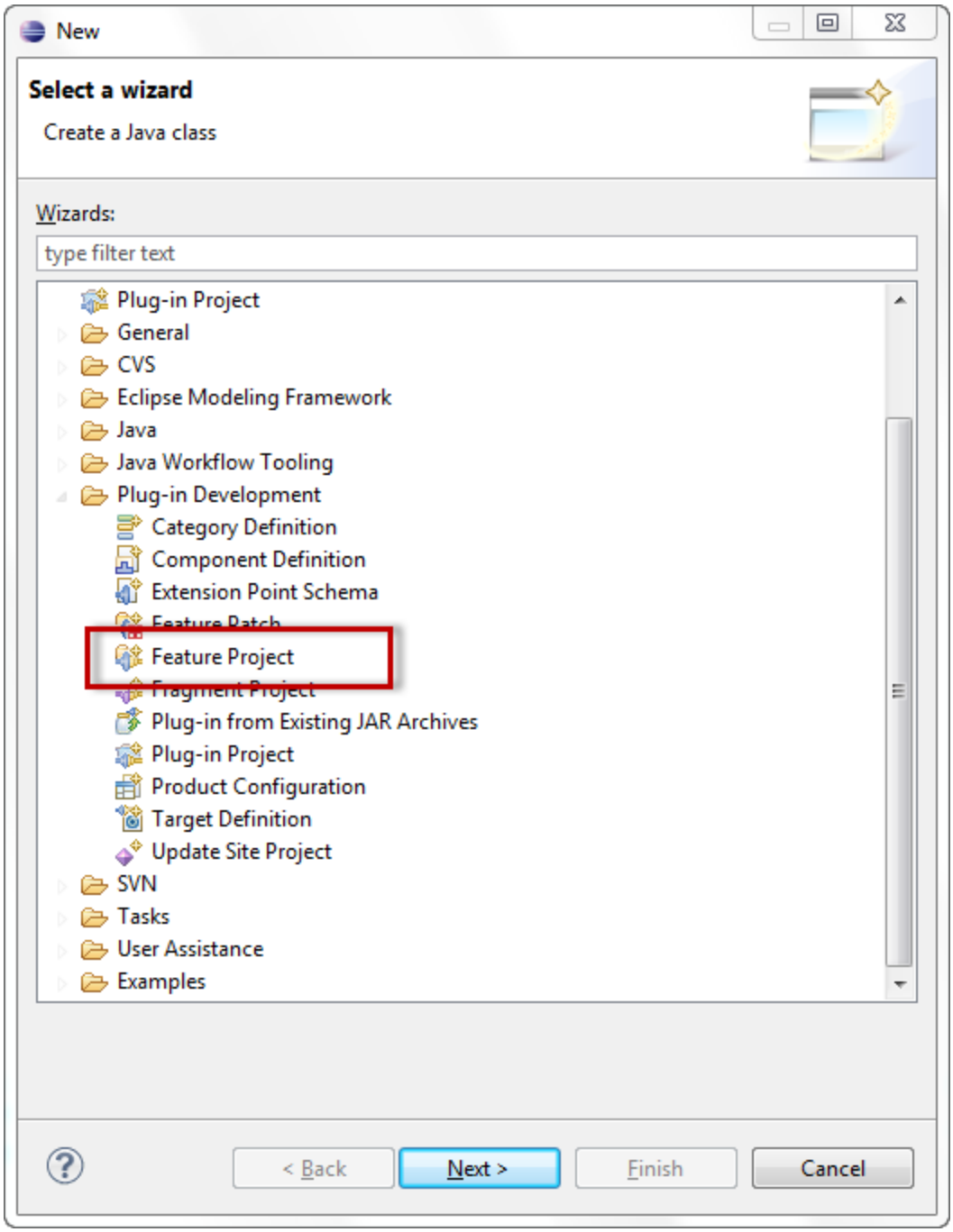
4. Select **Other** from the sub-menu.



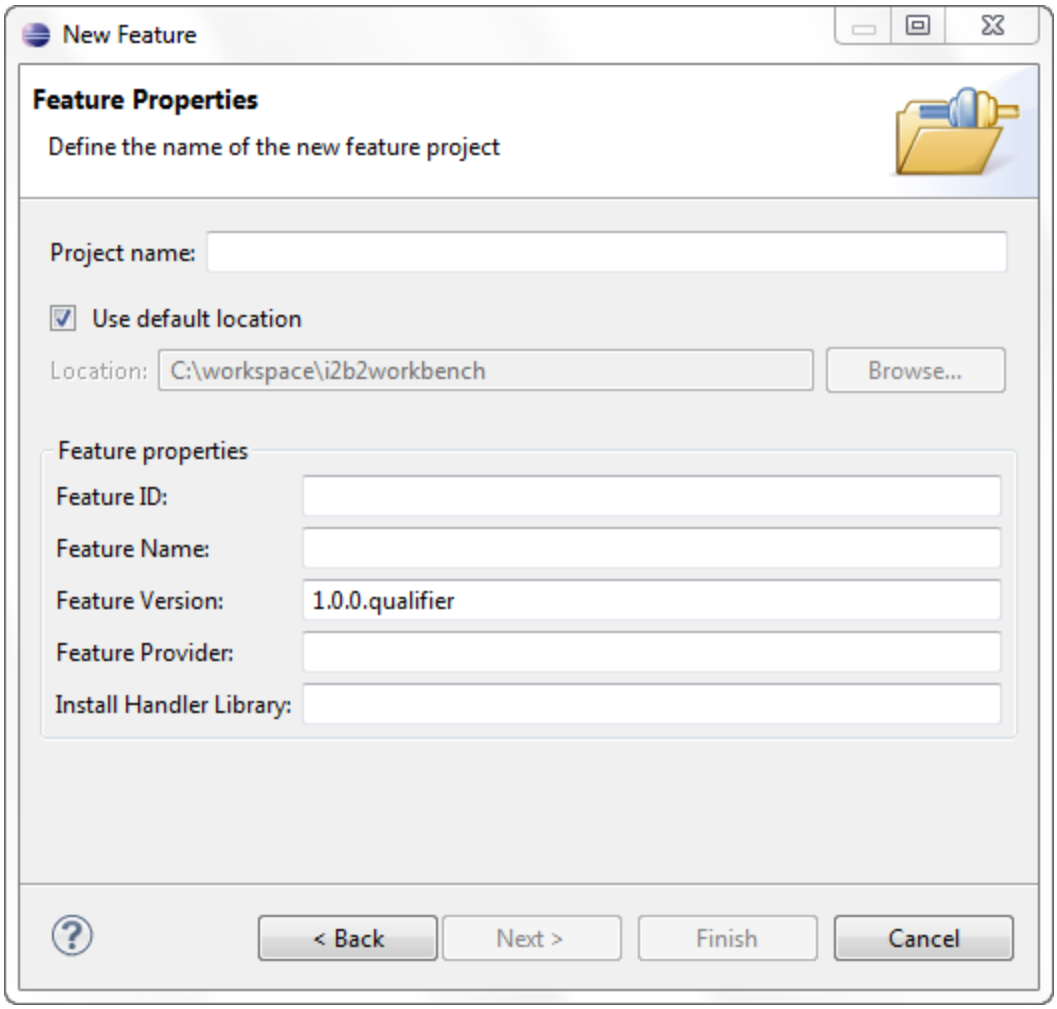
5. The **New** dialog box will open to the **Select a Wizard** page.



6. Double click on **Plug-in Development** to open the folder.
7. Click on **Feature Project**.



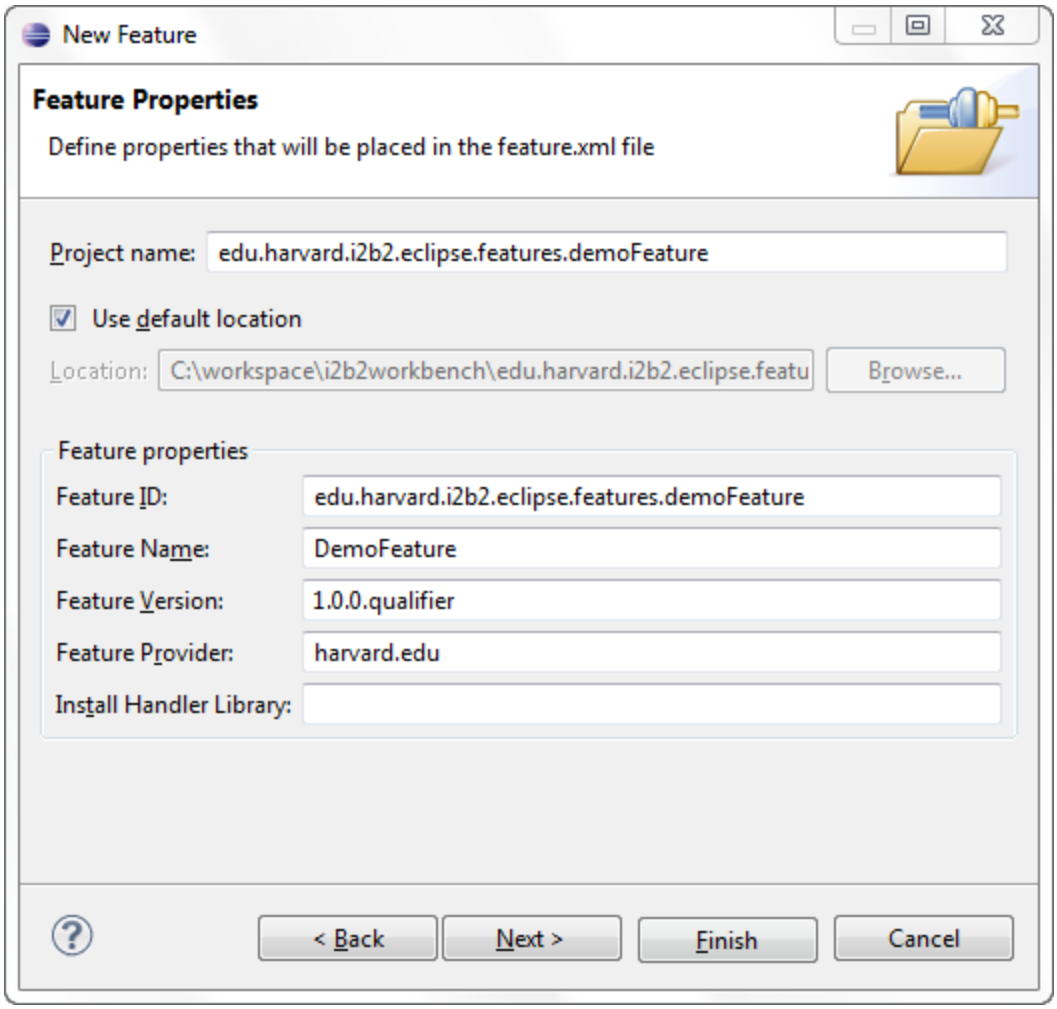
8. Click on the **Next** button.
9. The **New Feature** dialog box will open to the **Feature Properties** page.



