

## CPSC 1060/1061: Introduction to Programming in Java

---

### Lab 01 - Creating, Compiling, and Executing Java Programs

#### Introduction

The purpose of this lab is to introduce you to a way of creating and executing Java programs. The Integrated Development Environment (IDE) *Eclipse* will be used to create, compile, and execute Java programs. Eclipse offers a graphical interface, syntax checking, and several other useful features that make creating and maintaining large Java applications easier than it otherwise would be.

#### Lab Objectives

By the end of the lab, you should:

- understand the general steps required to create, compile, and execute Java programs;
- be able to use an IDE (specifically Eclipse) to create and execute Java programs;
- be able to submit project files to the course website.

#### Prerequisites

At this point, you need only understand that a Java program begins life as one or more source code files which are then compiled into class files containing bytecode. The bytecode is then executed by the JVM. Basic computer literacy and familiarity with Microsoft Windows is also needed for the lab. You need to know how to create folders and files in Windows, and upload files using a Web browser.

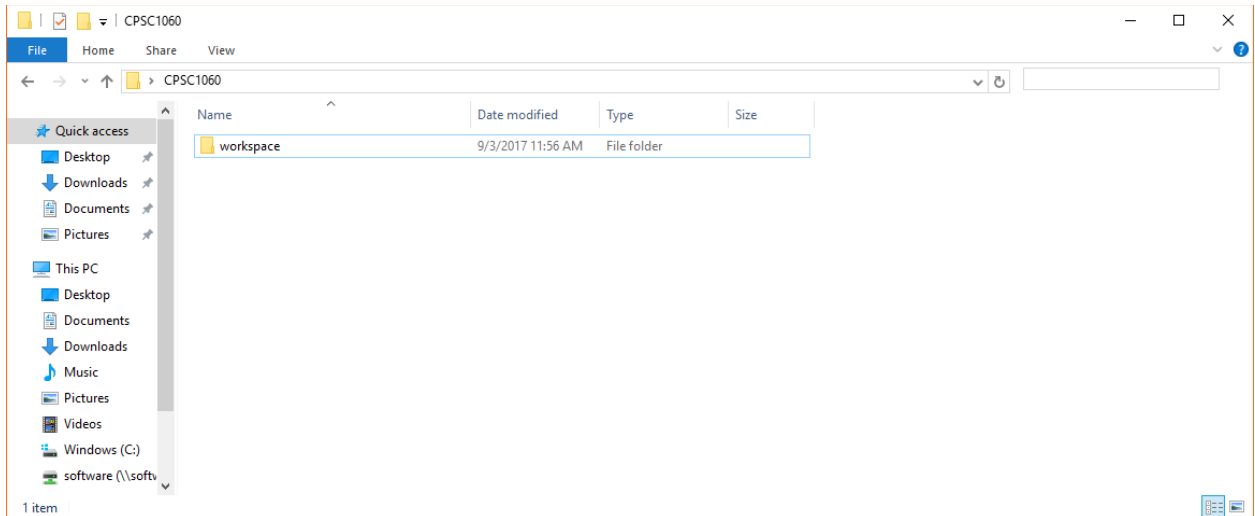
**Note:** The words “folder” and “directory” will be used interchangeably in this document. Directories (folders) are used to organize information on a computer. Directories may contain files or other directories. They are arranged into tree-like hierarchies, with each drive on the computer (for instance, the **C:** drive) having a root directory that branches into multiple subdirectories.

#### What to Submit

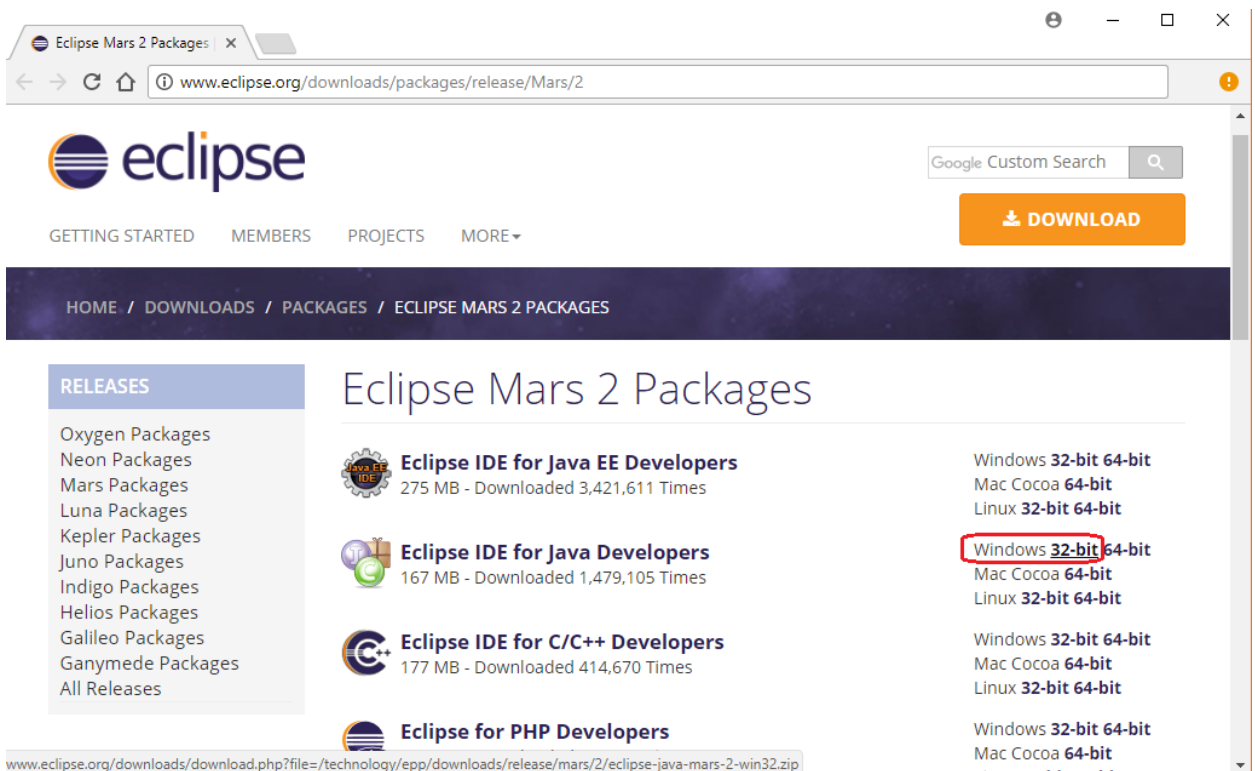
In the lab, you will create a small Java program. The source file (ending in **.java**) should be submitted to the course website. More instructions on how to do this are included in the last part of this document.

