Tool Plug-in Tutorial

Adding a measure tool







Table of Contents

1 Introduction	3
2 Create a New Plug-in	4
3 Manifest	7
4 Import Icons and Pointers	9
5 Extensions	11
6 Tool Definition	14
7 Tool Implementation	16
8 Testing the Plugin	19
9 Question and Answer	21
10 What to do Next	22

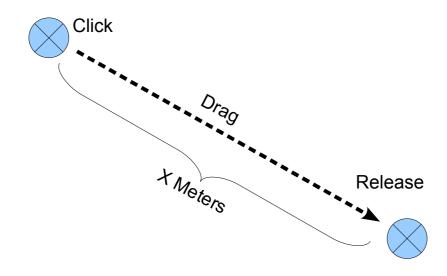
I Introduction

The uDig application is built on the "Eclipse Rich Client Platform" that offers an alternative to traditional application framework design. The RCP Platform is customized through **extensions** that contribute to **extensions points**. The good news is that everything is consistent, everything from adding a tool to creating an application is done in the same manner.

After completing this tutorial, you will have gained the skills to:

- · Create a new Plugin
- Define a new Extension
- Implement a new Tool Extension
- Update the map from within a tool

We are going to create a tool that returns the distance between the point the mouse is clicked and the point the mouse is released.



2 Create a New Plug-in

In this section we are going to create a new plug-in. Eclipse plug-ins are managed bundles of code (often packaged up as a jar or folder).

Our focus in this section is on creating a plug-in and getting the name and version information correct. We will also provide a list of uDig plug-ins we need in order to make a good tool.

An Eclipse class called **Platform** is responsible for loading up our plug-in when it is needed. The Platform class will use the information we provide to make sure all our requirements are met. Plug-ins are loaded up into separate class loaders; and Java class loader restrictions are in place so you really can only talk to plug-ins you depend on!

- 1. Open up Eclipse using the workspace configured for uDig SDK development.
- 2. Select **Window > Open Perspective > Other** and choose the **Plug-in Development** perspective from the list.

The Plug-in Perspective will contain a few Views and Editors you may not be familiar with from day to day Java programming.

CVS Repository Exploring		
∰ Debug		
🖏 Java (default)		
Java Browsing		
Java Type Hierarchy		
(1) Planning		
Plug-in Development		
Resource		
E ⁰ Team Synchronizing		
L		

3. Choose File > New > Project... from the menu bar.

4. Select **Plug-in Project** and press **Next**.

New Project	_ 🗆 🗙
Select a wizard	->
Create a Plug-in Project	
<u>W</u> izards:	
type filter text	
Java Project from Existing Ant Buildfile Java Project from Existing Ant Buildfile Plug-in Project CVS Eclipse Modeling Framework Eclipse Modeling Framework Dug-in Development Examples	
? < Back	Cancel

5. Create a name for the plug-in by entering **Project Name:** net.refractions.udig.tutorials.distancetool

New Plug-in Project		
Plug-in Project		
Create a new plug-in	project	
Project name: net re	fractions.udig.tutorials.distancetool	
Use default locati		
	orkspace\net.refractions.udig.tutorials.distancetool	Browse
Project Settings		
📝 Create a Java proj	ect	
Source folder:	src	
Output folder:	bin	
Target Platform		
This plug-in is target	ed to run with:	
Eclipse versi	on: 3.6 🔻	
🔘 an OSGi frar	nework: Equinox 👻	
Working sets		
Add project to w	orking sets	
Working sets:		Select
?	< Back Next > Finish	Cancel
<u> </u>		

6. Press the Next Button

7. Accept the default values for Plug-in Content.

New Plug-in Project				23
Content Enter the data required to	generate the plug-in.		1	
Properties				
ID:	net.refractions.udig.tutorials.distancetool			
Version:	1.0.0.qualifier			
Name:	Distancetool			
Provider:				
Execution Environment:	JavaSE-1.6	▼ Envir	onment	s
Activator: net.refrac This plug-in will make Enable API Analysis	tions.udig.tutorials.distancetool.Activator e contributions to the UI			
Rich Client Application Would you like to create	a rich client application?	© Yes	No	,
?	< Back Next > Finish		Cancel	

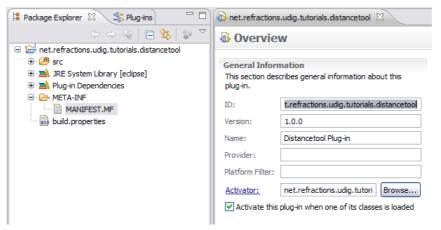
8. Press Finish to create your new project.