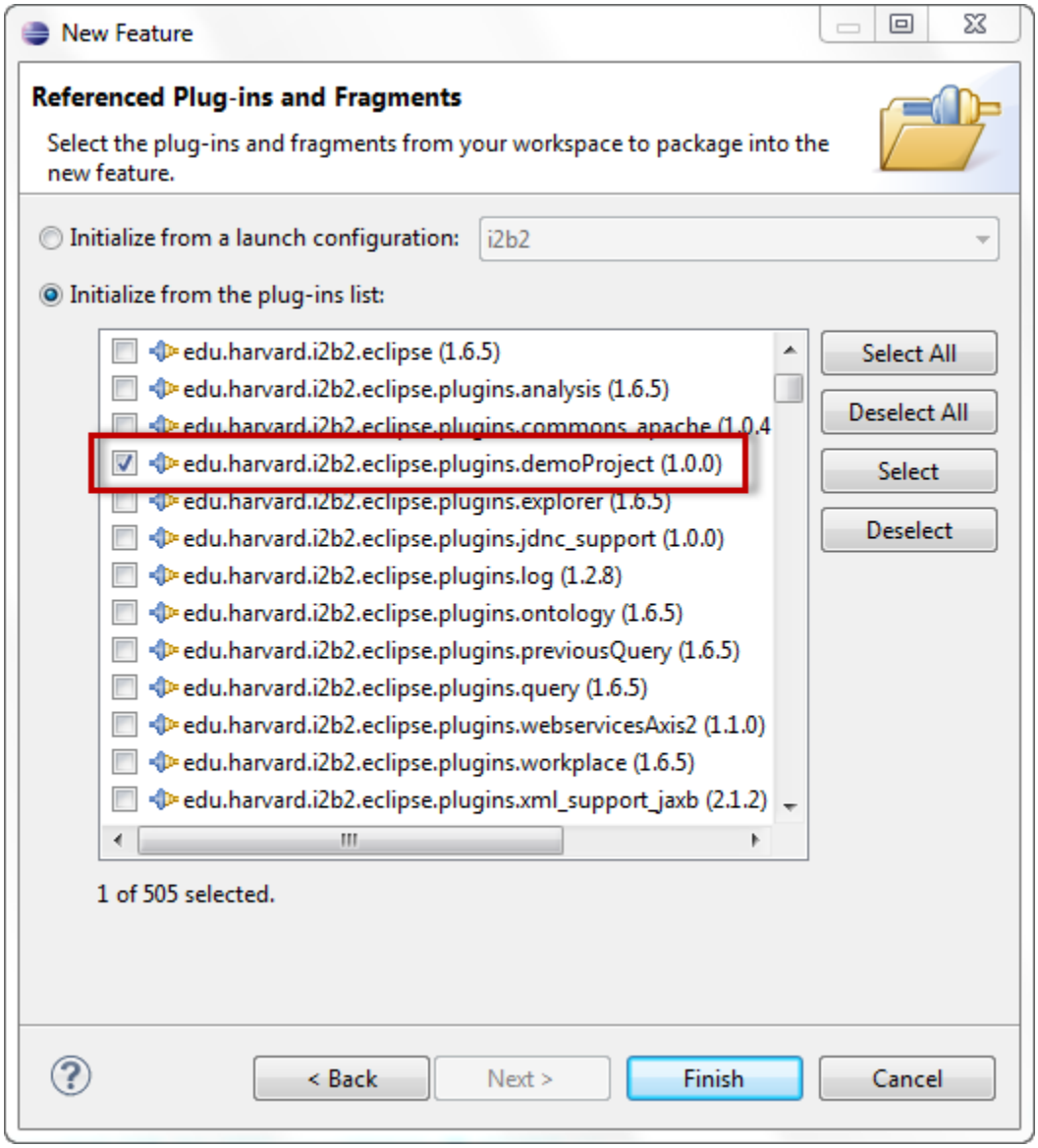
10. At the **Project Name** field enter the name of the new feature project.

As you enter the Project Name the Feature ID and Feature Name fields will automatically be populated.

11. Enter the provider of this feature at the **Feature Provider** field.

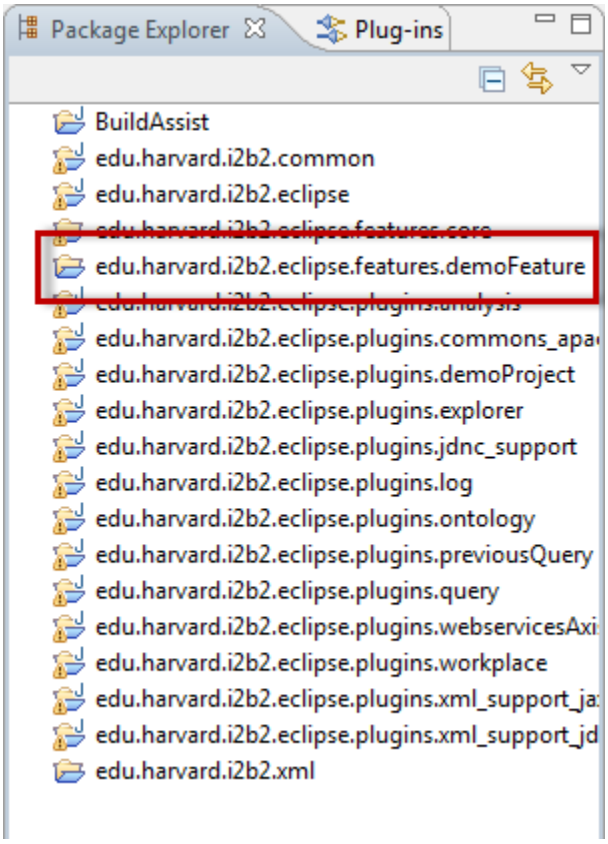


12. Click on the **Next** button.
13. The **Referenced Plug-ins and Fragments** page will open.
14. Select the plug-in(s) you want to include in the feature.



If the custom plug-in you created does not appear in the list try building it again and make sure you refresh and build your workspace.

15. Click on the **Finish** button.
16. The feature project is created and now appears in your workspace.



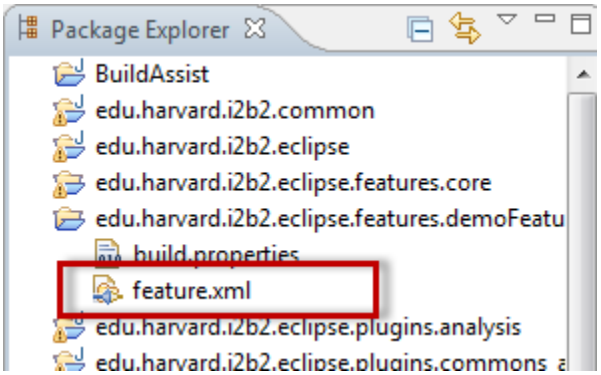
6.2. Feature Manifest (feature.xml)

Eclipse automatically creates a manifest or central file that contains all the information needed for the feature to work. This file is called the **feature.xml** and contains license and copyright information as well as the required plug-ins, dependencies and builds information.

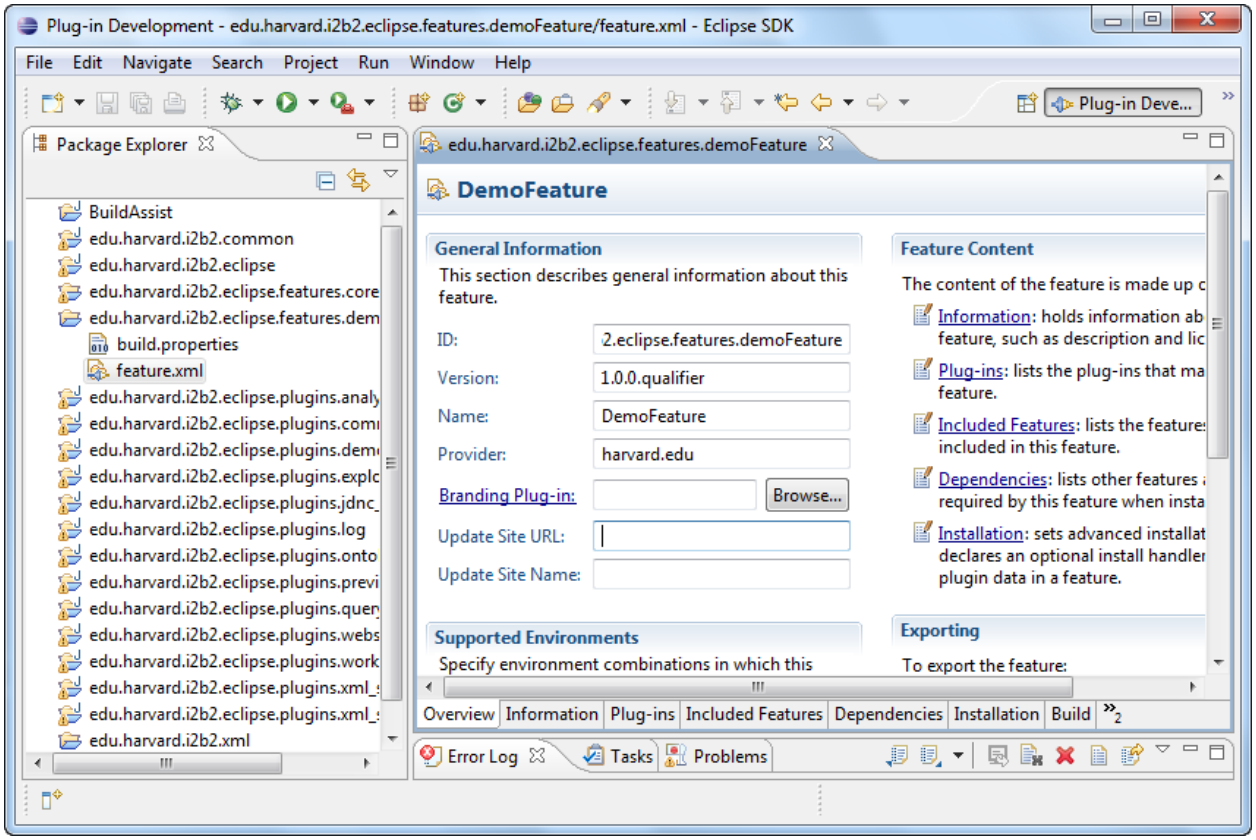
6.2.1. Opening the Feature Manifest Editor