#### Part I - Using Eclipse to create your first Java program

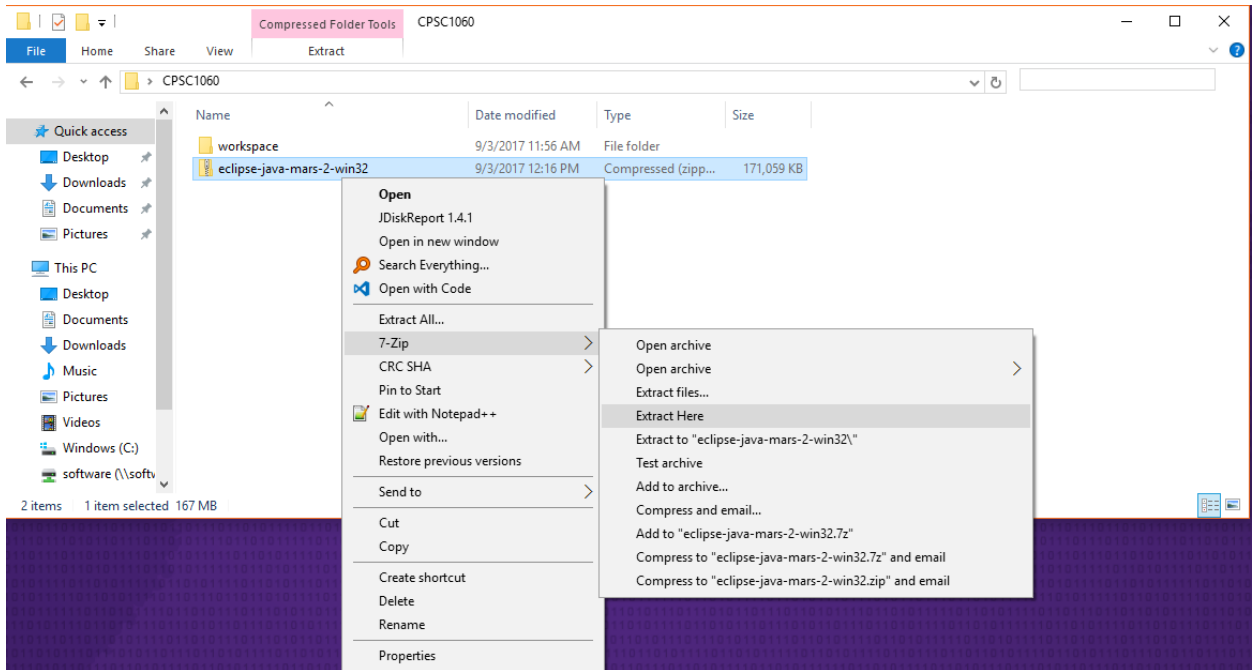
1. Before you start becoming familiar with *Eclipse*, create a folder on the desktop in your lab computer called **CPSC1060**. Go inside the CPSC1060 folder you created on your desktop, and create a directory called **workspace**. Please note these instructions are designed for use on a computer in your registered lab; therefore you may need to adjust these instructions if you are working on a computer outside of your registered lab.