3 Manifest

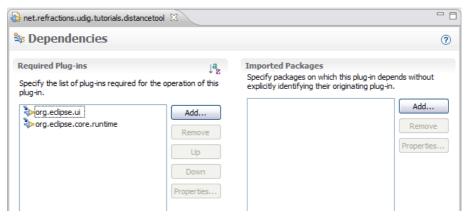
Your new project is both: a plug-in project (with a MANIFEST.MF file for the Eclipse Platform class to read); and a Java project with .classpath file for the eclipse compiler to read.

Lets have a look at what information is MANIFEST.MF:

1. In the **Package Explorer** navigate to the plug-in created in the previous section. Find the file **META-INF/MANIFEST.MF** and double click to open the plug-in manifest editor.



- 2. The overview tab shows much of the information you entered when you created the plug-in project.
- Switch to the Dependencies tab. (the tabs are located at the bottom of the editor area)



4. Click on the **Add** button under Required Plug-ins.

5. Select the **net.refractions.udig.project.ui** plug-in from the list.

project.ui	
Matching items:	
net.refractions.udig.project.ui (1.2.0.201008151814) het.refractions.udig.project.ui.editor (1.2.0.20100815181	
net.refractions.udig.project.ui	

6. At this point you need to save your work (using File > Save).

net.refractions.udig.tutorials.distanceto	ol 83	
Dependencies		C
Required Plug-ins Specify the list of plug-ins required for the plug-in.	$\downarrow^a_{\mathbb{Z}}$ e operation of this	Imported Packages Specify packages on which this plug-in depends without explicitly identifying their originating plug-in.
org.eclipse.ui org.eclipse.core.runtime org.eclipse.core.runtime org.eclipse.core.runtime	Add Remove Up Down Properties	Add Remove Properties

7. Changes to the plug-in project (ie MANIFEST.MF file) are used to update the Java project (ie .classpath file). If you do not save your work you will not be able find the tool classes used later in the tutorial.

Use "*project.ui" to quickly find the needed plug-in.

4 Import Icons and Pointers

You can use normal image files to define tool bar icons, and cool pointers.

By convention icons are organized into the following directories:

icons/etool16	Enabled icon used in application tool bar.
↔	(16x16 left and top clear)
icons/pointers	Used to define a cursor (32x32)

Lets download the images used for this tutorial:

- 1. Download the following file: <u>http://udig.refractions.net/files/tutorials/distance_icons.zip</u>
- 2. Select File > Import to open up the Import wizard

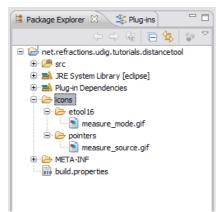
3. Choose General > Archive File and press Next

 Import Select Import resources from an archive file into an existing project. Select an import source: type filter text General Archive File Breakpoints Existing Projects into Workspace File System Preferences CVS CVS Mylyn Plug-in Development Team 		
elect Import resources from an archive file into an existing project. Select an import source: type filter text General General General File System File	ct.	
Select an import source:		
type filter text		
Archive File Breakpoints File System File System CVS CVS Point CVS Point CVS Point CVS Point CVS		
0	< Back Next >	Enish Cancel

If you are using this work book in a lab setting you will find the file on your DVD. Fill in the following details on the Archive File page: From archive file: distance_icons.zip Into folder: net.refractions.udig.tutorials.distancetool

Import	_ 🗆 🗙
Archive file	
Import the contents of an archive file in zip or tar format from the local file system.	Į.
From archive file: C:\Documents and Settings\Jody\My Documents\My Downloads'	Browse
Filter Types Select All Deselect All	
Into folder: net.refractions.udig.tutorials.distancetool	Bro <u>w</u> se
Qverwrite existing resources without warning	
⑦ < <u>Back</u> <u>Next</u> > Einish	Cancel

- 5. Press Finish
- 6. An icons directory will be created.



