

Developing Java SWT Applications

A Quick Start

<http://www.RCP-Vision.com>
Email: info@rcp-vision.com

SWT Applications in minutes

In this short presentation we will learn how to
develop and deploy
a simple SWT Application in
a matter of minutes.

Requirements

The setup process however is not included in these minutes, because you will have to do some download and setting process.

Anyway this is a one-time task!

Requirements

Ensure you have a Java JDK installed
(Java 1.5 or higher)

Download Eclipse RCP distro

Go to the Eclipse download page:

<http://www.eclipse.org/downloads>

and download the RCP version of Eclipse

Downloads: 225,387

	<p>Eclipse for RCP/Plug-in Developers (175 MB) A complete set of tools for developers who want to create Eclipse plug-ins or Rich Client Applications. It includes a complete SDK, developer tools and source code, plus Mylyn, an XML editor and the Eclipse Communication Framework. More... Downloads: 136,270</p>	<p>Windows Mac OS X Linux 32bit Linux 64bit</p>
---	---	---

	<p>Eclipse IDE for Java and Report Developers (195 MB) JEE tools and BIRT reporting tool for Java developers to create JEE and Web applications</p>	<p>Windows Mac OS X</p>
---	--	-----------------------------

Extracting Eclipse RCP IDE

Extract the downloaded file
(e.g. “eclipse-rcp-ganymede-SR1-win32.zip”)
somewhere on your local drive (e.g. on C:\)

Note: do not use the the Compressed Folder
unzipper shipped with the operating system since
it may not extract hidden files

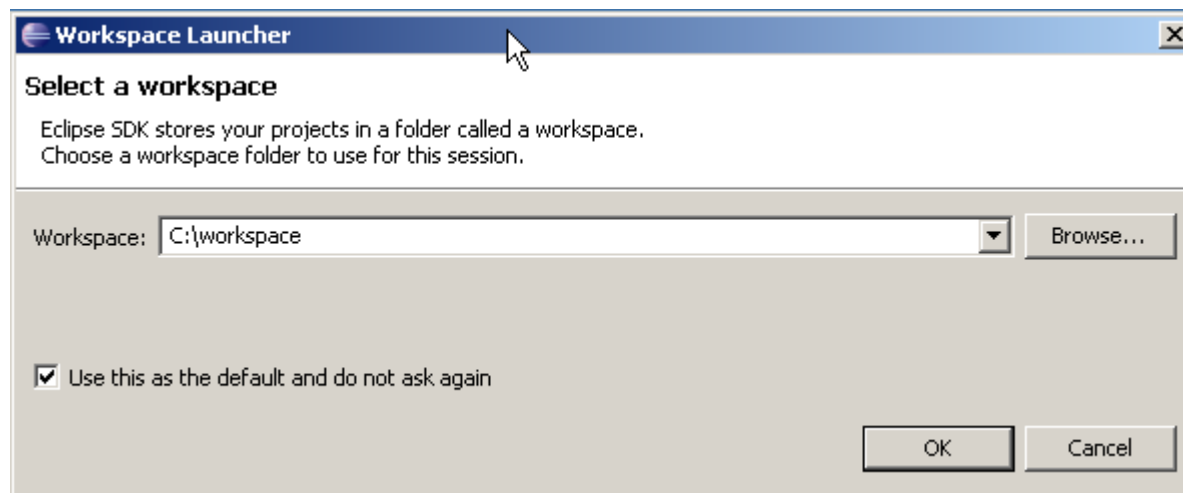
Running Eclipse RCP IDE

Run the Eclipse executable file

(e.g. C:\eclipse\eclipse.exe)

