CISC 275: Introduction to Software Engineering

Lab 1:

Getting Started with

eclipse

Charlie Greenbacker
University of Delaware
Fall 2011

Introduction

- Teaching Assistant: Charlie Greenbacker
- Email: charlieg@cis.udel.edu
- Website: http://www.cis.udel.edu/~charlieg
- Office hours in Smith 103: TBA
- Weekly labs will introduce new tools, etc. for use in class & labs
- Lab slides will always be posted to my website on the Teaching page

Overview

- What is Eclipse?
- Starting up Eclipse
- Walkthough of interface
- Creating a new project
- Creating a new class
- Entering code
- Running a Java program
- Other helpful features
- Lab exercise

What is Eclipse?

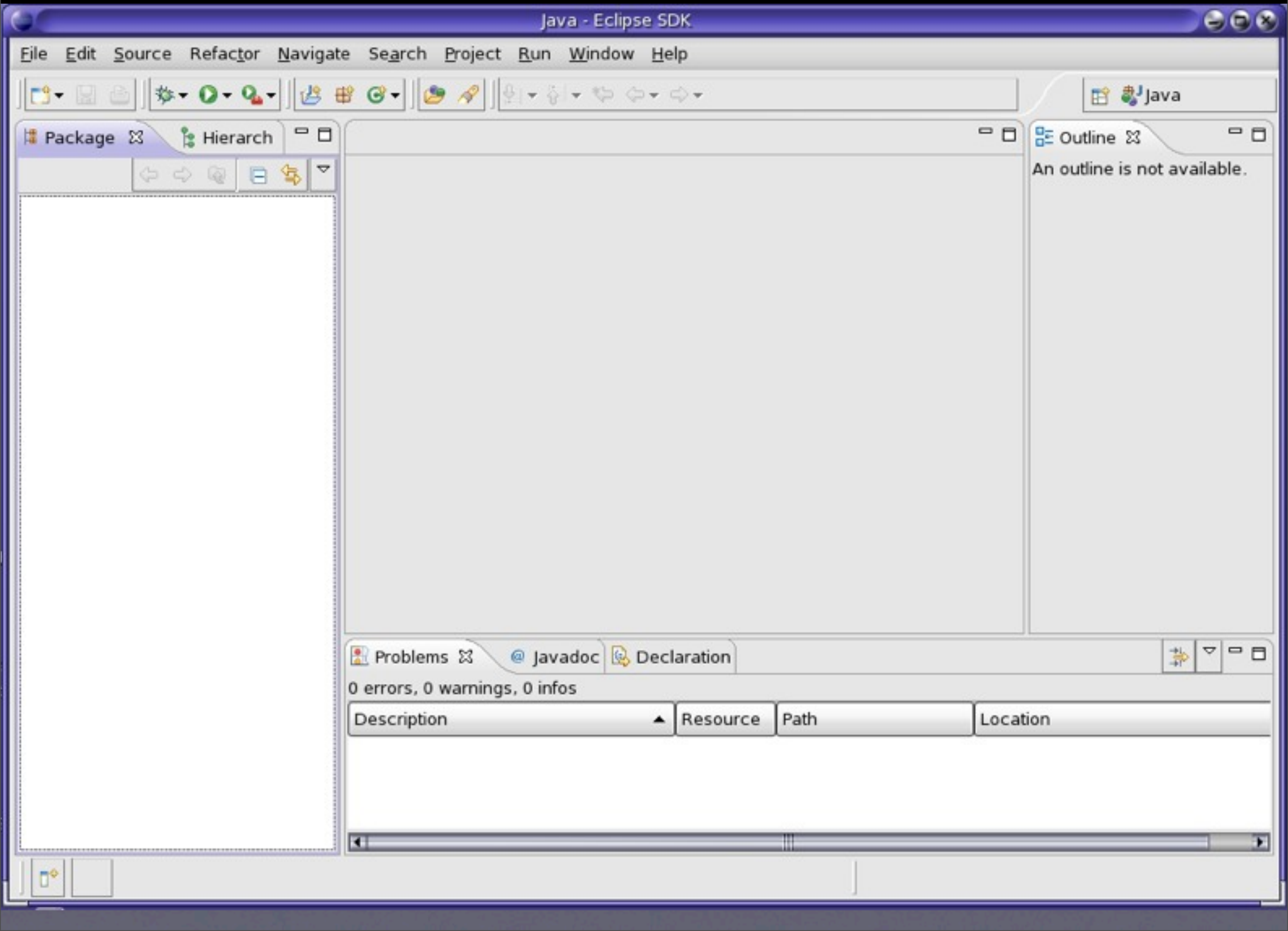
- An integrated development environment (IDE)
  - Source code editor + compiler/interpreter + build tools + debugger & more
- Organizes & manages software projects
- Lots of automated tools to make the programmer's job easier
- Specifically designed for Java but also works with other languages
- Analogy: IDE is to text editor as word processor is to typewriter
- Many plug-ins available to extend feature set

