Tutorial 0.1: Eclipse Tutorial

Tutorial objectives

The current tutorial is a step-by-step guide to learning Java using the Eclipse Editor. It is meant for complete beginners.
It will show you first, how to install java and Eclipse on your own machines.
By the end of this tutorial you will be able to start a new project, write your first Java program and run it. This knowledge will be essential in due course.

Why learn Java with Eclipse?

There are many ways to learn how to program in Java. There are advantages to learning Java using the Eclipse integrated development environment (IDE). Some of these are listed below:

Eclipse provides a number of aids that make writing Java code much quicker and easier than using a text editor. This means that you can spend more time learning Java, and less time typing and looking up documentation. Eclipse is constantly assisting while coding. The biggest advantages in eclipse are therefore:

1. **Code Completion**, instead of digging through documentation you should be able to tab your way through methods and save yourself a lot of writing
2. **Refactoring**, Global Find and Replace is no replacement for good refactoring support, that starts with renaming functions, variables, classes, ...
3. **Syntax Checking**, helping you out with writing correct code while you type

Note: The right version of Java and Eclipse are ***already installed on your lab machines***. The following instructions are for installing them on your own machines.

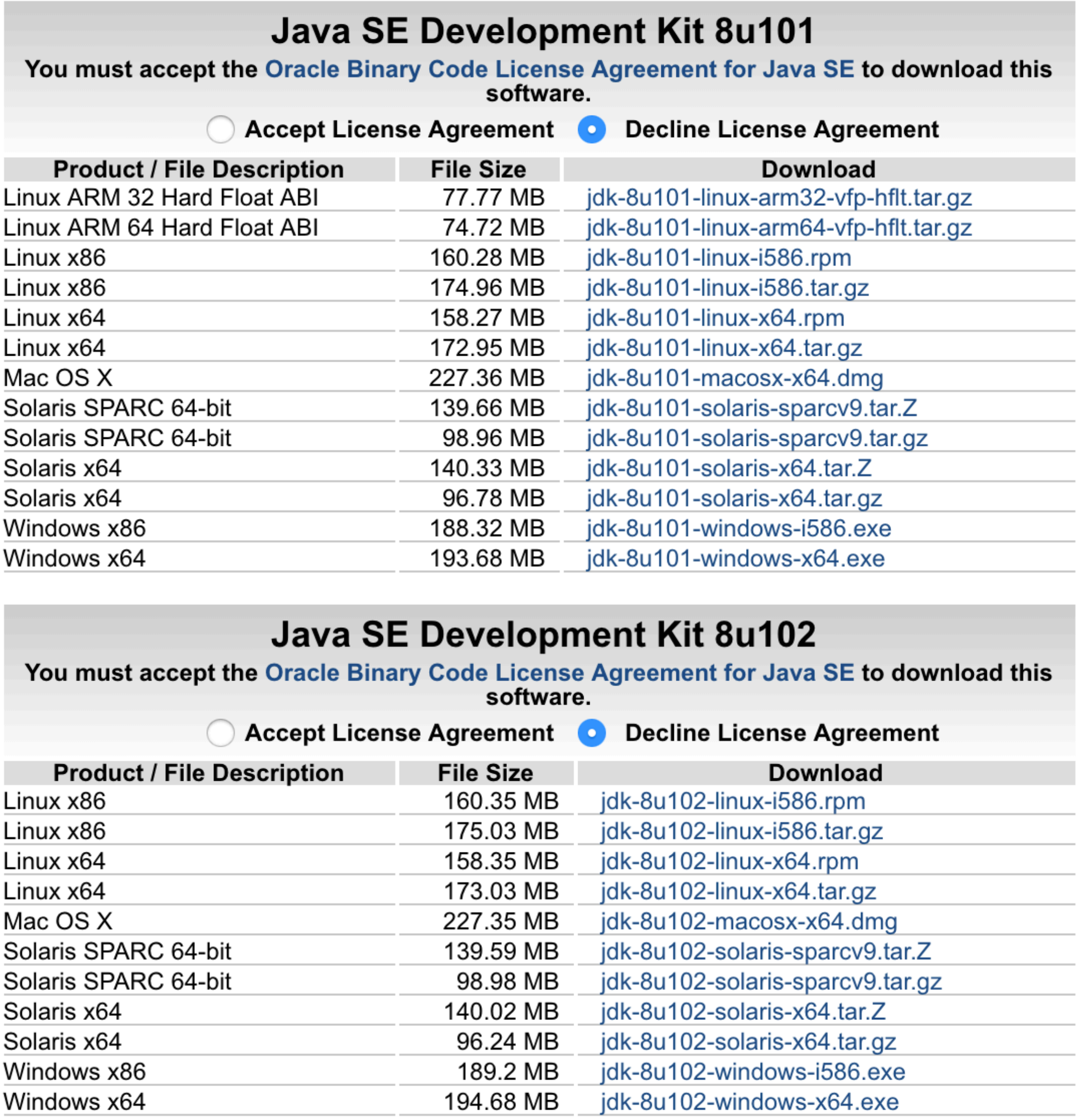
Before being able to use Eclipse, you need first to install Java.