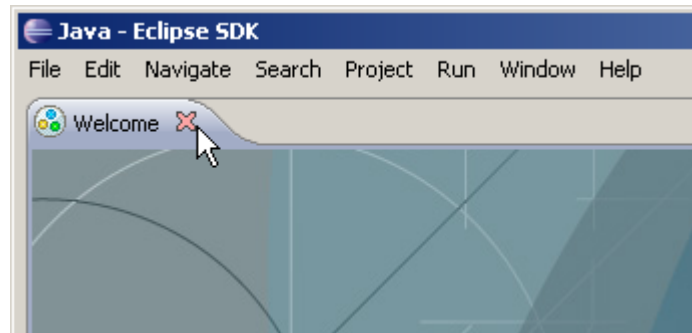
Running Eclipse RCP IDE

Choose a workspace location



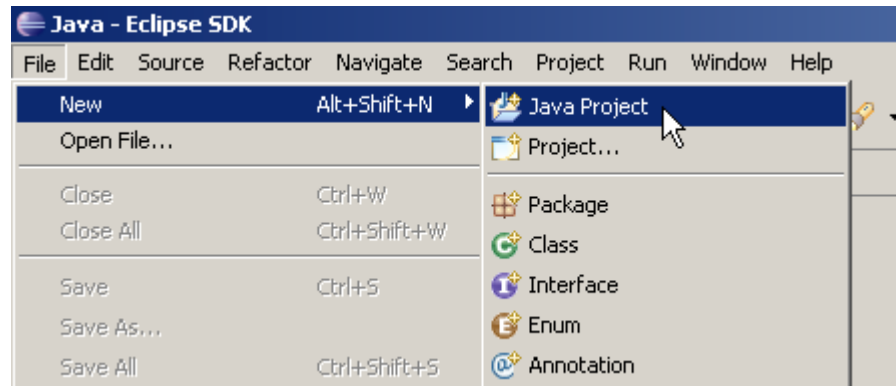
Running Eclipse RCP IDE

and close the Welcome page



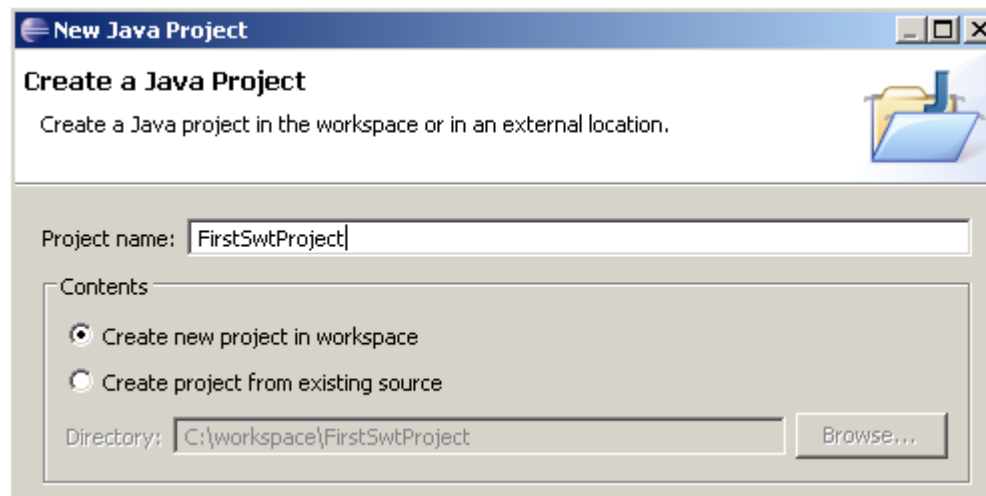
Create a Java Project

File, New, Java Project



Create a Java Project

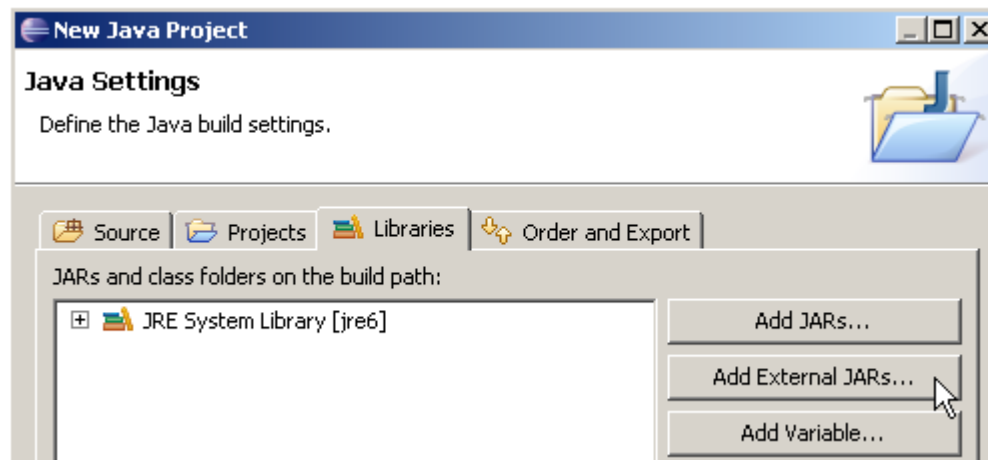
Enter a name for the Project
(e.g. FirstSwtProject)