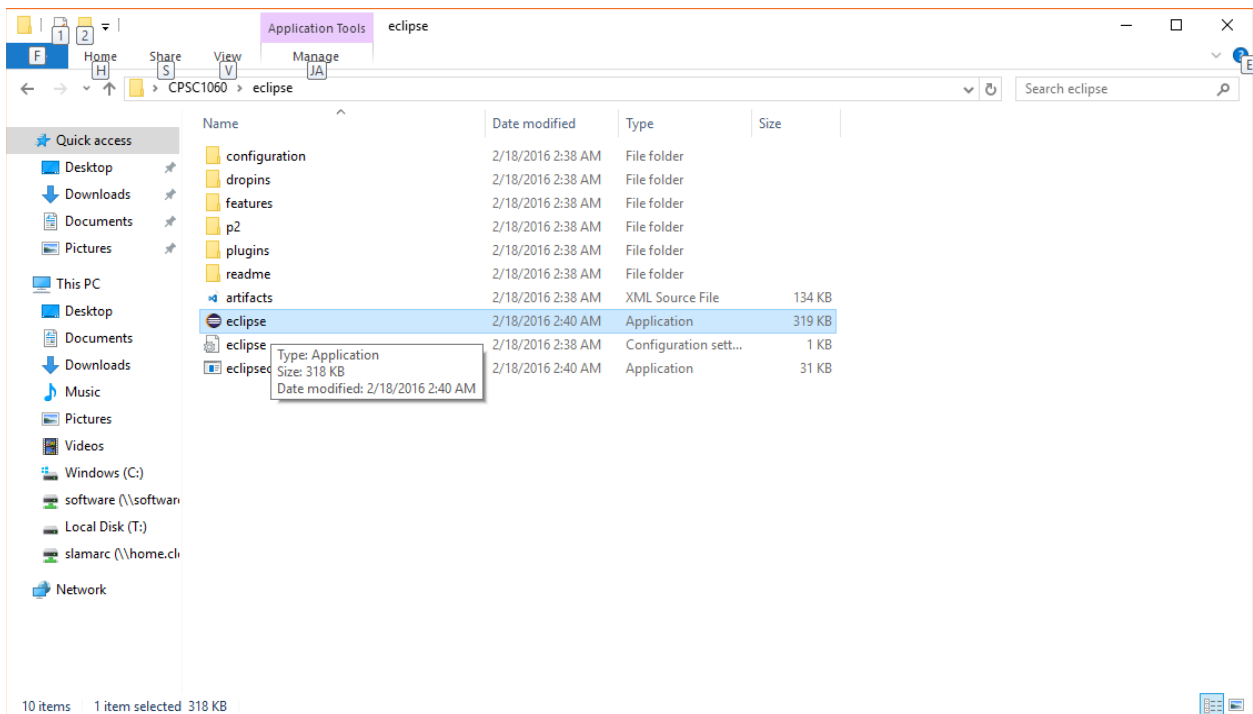
2. Afterwards, download the Eclipse IDE for Java Developers, Mars 2 package (32-bit) from <http://www.eclipse.org/downloads/packages/release/Mars/2>, and move its **.zip** file to your **CPSC1060** folder. You may need to wait a little bit for the download to complete. Do not download the 64-bit version of Eclipse, since that version will not work on our lab machines (our lab machines are running a 32-bit version of the Java Runtime Environment (JRE), which means the 64-bit version of Eclipse won't be able to run on our lab machines).



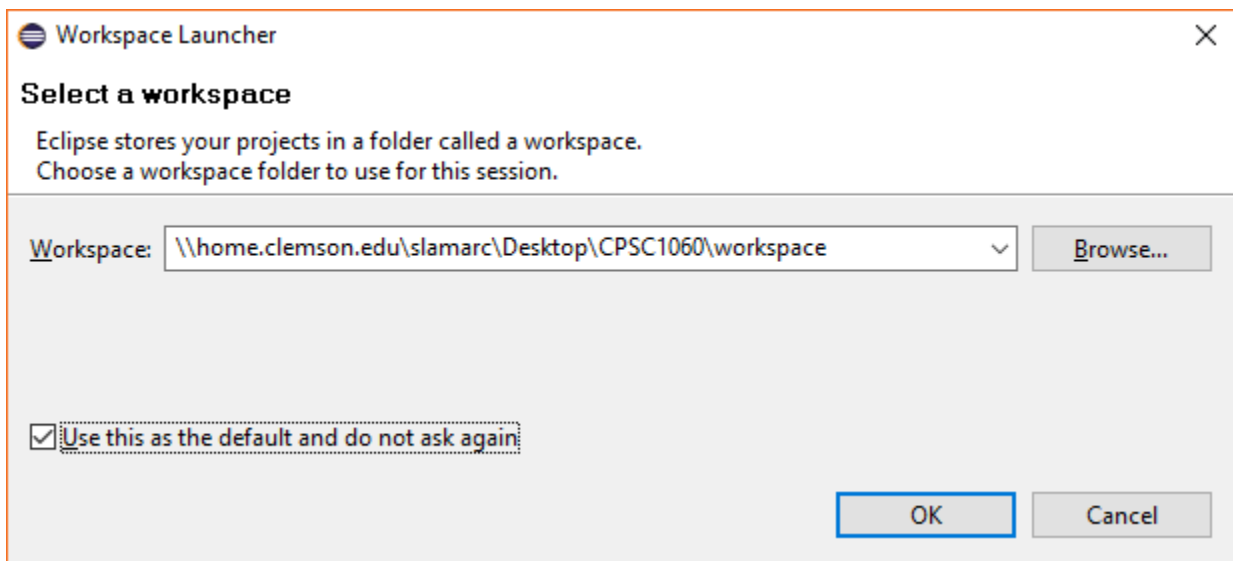
3. Once you download Eclipse, open up the CPSC1060 folder, right click on the Eclipse **.zip** file, select **7-zip**, and click **Extract Here**, and wait for the extraction to complete (it may take a few minutes to unzip the file).