Downloading and Installing Java

1. Visit the java download page: http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
2. Click on "**Accept License Agreement**" (Figure 1).
3. Scroll down until you find Java SE Development Kit 8u102, and download JDK (Figure 1).
4. Double-click the downloaded file (e.g. jdk-8u102-macosx-x64.dmg) and follow the instructions
5. When done check your java installation by opening a Terminal window (on mac) or a Command prompt (on Windows) and type: ***Java –version.*** The output should be similar to (Figure 2)

Note: Please note that you should select the java version corresponding to your operating system, e.g, Windows, Linux, etc.

The ‘*jdk-8u102-macosx-x64.dmg*’ file is for MacOS.



Java SE Development Kit 8u101

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

○ Accept License Agreement ● Decline License Agreement

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.77 MB	jdk-8u101-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.72 MB	jdk-8u101-linux-arm64-vfp-hflt.tar.gz
Linux x86	160.28 MB	jdk-8u101-linux-i586.rpm
Linux x86	174.96 MB	jdk-8u101-linux-i586.tar.gz
Linux x64	158.27 MB	jdk-8u101-linux-x64.rpm
Linux x64	172.95 MB	jdk-8u101-linux-x64.tar.gz
Mac OS X	227.36 MB	jdk-8u101-macosx-x64.dmg
Solaris SPARC 64-bit	139.66 MB	jdk-8u101-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	98.96 MB	jdk-8u101-solaris-sparcv9.tar.gz
Solaris x64	140.33 MB	jdk-8u101-solaris-x64.tar.Z
Solaris x64	96.78 MB	jdk-8u101-solaris-x64.tar.gz
Windows x86	188.32 MB	jdk-8u101-windows-i586.exe
Windows x64	193.68 MB	jdk-8u101-windows-x64.exe