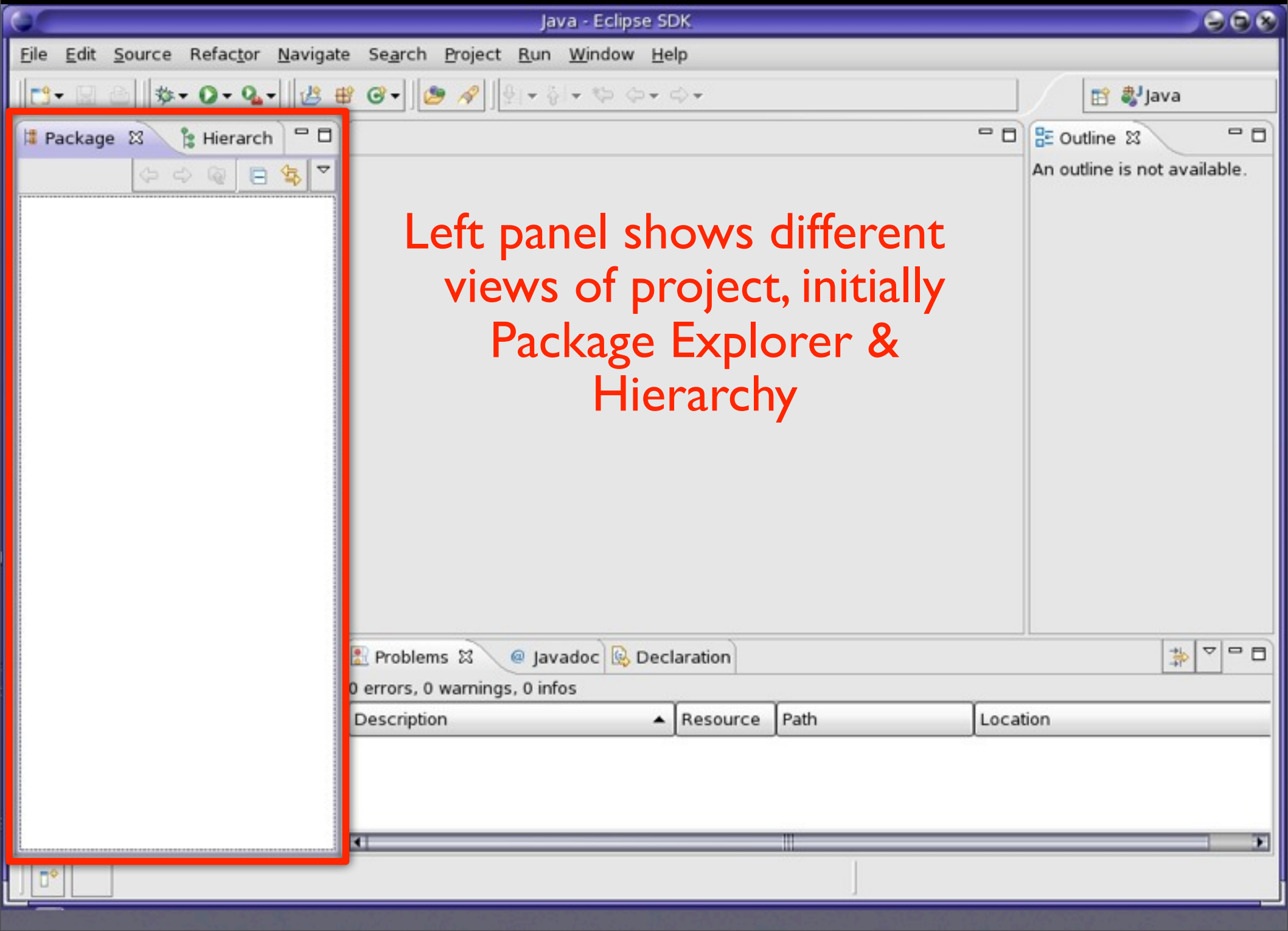
Starting up Eclipse

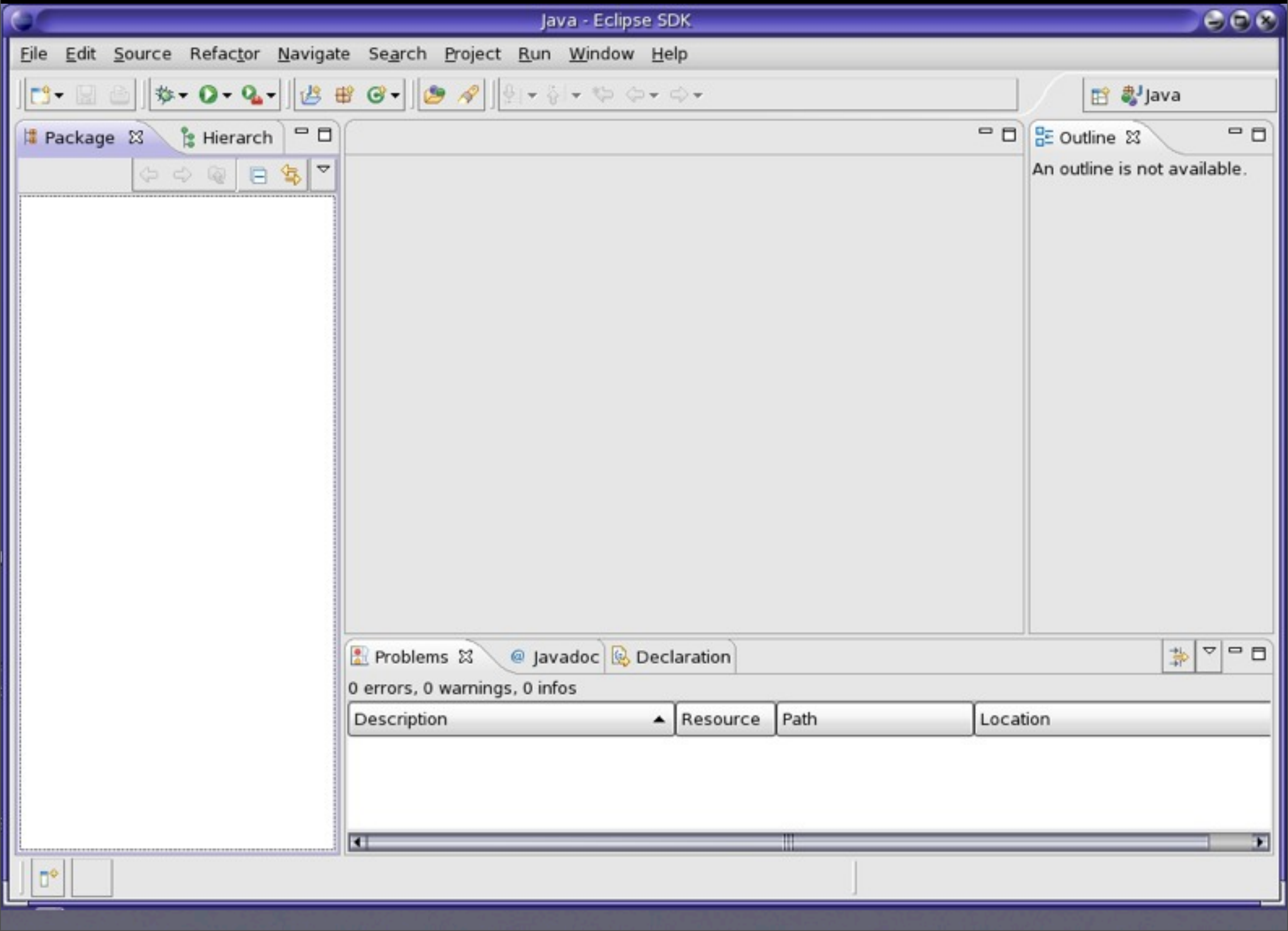
- Eclipse is pre-installed on lab machines
  - Albeit an older version (3.3.2) from 2008
- Likely missing from desktop or Launch menu
- To start Eclipse, run 'eclipse' from terminal
- The first time you run Eclipse, you may be asked to "Select a Workspace"... the suggested location is fine... click "Use as default" and OK
- **You are strongly encouraged to install Eclipse on your laptop & bring it to lab!**

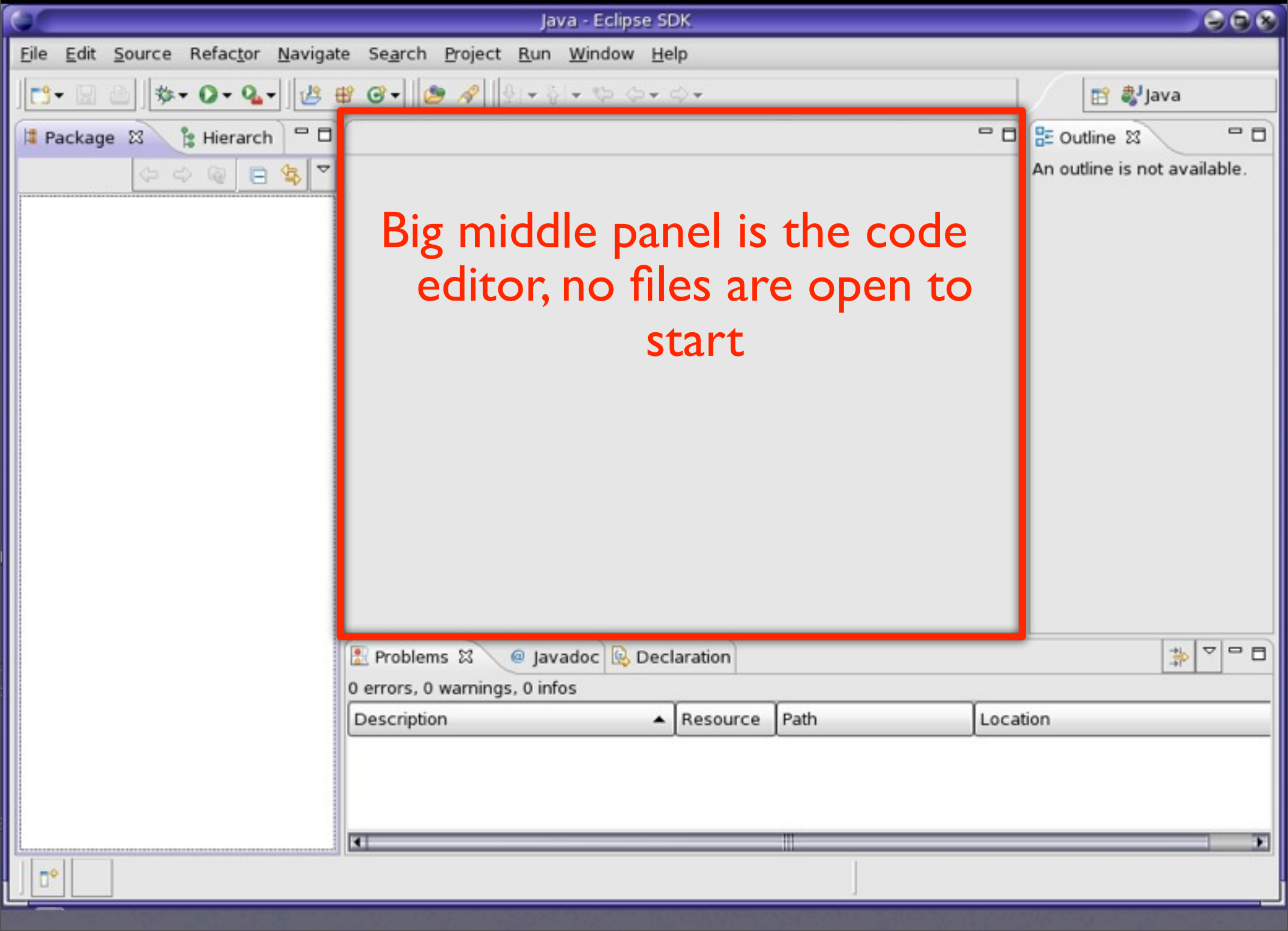
Walkthrough of interface

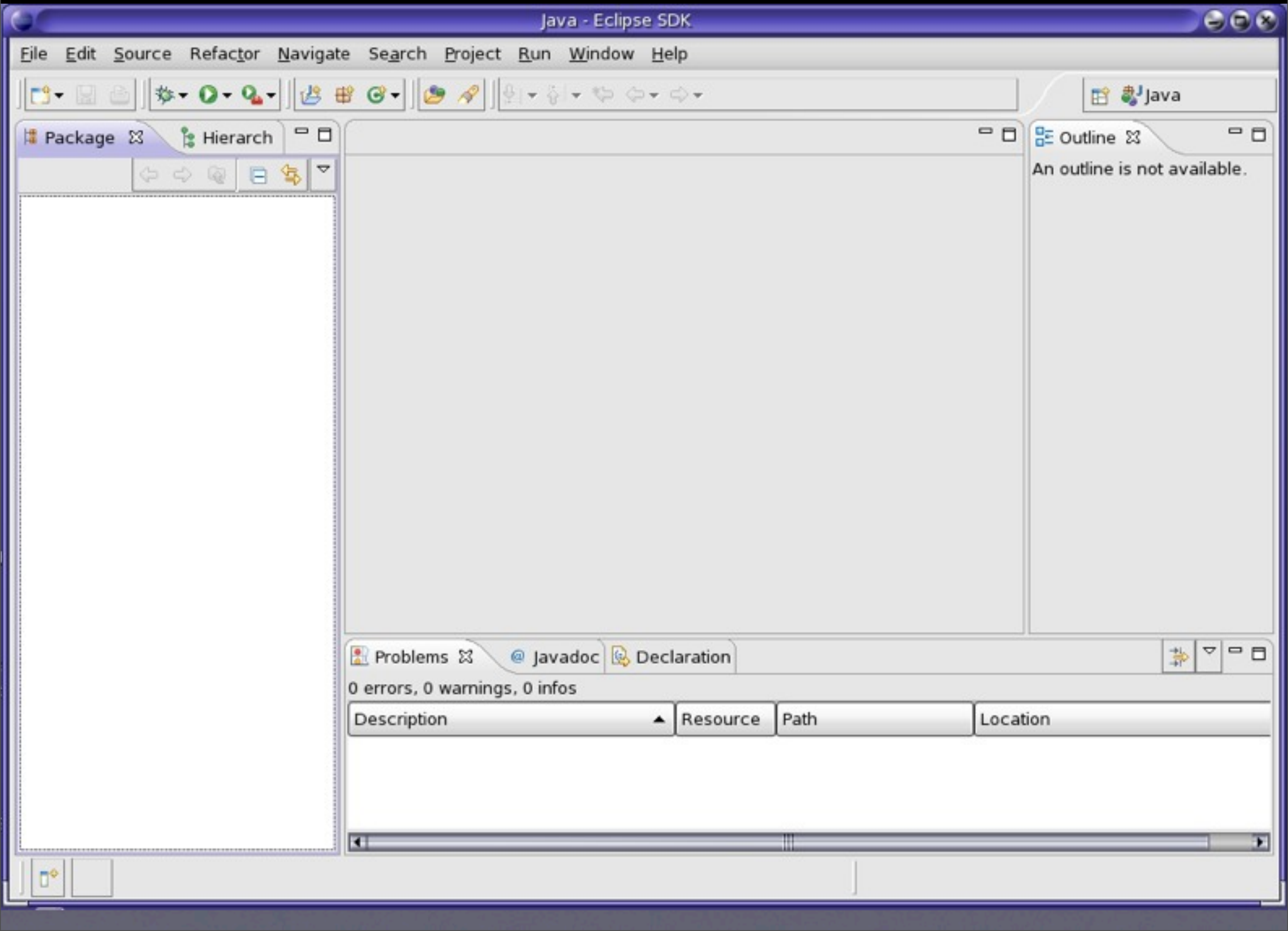
- Here's a snapshot of the initial Eclipse window:

