

SDK Quickstart

Set Eclipse for uDig Plug-in Development



Table of Contents

1 Introduction.....	3
2 Downloads.....	4
3 Eclipse SDK Installation.....	5
4 Eclipse Workspace.....	7
5 Eclipse Preferences.....	8
6 uDIG SDK.....	9
7 Running uDig.....	12
8 What to Do Next.....	15

I Introduction

This workbook is aimed at those doing plug-in development against the uDig platform. Follow these instructions to quickly set up a development environment for working on your own plug-ins.

If you would like to work on uDig itself please visit the project [Policies and Procedures](#).

Eclipse is familiar to most developers as a Java Integrated Development Environment (IDE). The Eclipse IDE can be extended with additional “capabilities” to work with alternate programming languages (like C++ or Ruby), or additional subject matter such as Java Enterprise Edition or in this case Eclipse Plug-in development.

In this Quickstart we are going to use the the Eclipse Plug-in Development capability; with the uDIG SDK as the target platform. This workbook covers setting up a development environment for working on your own plug-ins.

If you have an existing Eclipse installation please do not skip this tutorial – we are going to very carefully set up a copy of Eclipse with a few more additional tools then you are perhaps used to.

The Eclipse developers themselves make use of the Plug-in Development Environment (PDE) day in and day out – so it has gotten a lot of polish over the years. In some respects it is more polished then the Java development environment (with custom editors for all kinds of little files).

Some of the terminology used when working with the PDE:

- Everything is a **Plug-in**
- A plug-in can implement an **extension**
- A plug-in can provide an **extension-point** for others

Please keep these ideas in mind – even an “Application” is considered an extension when working with the Eclipse Platform.

2 Downloads

We are going to start by downloading all the software we need; we will be able to proceed with installation as we wait for some of the larger downloads.


If you are using this work book in a lab setting you will find these downloads available on your course DVD.

1. For uDig 1.2 we are going to download the latest SDK from here:
<http://udig.refractions.net/download/>

Normally you would grab the latest stable SDK from the public uDig download page.

At the time of writing the latest uDig SDK was:
udig-1.2.2-sdk.zip

2. Visit <http://www.eclipse.org/downloads> and click on the link for “Eclipse Modeling Tools” and download the appropriate



Eclipse Modeling Tools (includes Incubating components) (367 MB)
This modeling package contains a collection of Eclipse Modeling Project components, including EMF, GMF, MDT XSD/OCL/UML2, M2M, M2T, and EMFT elements. It includes a complete SDK, developer tools and source code. Note that the Modeling package includes some **incubating** components, as indicated by feature numbers less than 1.0.0 on the feature list. [More...](#)
Downloads: 1,264

Windows **32bit**
Mac Carbon **32bit**
Mac Cocoa **32bit 64bit**
Linux **32bit 64bit**

Tested with **Eclipse 3.7.0 Indigo Packages:**
[eclipse-modeling-indigo-win32.zip](#)

3. We have prepared an “dropins” download in the following folder:
<http://udig.refractions.net/files/downloads/extras/>

This download includes a developers guide for udig, platform language packs, and a resource bundle editor.

At the time of writing: **dropins-3.7.0.zip**

GDAL support is provided by the imageio-ext project started by GeoTools alumni.

4. Download a Java Runtime Environment from this folder:
<http://udig.refractions.net/files/downloads/jre/>

This is a special JRE that has been extended with Java Advanced Imaging and Image IO and GDAL for spatial image formats.

At the time of writing: **jre1.6.0_25.win32_gdal_ecw.zip**

Please ensure your eclipse and jre match (both win32 – or both win64).

5. Download Eclipse RCP Delta Pack

Download the RCP-Delta Pack from <http://download.eclipse.org/eclipse/downloads/eclipse3x.html> you will need to choose the "Latest Release" build; and the scroll down to the "Delta Pack" link to download.

Eclipse-3.7-delta-pack.zip at the time of writing

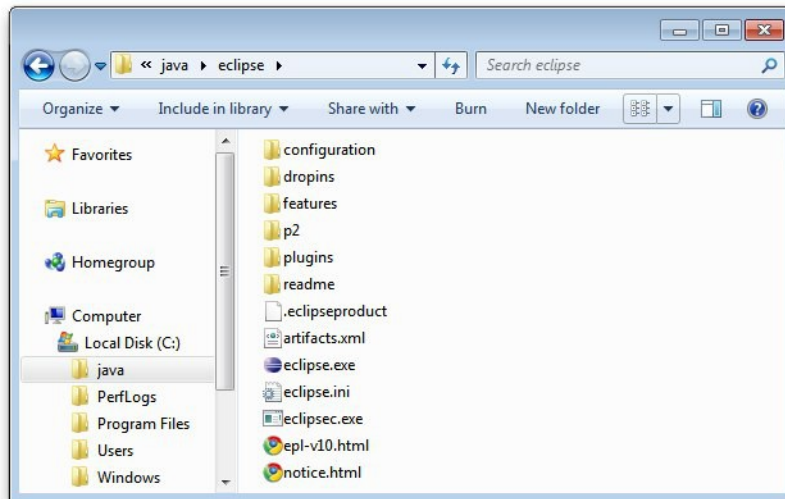
3 Eclipse SDK Installation

Hopefully by now your eclipse download has finished and we can begin to installation.

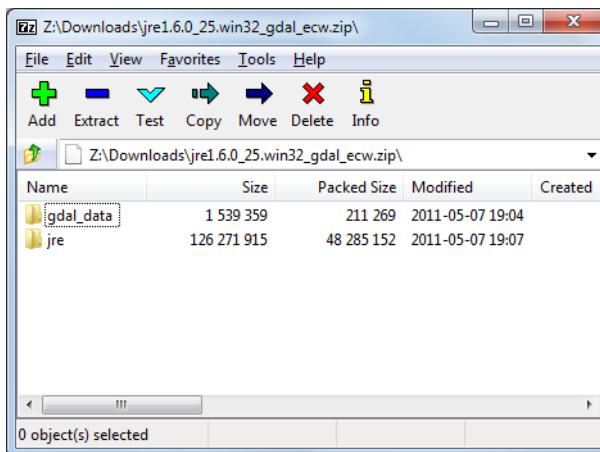
1. Create a folder: [C:\java](#)
2. Unzip the downloaded **eclipse-modeling-indigo-win32.zip** file to your [C:\java](#) directory – the folder **C:\java\eclipse** will be created

WARNING windows support for unzipping does not support long filenames.

