CSCI 1301: Introduction to Computing and Programming

Lab 01 - Creating, Compiling, and Executing Java Programs

Introduction

The purpose of this lab is to introduce you to 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.

The lab will also walk you through the process of submitting your completed lab assignment to ***eLC***.

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 ***eLC*** website using a Web browser.

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, use a text editor, 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.

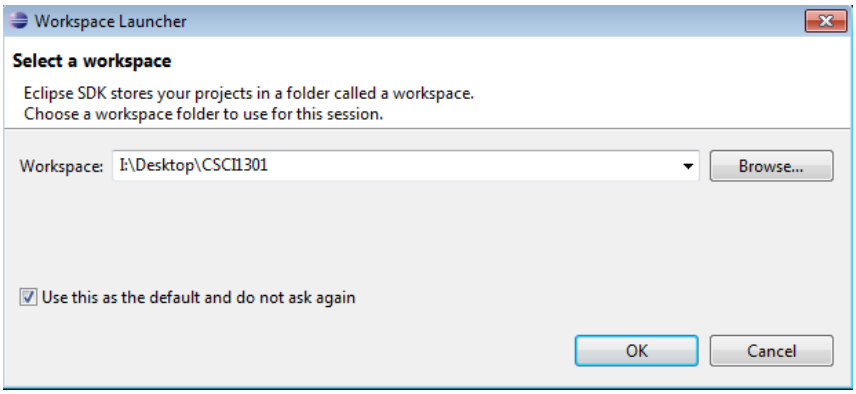
What to Submit

In the lab, you will create one small Java program. The source file (ending in `.java`) should be submitted to the course eLC site. Detailed instructions on how to do this are included as the second part of this document.

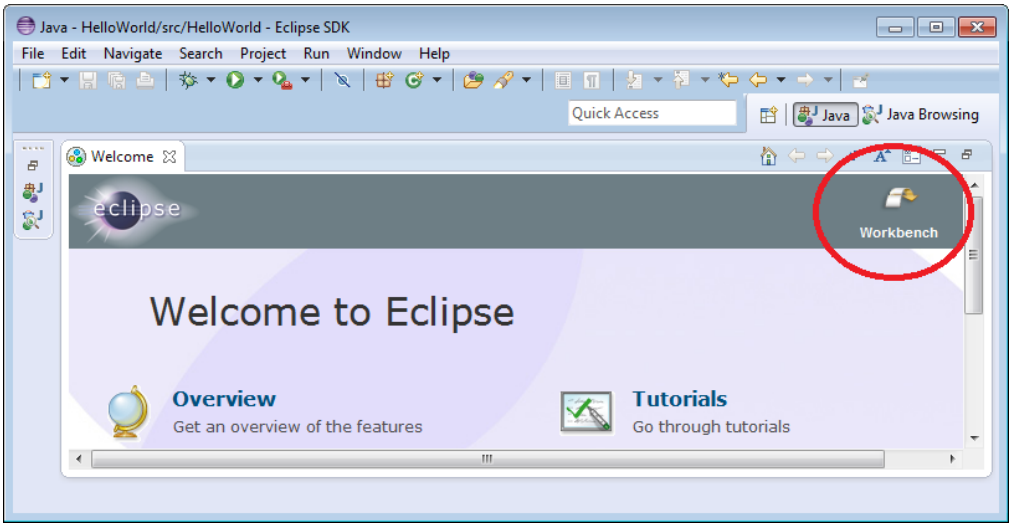
Part I - Using Eclipse to create your first Java program

1. Before you start becoming familiar with ***Eclipse***, create a folder on the desktop called **CSCI1301**.

2. Afterwards, click on the ***Eclipse*** icon on the desktop. ***Eclipse*** will prompt for the folder (workspace) in which the new project will be saved. Click on the Browse button to locate the folder **CSCI1301** that you just created in step 1. ***Note:*** if you use a lab machine, your workspace is at `I:\Desktop\CSCI1301`, but on another Windows machine outside the lab you may need to use another location like `C:\Desktop\CSCI1301`.