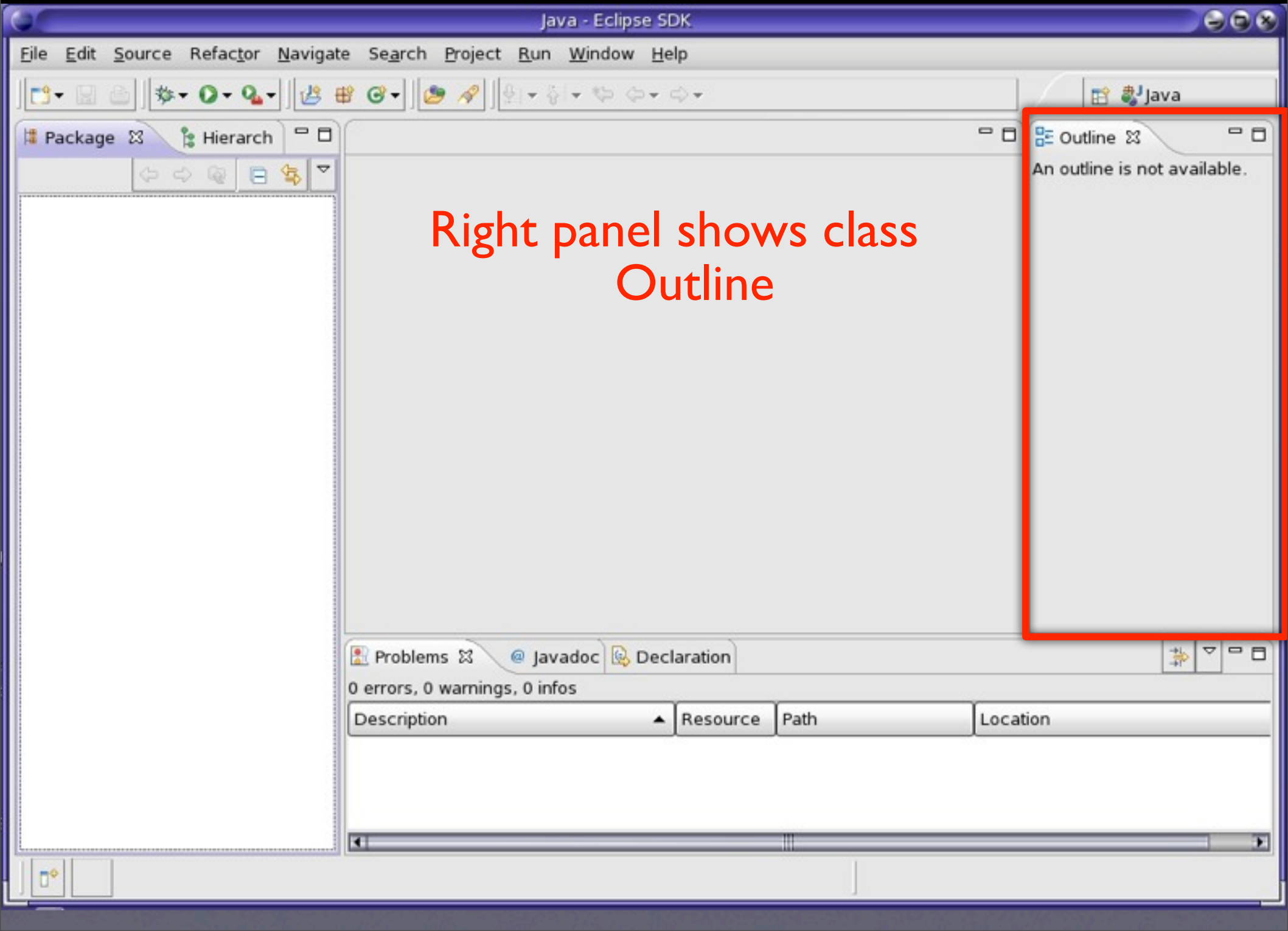

Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package
Hierarch
Outline
An outline is not available.
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location

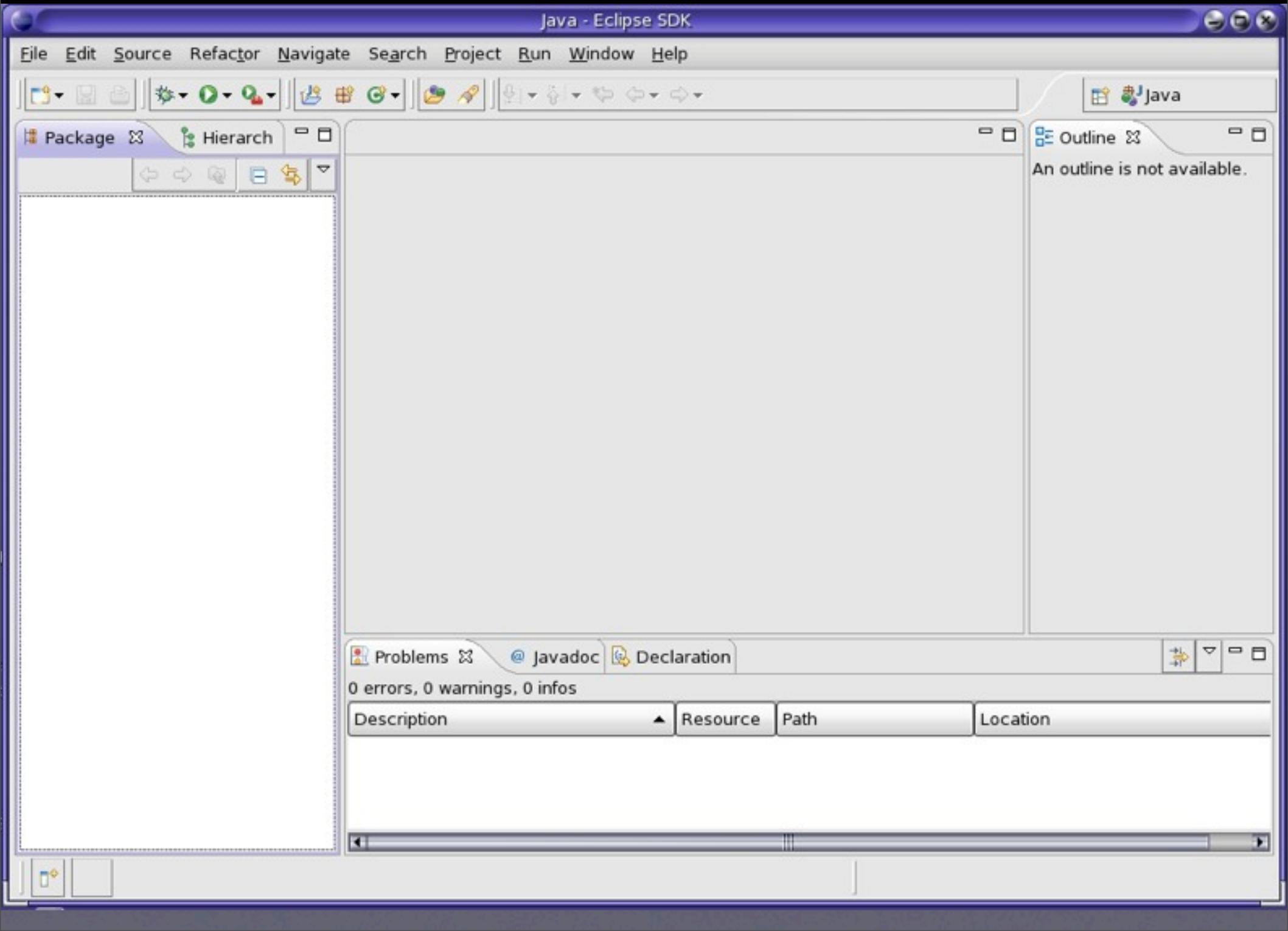

Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package
Hierarch
Outline
An outline is not available.
Left panel shows different views of project, initially Package Explorer & Hierarchy
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location