The following outlines how to open the **Feature Manifest Editor**. In this example you will be using the feature you created in the earlier section called; *Creating a New "Feature" Plug-in*.

1. Open your feature's folder by double clicking on its name.
2. Double click on the **feature.xml** file.

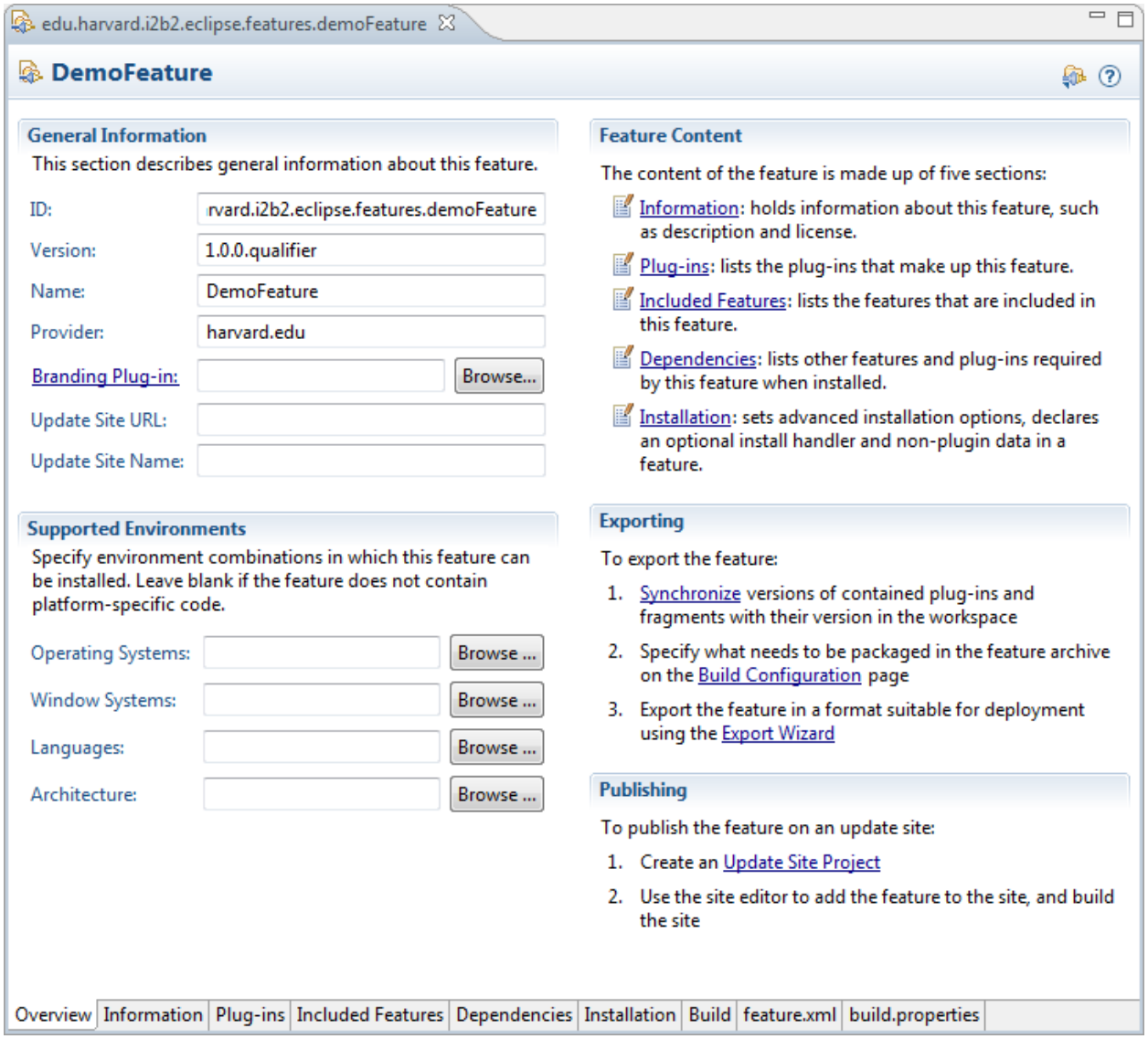


3. The **Feature Manifest Editor** will open to the **Overview** page.



6.2.2. Overview Tab

The **Overview** page contains basic information about the feature.

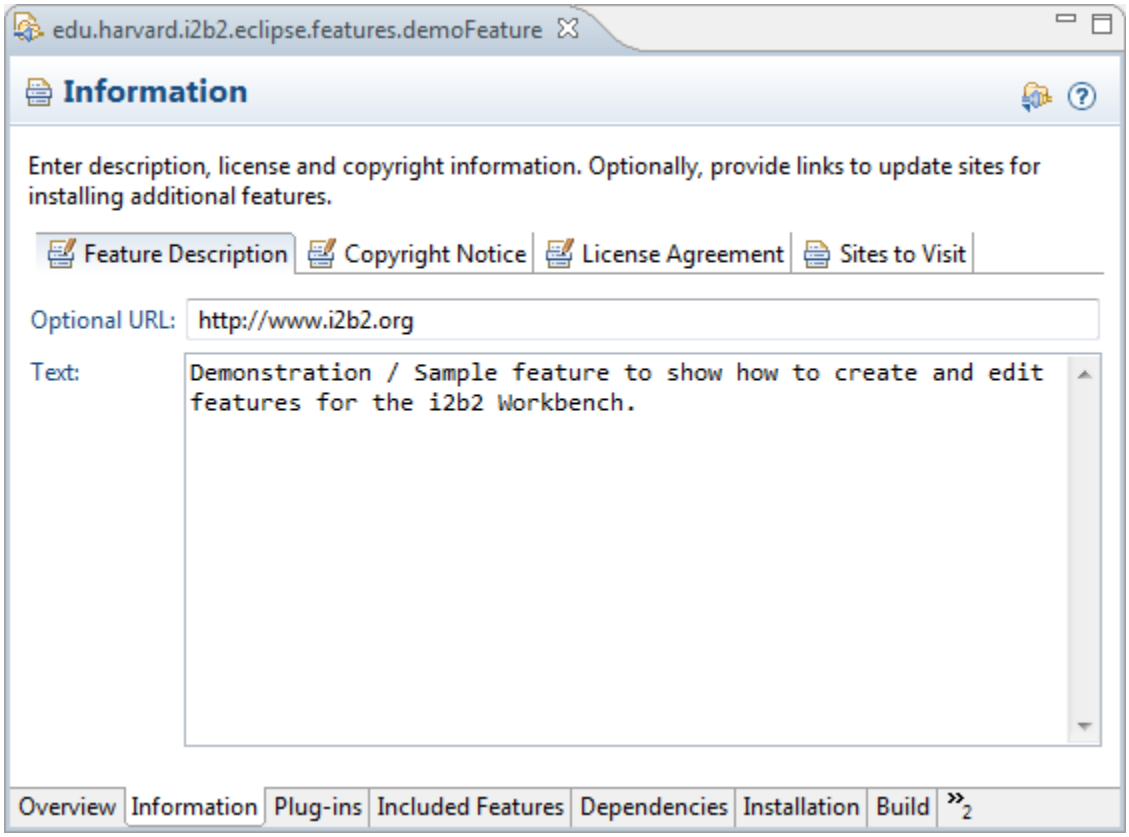


6.2.3. Information Tab

The **Information** page contains several pages for entering a *description*, *copyright notice*, *license agreement* and *sites to visit*. All of this information is visible to the user when they install the feature from within the i2b2 Workbench.