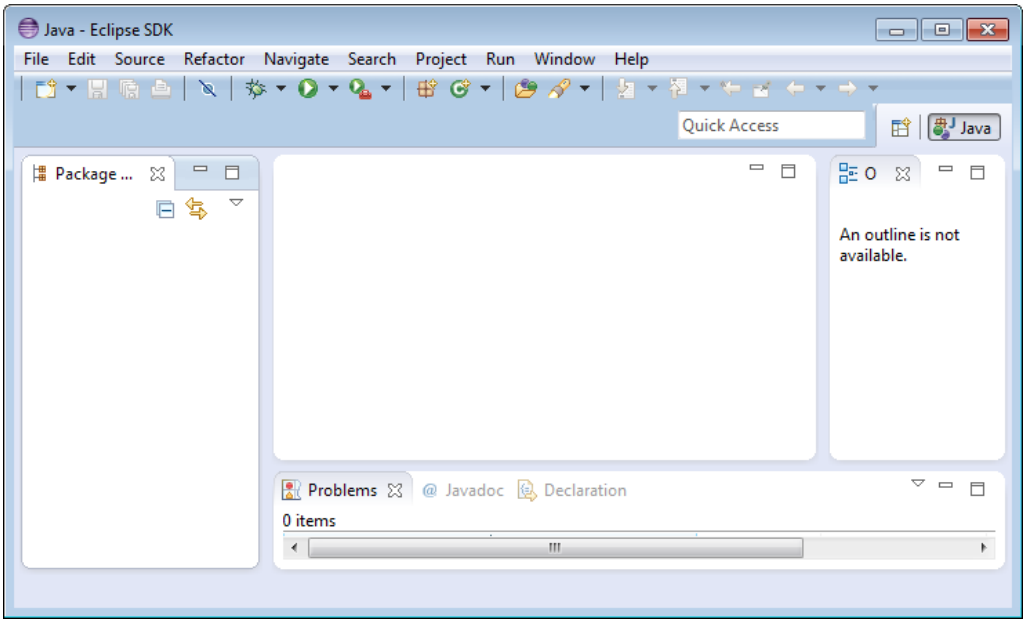


3. After you click ***OK***, you will see the following window or something similar (otherwise click on the ***Welcome*** in the ***Help*** menu):



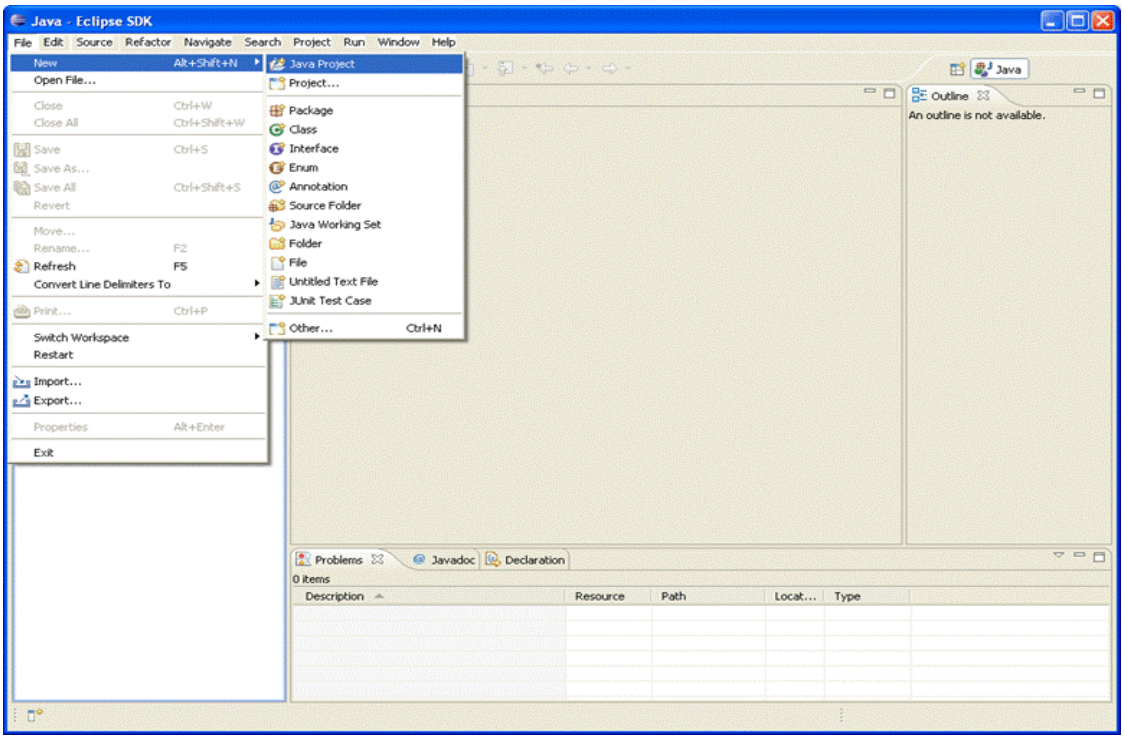
Click on the Workbench icon to start the ***Eclipse*** IDE.