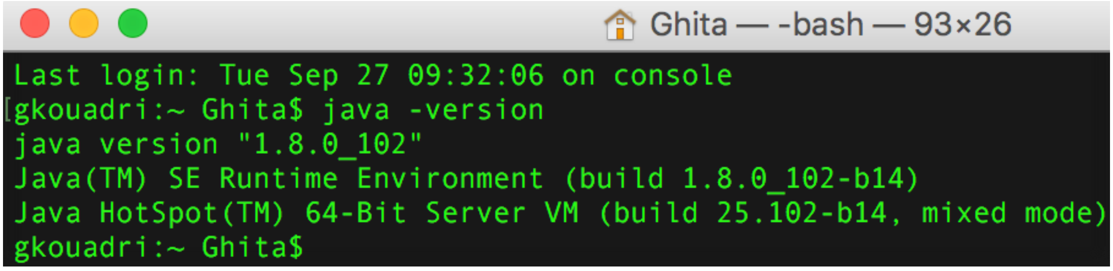
Java SE Development Kit 8u102

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

○ Accept License Agreement ● Decline License Agreement

Product / File Description	File Size	Download
Linux x86	160.35 MB	jdk-8u102-linux-i586.rpm
Linux x86	175.03 MB	jdk-8u102-linux-i586.tar.gz
Linux x64	158.35 MB	jdk-8u102-linux-x64.rpm
Linux x64	173.03 MB	jdk-8u102-linux-x64.tar.gz
Mac OS X	227.35 MB	jdk-8u102-macosx-x64.dmg
Solaris SPARC 64-bit	139.59 MB	jdk-8u102-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	98.98 MB	jdk-8u102-solaris-sparcv9.tar.gz
Solaris x64	140.02 MB	jdk-8u102-solaris-x64.tar.Z
Solaris x64	96.24 MB	jdk-8u102-solaris-x64.tar.gz
Windows x86	189.2 MB	jdk-8u102-windows-i586.exe
Windows x64	194.68 MB	jdk-8u102-windows-x64.exe

Figure 1: JDK environment download page



```
Last login: Tue Sep 27 09:32:06 on console
gkouadri:~ Ghita$ java -version
java version "1.8.0_102"
Java(TM) SE Runtime Environment (build 1.8.0_102-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.102-b14, mixed mode)
gkouadri:~ Ghita$
```

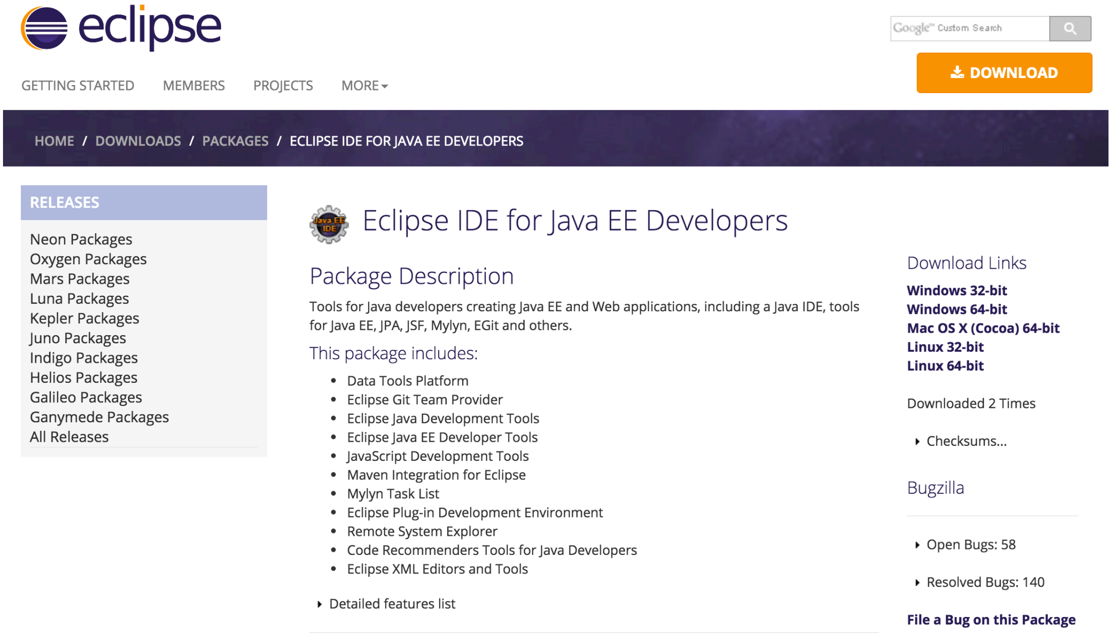
Figure 2: Checking Java installation

Downloading and Installing Eclipse IDE (Integrated Development Editor)

1. Visit the Eclipse Download Page: http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-

developers/neon1. This year we are using Eclipse for EE Developers, Neon version.

2. Download the version of Eclipse corresponding to your operating system (Figure 3).
3. Double click on the downloaded file and follow the instructions on the screen.



Eclipse IDE for Java EE Developers

Package Description

Tools for Java developers creating Java EE and Web applications, including a Java IDE, tools for Java EE, JPA, JSF, Mylyn, EGit and others.