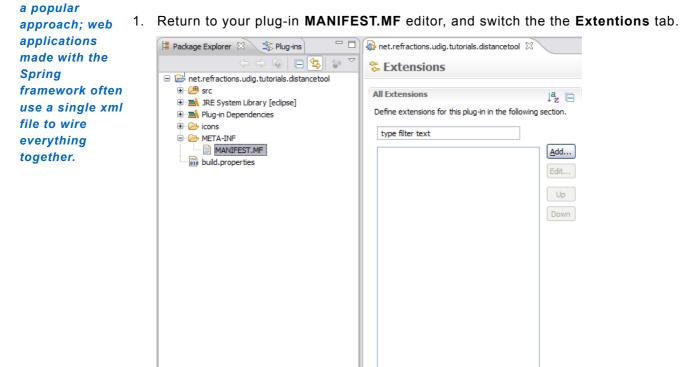
5 Extensions

Programming with extensions is accomplished in two parts; filling in a bit of information (letting the Eclipse Platform class know what you are up to); and then implementing a Java class to do the work.

The Platform class acts as a mediator; hooking up the distance tool we define here to the uDig application that will display it on the tool bar and make use of the tool when the user asks.

Overview Dependencies Runtime Extensions Extension Poin

The information we provide is stored in a file called plugin.xml; the Platform class reads in *This is becoming* all the plugin.xml files when the application is started it and wires up everything.



2. Press the Add button

3. Select **net.refractions.udig.project.ui.tool** from the list of extension points.

🔿 New Extension
Extension Point Selection
Create a new Tool extension.
Extension Points Extension Wizards
Extension Point filter:
-1 net.refractions.udig.project.style
-1 net.refractions.udig.project.ui.editorInputs
- net.refractions.udig.project.ui.featureEditor
- net.refractions.udig.project.ui.itemProviderAdapterFactories
-Inet.refractions.udig.project.ui.tool
Internet.refractions.udin.nroiect.ui.toolMananers
✓ Show only extension points from the required plug-ins
Extension Point Description: Tool
The tool extension point allows third-party developers to develop new tools for UDIG and is one of the most used points of extension. There are three different type of tools and two ways of grouping tools. The three type of tools are as follows:
Available templates for tool:
(

- 4. Click Finish
- The id and name provided here is used by the Platform class to log errors associated with your tool.

There is a bug with icon lookup which is why we made the id empty. Enter the following Extention Details: ID: <Make it Empty> Name: Distance Tool Example

Extensions	Ja 🖃	Extension Details	
efine extensions for this plug-in in the following	g section.	Set the properties of the selected extension. Required fields are denoted by "*".	
type filter text		ID:	
······ = net.refractions.udig.project.ui.tool	<u>A</u> dd	Name: Distance Tool Example	
	Edit		
	Up	Show extension point description	
	Down	Open extension point schema	

6. Change to the build tab of your MANIFEST.MF editor. Make sure your **icons** folder and **plugin.xml** is checked as part of the Binary Build.

Binary Build Select the folders and files to include in the binary build.
✓ .dasspath ✓ .project ✓ >META-INF ✓ >bin Image: build.properties ✓ >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Extra Classpath Entries
Overview Dependencies Runtime Extensions Extension Points Build MAI

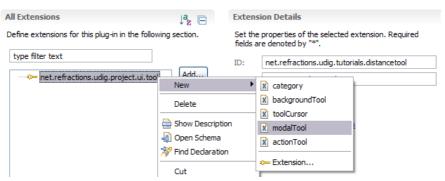
7. This step will ensure that the icons are included when the plug-in is bundled up into a jar. Proceed to the next section and we will define our distance tool.

6 Tool Definition

We can now use the extensions tab to define our distance tool. A single plug-in may define multiple tools, and indeed provide contributions to several extension points.