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

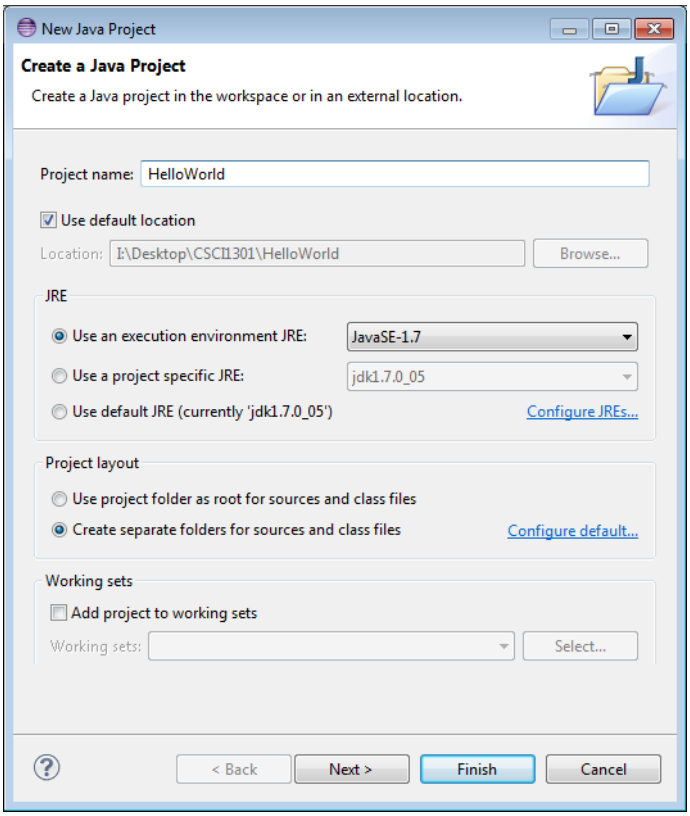


4. In ***Eclipse***, a project is a collection of one or more Java source code files (.java) saved in the folder ***src*** under the project's folder located in the workspace you specified in step 2.

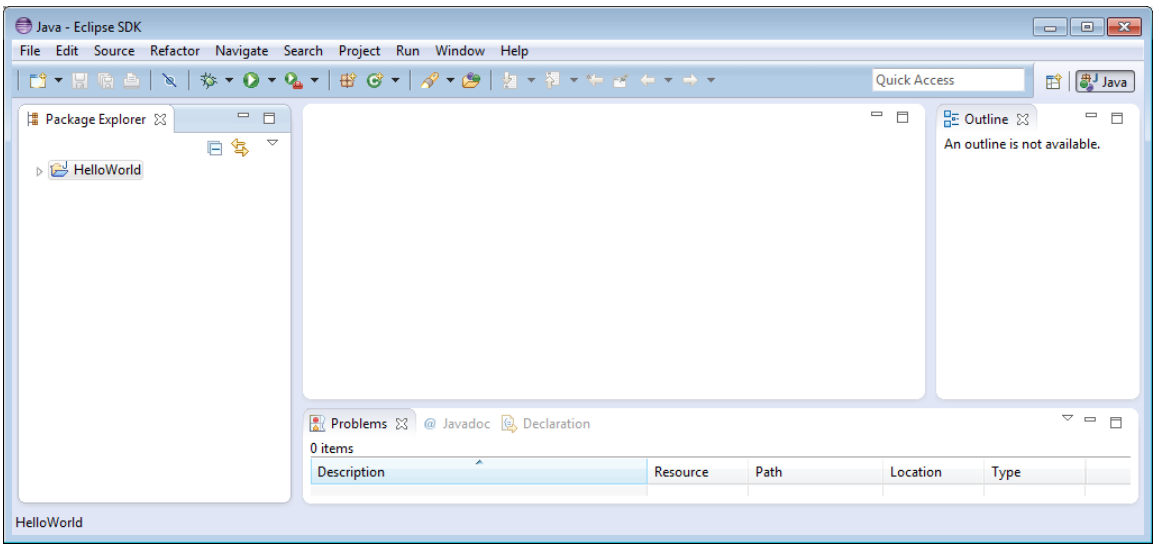
   Click on ***New/Java Project*** in the ***File*** menu or ***New Java Project*** button in the main toolbar to create your first project in Eclipse