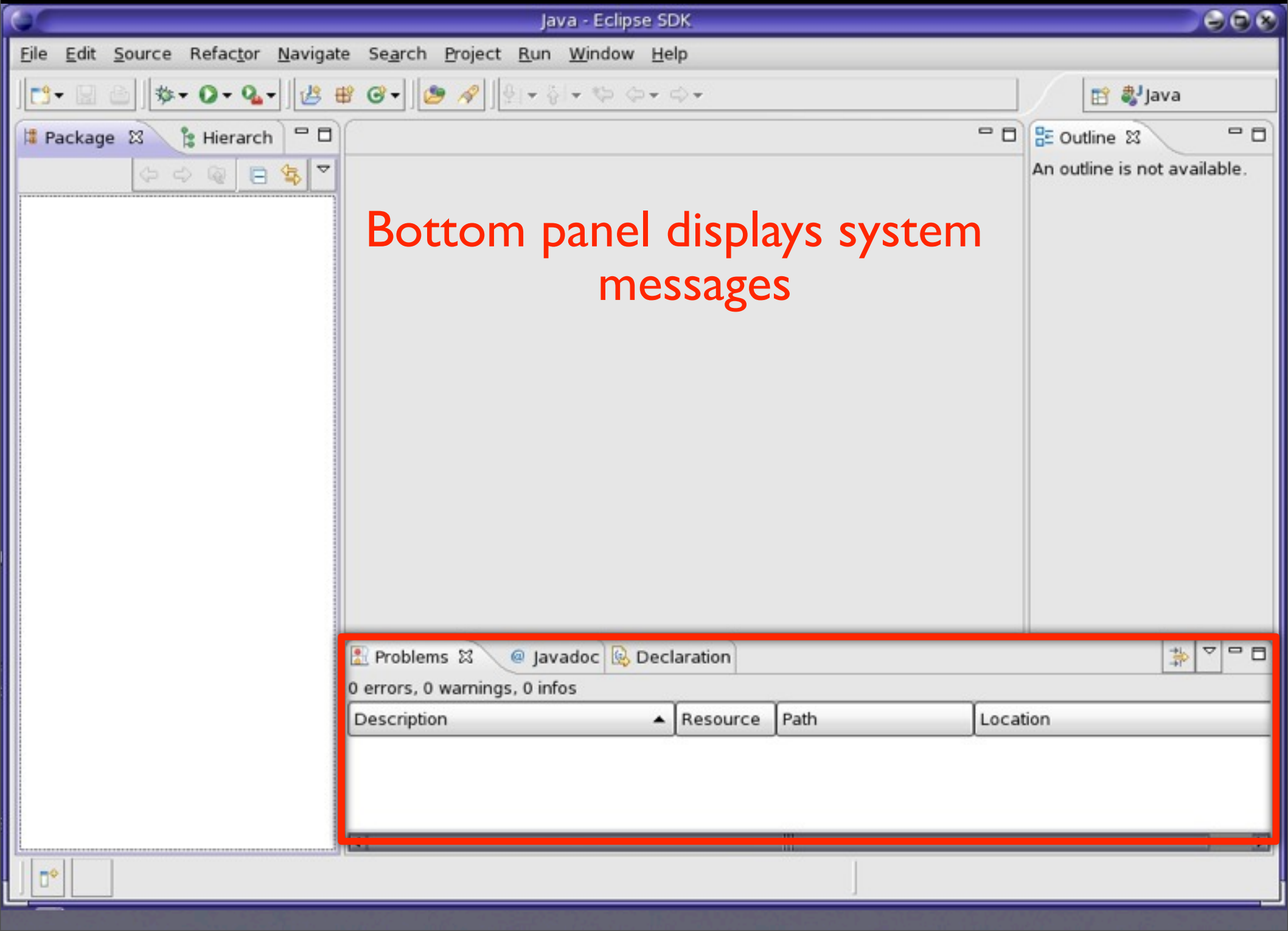

Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package
Hierarch
Outline
An outline is not available.
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location

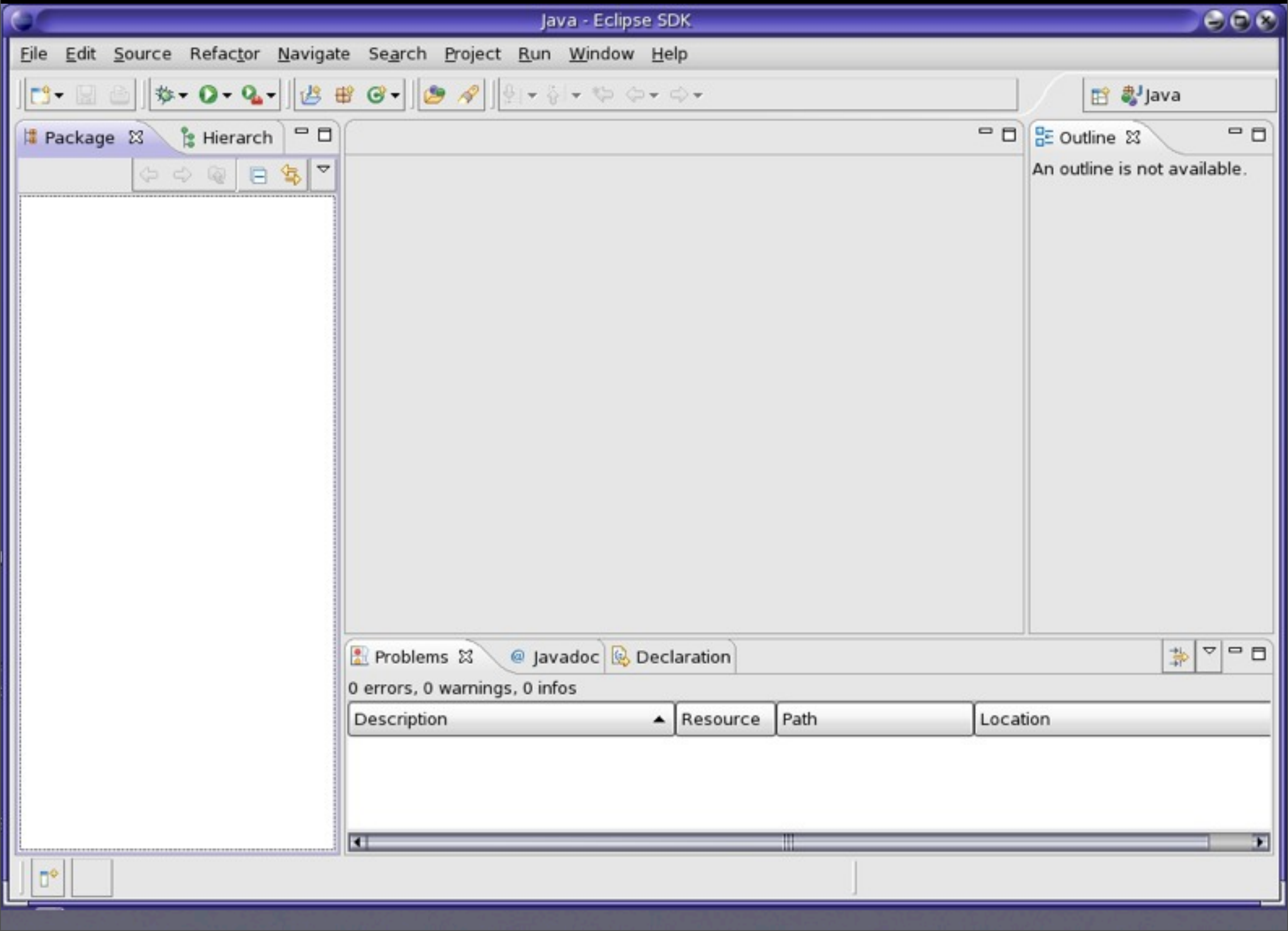

Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package
Hierarch
Outline
An outline is not available.
Big middle panel is the code editor, no files are open to start
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location

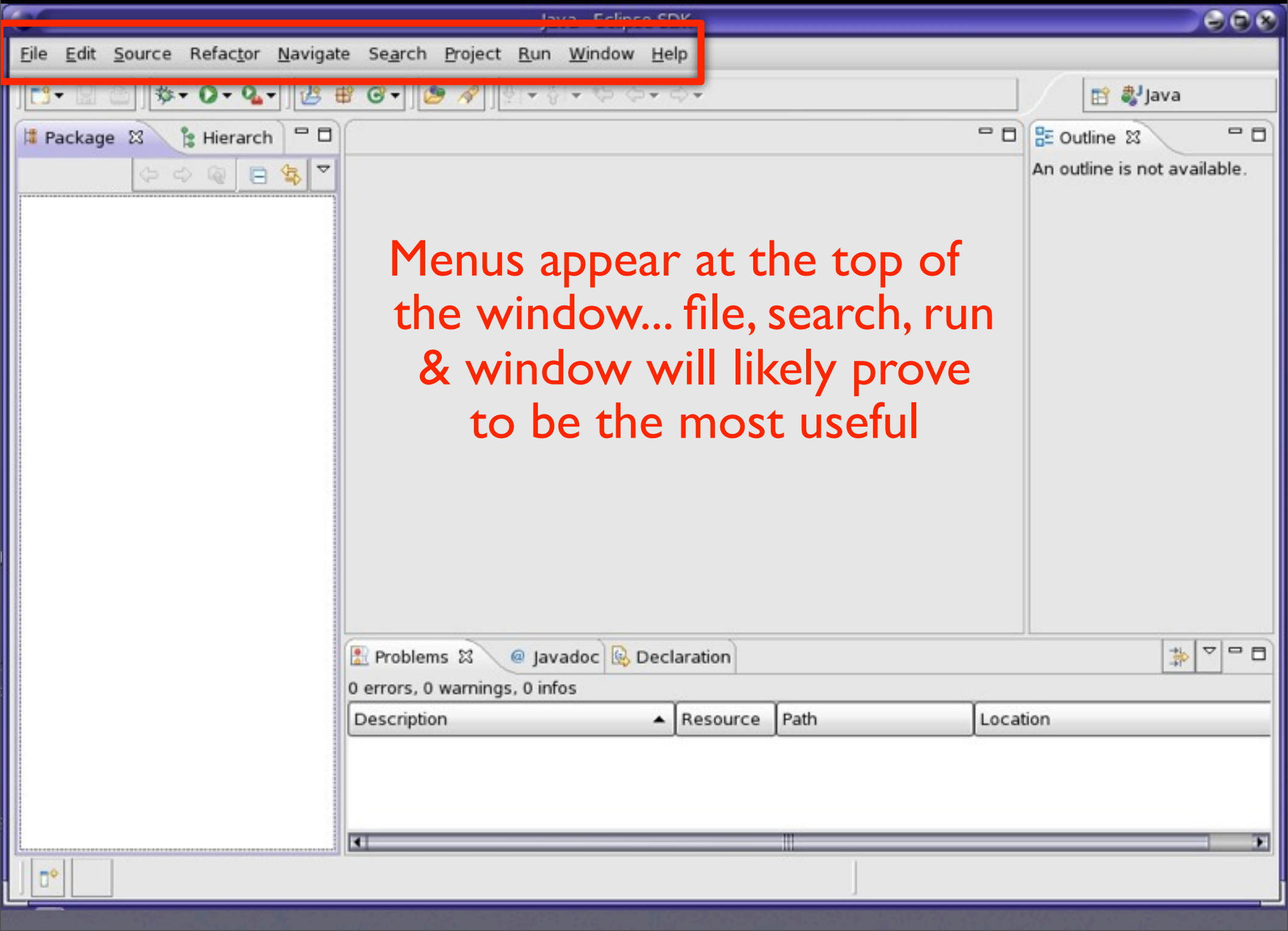

Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package
Hierarch
Outline
An outline is not available.
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location