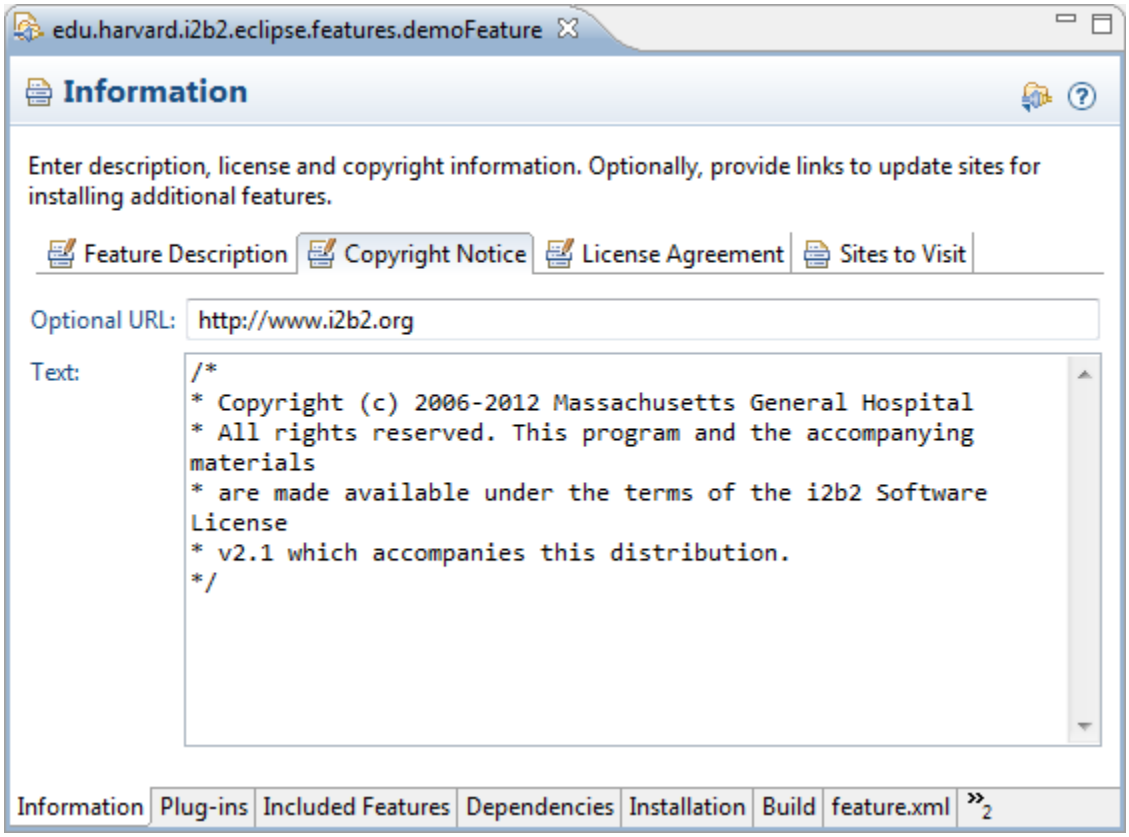
6.2.3.1. FEATURE DESCRIPTION

The first page to appear when selecting the *Information* page is the **Feature Description**. This page allows you to enter a brief description that users will see when installing this feature. You can also enter a URL address of where the user can go to see additional information.



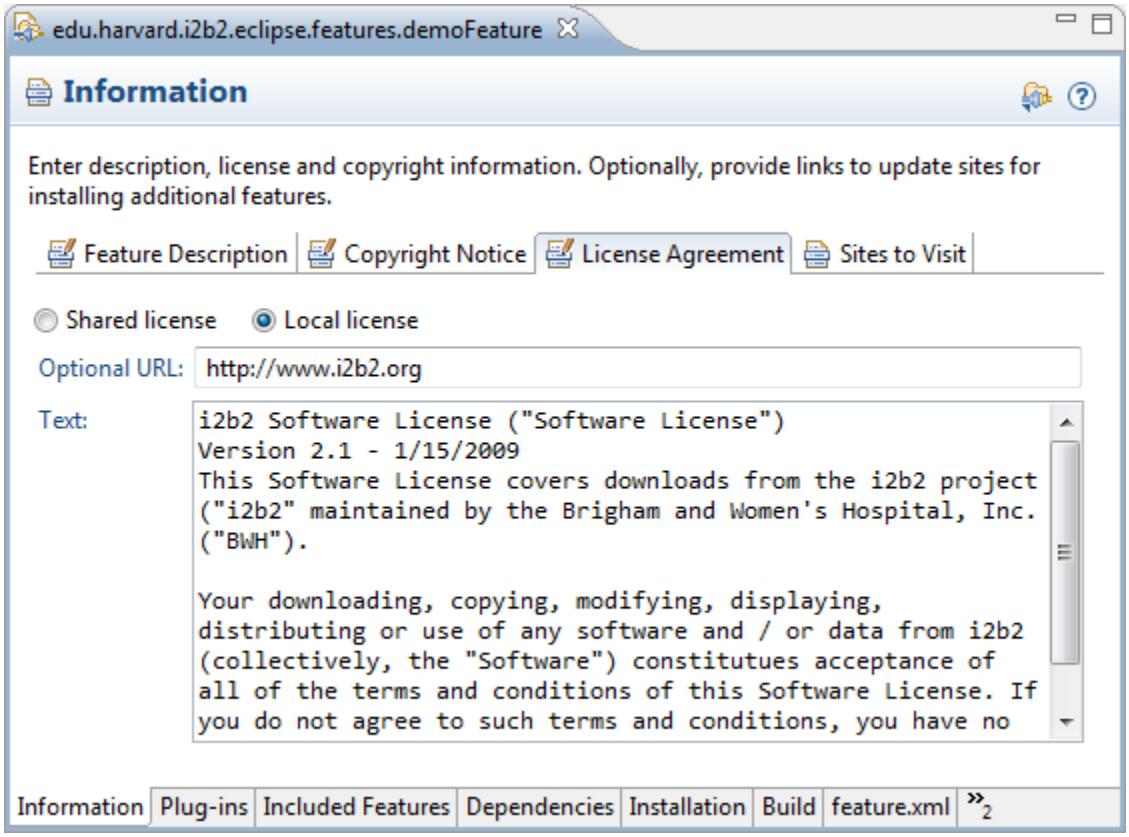
6.2.3.2. COPYRIGHT NOTICE

Clicking on the tab for the **Copyright Notice** page will allow you to enter any copyright information that is required for this feature.

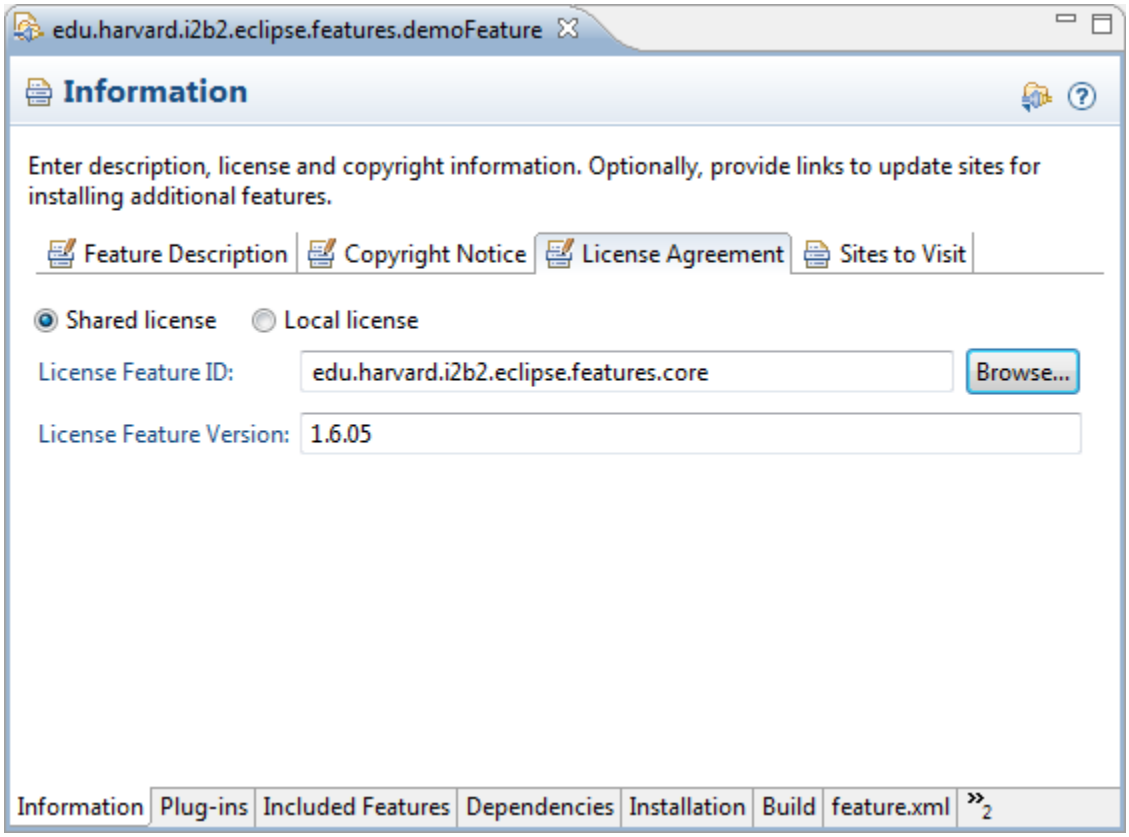


6.2.3.3. LICENSE AGREEMENT

The information you enter on the **License Agreement** page will appear during the installation of the feature in the i2b2 Workbench. Users will have to accept the license in order to continue with the installation process. If they do not want to accept the license then they will not be able to install the feature.