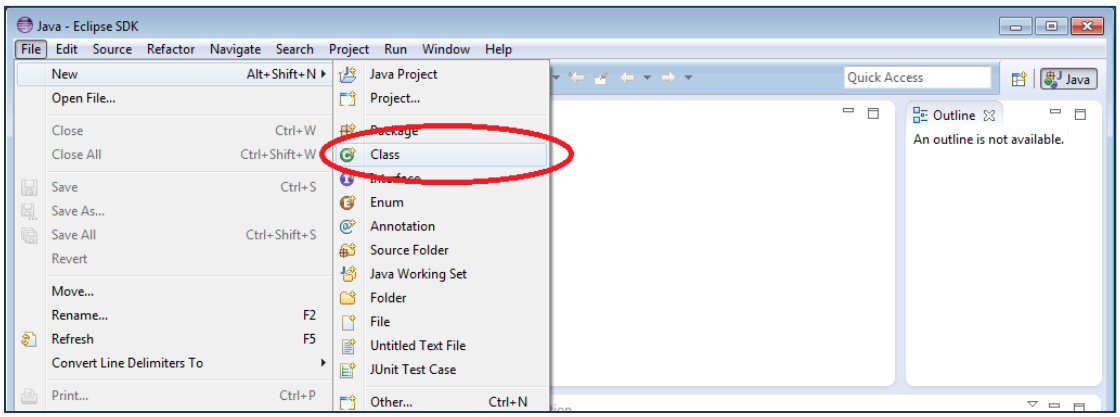
5. In the ***New Java Project*** window, type **HelloWorld** in the ***Project name*** textbox