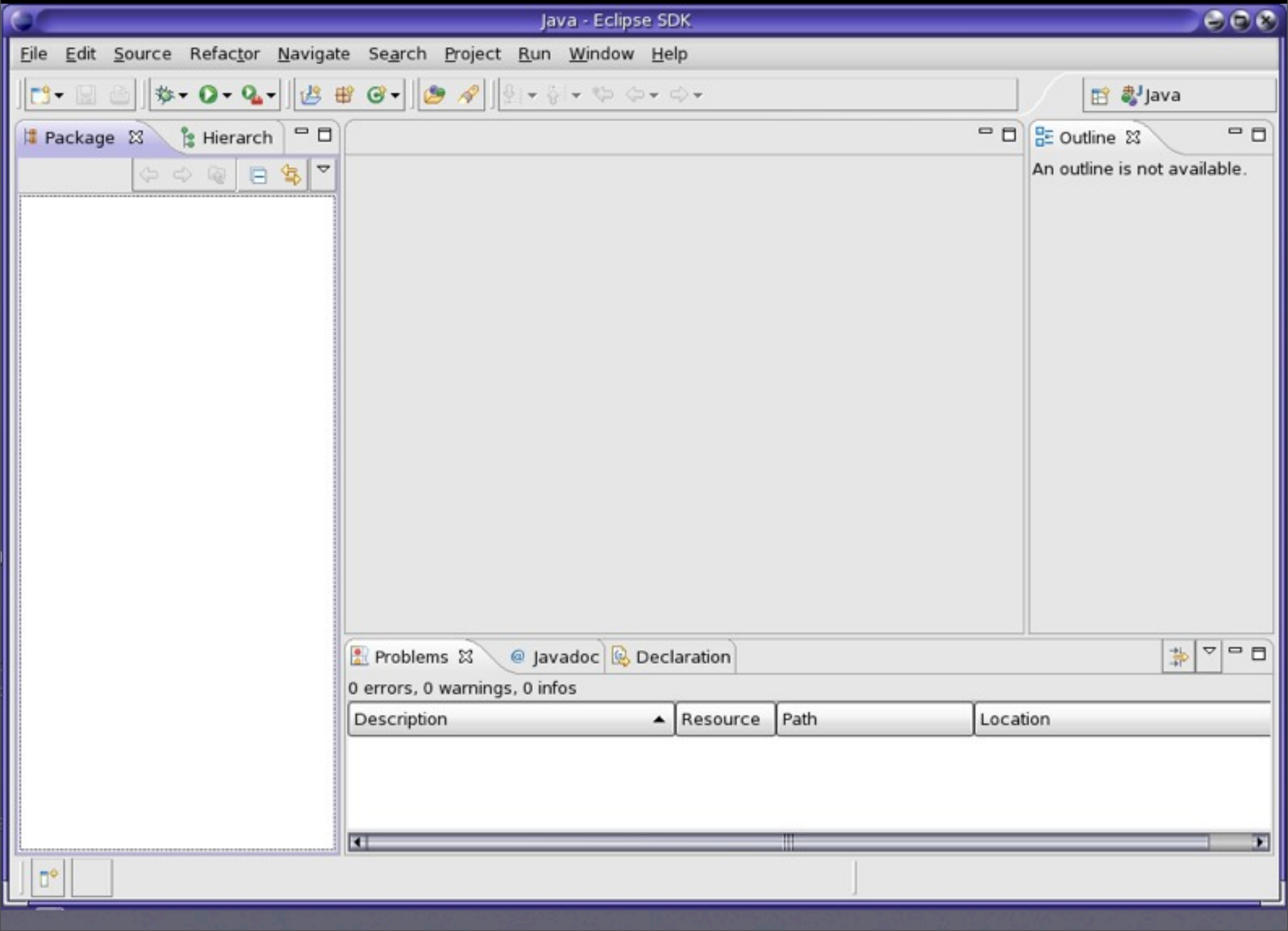

Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package
Hierarch
Outline
An outline is not available.
Right panel shows class Outline
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location


Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package
Hierarch
Outline
An outline is not available.
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location


Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package
Hierarch
Outline
An outline is not available.
Bottom panel displays system messages
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location

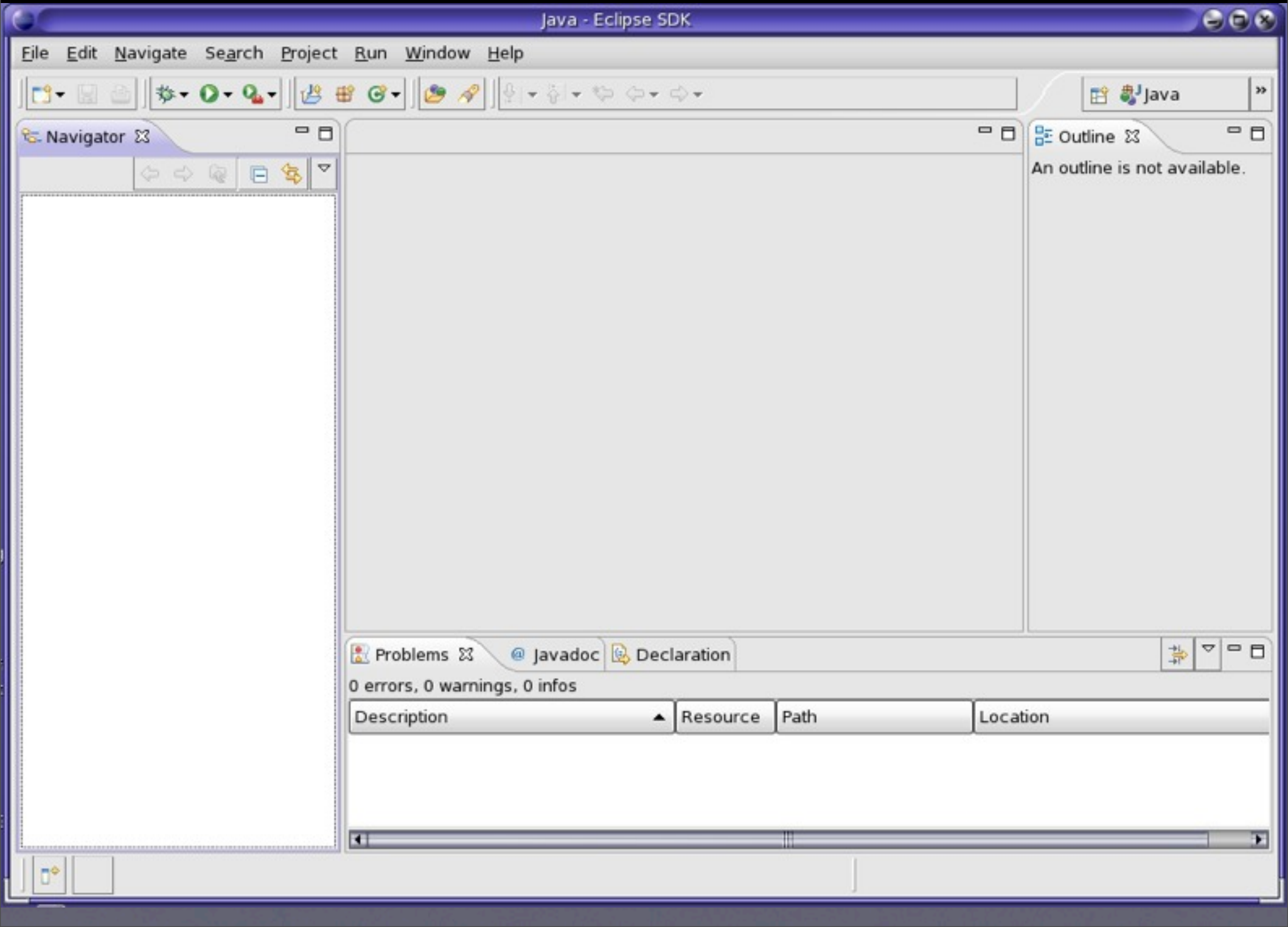

Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package
Hierarch
Outline
An outline is not available.
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location


File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package
Hierarch
Outline
An outline is not available.
Menus appear at the top of the window... file, search, run & window will likely prove to be the most useful
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location


Java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package
Hierarch
Outline
An outline is not available.
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location