To create a new tool:

1. Right click on newly added **net.refractions.udig.project.ui.tool** in the Extensions tab, and select **New > modalTool**



- 2. Fill in the following value for id: net.refractions.udig.tutorials.distancetool.
- Enter a tool tip message into tooltip: Measure the surface distance between two points
- 4. For icon, click on the browse button and select: icons/etool16/measure_mode.gif

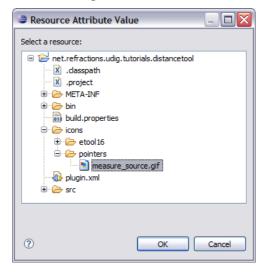
🕏 Resource Attribute Value	×
Select a resource:	
 net.refractions.udig.tutorials.distancetool dasspath project META-INF bin bin build.properties cons cons cons measure_mode.gif plugin.xml src 	
⑦ OK Cancel	

- 5. Fill in the following for name: Distance
- 6. For onToolbar: select true from the list
- 7. For the tool categoryId: net.refractions.udig.tool.category.info

8. Save your work.

All Extensions	↓ ^a z 🖻	Extension Eleme	nt Details	
Define extensions for this plug-in in the following section.		Set the properties	of "modalTool". Required fields are denote	ed by "*".
type filter text		id*:	net.refractions.udig.tutorials.distancet	ool.modalToc
net.refractions.udig.project.ui.tool	Add	tooltip:	Measure the surface distance between	two points
Distance (modalTool)	Remove	<u>class:</u>		Browse
		icon:	icons/etool16/measure_mode.gif	Browse
	Up	name:	Distance	
	Down	onToolbar:	true	*
	DOWI	categoryId:	net.refractions.udig.tool.category.info	
		commandIds:		
		toolCursorId:		
		commandHandler:		Browse
		selectionProvider:		Browse

- 9. We are going to add a child element that specifies the cursor.
- 10. Right click on **Distance** and select **New > cursor**.
- 11. For the image: use the browse button to select: icons/pointers/measure_source.gif



12. Fill in the location of the hot spot where the user clicks: hotSpotX: 10 hotSpotY: 10

All Extensions	↓ ^a z ⊟	Extension	Element Details
Define extensions for this plug-in in the following	section.	Set the pro by "*".	perties of "cursor". Required fields are denoted
type filter text	<u>A</u> dd	image:	icons/pointers/measure_source.c
i → Hettern dealistatig, pojectian don i → Distance (modalTool)	Edit	hotspotX: hotspotY:	10 10
	Up	id:	
	Down		

13. Save your work before continuing to the next section.

7 Tool Implementation

Now that all the description is out of the way we can implement the DistanceTool class.

- 1. Return to the **Distance (modalTool)** element (it is child of net.refractions.udig.project.ui.tool)
- 2. Enter in the following for **class**: net.refractions.udig.tutorials.distancetool.DistanceTool

All Extensions	↓ ^a z ⊡	Extension Elemen	nt Details
Define extensions for this plug-in in the following se	_	Set the properties ("*",	of "modalTool". Required fields are denoted by
type filter text		id*:	net.refractions.udig.tutorials.distancetool.mo
end e	Add	tooltip:	Measure the surface distance between two p
Distance (modalTool)	Remove	class:	Browse
		icon:	icons/etool16/measure_mode.gil Browse
	Up	name:	Distance
	Down	onToolbar:	true 💌
		categoryId:	net.refractions.udig.tool.category.info
		commandIds:	
		toolCursorId:	
		commandHandler:	Browse
		selectionProvider:	Browse

- 3. Click the **class** link shown above.
- 4. A **New Java Class** wizard is opened, the needed details should have be already filled in for you.

Constructors from superclass: uncheck

New Java Class		
Java Class Create a new Java o	class.	
Source folder:	net.refractions.udig.tutorials.distancetool/src	Browse
Package:	net.refractions.udig.tutorials.distancetool	Browse
Enclosing type:		Browse
Name:	DistanceTool	
Modifiers:	public default private protected abstract final static	
Superclass:	net.refractions.udig.project.ui.tool.SimpleTool	Browse
Interfaces:		Add
		Remove
Which method stub	os would you like to create?	
	public static void main(String[] args)	
	Constructors from superclass	
Description	Vinherited abstract methods	
Do you want to add	I comments? (Configure templates and default value <u>here</u>)	
?	Finish	Cancel