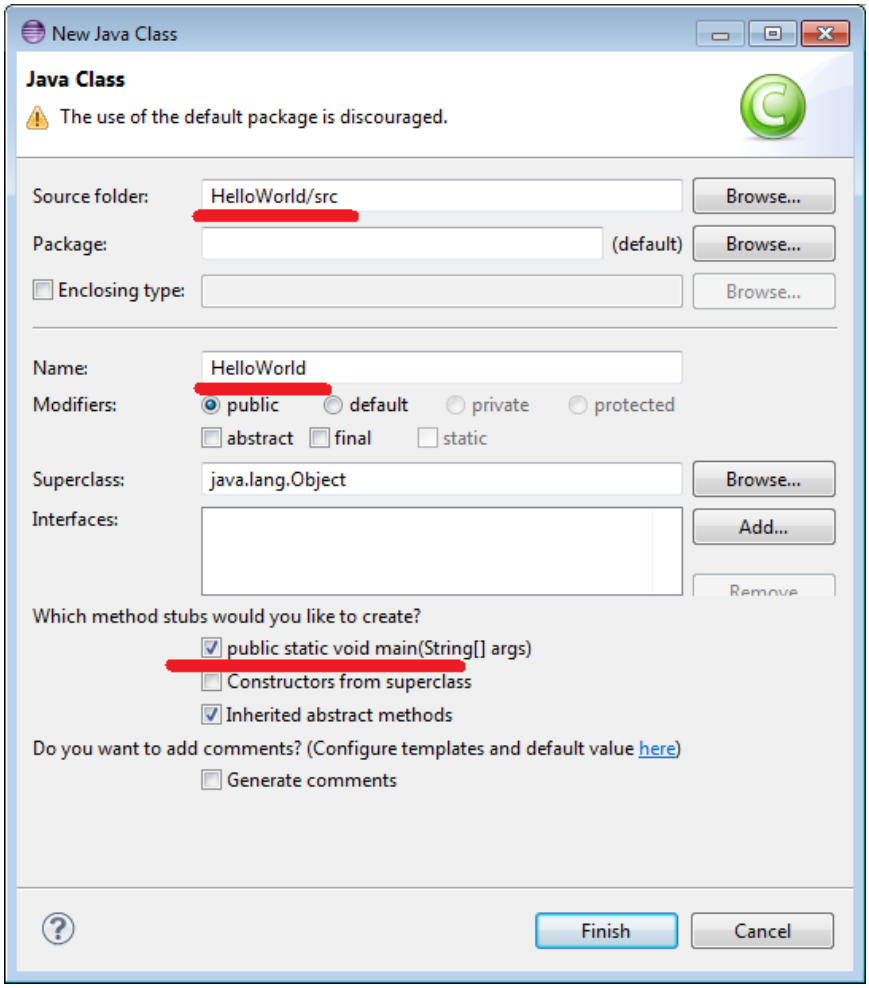
6. Click on the ***Finish*** button, and after few seconds ***Eclipse*** should have created a new empty Java project. When you instruct Eclipse to create a new project, it creates a folder with the name of your project in the workspace that you specify in step 2. In this example, ***Eclipse*** creates the folder ***HelloWorld*** in the folder **CSCI 1301**. 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 stored the Java byte code (.class files) of your project. Afterwards, you will see a window like this:



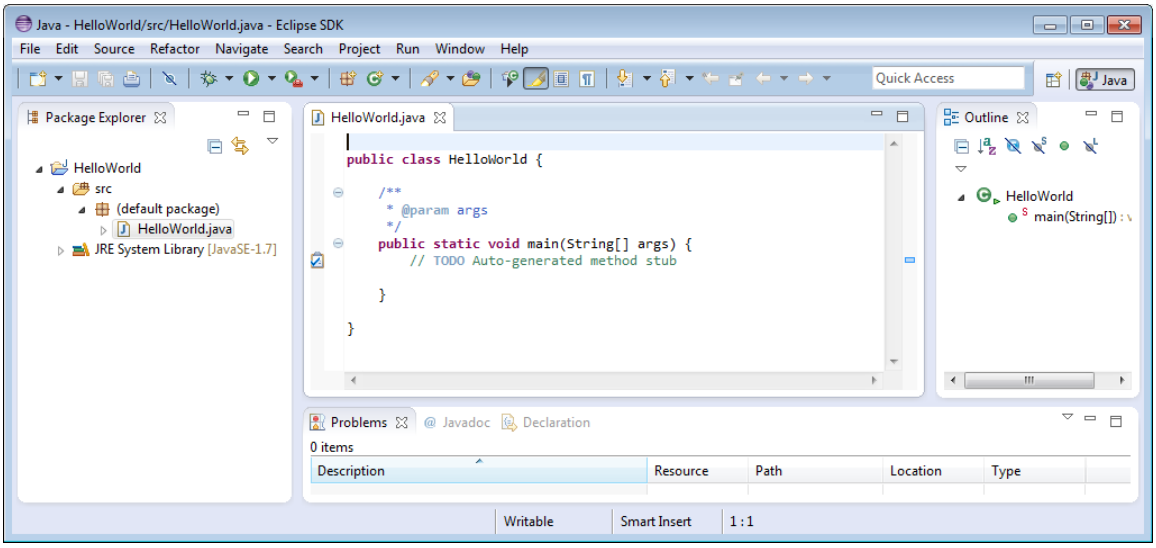
7. The next step is to create the class ***HelloWorld*** within your project. To do so, click on ***New/Java Class*** in the ***File*** menu or on the ***New Java Class*** button in the main toolbar to create the class file.