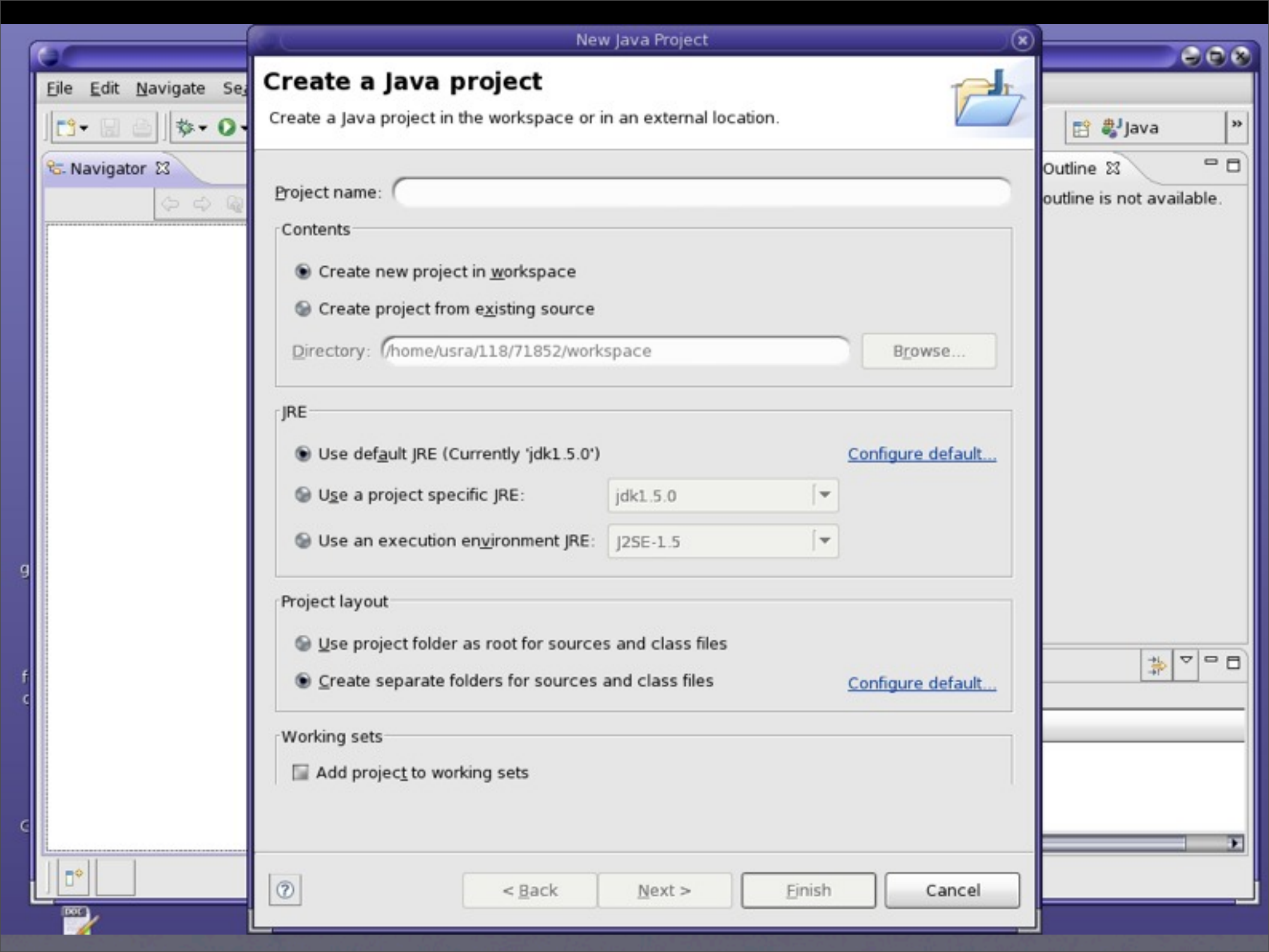
Walkthrough of interface

- I find the Navigator view to be the most helpful, far more useful than Package Explorer or Hierarchy
- Close these other views by clicking on the X for each in the right panel
- Open the Navigator view by clicking on Window/ Show View/Navigator
- Your Eclipse window should now look like this:


Java - Eclipse SDK
File Edit Navigate Search Project Run Window Help
Java
Navigator
Outline
An outline is not available.
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location

Creating a new project

- Click on File/New/Java Project
- Enter as project name: HelloWorldApp
- We have the option to select from different versions of Java if available on the system (everyone should be using at least Java 1.6 by now)
- Let Eclipse manage folders for source code & class files (default option on new versions)
  - On lab machines, in New Java Project dialog, under Project Layout, be sure to select "Create separate source and output folders"
- The new project dialog looks like this:

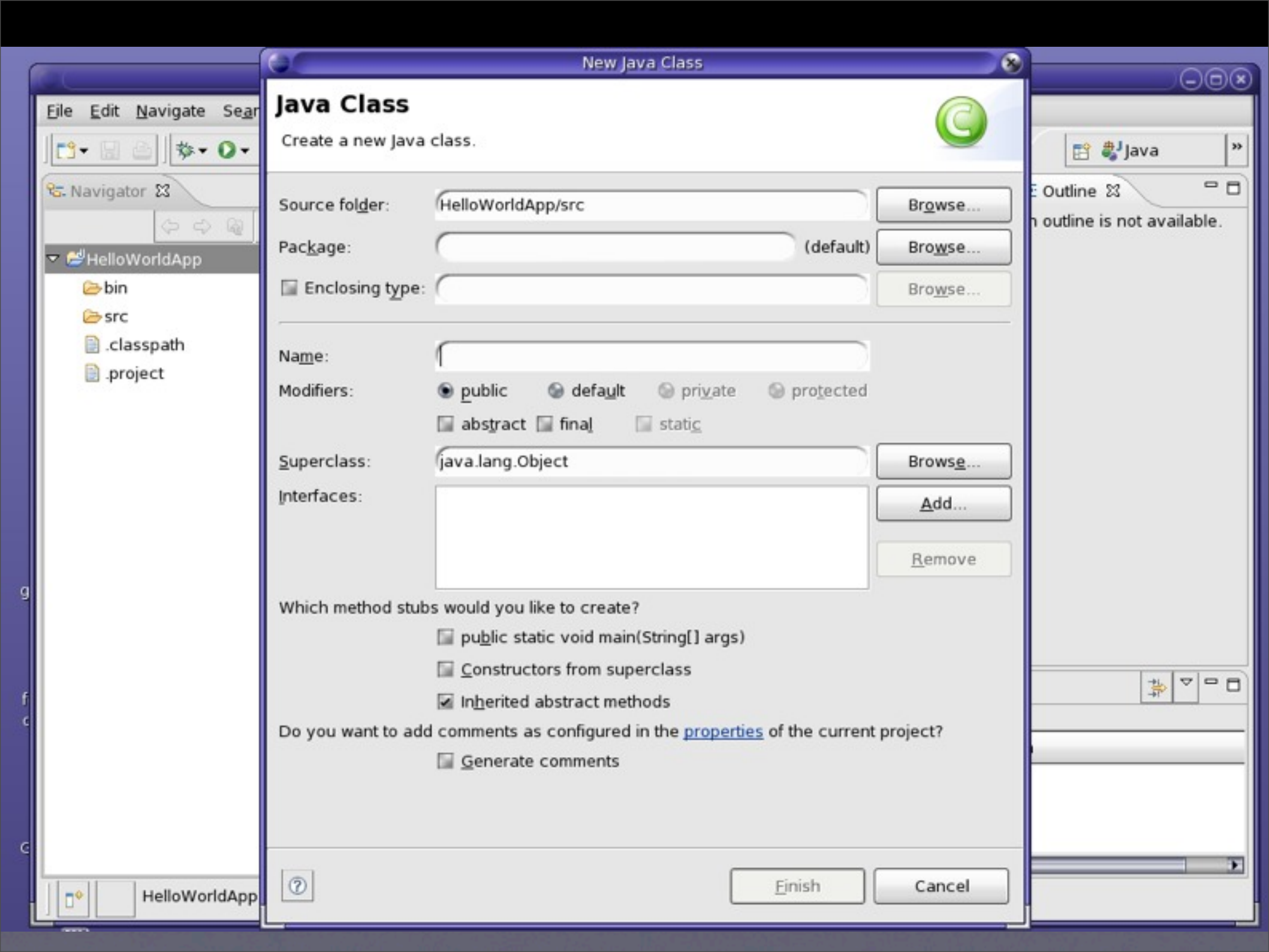

New Java Project
Create a Java project
Create a Java project in the workspace or in an external location.
Project name:
Contents
Create new project in workspace
Create project from existing source
Directory: /home/usra/118/71852/workspace
Browse...
JRE
Use default JRE (Currently 'jdk1.5.0')
Configure default...
Use a project specific JRE: jdk1.5.0
Use an execution environment JRE: J2SE-1.5
Project layout
Use project folder as root for sources and class files
Create separate folders for sources and class files
Configure default...
Working sets
Add project to working sets
< Back
Next >
Finish
Cancel
File Edit Navigate
Navigator
Java
Outline
outline is not available.