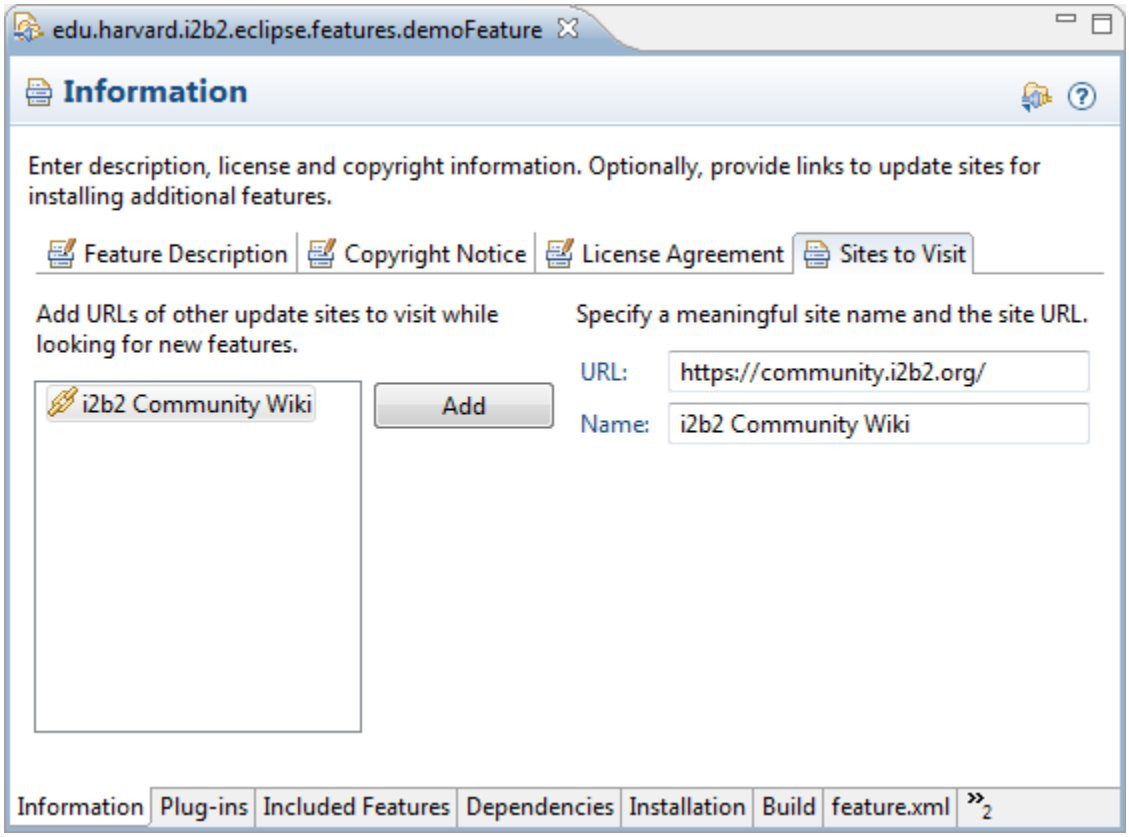


Depending on your version of Eclipse you may have the option to "Share" your license with all the plug-in features that you develop.



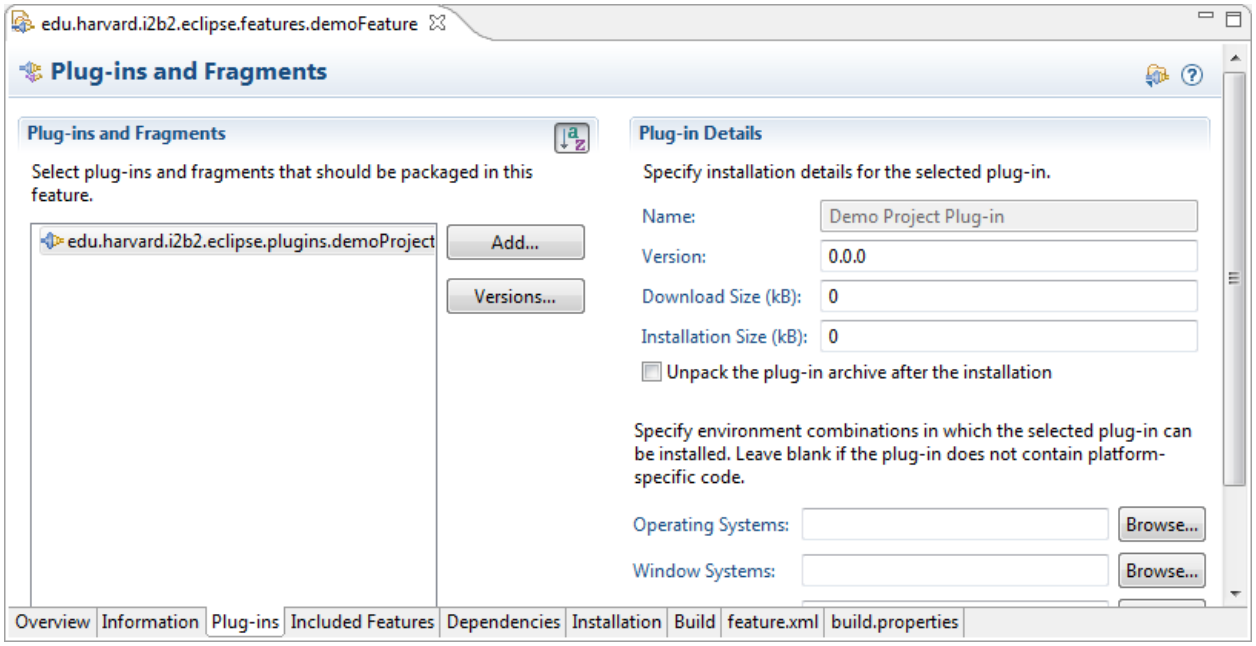
6.2.3.4. SITES TO VISIT

The **Sites to Visit** page is an optional page that you can use to provide your users with a list of additional update sites.



6.2.4. Plug-ins Tab

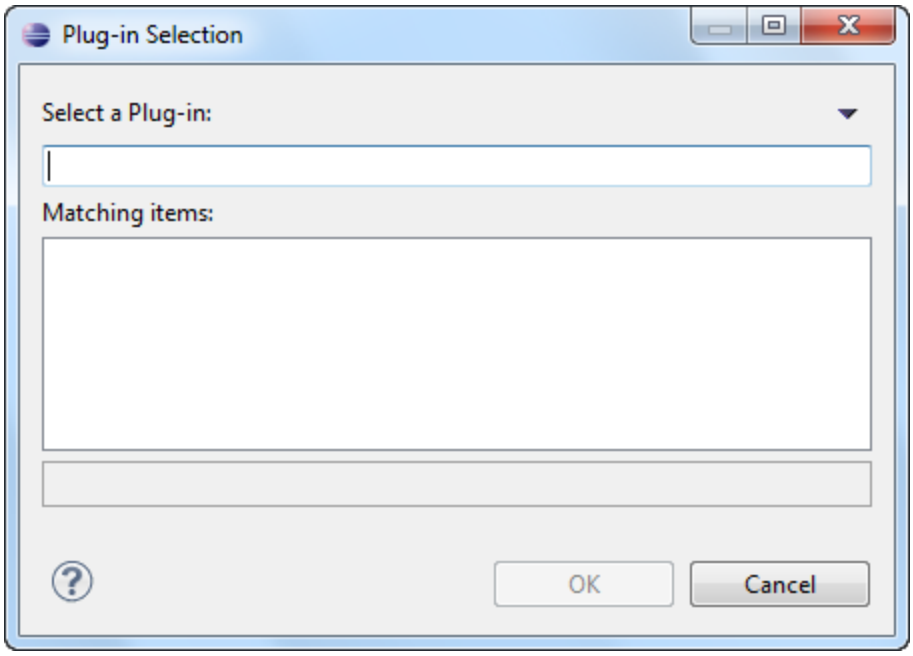
The plug-ins and fragments that are included in the feature are listed on the **Plug-ins and Fragments** page. The list is populated with those plug-ins selected when the feature was created.



6.2.4.1. ADDING PLUG-INS AND FRAGMENTS

The following explains how to add a plug-in and / or a fragment that will be included in this feature.

1. Click on the **Add** button in the **Plug-ins and Fragments** section.
2. The **Plug-in Selection** dialog box will open.



3. At the **Select a Plug-in** prompt type in the name of the plug-in or fragment you want to add.

As you begin typing the name, a list of available plug-ins and fragments will appear in the Matching Items box.

4. Select the plug-in and / or fragment you want to add.
5. Click on the **OK** button.
6. The plug-in and / or fragment will be added to the list.

6.2.4.2. REMOVING PLUG-INS AND FRAGMENTS

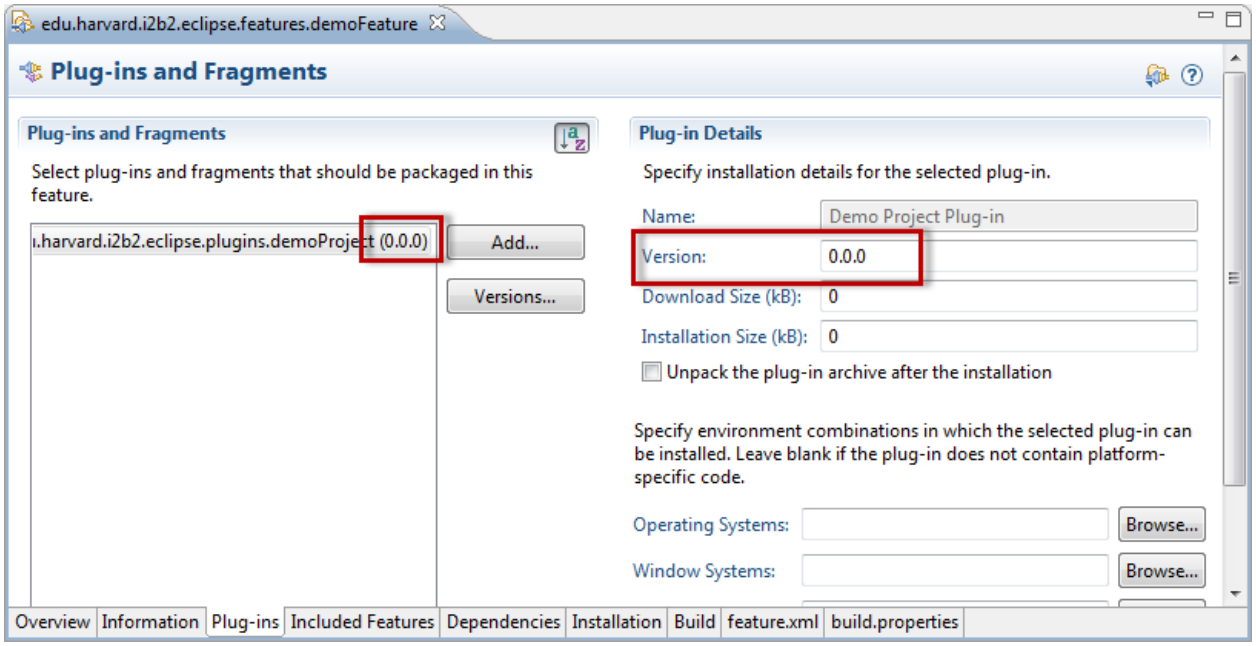
The following explains how to remove a plug-in and / or a fragment from the list of items to be included. Once it is removed it will no longer be packaged with the feature when it is built.