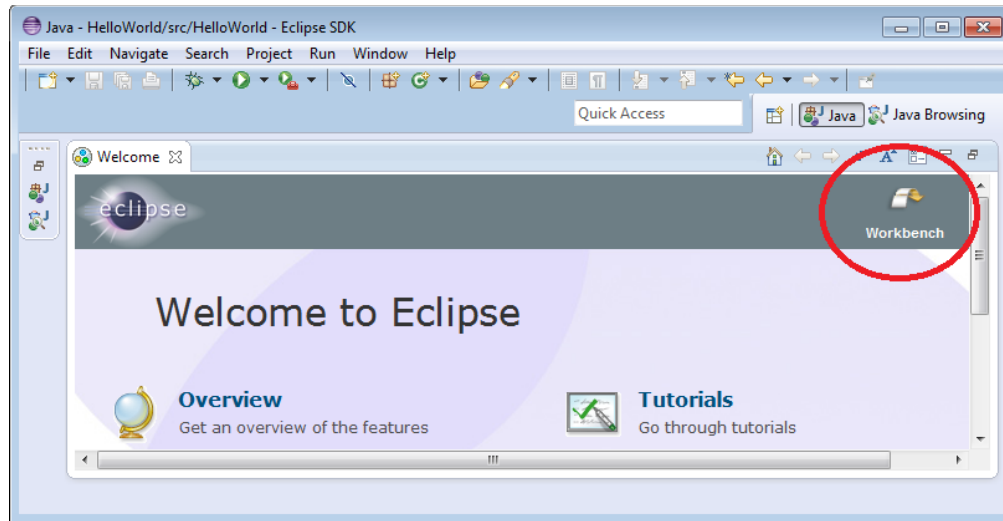
- After the extraction completes, you should have a folder called eclipse (or starts with the name eclipse) in your CPSC1060 folder. Go into that eclipse folder and you should see files similar to what is below. Double click on the **eclipse** program icon, highlighted below, to run Eclipse (you may also want to create shortcut on your Desktop to eclipse since we will be using it throughout the semester).



5. Once you run Eclipse, it will ask you to setup a workspace, which is a place that Eclipse will use to save the files that you work on. Important note: you should not store anything important on the C: drive of the lab machines since files stored here will be deleted when the computer resets. Instead, you should save things to your CPSC1060 desktop folder, which is on a networked drive (\\home.clemson.edu). To setup your workspace on Eclipse, when the window below appears, hit the browse button and navigate to select the **workspace** folder you created in the **CPSC1060** folder (which is on the desktop), and check the “Use this as the default ...” box. Your workspace may be similar to what is below (your username will be different). Make sure you write down and remember where your workspace is since this is the folder that will contain the source code for labs and projects you’ll be working on throughout the semester. Also, it is always important and your responsibility to back up your data, and you may do this by downloading your files to a USB drive (you’ll need to bring to your own to lab to do this), emailing your source files to yourself, or use Clemson’s Box cloud storage account (details at <https://www.clemson.edu/online/tools/box.html>).

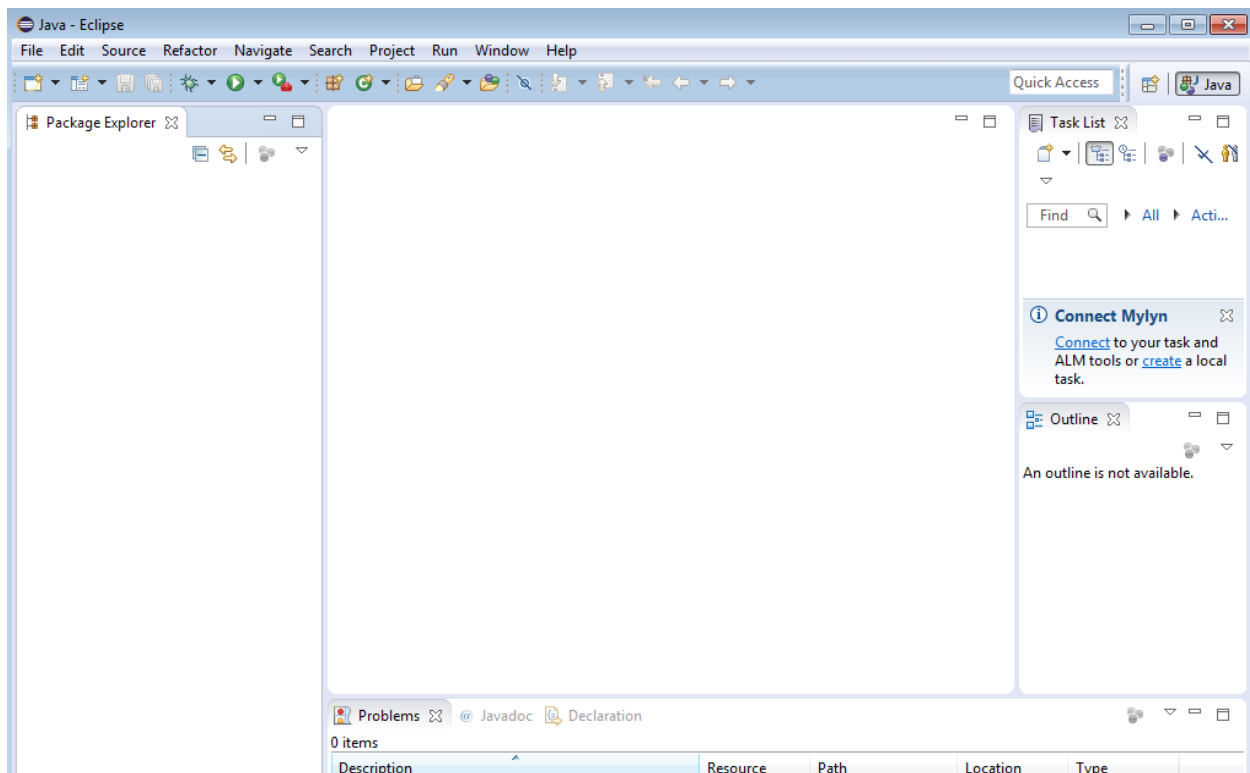


6. After you click **OK**, you will see the following window or something similar (otherwise click on the **Welcome** in the **Help** menu):