If you need a good program to unzip archive files try: www.7-zip.org

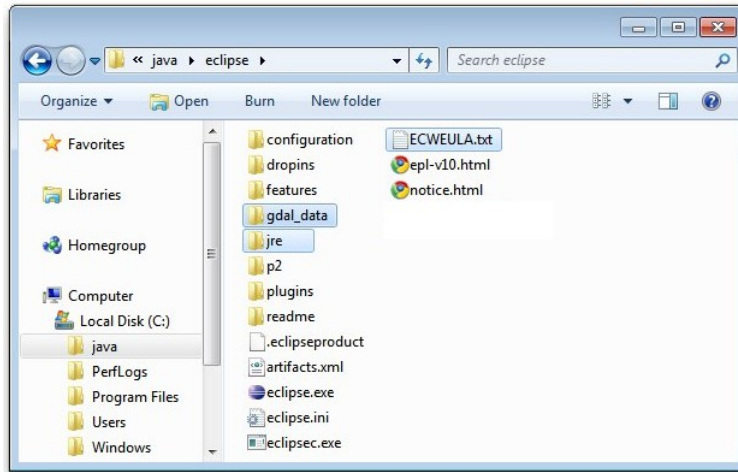


3. Open up the jre zip file Drag from the 7zip window.



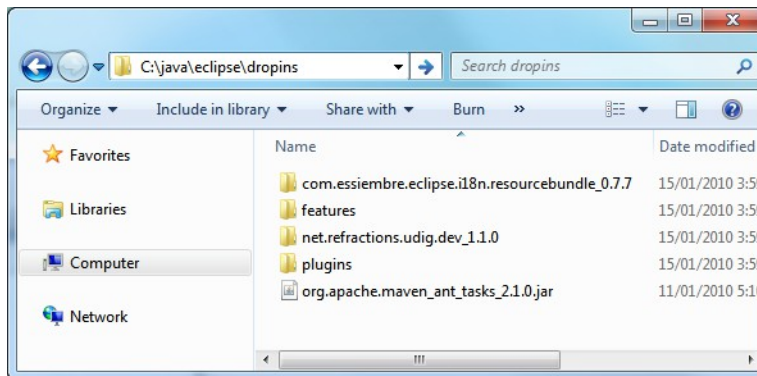
The folder must be called "jre" for the eclipse.exe to recognize it

4. Drop into your C:\java\eclipse



Be careful when extracting this zip not to get something silly like dropins/dropins

5. Unzip the **dropins-3.7.zip** file to your eclipse/dropins folder. The download will add additional plug-ins and features to to your eclipse installation without getting mixed up with the bare-bones eclipse.



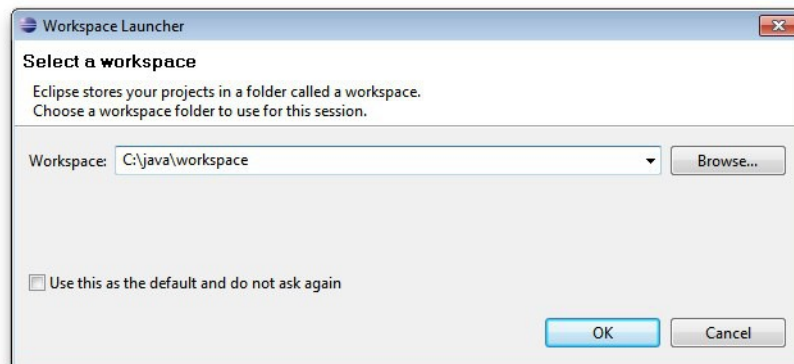
You may want to modify the "eclipse.ini" file to specify additional command line options such as more memory.

6. Navigate to **C:\java\eclipse** and right-click on the **eclipse.exe** file and select **Send To->Desktop (create shortcut)**.