1. Using the *right mouse button*, click on the name of the plug-in or fragment to be deleted.
2. A pop-up menu will display.
3. Select **Delete** from the pop-up menu.

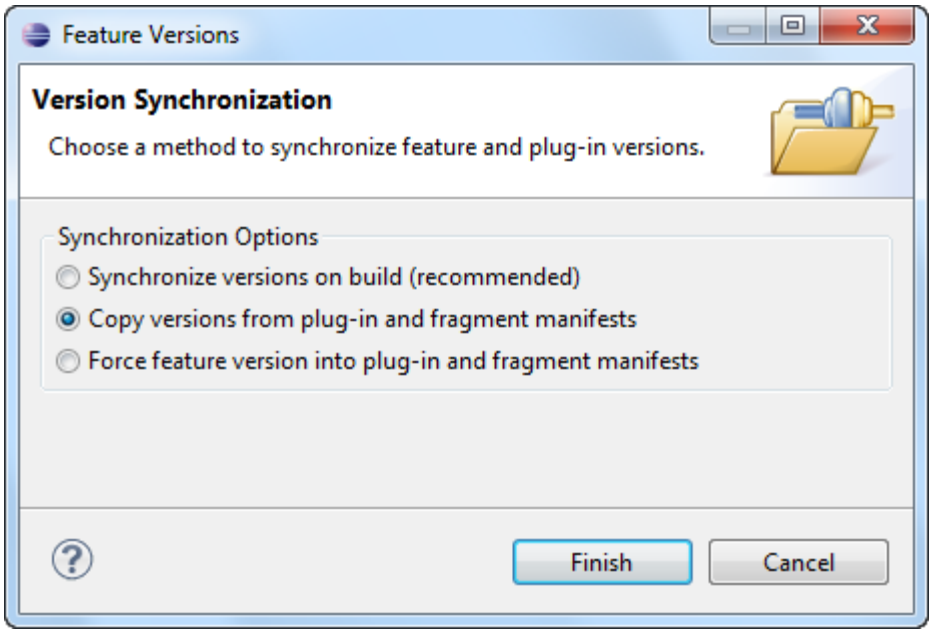
4. The item will be removed from your list.

6.2.4.3. UPDATE THE VERSION OF PLUG-INS AND FRAGMENTS

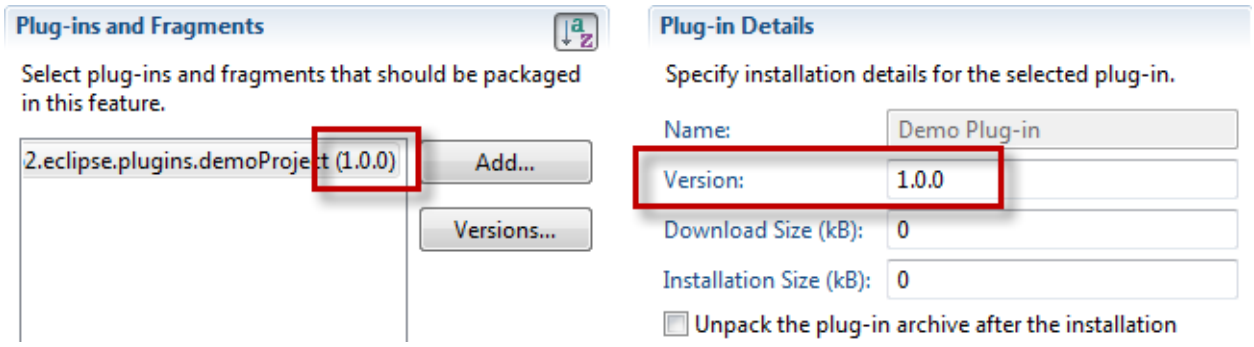
When a feature is first created the initial version of the plug-ins / fragments in the feature is set to 0.0.0. Therefore, it is important that you update them to reflect the correct version information.



1. Select all the plug-ins / fragments in the list.
2. Click on the **Versions** button.
3. The **Features Version** dialog box will open.



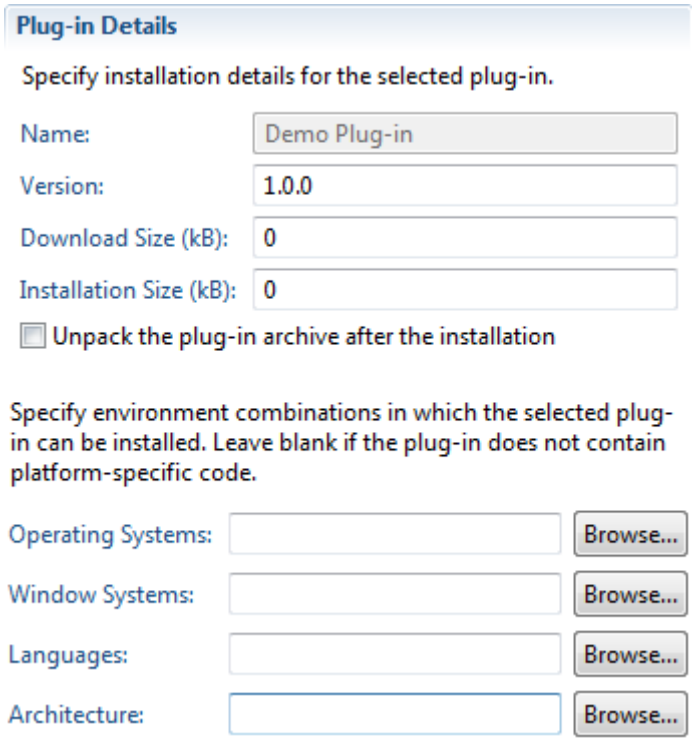
4. Select **Copy versions from plug-in and fragment manifests**.
5. Click on the **Finish** button.
6. The plug-ins / fragments will be updated with the version information in each of their respective manifests.



6.2.4.4. UPDATE PLUG-IN DETAILS

The **Plug-in Details** section is located in the top right corner of the *Plug-ins page*. The information that appears is dependent upon which plug-in is highlighted in the list on the left of the page. If you only have one plug-in or fragment then it will automatically display the information for it.

The *name* and *version* information will default from the plug-in. You can add *download* and *installation* sizes as well as any *environment information* that is specific to the plug-in. All of this information is visible to the user when they are installing the feature from within the i2b2 Workbench.

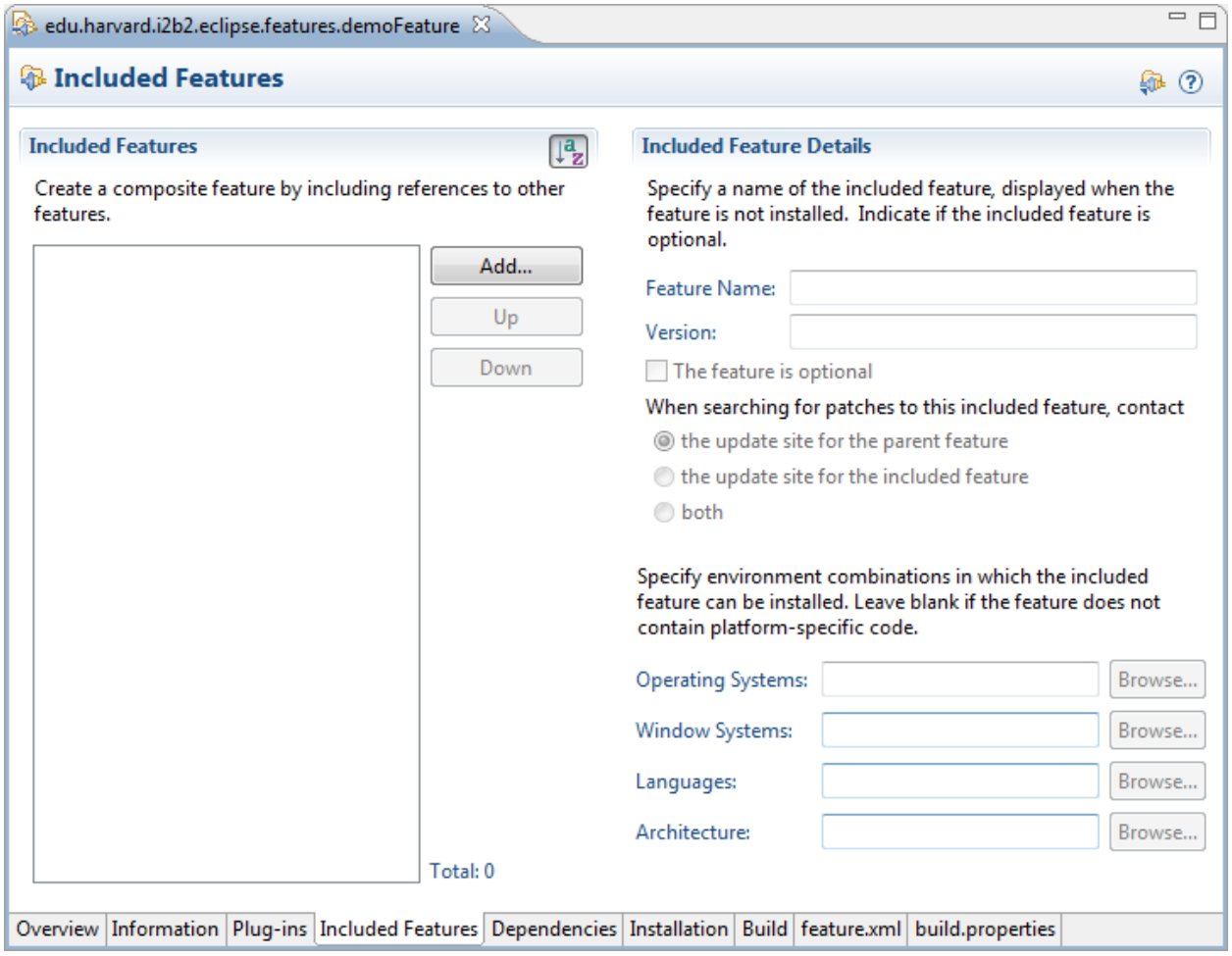


It is important to identify any environment constraints so the user is aware of them before they install the feature on their current platform.