and press the Next button

Create a Java Project

Select the “Libraries” tab

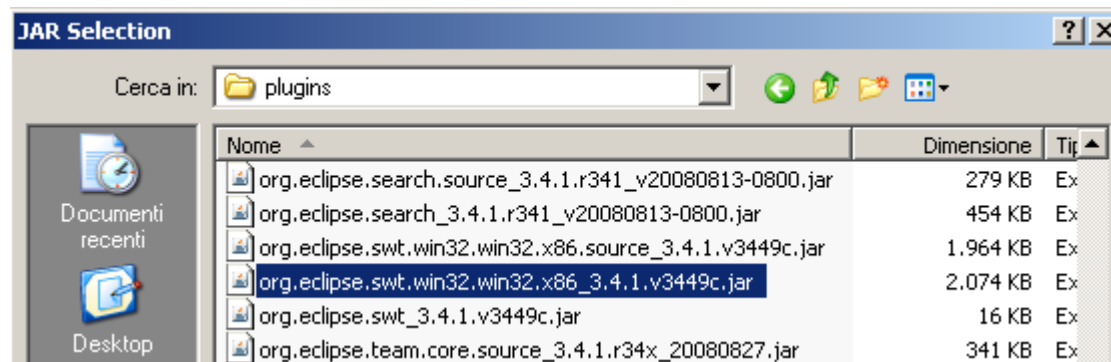


and press the “Add External JARs ...” button

Create a Java Project

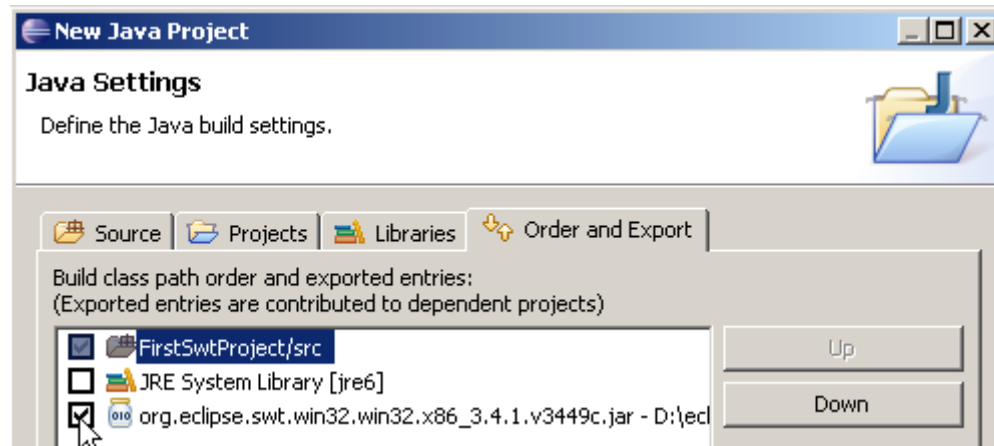
Browse and select the file

<eclipse_dir>/plugins/
org.eclipse.swt.win32.win32.x86_3.4.1.v3449c.jar



Create a Java Project

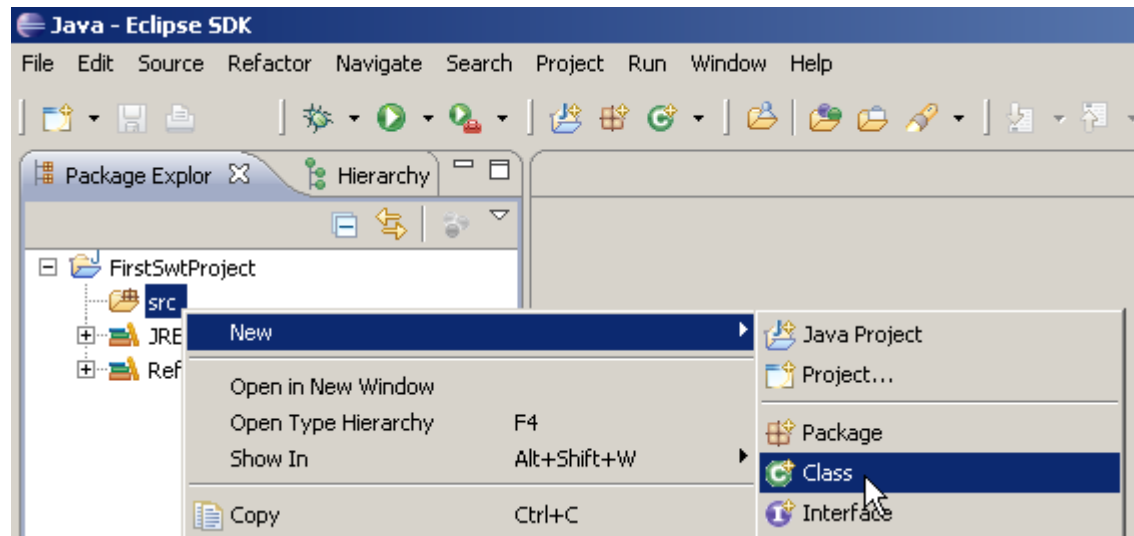
Select the “Order and Export” tab



and check the jar file we just added
(this will export this jar file in the deploy phase)
and press the “Finish” button

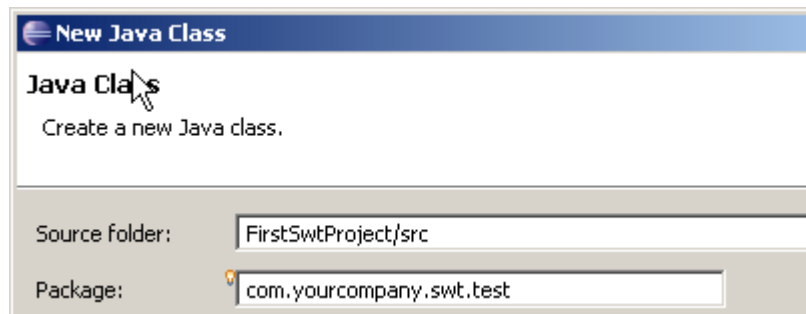
Create a Java Class

Expand the Project tree,
select the “src” folder
and right-click New, Class



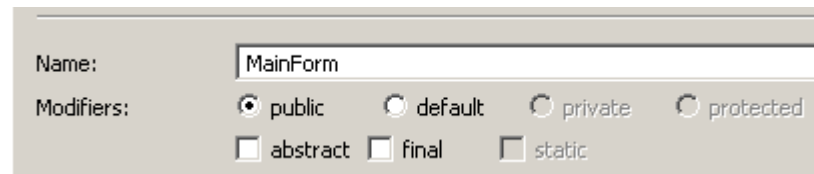
Create a Java Class

Enter a package name for the new class
(e.g. “com.yourcompany.swt.test”)



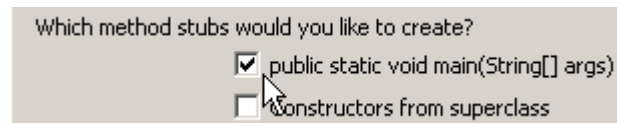
Create a Java Class

Enter a name for the new class
(e.g. “MainForm”)



A screenshot of the 'Name' and 'Modifiers' fields in the 'Create New Class' dialog. The 'Name' field contains the text 'MainForm'. The 'Modifiers' section has radio buttons for 'public' (selected), 'default', 'private', and 'protected'. Below these are checkboxes for 'abstract', 'final', and 'static', all of which are currently unchecked.

and make it executable
(checking “public static void main(String[] args)”)



A screenshot of the 'Which method stubs would you like to create?' dialog. It has two checkboxes: 'public static void main(String[] args)' (checked) and 'Constructors from superclass' (unchecked). A mouse cursor is pointing at the checked checkbox.