This package includes:

- Data Tools Platform
- Eclipse Git Team Provider
- Eclipse Java Development Tools
- Eclipse Java EE Developer Tools
- JavaScript Development Tools
- Maven Integration for Eclipse
- Mylyn Task List
- Eclipse Plug-in Development Environment
- Remote System Explorer
- Code Recommenders Tools for Java Developers
- Eclipse XML Editors and Tools

Detailed features list

Downloaded 2 Times

Open Bugs: 58

Resolved Bugs: 140

Figure 3: Eclipse download site

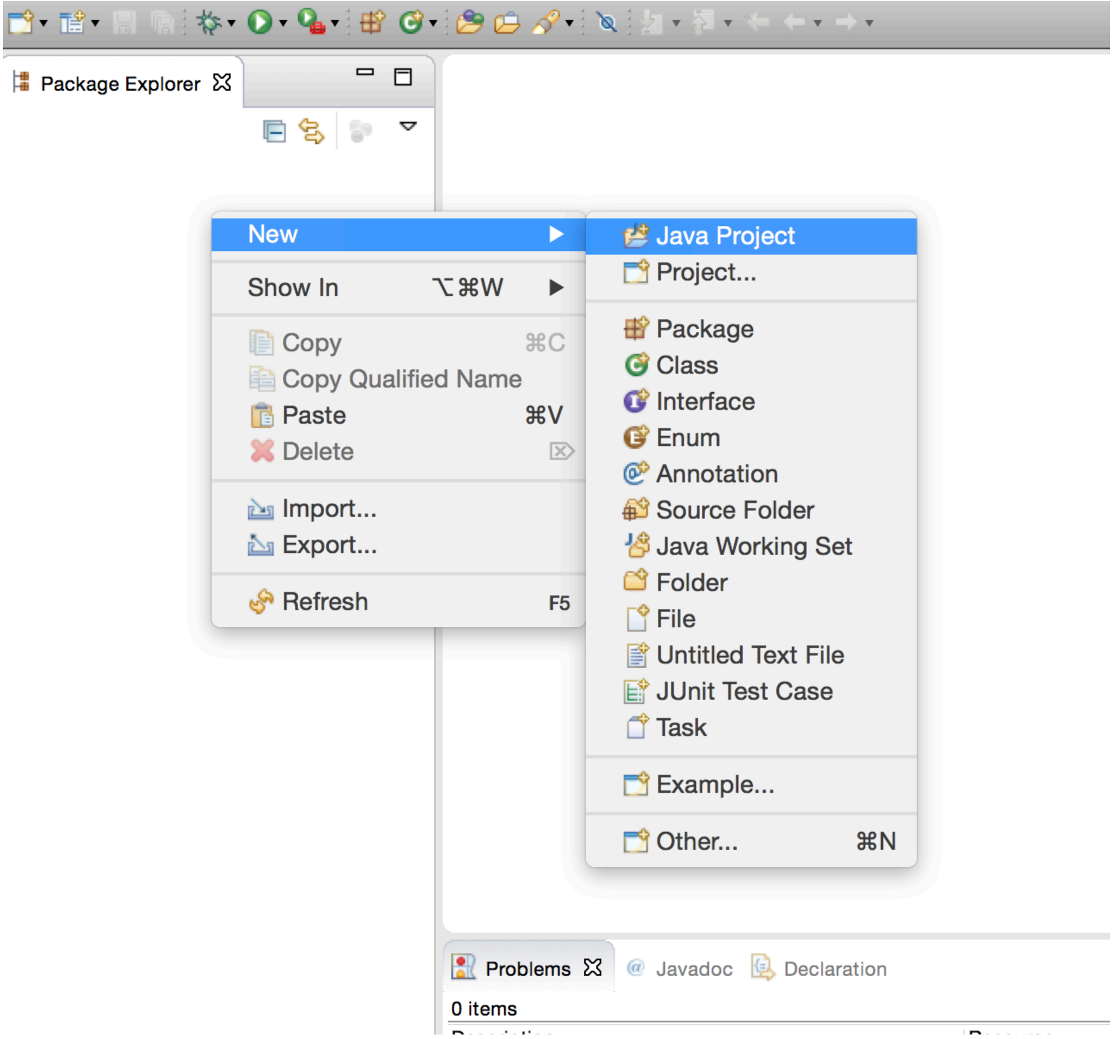
4. Once Eclipse installed. Make sure it is set-up to use the right version of Java virtual machine (JVM). For that follow the instructions on the following page: https://wiki.eclipse.org/Eclipse.ini

By now, your should have everything you need to run your first java program.