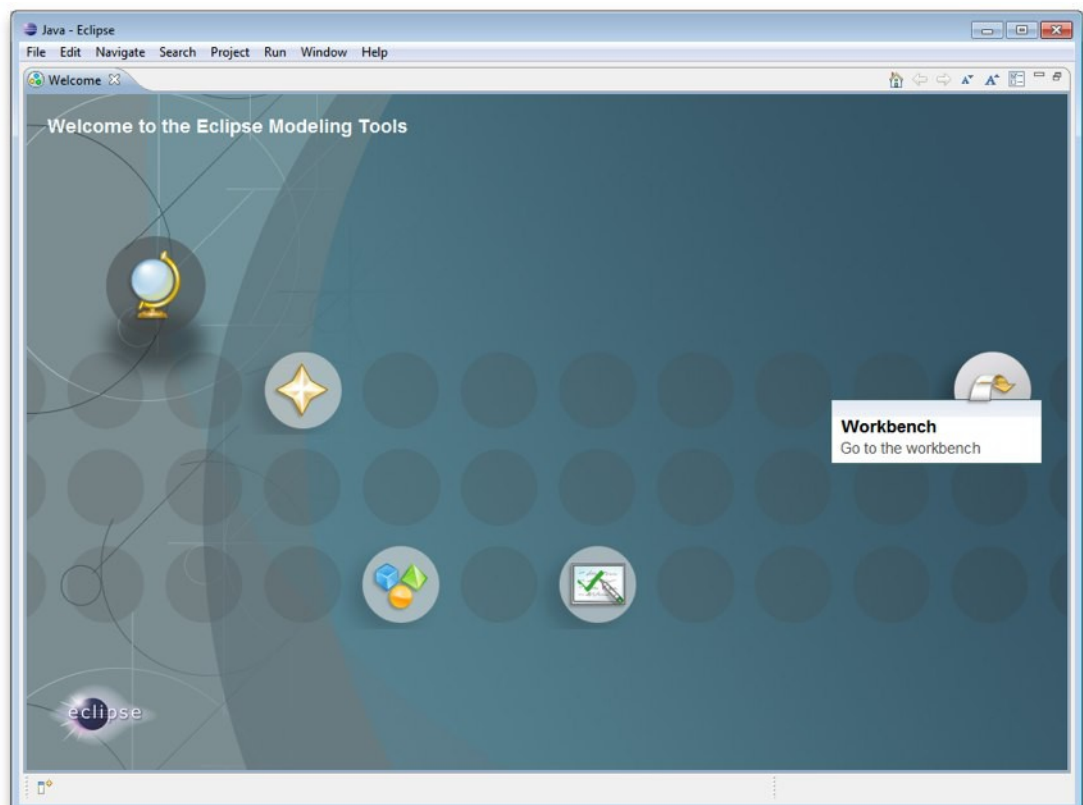
4 Eclipse Workspace

The Eclipse IDE keeps track of what you are doing in a workspace. You can have several workspaces (often for different projects) each with its own configuration.

1. Double click on your desktop short cut to start up eclipse. When you start up eclipse for the first time it prompt you for a workspace.
2. Choose a workspace for your sdk development:



3. Wait a few moments while eclipse starts up.
4. On the Welcome view press the **Workbench** button along the right hand side.



5 Eclipse Preferences

We have a few global settings to configure before we can proceed.

We are waiting for a Mac OSX JRE to be available before using Java 7.

1. Open up **Window > Preferences** from the menu bar.
2. Navigate to the **Java > Compiler** page and change:
 - Compiler compliance level: 1.6
3. Check the **Java > Installed JREs** page:

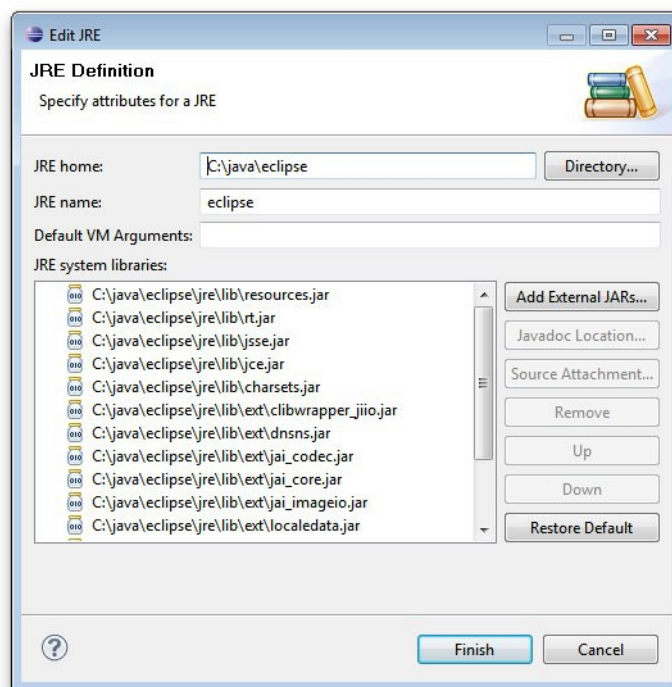
- Should have Location: C:\java\eclipse

If the setting is not correct you can use the **Add..** button and create a JRE entry for C:\java\eclipse

- You can press **Edit** to look at the installed JRE.

Regardless of platform we are interested in making sure **jai_core.jar**, **jai_imageio.jar** and **jai_codec.jar** are available.

On OSX we are not able to provide a JRE for you to download – uDig will install JAI and ImageIO when run for the first time.

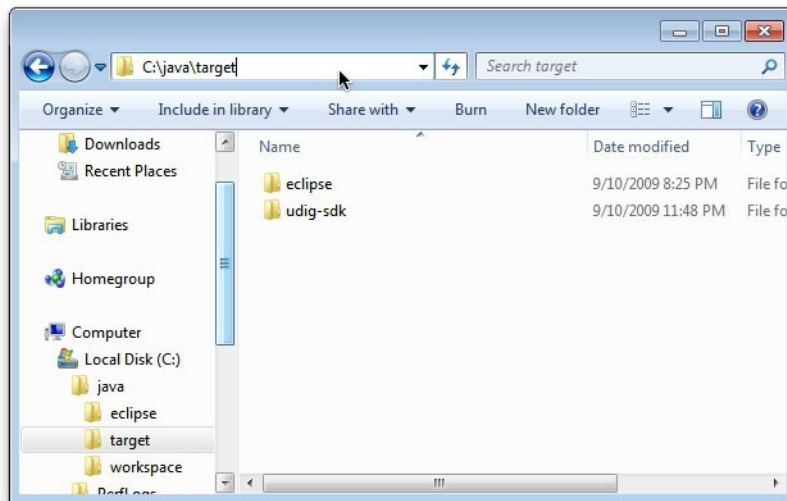


- As a result of these changes, Eclipse may prompt you to perform a clean build. Simply accept this request; it won't take long since we don't yet have any source code.

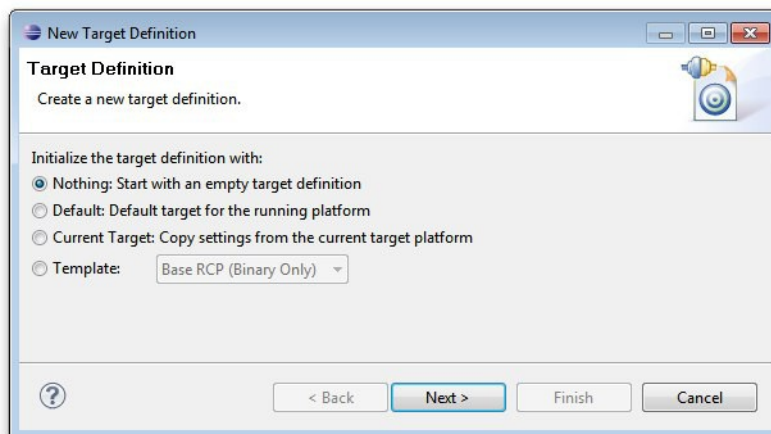
6 uDIG SDK

We are now going to unpack the udig sdk and use it as our plug-in target platform.