then press the “Finish” button

Edit the Java Class

Fill the MainForm class with the following code

```
public class MainForm {
    private static int n = 0;
    public static void main(String[] args) {

        Display display = new Display();
        Shell shell = new Shell(display);
        shell.setText("Click counter");
        shell.setBounds(100, 100, 200, 100);
        shell.setLayout(null);

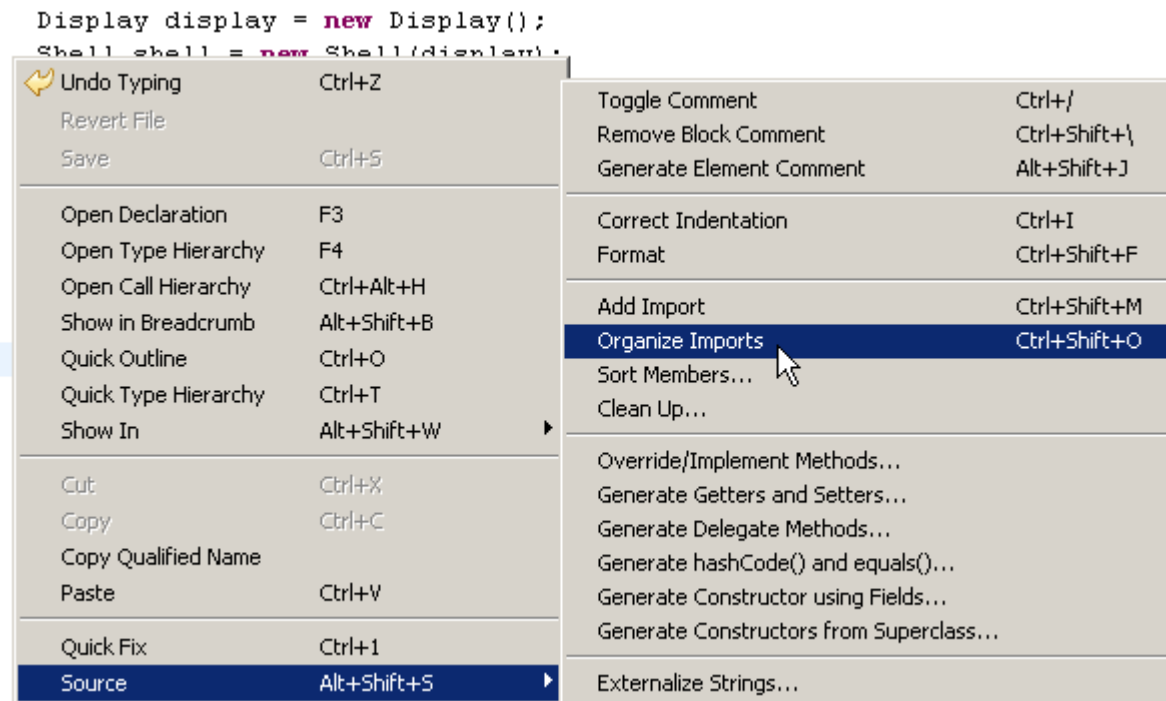
        final Label label = new Label(shell, SWT.PUSH);
        label.setBounds(120, 20, 30, 30);

        final Button button = new Button(shell, SWT.PUSH);
        button.setBounds(10, 10, 80, 30);
        button.setText("Click Me");
        button.addSelectionListener(new SelectionAdapter() {
            public void widgetSelected(SelectionEvent event) {
                n++;
                label.setText(""+n);
            }
        });

        shell.open();
        while (!shell.isDisposed()) {
            if (!display.readAndDispatch())
                display.sleep();
        }
        display.dispose();
    }
}
```