Creating a new project

- Click on Finish to create the new project
- Click on the arrow next to HelloWorldApp in the Navigator to see the src & bin directories generated by Eclipse

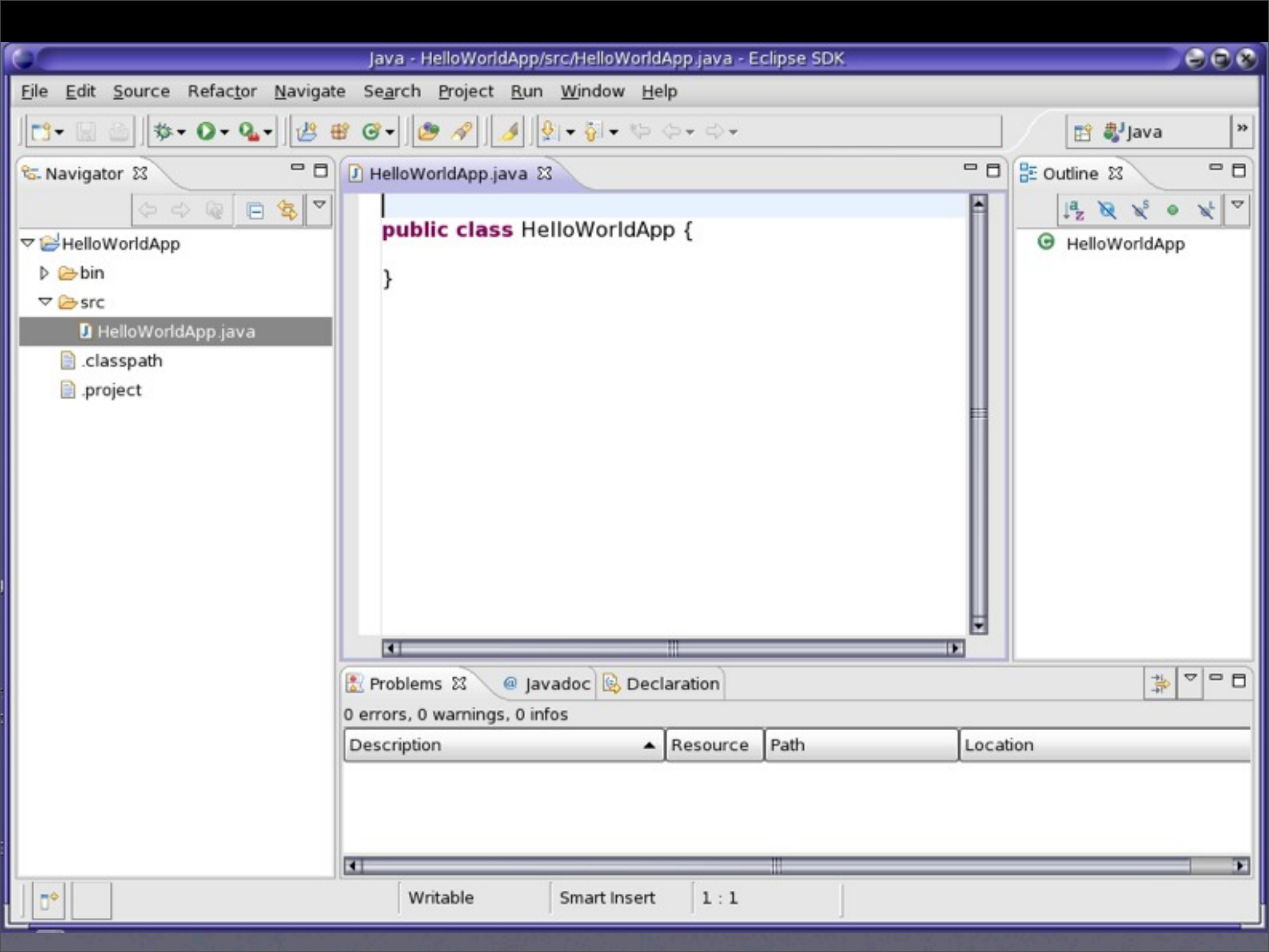
Creating a new class

- Right-click on HelloWorldApp in Navigator
- Select New/Class
- Enter as name: HelloWorldApp
- Ignore default package warnings
- We have some options for class modifiers, superclass, interfaces & generating method stubs, but the defaults are fine for now
- Click Finish to create the new class
- The new class dialog looks like this:


New Java Class
Java Class
Create a new Java class.
Source folder:
HelloWorldApp/src
Browse...
Package:
(default)
Browse...
Enclosing type:
Browse...
Name:
Modifiers:
public
default
private
protected
abstract
final
static
Superclass:
java.lang.Object
Browse...
Interfaces:
Add...
Remove
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
Inherited abstract methods
Do you want to add comments as configured in the properties of the current project?
Generate comments
Finish
Cancel
File
Edit
Navigate
Navigator
HelloWorldApp
bin
src
.classpath
.project
Java
Outline
outline is not available.

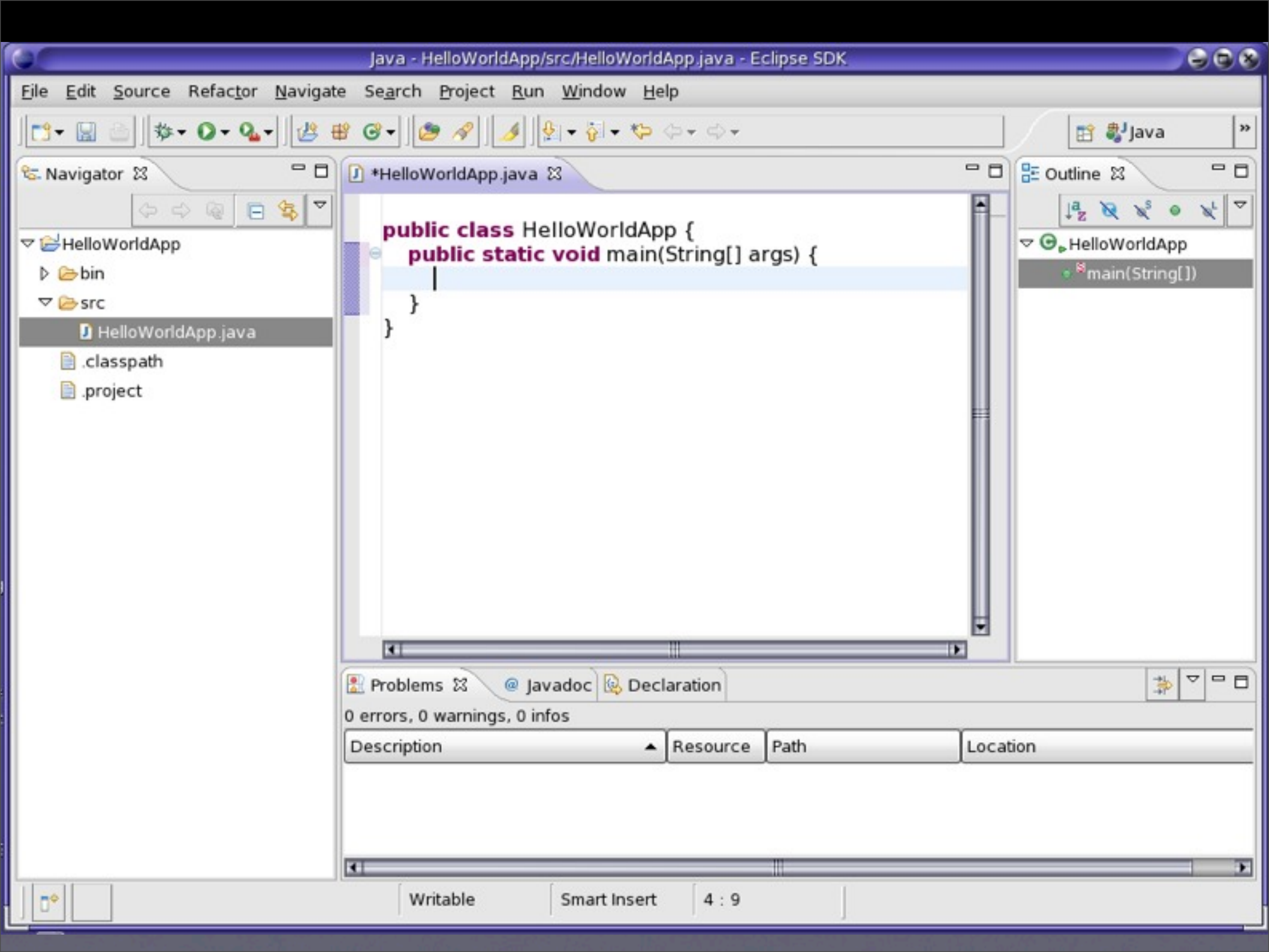
Entering code

- With the shell of a new class automatically generated by Eclipse, we can start filling in the pieces...


Java - HelloWorldApp/src/HelloWorldApp.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Navigator
HelloWorldApp
bin
src
HelloWorldApp.java
.classpath
.project
HelloWorldApp.java
public class HelloWorldApp {
}
Outline
HelloWorldApp
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location
Writable
Smart Insert
1 : 1

Entering code

- Inside the class definiton, hit Tab & start typing:
  `public static void main(`
- Notice how Eclipse automatically inserts a close parenthesis & adds main to the Outline
- Continue inside the parentheses with:
  `String[] args`
  - Eclipse inserted a close bracket too
- Type an open curly brace & hit Enter... Eclipse auto-indents & adds a close curly brace
- Your window should now look like this:

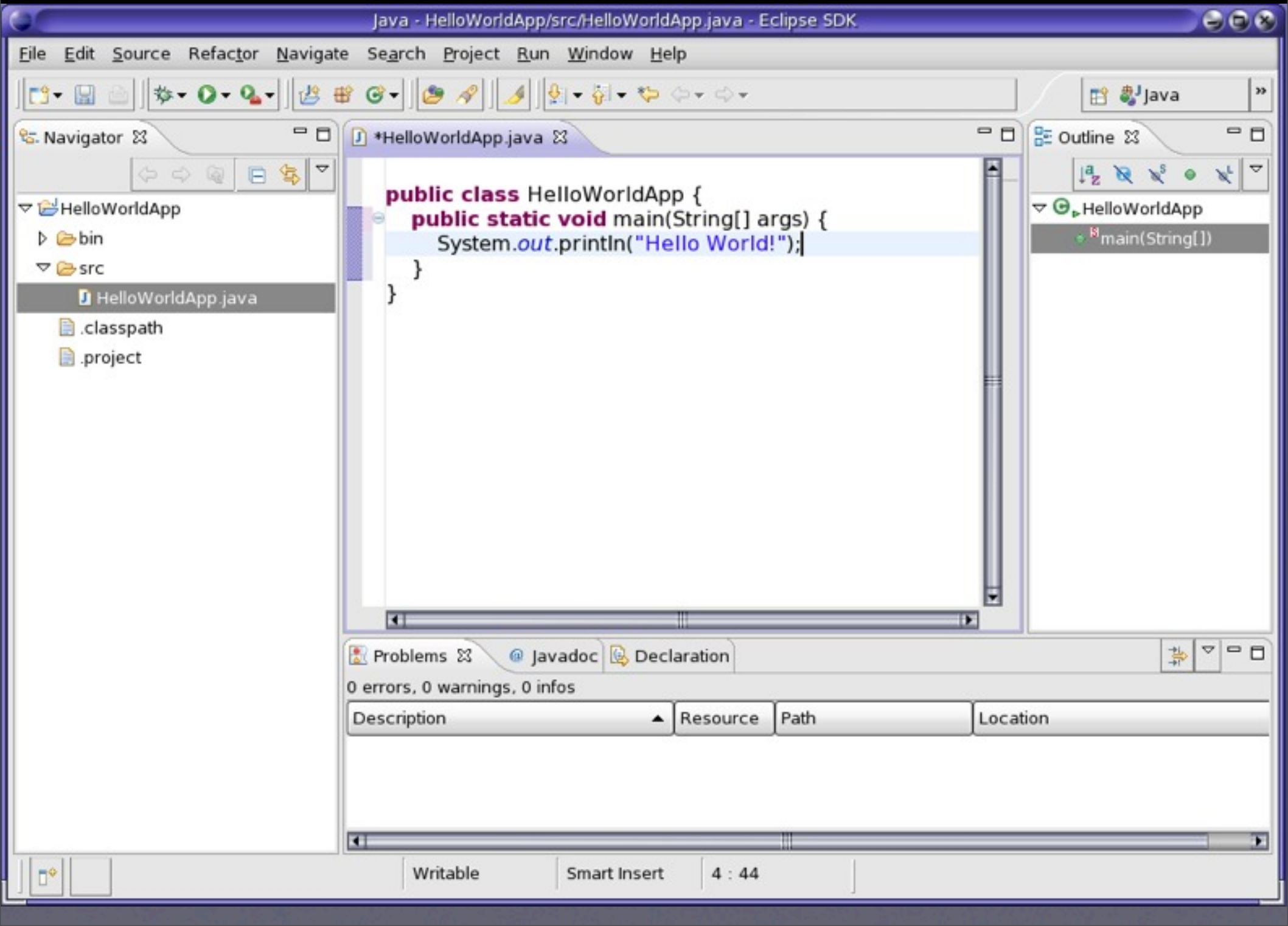

Java - HelloWorldApp/src/HelloWorldApp.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Navigator
HelloWorldApp
bin
src
HelloWorldApp.java
.classpath
.project
*HelloWorldApp.java
public class HelloWorldApp {
public static void main(String[] args) {
}
}
Outline
HelloWorldApp
main(String[])
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location
Writable
Smart Insert
4 : 9

Entering code

- Now start typing `System.`
- See the friendly help menu listing lots of options to choose from to complete the method call
- In this case, we want the `out` PrintStream, so select it or just continue typing
- Again, typing a period after `out` opens a new list of suggested methods
- We'll select `println`, Eclipse will automatically add the parentheses, and if this method had any parameters, it would generate those too

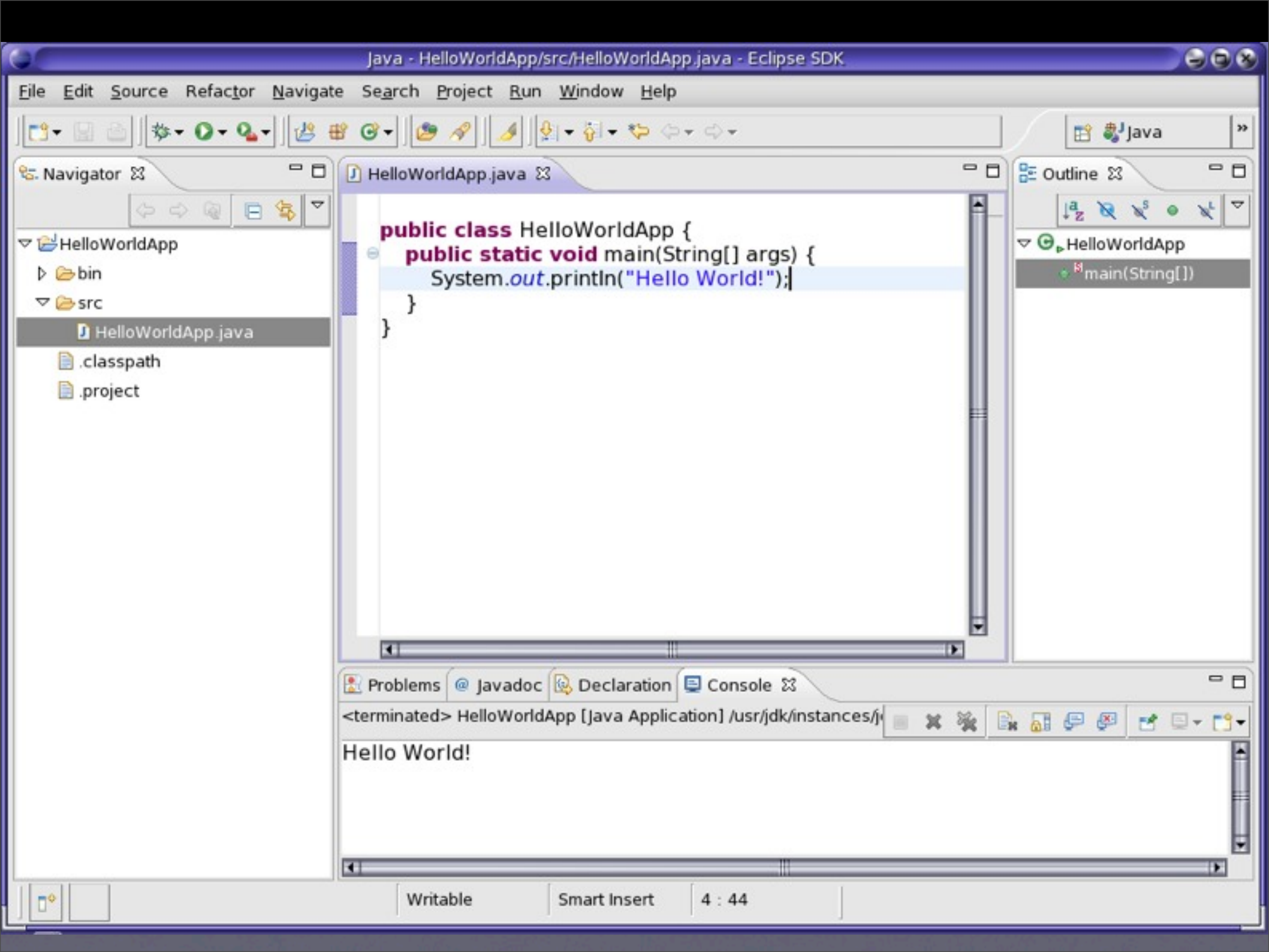
Entering code

- Type “Hello World!” in double-quotes inside the parentheses for `System.out.println()`
- Add a semi-colon to the end of the line
- Your window should now look like this:


Java - HelloWorldApp/src/HelloWorldApp.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Navigator
HelloWorldApp
bin
src
HelloWorldApp.java
.classpath
.project
*HelloWorldApp.java
public class HelloWorldApp {
public static void main(String[] args) {
System.out.println("Hello World!");
}
}
Outline
HelloWorldApp
main(String[])
Problems
Javadoc
Declaration
0 errors, 0 warnings, 0 infos
Description
Resource
Path
Location
Writable
Smart Insert
4 : 44

Running a Java program

- Now we're ready to test out our darling program
- We can run the Java application directly inside of Eclipse, no need to invoke javac from the console
- Select Run/Run As/Java Application
- You will be prompted to save (always a good idea)
- Just like calling javac from the terminal, output & errors displayed in console message panel
- Should look something like this:


Java - HelloWorldApp/src/HelloWorldApp.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Navigator
HelloWorldApp
bin
src
HelloWorldApp.java
.classpath
.project
HelloWorldApp.java
public class HelloWorldApp {
public static void main(String[] args) {
System.out.println("Hello World!");
}
}
Outline
HelloWorldApp
main(String[])
Problems
Javadoc
Declaration
Console
<terminated> HelloWorldApp [Java Application] /usr/jdk/instances/j
Hello World!
Writable
Smart Insert
4 : 44

Other helpful features

- Eclipse can automatically detect & fix many common errors
- To test this, change `println` to `println2`
- A light bulb/red X appears to the left of the line
- Clicking this icon brings up a list of suggested fixes, including changing back to `println`

Other helpful features

- Notice that when Eclipse detects an error or errors in your code, a red box will appear at the top along the right side of your code
- Red dashes indicate the lines of code containing these errors
- There are similar yellow dashes for warnings
- Fixing the error replaces the light bulb/red X icon with a blue dot, which will disappear the next time the program is successfully run

Other helpful features

- For another example, add these lines of code:
```
String temp = "hello world";
temp.indexOf(3, "world");
```
- Eclipse will detect an error in the second line & suggest some possible fix actions
- Multiple versions of `String.indexOf()` have different sets of parameters
- Eclipse suggests removing arguments to match `indexOf(String)` or `indexOf(int)`, or swapping the order of the arguments, which is what we want for `indexOf(String, int)`

Other helpful features

- Eclipse can also organize & manage your import statements
- It can automatically generate stubs for constructors & other methods
- You'll pick up some advanced features & plug-ins for Eclipse throughout the semester

Lab exercise

- Most labs include an exercise designed to demonstrate an understanding of the material
- Today, I just want to see that you can run a Java program in Eclipse
  - Enter the "Hello World" code
  - Run it as a Java application
  - Show me the output before you leave the lab
- I always look for comments & will often deduct points for poorly documented code