1. Extract the udig sdk download into C:\java\target
2. Extract the eclipse-delta-pack download into C:\java\target
3. You will be left with the following directory structure:

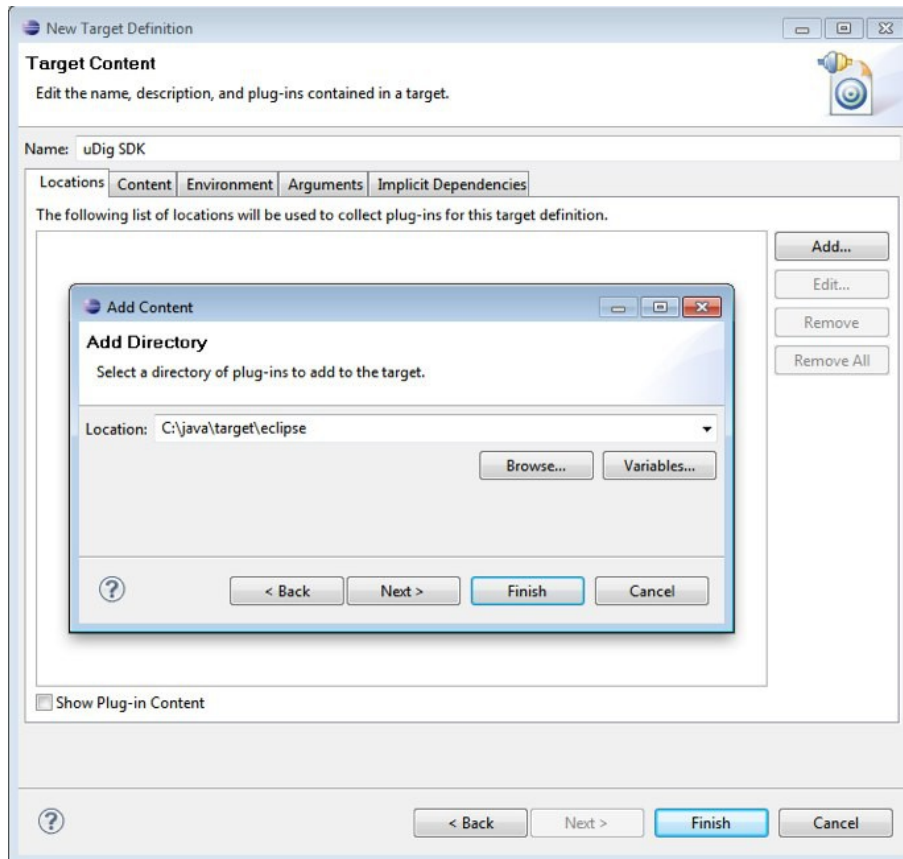


4. Go back to eclipse and open **Window>Preferences**.
5. Select the **Plugin Development > Target Platform** page.
6. Press the **Add** button to start a new target definition
7. Choose **Nothing** as we are going to define a target from scratch

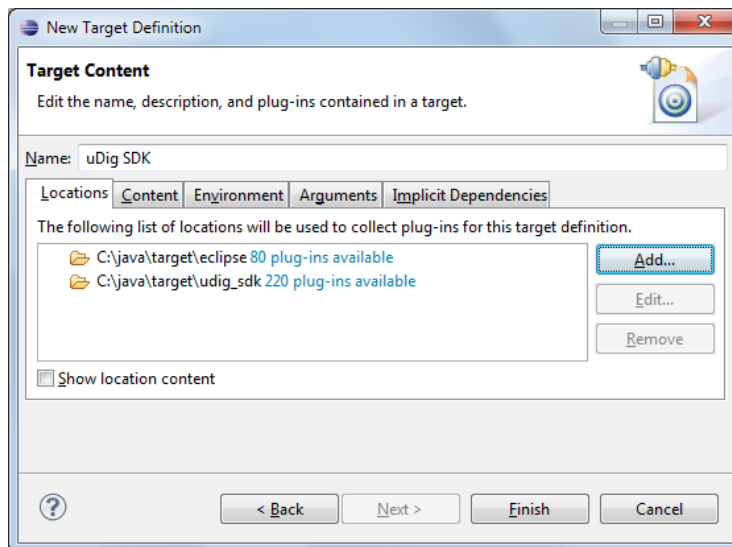


8. Fill in the name “uDig SDK” for this target

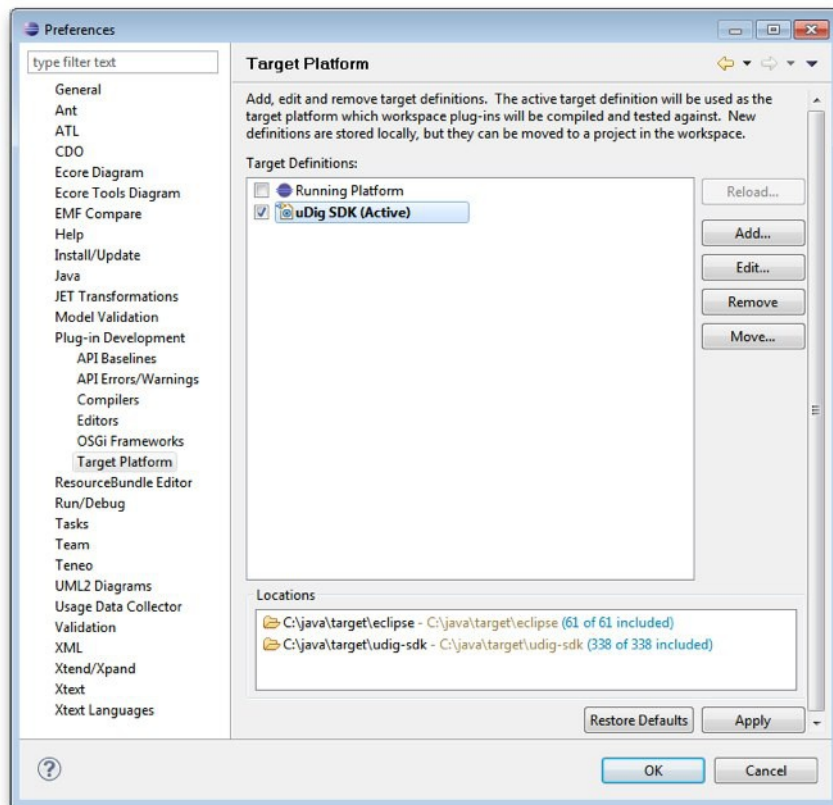
9. Press the **Add** button and add a Directory. Choose the C:\java\target\eclipse directory where you unpacked the eclipse-delta-pack