Edit the Java Class

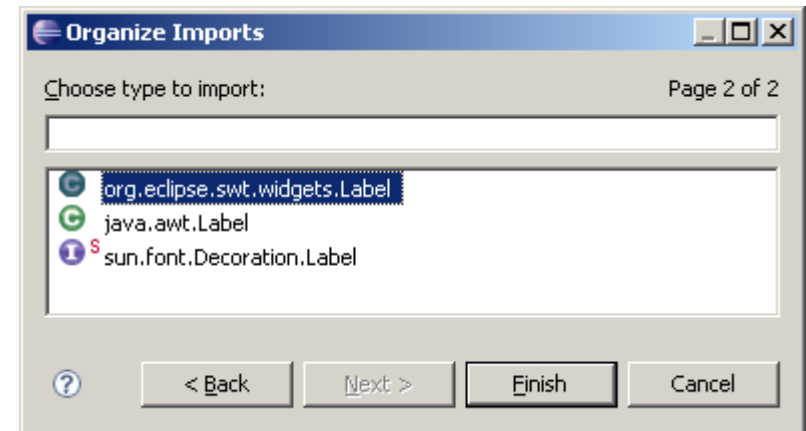
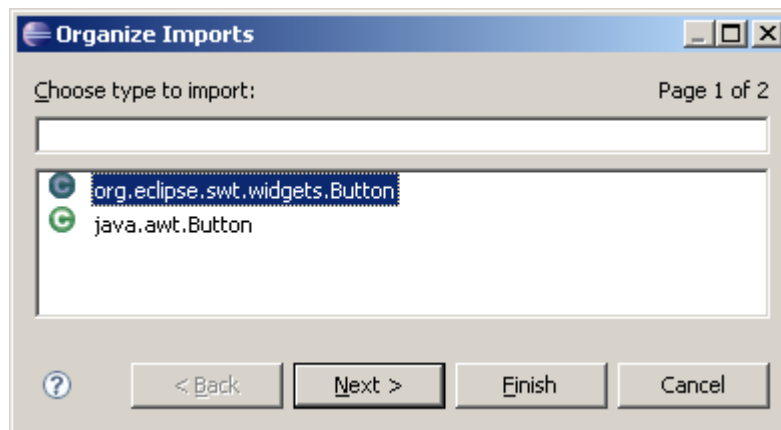
In order to solve the errors that are showed:
right-click on the editor area and perform a
“Source, Organize imports” operation



Edit the Java Class

Choose to import the SWT Button and Label class

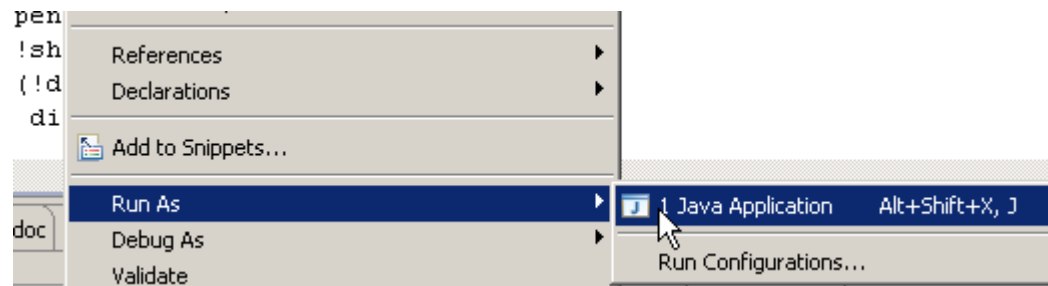
(“org.eclipse.swt.widgets.Button” and
“org.eclipse.swt.widgets.Label”)



Run the Java Class

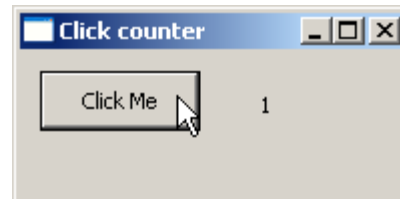
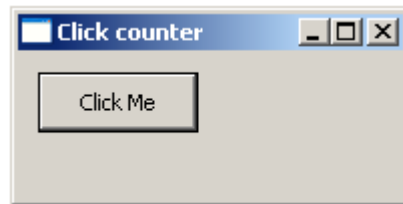
Save the class (File, Save)
then

Right-click on the editor area and perform a
“Run as, Java Application”



Run the Java Class

You should see a form with a button
and a label showing the number
of times the button was clicked