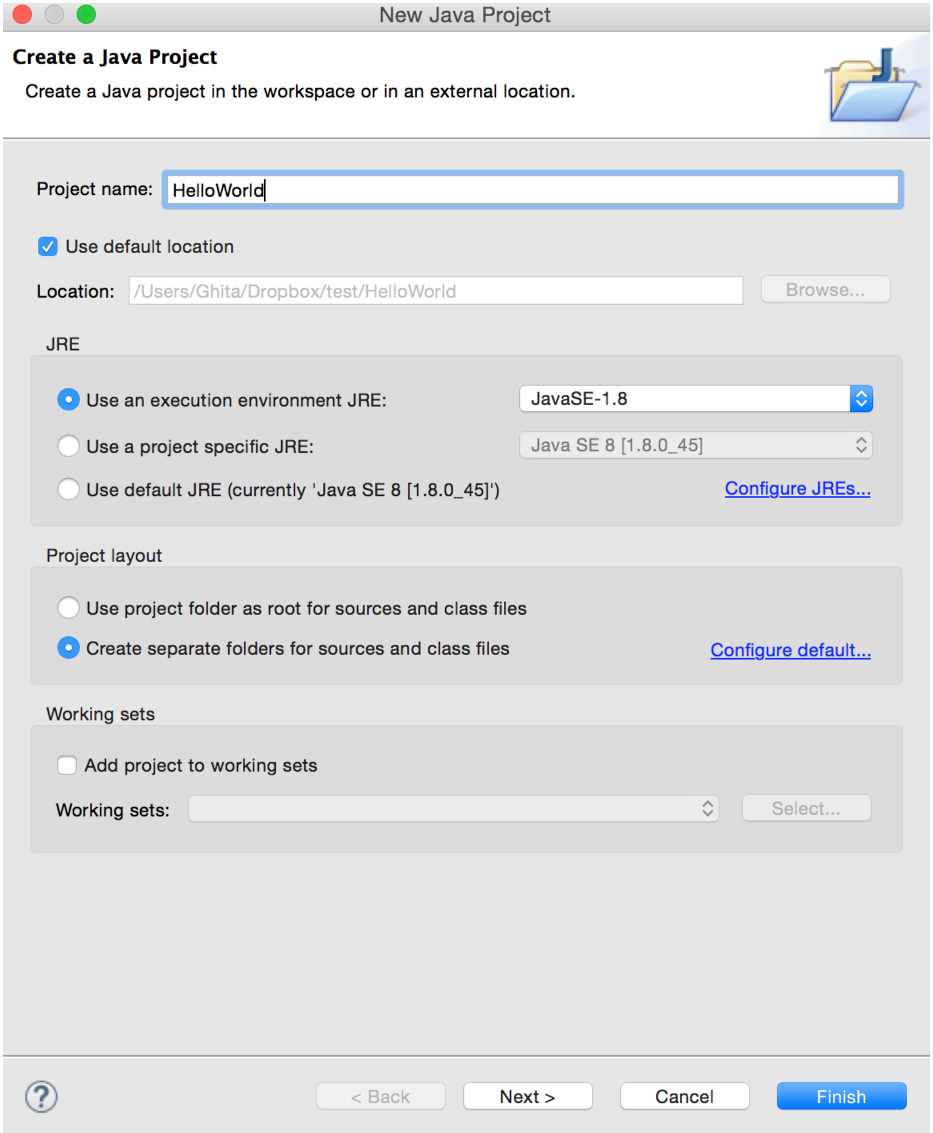
Exercise 1 First Java application in Eclipse

We're going to start off by coding the traditional Java Hello World program!

First, we need to create a Java project before we can begin coding. A Java project is just a bunch of Java files and other files needed for your program to work. Every time you want to work on something new, I recommend creating a new project for it, as it's a good habit to have. So, if you haven't already, go start up Eclipse, and hit ok when the Workspace Launcher box appears.