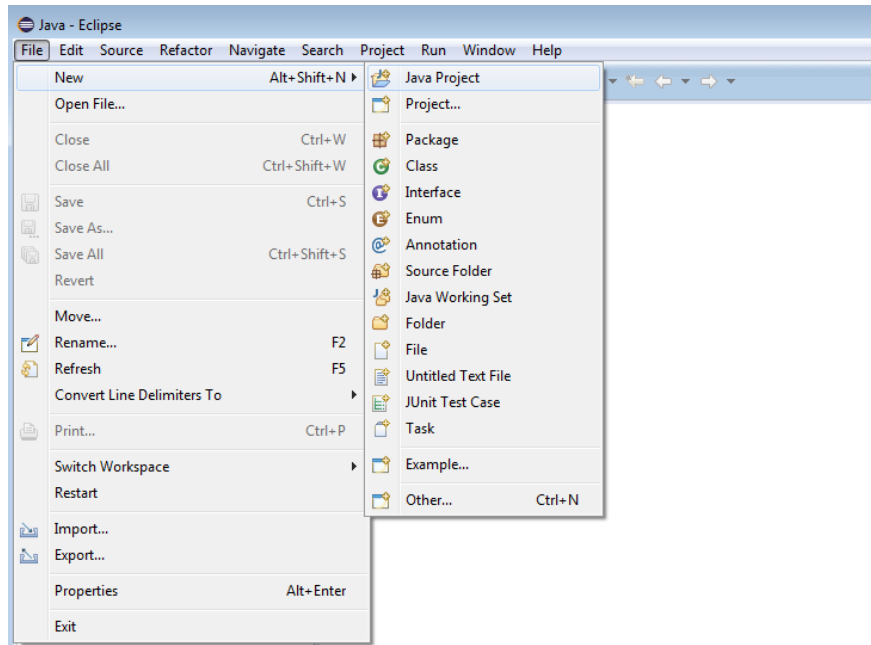
Click on the Workbench icon to start the *Eclipse* IDE.

After a few seconds, you will see a window somewhat like this:

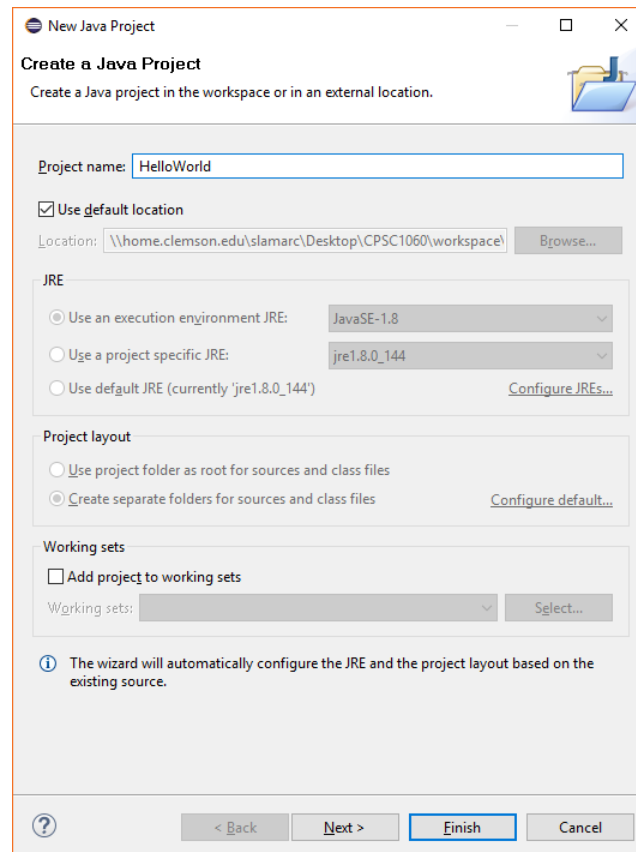


7. In *Eclipse*, a project is a collection of one or more Java source code files (.java) saved in the folder *src* under the project's folder located in the workspace you specified in step 2.

Click on *New/Java Project* in the *File* menu or *New Java Project* button in the main toolbar to create your first project in Eclipse