6.2.5. Included Features Tab

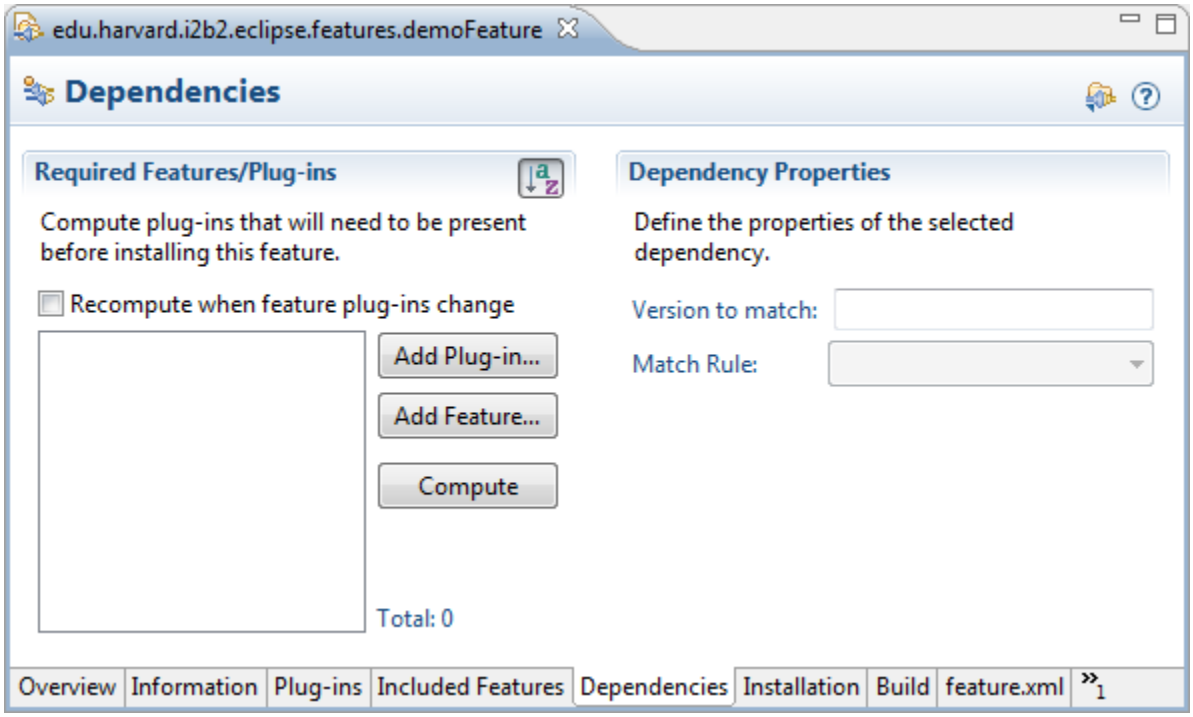
A list of sub-features to be included with the feature is defined on the **Included Features** page.

"Sub-feature" is a feature nested inside a feature.



6.2.6. Dependencies Tab

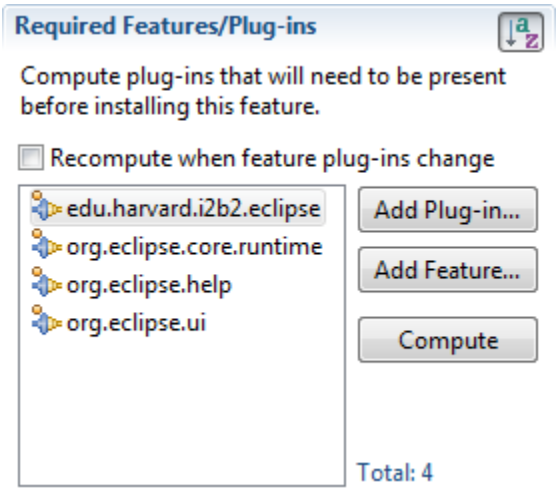
All the plug-ins and other features that are required for the feature to work are listed on the **Dependencies** page. If an item on the list is missing then the feature will not load when the user tries to install it via the i2b2 Workbench.



The dependencies page is not populated when you first create the feature. Therefore, it is important that you add them before building and deploying it.

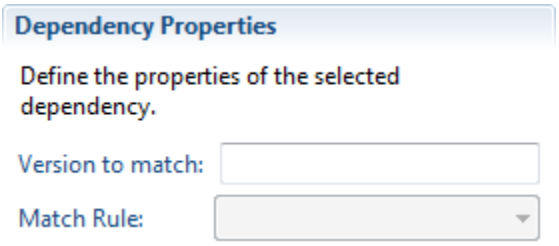
There are two ways to add dependencies:

1. Manually — You select which plug-ins or features to include by clicking on the **Add Plug-in** or **Add Feature** buttons.
2. Compute — This method will populate the list of dependencies based on the requirements defined by the plug-ins listed on the *Plug-ins* page. Simply click on the **Computer** button to update the list.



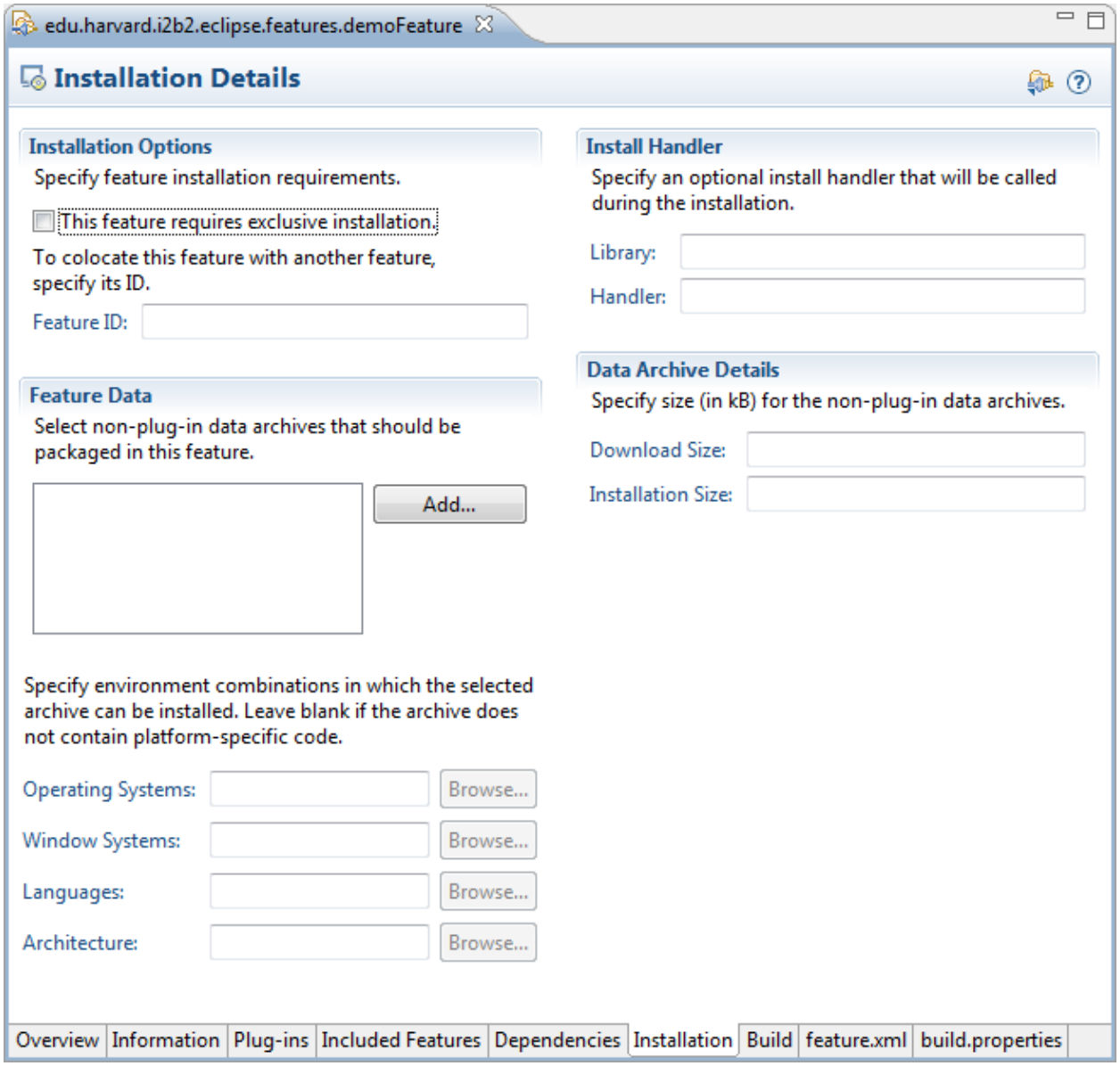
6.2.6.1. DEPENDENCY PROPERTIES

On the left side of the *Dependencies* page is a section called **Dependency Properties**. It is here that you can define a **Version to match** and **Match Rule**. The feature will not load if the version and the defined match criteria are not met.