5. Press Finish

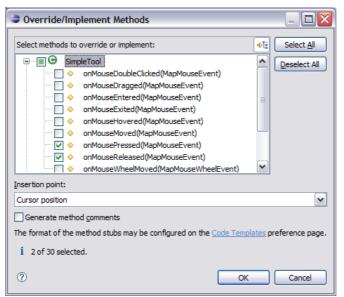
6. The following file will be created for you.

```
package net.refractions.udig.tutorial.distancetool;
import net.refractions.udig.project.ui.tool.SimpleTool;
public class DistanceTool extends SimpleTool {
```

7. Add the following field, to jot down where the user clicked.

```
import com.vividsolutions.jts.geom.Coordinate;
public class DistanceTool extends SimpleTool {
    /** records where in the world the user clicked */
    Coordinate start;
```

- 8. Right click on the editor and select Source > Override/Implement Methods
- 9. Expand SimpleTool node and check the following:
 - Check onMousePressed(MapMouseEvent)
 - Check onMouseReleased(MapMouseEvent)



- 10. Click on the **OK** button to create these methods.
- 11. Implement the onMousePressed(MapMouseEvent) method

```
@Override
protected void onMousePressed(MapMouseEvent e) {
    start=getContext().pixelToWorld(e.x, e.y);
}
12. Implement the onMouseReleased(MapMouseEvent) method.
```

We are using the utility **JTS** class to calculate the distance between two coordiantes.

```
@Override
protected void onMouseReleased(MapMouseEvent e) {
    Coordinate end=getContext().pixelToWorld(e.x, e.y);
    try {
        double distance=JTS.orthodromicDistance(
        start, end,
        getContext().getCRS() );
        displayOnStatusBar(distance);
    } catch (Exception el) {
        displayError();
     }
}
```

Implement the displayOnStatusBar(double) method. We need to

```
private void displayOnStatusBar(double distance) {
       final IStatusLineManager statusBar =
       getContext().getActionBars().getStatusLineManager();
   if( statusBar==null ) {
              return; // shouldn't happen if the tool is being used.
   int totalmeters=(int)distance;
      final int km=totalmeters/1000;
      final int meters=totalmeters-(km*1000);
      float cm = (float) (distance-totalmeters)*10000;
      cm = Math.round(cm);
      final float finalcm=cm/100;
      getContext().updateUI(new Runnable() {
               public void run() {
                      statusBar.setMessage(
                      "Distance = "+km+"km "+meters+"m "+finalcm+"cm"
                  );
              }
      });
```

14. Implement the displayError () method

```
private void displayError() {
    final IStatusLineManager statusBar =
        getContext().getActionBars().getStatusLineManager ();
    if( statusBar==null )
        return; // shouldn't happen if the tool is being used.
    getContext().updateUI(new Runnable() {
        public void run() {
            statusBar.setErrorMessage("Unable to calculate the distance");
        }
    });
}
```

- 15. The file will not compile as we have a few imports to sort out.
- 16. Press **Ctrl-Shift-o** to import any needed classes. You will have one conflict to sort out, please choose: org.geotools.geometry.jts.JTS
- 17. Save your file, this should refresh the project and clean up any remaining error message.

8 Testing the Plugin

We can now run uDig and try out your new plug-in:

- 1. Select **Run > Run Configurations...** from the menu bar and choose the configuration you set-up in the previous tutorial
- 2. Go to the **Plug-ins** tab and check that "Launch With" is set to "plug-ins selected below only". The actual plugins selected were defined by the udig.product we are going to add our new plugin to this list next.
- 3. Select your new plugin, listed at the top under "Workspace"

📄 Main 🕬= Arguments ᡝ Plug-ins 🛛 📴 Configuration 🚔 Tracing 🕅	Environment	Common	
Launch with: plug-ins selected below only Default Star	t level: 4	Default Aut	o-Start: false 🔻
type filter text			Select All
Plug-ins	Start Level	Auto-Start	Deselect All
♥ ♥ Workspace ♥ ♥ net.refractions.udig (1.2.0.201008151814) ♥ ♥ net.refractions.udig.tutorials.distancetool (1.0.0.qualifier) ● ♥ Target Platform	default 0	default default 🔻	Add Working Set Add Required Plug-ins
	default default	default default	Restore Defaults
▼ javax.servlet (2.5.0.v200910301333) ▼ javax.servlet(jsp (2.0.0.v200806031607)	default default	default default	- Only show selected
III.		•	327 out of 628 selected
 Include optional dependencies when computing required plug-ins Add new workspace plug-ins to this launch configuration automatically 			
Validate plug-ins automatically prior to launching			Validate Plug-ins
		Ap	pply Revert

- 4. Click Run.
- 5. After the application has started up we can put together a map to try out the distance tool.
- 6. Select File > New > New Map
- 7. Change to the **Web** view and choose **WMS: jpl nasa** on the page. Select the **Blue Marble Next Generation** layer and press **Finish**.