10. Press **Add** again and add the C:\java\target\udig-sdk directory.



11. Press Finish to complete the “uDig SDK” target platform. Please tick the checkbox next to the “uDig SDK” so it will be the Active target platform.



12. Press **OK**

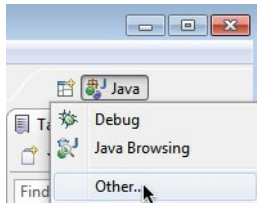
At this point all the source code for the Eclipse and uDig plug-ins are available. We can now start working on uDig plug-ins, but before we do that lets try running the application.

If you get a message indicating the SDK is newer than the version of eclipse you are running – then these instructions are out of date!

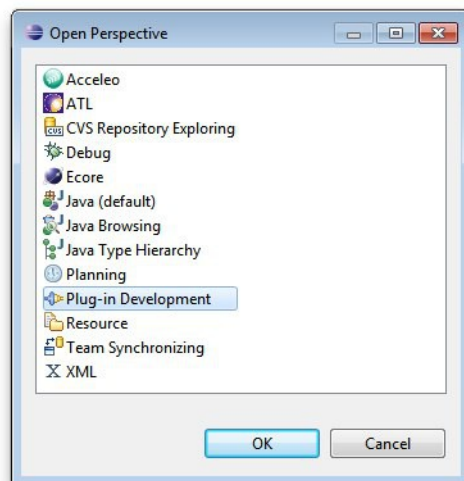
7 Running uDig

With all this in place we can now run the uDig application from your development environment. This is a good way to test that everything is installed correctly.

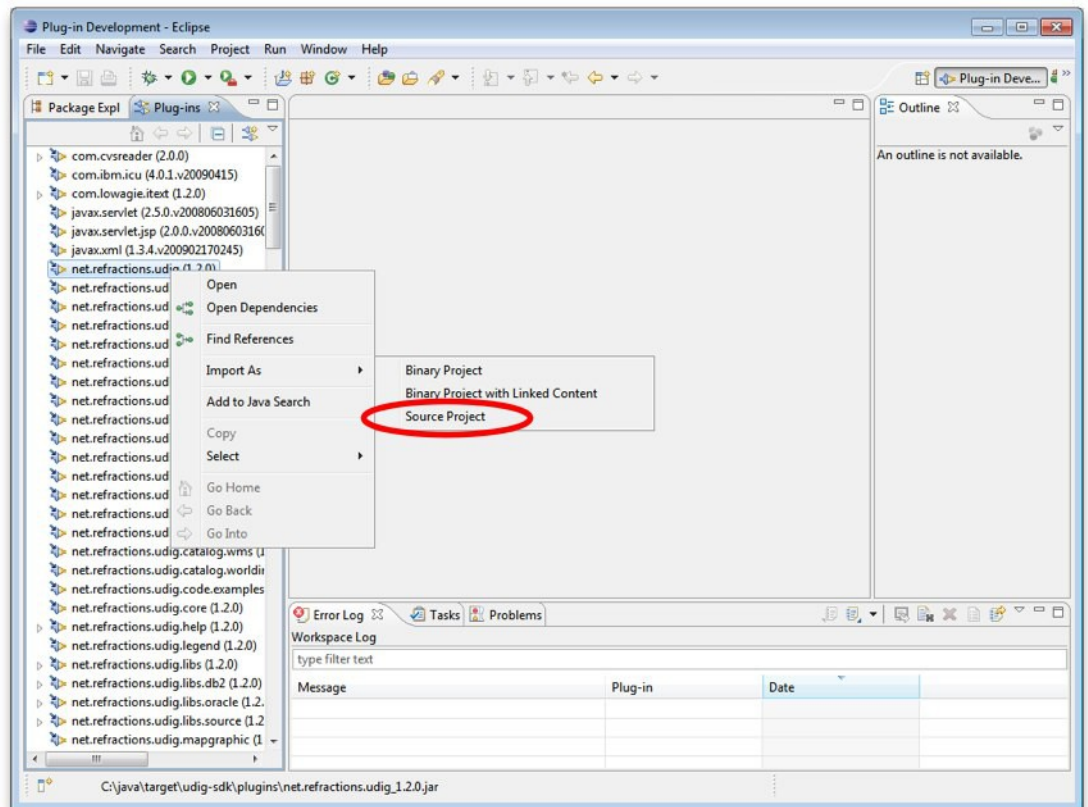
1. To start out with we will switch to the **Plug-in Development** perspective; in the top right of the toolbar you can choose **Other** to open the **Open Perspective** dialog