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 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.

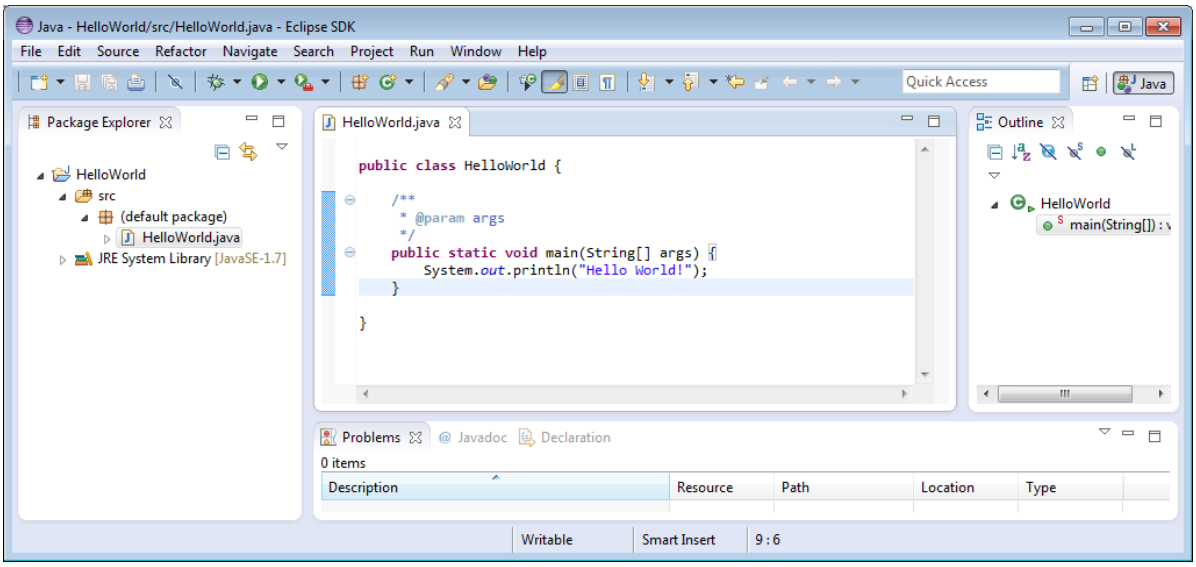
8. Within the main method, replace the line:

   //// **TODO Auto-generated method stub**

   by

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

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



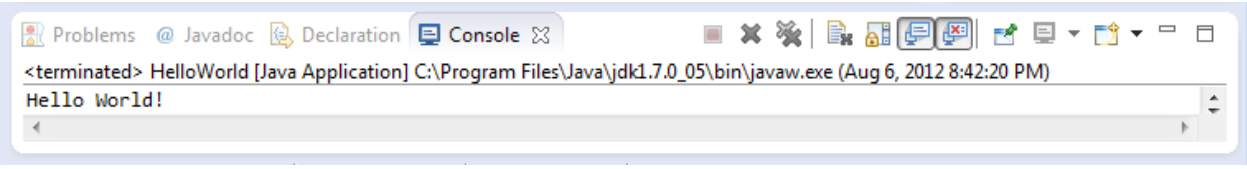
Notice that the phrase "Hello World!" is in blue, which means it is a String literal. A String literal in Java is a sequence of characters enclosed between double quotes. In this example, the String literal is used to display a greeting message to the user.

9. 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 on the + button of the item ***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.

10. 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!"



11. Before you exit ***Eclipse***, 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 - eLC Submission and Grading

After you have completed and thoroughly tested your program, upload and submit the file **`HelloWorld.java`** to eLC. The steps for submitting the file are found here: http://cobweb.cs.uga.edu/~cs1301/elcSubmission.pdf. Always double check that your submission was successful on eLC!

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's late period ends (48 hours after the deadline) or if they do not compile, then a grade of 0 will be assigned.
- Deductions for late submissions and absences will be deducted in a way consistent with the course syllabus.
- The programs will be evaluated to determine if they output the correct Strings.