Right click on the left side panel, go to new, and select Java Project. (If you don't see that immediately, just hit Project... and you'll see more project types. Java Project should be at the very top, but if not, go to the folder that says Java, and you should see Java Project in there.)



New Java Project

Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: HelloWorld

☑ Use default location

Location: /Users/Ghita/Dropbox/test/HelloWorld

JRE

● Use an execution environment JRE: JavaSE-1.8

○ Use a project specific JRE: Java SE 8 [1.8.0_45]

○ Use default JRE (currently 'Java SE 8 [1.8.0_45]')

Project layout

○ Use project folder as root for sources and class files

● Create separate folders for sources and class files

Working sets

☐ Add project to working sets

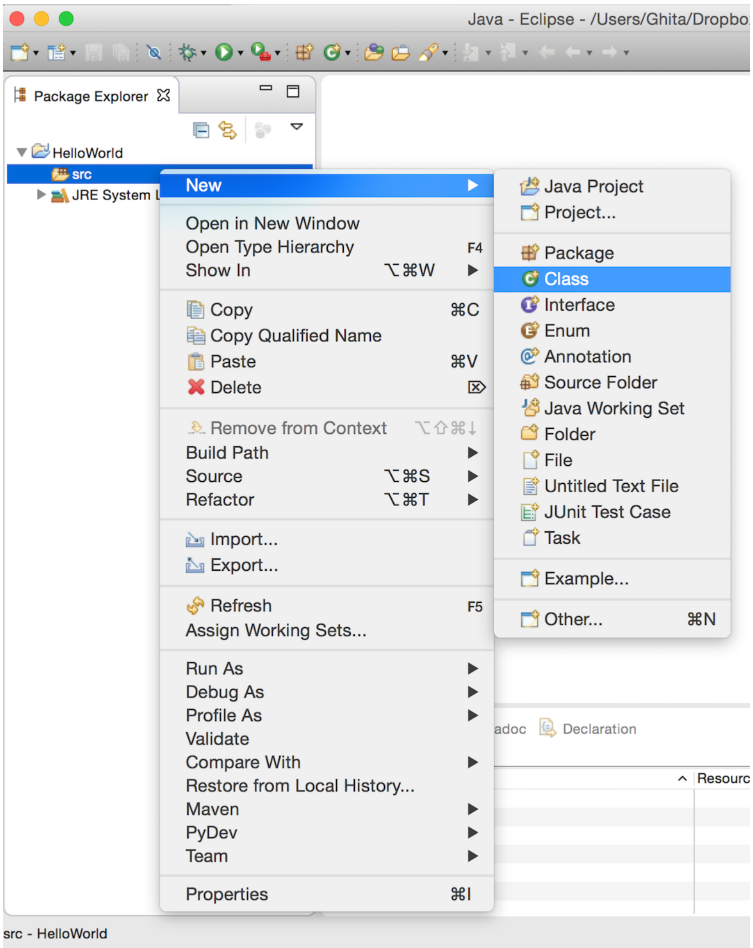
Working sets:

Great! Now we have a bunch of options to set for our new Java Hello World project. All of these options can quickly become overwhelming, so once again we won't go into any details. Name your new project Hello World, leave everything else how it was, and then hit Finish.

The form will close and you should see your new project for your Java Hello World program on the side panel.