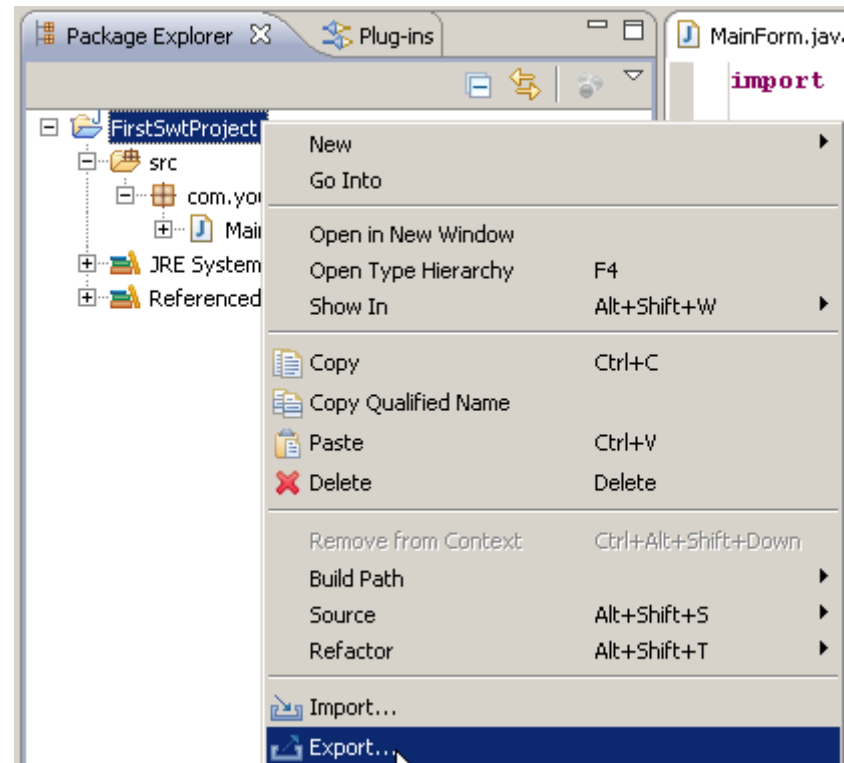
Deploy the Application

Now that we have developed our (tiny) application
how do we deploy it to a client ?

First of all there is a requisite: on the client there
must be a Java Runtime Environment installed