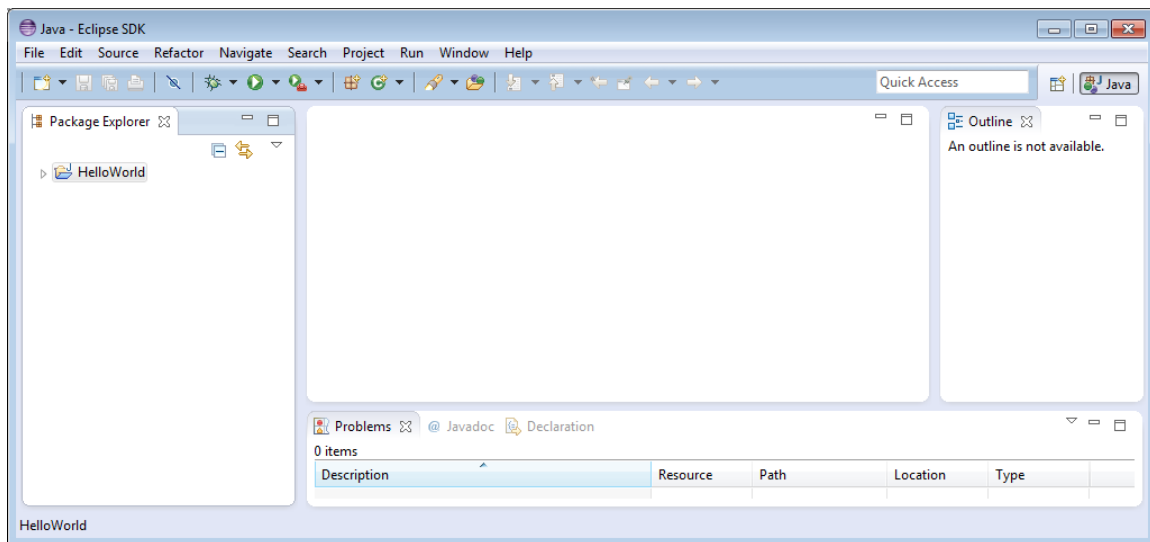


8. In the **New Java Project** window, type **HelloWorld** (no spaces) in the **Project name** textbox

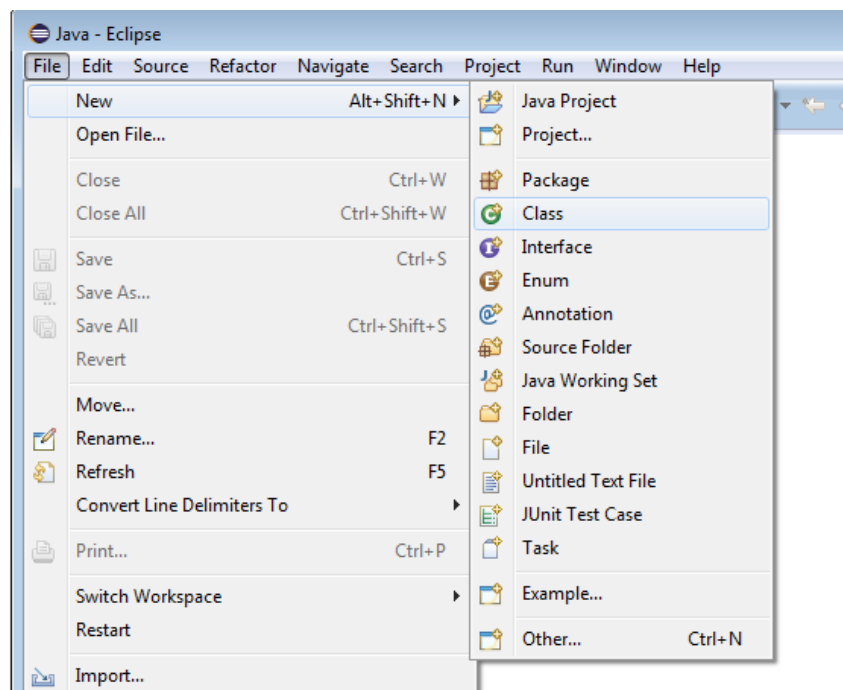


9. Click on the **Finish** button, and after few seconds **Eclipse** should have created a new empty Java project. When you instruct Eclipse to create a new project, it creates a folder with the name of your project in the workspace that you specified in an earlier step. In this example, **Eclipse** creates the

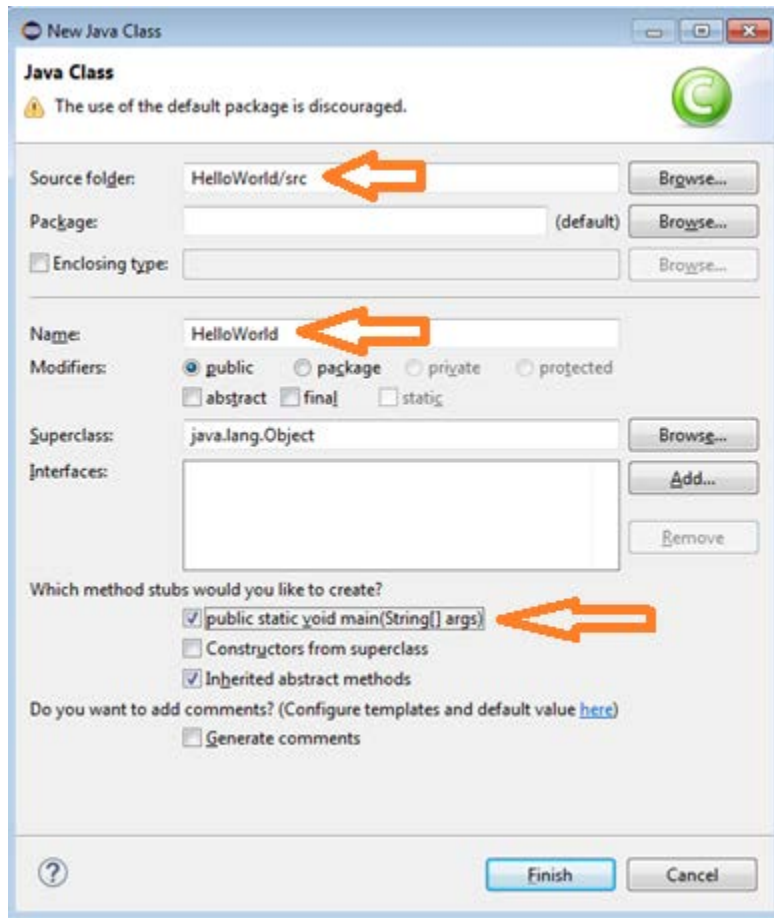
folder **HelloWorld** in the workspace folder. In this folder, **Eclipse** also creates a folder called **src** where the Java source files of the project will be saved and another folder **bin** that will store the Java byte code (.class files) of your project. Afterwards, you will see a window like this:



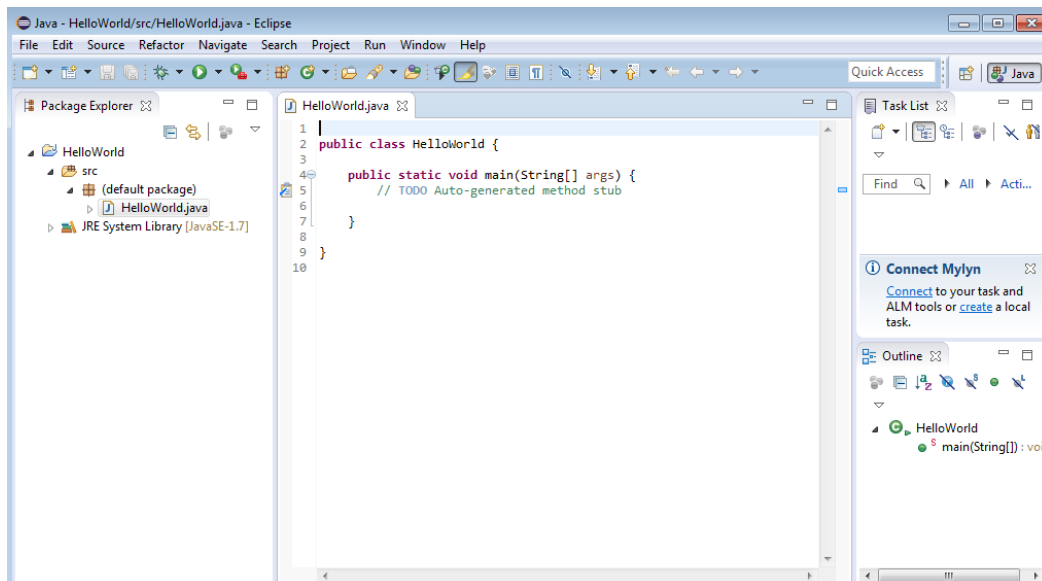
10. The next step is to create the class **HelloWorld** within your project. To do so, make sure the **HelloWorld** project is highlighted (click on it to highlight), and then click on **New/Java Class** in the **File** menu or on the **New Java Class** button in the main toolbar to create the class file.



This will open up the **New Java Class** window. Select HelloWorld/src as the source folder if it is not already specified. Type HelloWorld in the **Name** textbox, check the checkbox **public static main ...** to create the main() method of the class and click **Finish**:



*Eclipse* then creates a template of the class HelloWorld and saves its Java source file in the *src* folder. Afterwards, *Eclipse* displays the code of the new class in the editor window under the HelloWorld.java tab. As you can observe, *Eclipse* had automatically included a template of the main method within the class ...





*Eclipse's* source editor offers some nice features that help you to enter and read Java source code in an easy manner. For example, *Eclipse* displays the source code of the class in the source editor window.

- Tabs are used to indent several parts of the program. This makes your code more readable and easier to debug.
- Words in purple are keywords in Java: words that belong to the Java's vocabulary. We will learn the specific function of these words throughout the course.
- Lines in light blue or green are comments that provide documentation to your program but are not part of the Java code meaning that comments will not be executed by the Java VM.
- Braces and parentheses come in pairs. If you place the cursor after a brace (and parentheses), Eclipse will enclosed its corresponding partner. This feature is useful when you need to find unmatched braces or parentheses in your code, which are common source of syntax errors.