Create a new class

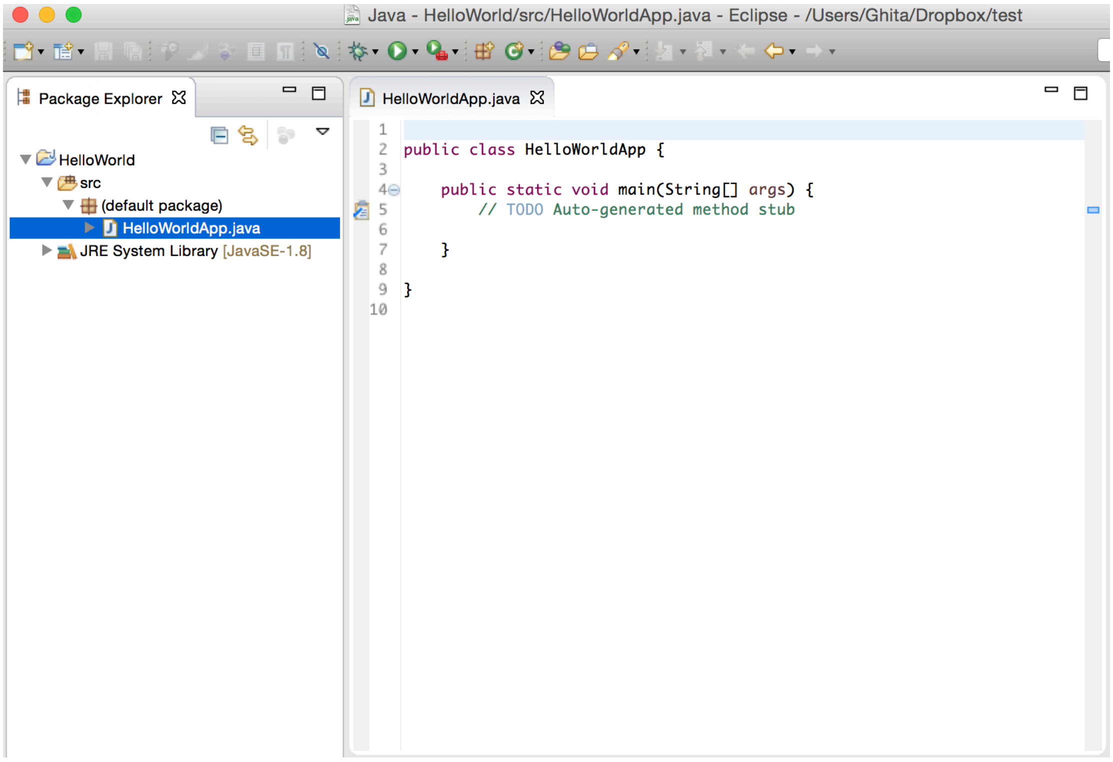
If you expand your new created project by clicking the 'triangle' sign next to the folder, you'll see a folder called *src* and also a *JRE* System Library. Again, don't worry about these things. Right click on the *src* folder, go to *new*, and select *Class*.



Give your new file a name. We're going to call it *HelloWorldApp* (no spaces!) where it asks for the name. Java files should always start with a capital letter as the first letter and should have no spaces, so remember that.

Also, check the box that says *public static void main(String[] args)*, because you'll need that. Then hit *Finish*.

Now you'll see your newly created file, and you'll see that some of the work has been done for you. Let's take a quick look at what it is you're seeing.


Java - HelloWorld/src/HelloWorldApp.java - Eclipse - /Users/Ghita/Dropbox/test
Package Explorer
HelloWorld
src
(default package)
HelloWorldApp.java
JRE System Library [JavaSE-1.8]
HelloWorldApp.java

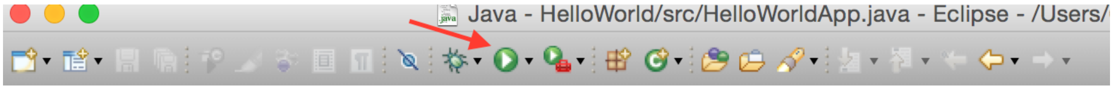
```

public class HelloWorldApp {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

    }

}

```

Add the following line of code inside the main method

System.out.println("Hello World!");

Run your application

In order to run your application, hit the run button (green button with a white triangle inside it).


Java - HelloWorld/src/HelloWorldApp.java - Eclipse - /Users/

Now you should see the message *Hello World!* Displayed in the Eclipse console. Congratulations! You run your first java application.

Exercise 2 A Closer Look at the "Hello World!" Application

Please read the following short tutorial to remind you of the different parts of the *HelloWorldApp* class already discussed in the lecture:

https://docs.oracle.com/javase/tutorial/getStarted/application/

~~~OOO~~~
~~~