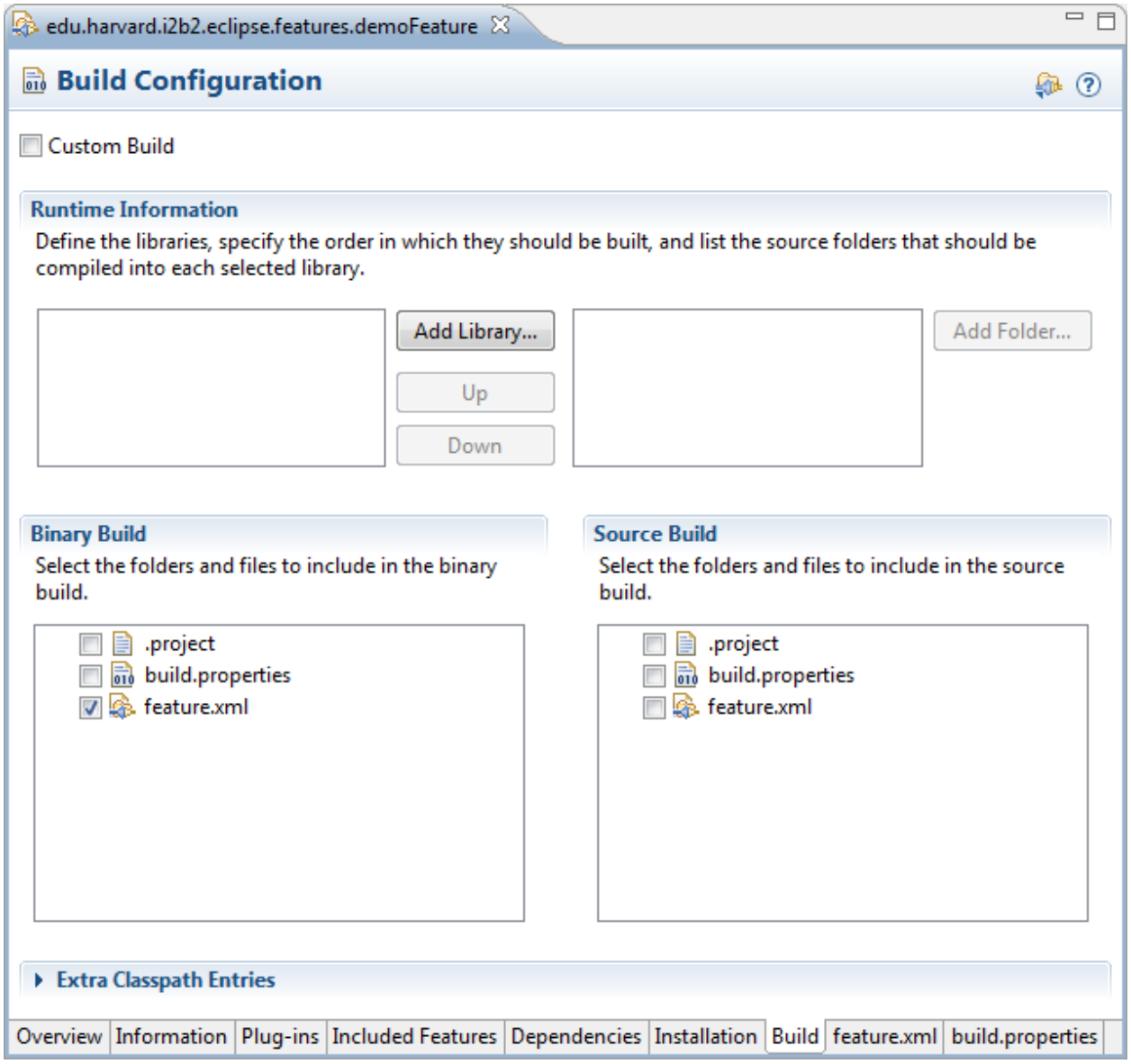
6.2.7. Installation Tab

Information regarding the installation of the feature is entered on the **Installation** page. This information is used by the *Update Manger* when a user installs the feature from within the i2b2 Workbench.



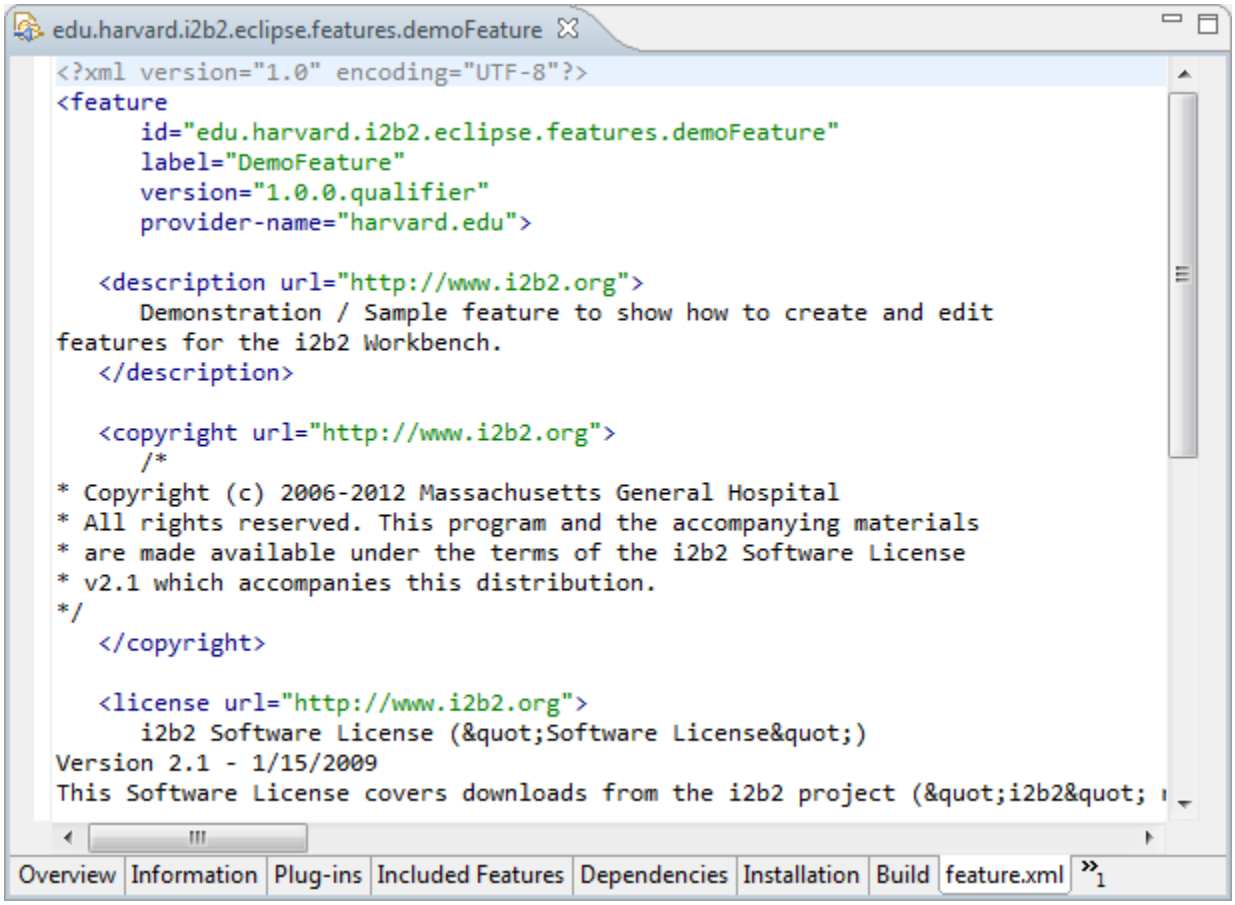
6.2.8. Build Tab

Information regarding the *libraries*, *folders* and *files* to include when building the feature are defined on the **Build Configuration** page.



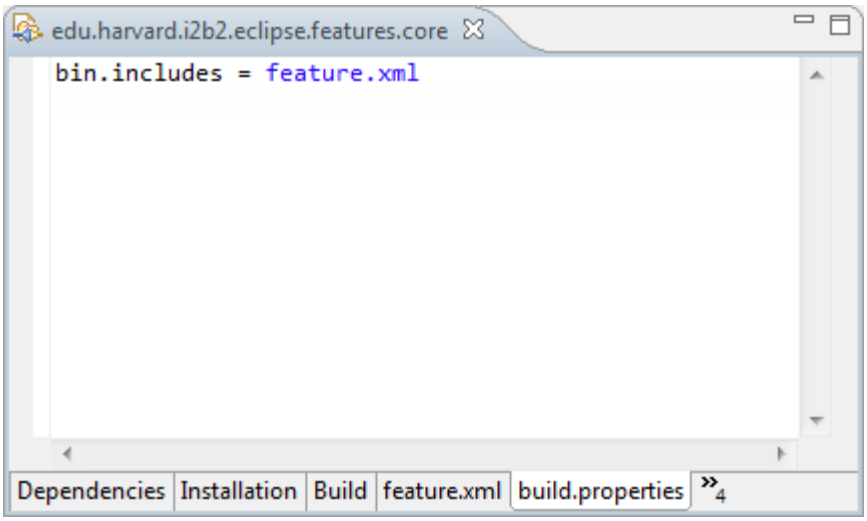
6.2.9. Feature.xml Tab

The **feature.xml** page is the XML for all the information defined on the other pages.



6.2.10. Build.properties Tab

The **build.properties** page is the properties file for the feature. The information that appears on this page correlates with what is defined on the **Build Configuration** page.


edu.harvard.i2b2.eclipse.features.core
bin.includes = feature.xml
Dependencies
Installation
Build
feature.xml
build.properties
»4

LICENSE

The i2b2 source code is licensed under the i2b2 Software License 2.1. This includes but is not limited to all code in the edu.harvard.i2b2.* package namespace.