- You can find your distance tool "under the i" since you placed it in the information tool category.
- 8. Select the Distance Tool () from the toolbar and drag from one point to another on the Map.



9. The distance will be displayed in the Map status bar.

9 Question and Answer

Here are some common questions:

Q: My distance tool does not show up

A: Did you open the information tool menu and look in the drop down list?

A: Check your plugin.xml file; make sure the "id" is unique; check that it does not conflict with the extension point id that contains it.

Q: Unable to calculate the distance!

A: The projection of your data must be defined, you may see this if you are working with a shapefile that does not have a ".prj" file defined.

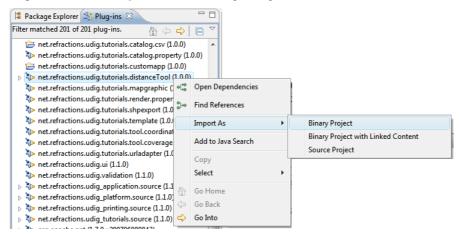
Q: Connection error has occurred

A: Sounds like the data you were looking for is unavailable, try a different WMS.

Q: How can I look at the source code examples

A: It is included in the SDK:

- 1. Make sure you have the Plug-in Development Perspective open
- 2. Open the Plugins tab and scroll down the the code examples
- 3. Right click and Import As a Binary Project



4. The project will be copied into your workspace.

10 What to do Next

Here are some additional challenges for you to try:

• Tools are organized into "Categories" each with their own keyboard short-cut, the Distance Tool is in the category "Information".

Create your own Category

All Extensions		↓ <mark>a</mark> [=		
Define extensions for this plug-i	n in tl	he following section.			
type filter text					
			٦.		
▲ ⁽¹⁾ net.refractions.udig.pro ▷ (→) Distance (modalTo		New	•	Х	backgroundTool
p is a bistance (modarro		Delete		X	category
		Delete		X	modalTool
		Show Description		x	toolCursor
	-	Open Schema		x	actionTool
	*	Find Declaration			
				0=	Extension
		Cut	. L.	_	

• Plug-ins have a life cycle (controlled by the Platform class); the start method is used to provide your Activator with a BundleContext (used to access resources like lcons). The ID for your Plugin is used so often that it is worth making a constant in your activator for others to refer to. This may already be done by the wizard.

public static final String PLUGIN_ID =
 "net.refractions.udig.tutorials.distancetool";

 Your activator can also be used to send log messages; and check debug options (from the TRACING page)

Run Configurations		×
Create, manage, and Create a configuration to	d run configurations Jaunch an Eclipse application.	
Image: Control of the second secon	Name: udig.product Main (bit-Arguments (*) Plug-ins (*) Configuration (*) Tracing Environment (*) Common Image: Image: Configuration (*) Tracing Environment (*) Common Image: Image: Configuration (*) Configuration (*) Tracing Image: Configuration (*) Config	
Filter matched 10 of 10 it	Apply Rejet	
?	Bun Clor	se

To enable this, add a ".options" file to your plug-in next to plugin.xml. The presense of a ".options" file tells the system that you have trace options available.

```
net.refractions.udig.tutorials.distancetool/debug=true
net.refractions.udig.tutorials.distancetool/debug/distance=true
```

At runtime use your Activator to check if tracing is turned on, put the following in the start method:

```
if( isDebugging() &&
    "true".equalsIgnoreCase(Platform.getDebugOption(PLUGIN_ID + "/debug"))){
    Status status = new Status(IStatus.INFO, PLUGIN_ID, "Distance Tool Started");
    getLog().log(status);
```

Now, output your distance results to the console log. This information is currently being displayed in the status bar (eg. 3420 km)

• Advanced: You can select the "Information" category by pressing "i". Hook up your category to a keyboard binding.