Lab Notes: Hello Java

UMD CS Department

1 Overview

This first of a series of labs is designed to:

1. Ensure that you have successfully installed all of the required software.
2. Verify that your installed software is working. And to
3. Gently introduce you to the Java programming language.

Ideally, these labs will give you an opportunity to work with other students and your Teaching Assistants in a more relaxed and personalized setting that what is possible in a large lecture.

1.1 Expectations and Conventions

Labs are generally distributed on Monday mornings, and are expected to be completed by Tuesday mornings (for the early, simple labs) and Wednesday afternoon (latest) for more complex and involved labs. At the start of your Discussion section, you will be able to use the CVS check-out mechanism to check-out the Lab (or Project). You will then use the ECLIPSE IDE to edit the source code, and then you will *submit* your modifications to the **Submit Server**, where you will see the results. Taken together, these actions are the general *workflow* that you will follow for labs and projects over the next sixteen weeks.[1]

2 Your task is . . .

For this first lab, you will modify the source code found in the `HelloWorld.java` file in order to have it print the message

```
"Hello World!"
```

[1]The only differences between projects are that projects may be distributed on various days of the week, are more complex, requiring more time, and are *expected to be completed by each student without outside assistance.*

Having made the necessary changes, follow your Teaching Assistant's instructions to *submit* your code for evaluation. When you are satisfied that it works (i.e., when you've passed all of the tests), you're done . . . it's as simple as that.

3 A word on the bits and pieces

Before you could "check out" your lab today the actual code had to exist in your directories, which are located on the Linux Lab cluster of machines. Checking out your code actually created a copy of that code, which is a collection of *files* residing in *directories*. This means that you only "own" a private copy of the code—changes that you make to that code are invisible to others.

Next, you had to understand the problem at hand, formulate a solution to it, and then express that solution in the JAVA programming language. The JAVA source code that resides in your *buffers* is *compiled* by the JAVA COMPILER, which is part of the ECLIPSE IDE that you installed prior to class. The compiler is a program that translates the text that resides in your `HelloWorld.java` text file into a `class` file, which is a file that contains code that is mostly unreadable to human beings but is *executable* by JAVA.

Finally, you submitted your code to the **Submit Server**, where a suite of automated tests were performed, and those results were recorded on a student-by-student basis. Note, these tests are also written in JAVA. Soon, you will learn how to write your own tests because testing code is proven effective way of learning how to write better code.

3.1 Diagnosing problems

Because this is a multistep process, things can go wrong along any one step. For example, attempting to type non-JAVA statements into the editor will likely result *syntax* errors. These are indicated by red-Xs in the buffer. Suggestions for fixing these errors are usually provided. As a rule: you should never accept another person or a program's suggestion about fixing your code unless you understand what is being proposed.

Besides syntax errors, you may enter a statement that is syntactically correct but is logically incorrect. ECLIPSE is not as helpful in finding these kinds of errors because their discovery requires a deeper understanding—specifically, one needs to understand the problem being solved and the solution being proposed in order to identify these kinds of errors. This is where you will spend most of your time in this class, because this is the biggest payoff.

Finally, these kinds of problems assume that you've correctly installed all of the software! For example, installing ECLIPSE is great, but forgetting to install the **UMD plug-in** that allows you to submit your work is a show-stopper . . . but, so is installing the *wrong* compiler (in this case, a version of JAVA less than 1.7).

It might seem overwhelming, but starting off on the right foot is one key to ensuring a successful semester, and that's why we have this first lab!