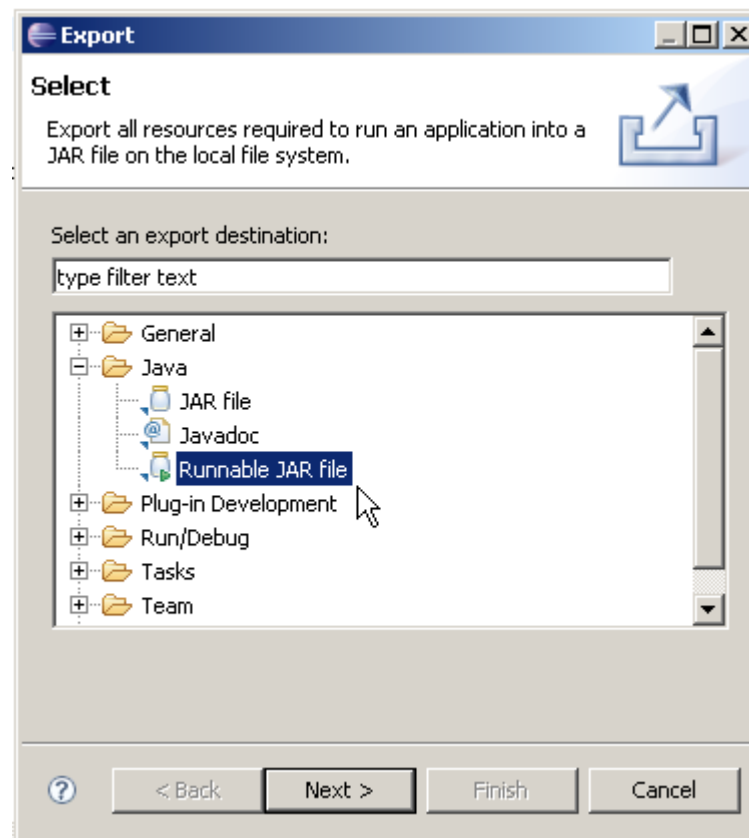
Deploy the Application

Right-click on the Project, Export



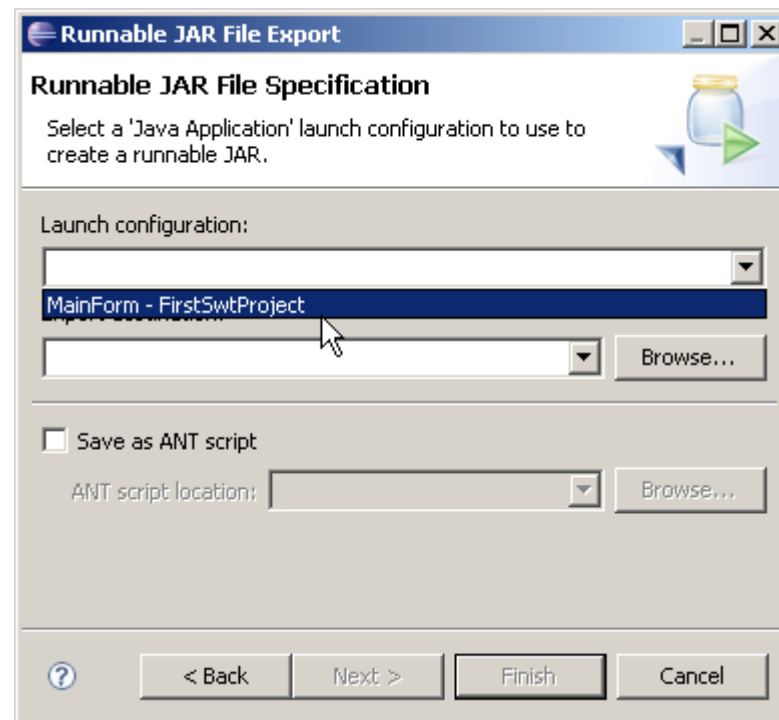
Deploy the Application

Expand the Java node,
select “Runnable JAR file”,
then press the Next button



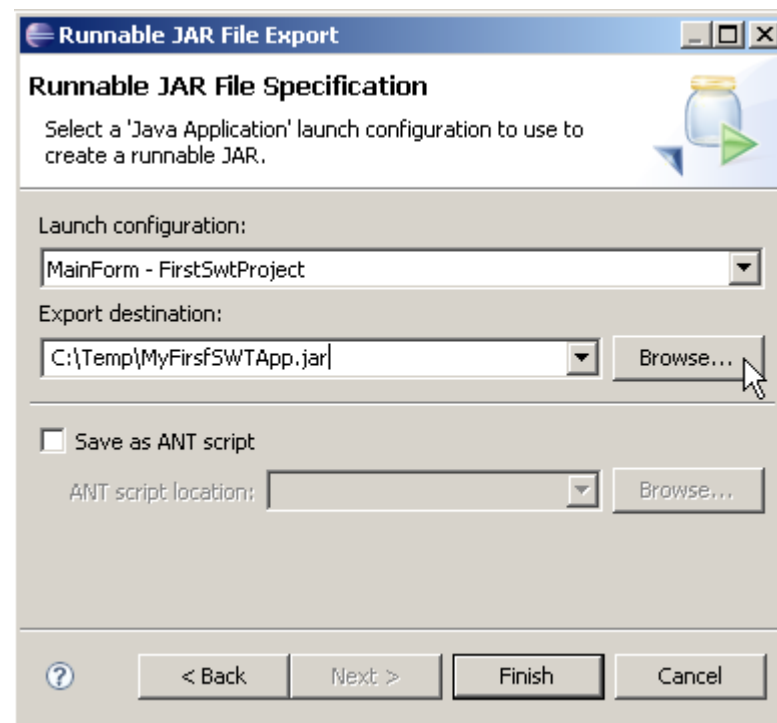
Deploy the Application

Select the (only) runnable Class



Deploy the Application

Browse and enter a name for the exported JAR (e.g. C:\Temp\MyFirstSWTApp.jar), then press the Finish button



Deploy the Application

Ok, we are finished!

Now just double-click the exported JAR file:

it is your executable Java SWT Application.

It can be copied and launched from any client with
a Java Runtime Environment installed!