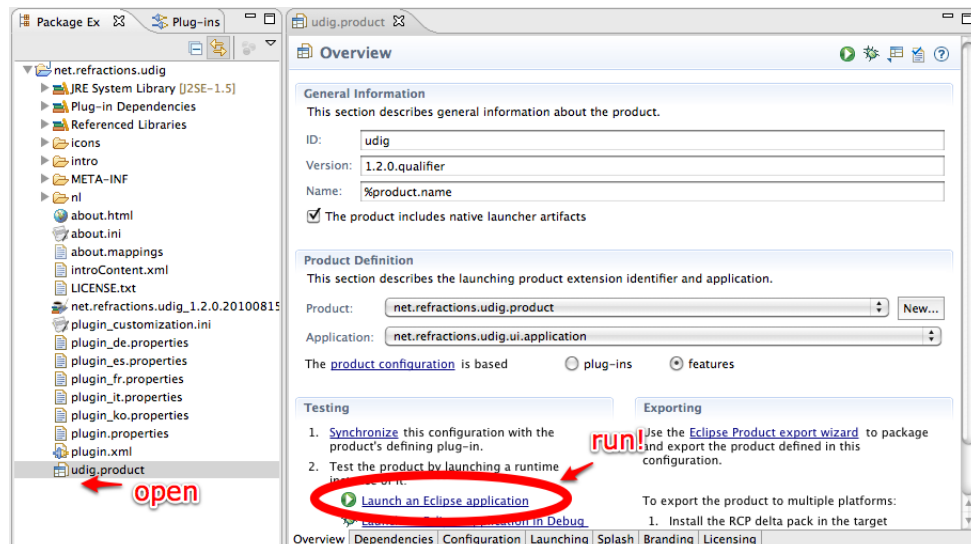
2. Chose **Plugin-Development** from the list and press **OK**



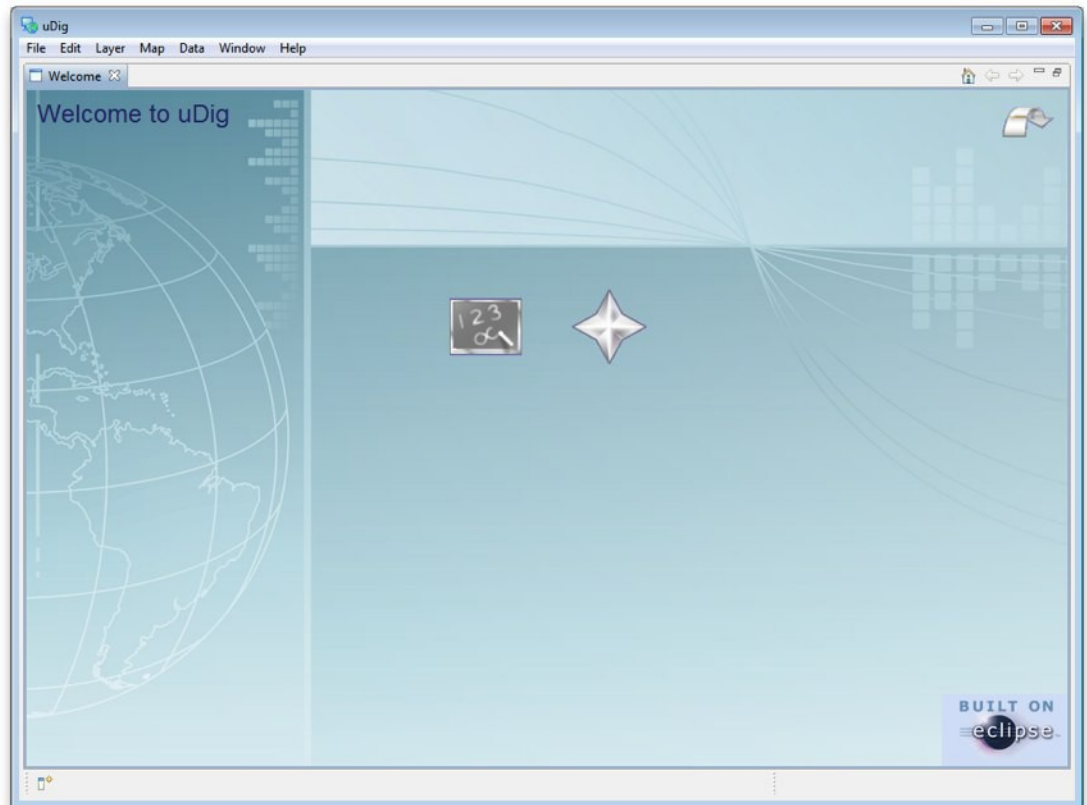
- Click on the **Plugins** view and right click on **net.refractions.udig** plug-in in the list. Select **Import As > Source Project** to copy the plugin into your workspace.



- Change to the **Package Explorer** view and open up the **net.refractions.udig** plugin and double click to open **udig.product**.
- From the overview tab of **udig.product** click on the **Launch Eclipse Application** link (it is located under testing as shown below).