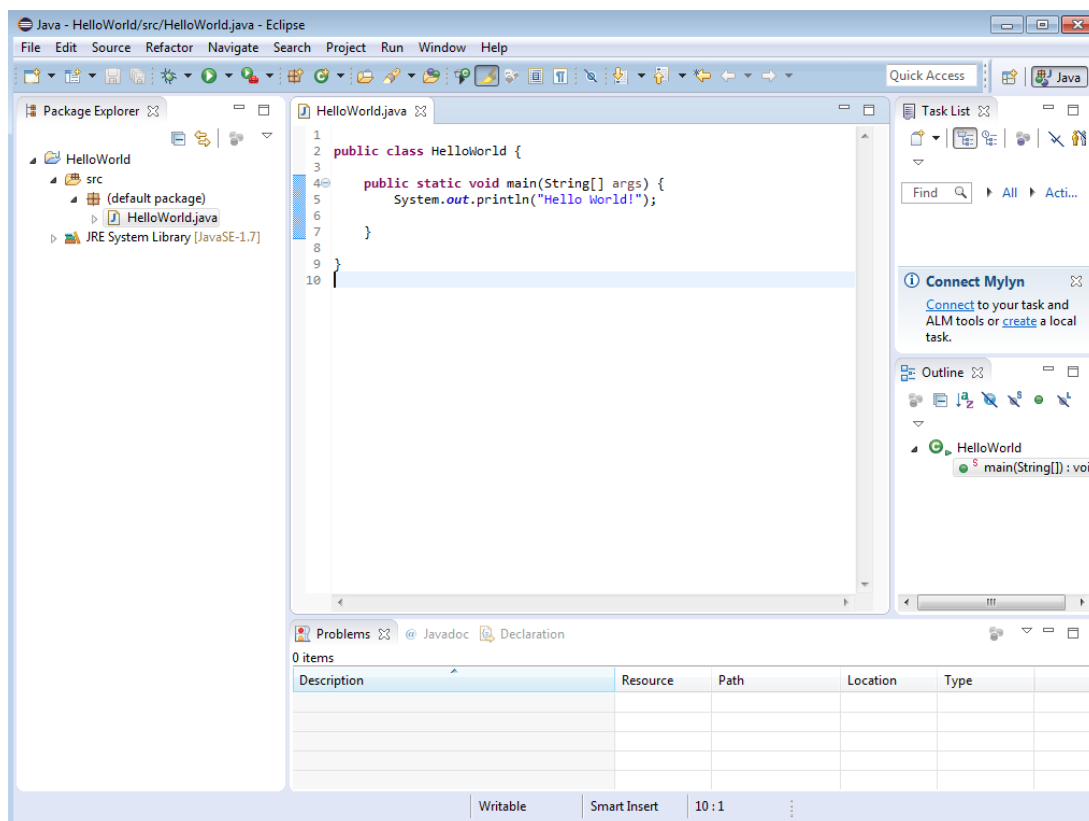
11. Within the main method, replace the line:

```
//// TODO Auto-generated method stub
```

by

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

Afterwards, the *HelloWorld.java* window should look like this:




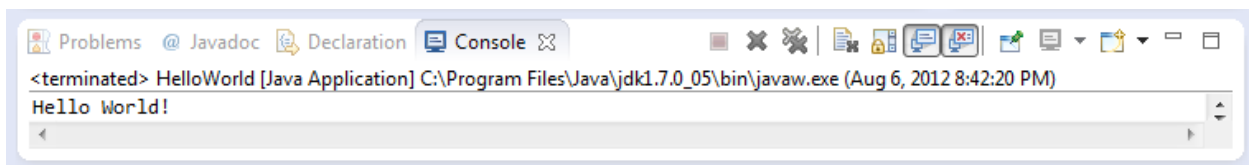
Notice that the phrase “Hello World!” is in blue, which means it is a String literal. A String literal in Java is a sequence of characters enclosed between double quotes. In this example, the String literal is used to display a greeting message to the user.

12. In the **Project** menu, check the option **Build Automatically** if it is not already checked. When this option is checked, **Eclipse** will run the **javac** compiler against all of the source code you have associated with this project every time you save a source Java file. If you prefer to compile your project manually, uncheck the **Build Automatically** option and click on the option **Build All** in the **Project** menu to have **Eclipse** run the **javac** compiler on all the source files in your project.

After the source code has been compiled, **Eclipse** will report all syntax errors found (if any) by the Java compiler under the item named **Errors** in the **Problems** tab in the lower window. If no syntax errors are reported in the **Items** tab then your source code has been successfully compiled and the .class file generated and saved in the folder **bin**. Otherwise, click (or double click) on the **Errors** to see the list of syntax errors founds in your Java source file.

**Eclipse** displays the description, the name of the file and line in the source file where a syntax error was found. Click on the line in the **Errors** item that displays the error, and this will take you to the line that contains the error in the editor window. After you locate the error, you must fix and compile the project again until your project is successfully compiled. If your source code has no syntax errors, the Java bytecode file HelloWorld.class is generated in the folder **bin** in the HelloWorld folder.

13. To run your first Java program, right-click on the default package in the Package Explorer and select **Run As > Java Application** or on the **Run as/Java Application** option of the **Run** menu. The **Console** view should appear at the bottom and display the message “Hello, World!”. You may also use the green Run button  to run your program.



14. Before you exit **Eclipse**, save your work, and click on **Close** in the file menu to close the project. If **Eclipse** prompts you to save the modifications, click on **Yes** if you want to save the last modifications you made. Finally, click on **Exit** in the **File** menu to exit.

## Part II - Submission and Grading

After you have completed and thoroughly tested your program, upload and submit the file **HelloWorld.java** (there's no need to submit **HelloWorld.class** or any .class files this semester) to our course website on Canvas under the Lab 01 assignment. You will need to locate your **HelloWorld.java** file on your computer in order to submit it. Please ask your lab instructor if you have any questions on how to find or submit this file. Always double check that your submission was successful for this lab and future assignments throughout the semester. How do you double check this? Simple, download the source file(s) you submitted on Canvas, read over the downloaded files from Canvas, and then compile, run, and test them again to make sure they work. If you submitted the wrong file, then resubmit the correct file (with multiple submissions, we will only grade the last submission). If you have any issues submitted your work on Canvas for this assignment or any other assignments, then you must email your lab instructor **BEFORE** that assignment is due.

The lab will be graded according to the following guidelines.

- A score between 0 and 100 will be assigned.
- If the source file(s) are not submitted before the specified deadline, or if they do not compile, or if the submitting student has an unexcused absence for a single lab period devoted to the assignment, then a grade of 0 will be assigned. (Note: students who show up to their first lab period and ***personally*** show their TA that they finished and submitted the week's lab assignment may be excused from their second lab period for that week.)
- The program will be evaluated to determine if its output is correct.