6. The application will now start!

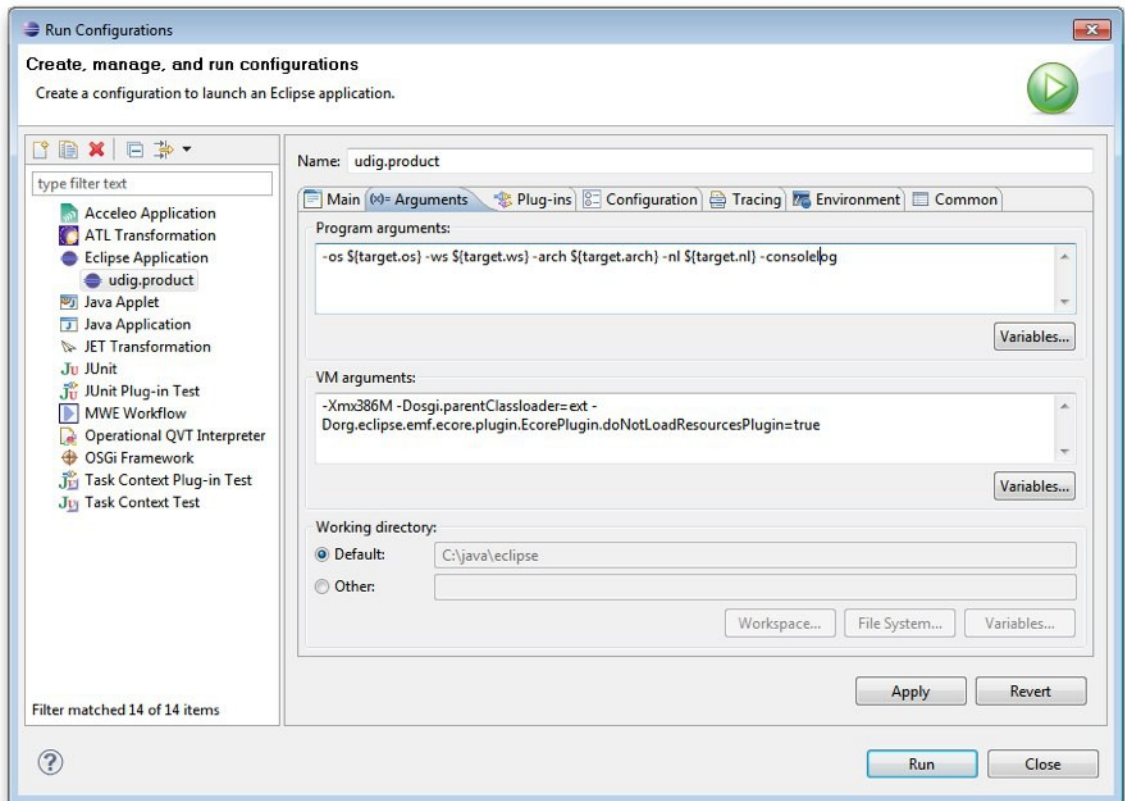


8 What to Do Next

Here are some additional things to try when running uDig.

- From Eclipse open up **Run > Run Configurations** to examine or customize configuration of uDig you are running. Many of these fields were filled in for you by the uDig.product.
- The number one tip is to go to the Arguments tab and enable console logging. To send log information to the console as uDig runs add **-consolelog** to your “program arguments”.

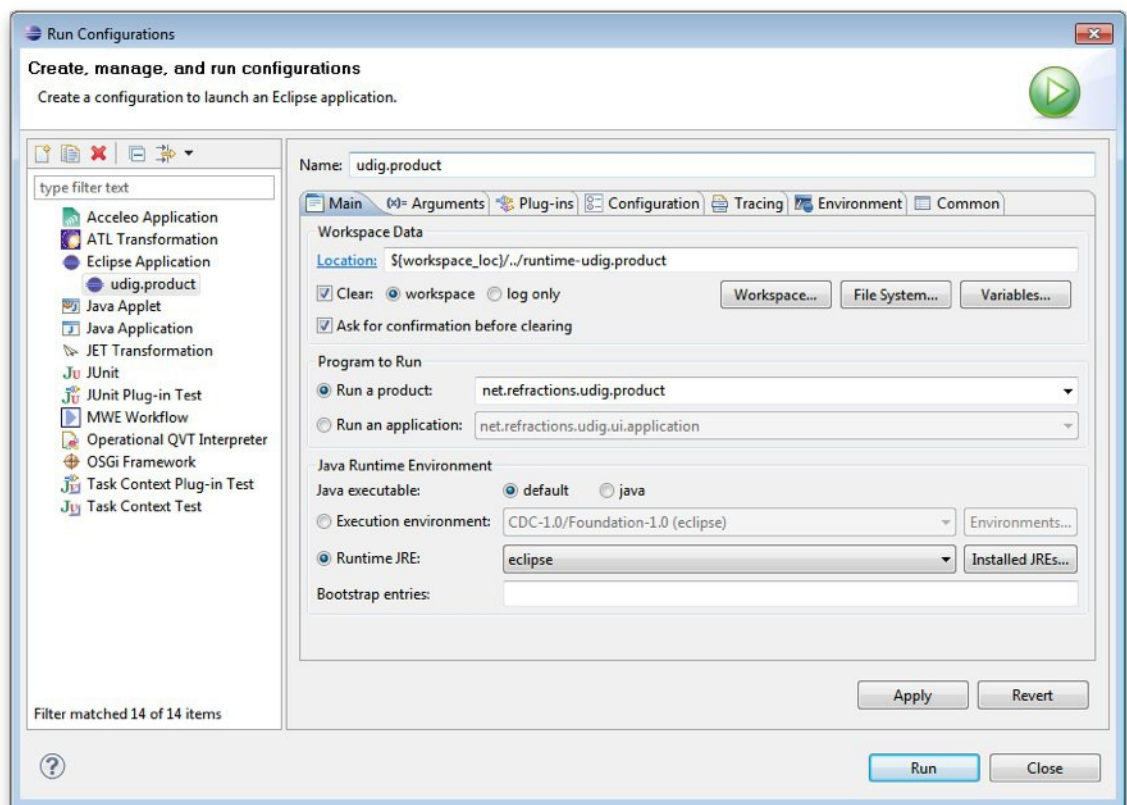
For documentation on these command line parameters check the eclipse help menu.



You can also review the VM arguments; including changing the amount of memory available to your uDig application **-Xmx512m** may be useful when working with large images?

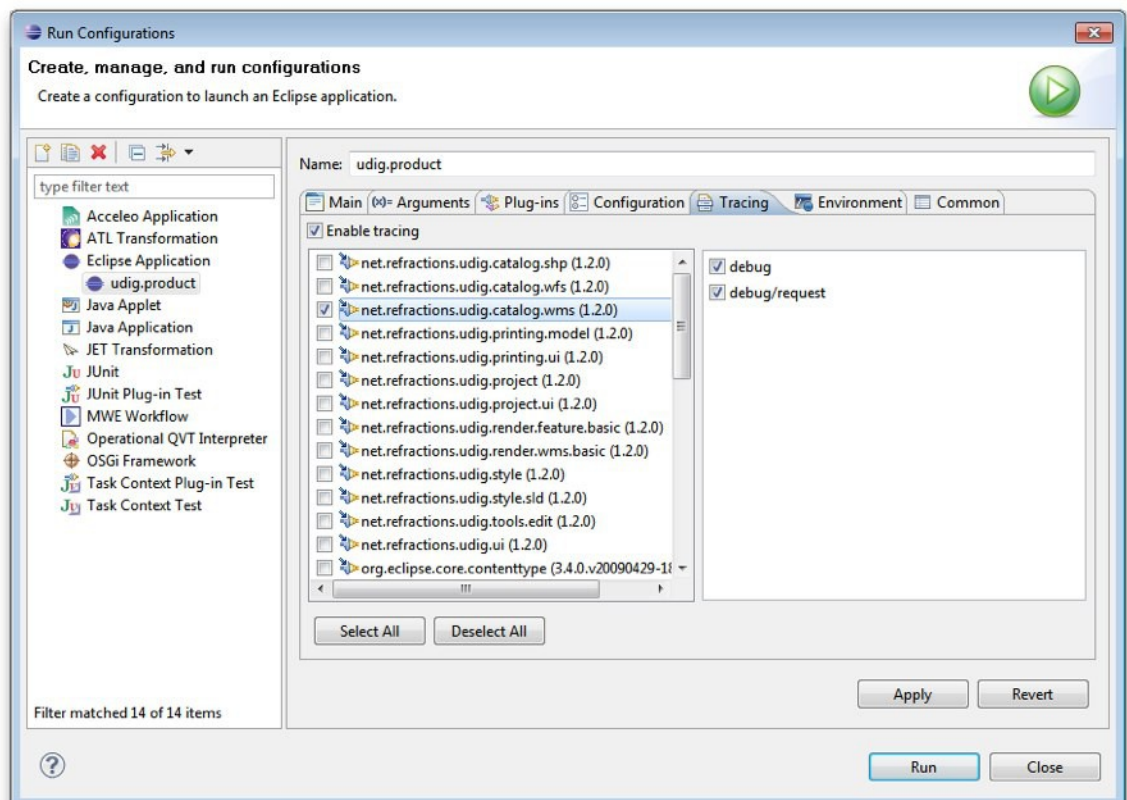
The **-Dosgi.parentClassload=ext** setting allows uDig to find JRE extensions such as Java Advanced Imaging and ImageIO.

- The running uDig application makes use of the “Workspace Data” folder defined in the Run dialog. Try checking **clear** and workspace in order to simulate starting uDig from a fresh install.



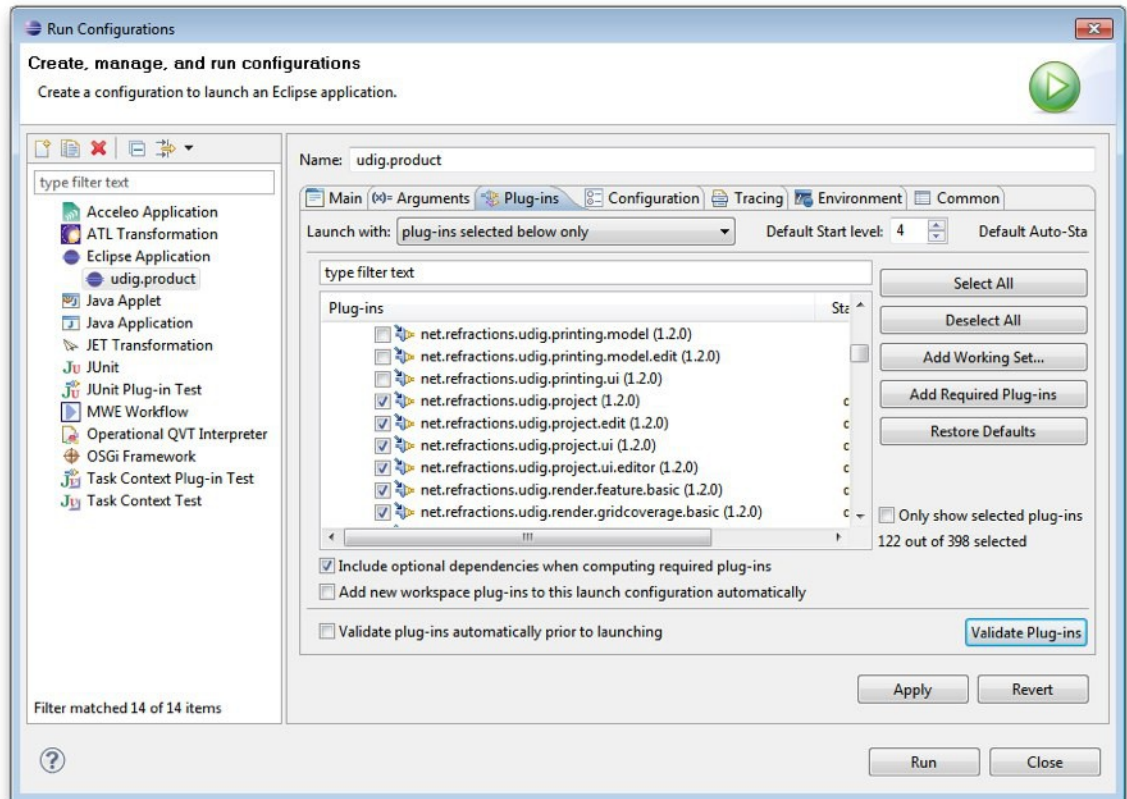
- Have a look on the Tracing tab of the Run dialog; you can control the amount of logging information produced (for example WMS logging is turned on below).

Tracing is especially useful when the -consolelog program argument is used.



- Have a look at the plug-ins tab and see if you can turn off: printing support.

Normally the plugins included by an application are organized into features – we will cover how to do this in the custom application tutorial.



- Advanced: When working on the uDig project itself we use the FindBugz tool to check for obvious mistakes prior to committing. You can add FindBugz to your environment using the following update site:
<http://findbugs.cs.umd.edu/eclipse>

