CSCE 155 - Java

Lab 1.0 - Introduction

Prior to Lab

In each lab there may be pre-lab activities that you are *required* to complete prior to attending lab. Failure to do so may mean that you will not be given full credit for the lab. For the first lab, there are no pre-lab activities.

Peer Programming Pair-Up

To encourage collaboration and a team environment, labs will be structured in a *pair programming* setup. At the start of each lab, you will be randomly paired up with another student (conflicts such as absences will be dealt with by the lab instructor). One of you will be designated the *driver* and the other the *navigator*.

The navigator will be responsible for reading the instructions and telling the driver what to do next. The driver will be in charge of the keyboard and workstation. Both driver and navigator are responsible for suggesting fixes and solutions together. Neither the navigator nor the driver is "in charge." Beyond your immediate pairing, you are encouraged to help and interact and with other pairs in the lab.

Each week you should alternate: if you were a driver last week, be a navigator next, etc. Resolve any issues (you were both drivers last week) within your pair. Ask the lab instructor to resolve issues only when you cannot come to a consensus.

Because of the peer programming setup of labs, it is absolutely essential that you complete any pre-lab activities and familiarize yourself with the handouts prior to coming to lab. Failure to do so will negatively impact your ability to collaborate and work with others which may mean that you will not be able to complete the lab.

1 Lab Objectives & Topics

At the end of this lab you should be familiar with the following

- The general lab environment & CSE system including its policies and expectations
- Retrieving lab code from Github
- Modifying, compiling and executing your first Java program using the Eclipse IDE (Integrated Development Environment)
- Using CSE's webhandin and web grader

2 Administrivia

2.1 Login & Consent Form

You will receive your login from the CSE System Administrators who will also give you an overview of the CSE system and its policies. Be sure to login and change your temporary password. Some departmental resources that you may find useful:

- CSE Website: http://cse.unl.edu
- UNL Computing Policy: http://www.unl.edu/ucomm/compuse/
- CSE Academic Integrity Policy: http://cse.unl.edu/academic-integrity-policy
- CSE System FAQ: http://cse.unl.edu/faq
- Account Management: https://cse-apps.unl.edu/amu/
- CSE Undergraduate Advising Page: http://cse.unl.edu/advising
- CSE Student Resource Center: http://cse.unl.edu/src

2.2 Lab Introduction

- Lab instructors and TAs
- Office Hours
- Student Resource Center (Avery 12)
- Lab policies

3 BYOD: Bring Your Own Device

Though you can use the lab computers for this activity, you will most likely want to use your own machine to do development independent of the CSE labs. To do this, you'll need to install a couple of things.

1. Go to Oracle's website and download and install the Java Development Kit (JDK). You may already have "Java" installed but likely it is the *Runtime Environment* (JRE) that allows you to run Java programs. The Dev Kit allows you to *develop* programs. You should install the latest version; as of this writing the current version is 13 and is available at: https://www.oracle.com/java/technologies/javase-jdk13-downloads.html (alternatively, you can use openJDK instead if you wish: https://openjdk.java.net/)
2. Got to https://www.eclipse.org/ and download/install the Eclipse IDE. It is perfectly fine to use an alternative IDE if you wish, but instructions for all labs will assume Eclipse.
3. Though not necessary for this lab, you should also sign up for a GitHub account: https://github.com/

4 Activity: Creating a New Project

4.1 Using Eclipse

Eclipse is an industry-standard IDE for Java development. There are several other popular and emerging IDEs available and you are welcome (and encouraged) to try them out and use them. For this course, we will primarily focus on Eclipse.

1. From Windows, start Java Eclipse
2. Choosing a "Workspace":
   - If you are using a lab computer, type the following in the text box: `Z:\Workspace` (the Z: drive is your personal file storage on CSE)
   - If you are using a personal computer, choose a workspace directory/folder where you want all of your future projects to exist.
3. Close the Welcome screen, and create a new project. Select: File → New → Project
4. Select "Java Project" when prompted by the New Project Wizard; click Next
5. Enter a name for your project (Lab01); click Finish
6. Open the Lab01 folder in the Package Explorer View and find the `src` folder

7. Right-click on the `src` folder and select "New Class" and name it `Hello`.
8. This should open a new source code editor. Make your program look like the following and save it.

```
1  public class Hello {
2
3    public static void main(String args[]) {
4      System.out.println("Hello World!");
5    }
6
7  }
```

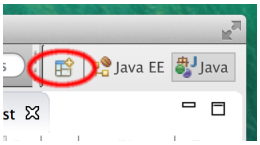
9. Run your program by clicking the "play" button in the tool bar

Congratulations on your first program!

5 Checking Out Code From Github

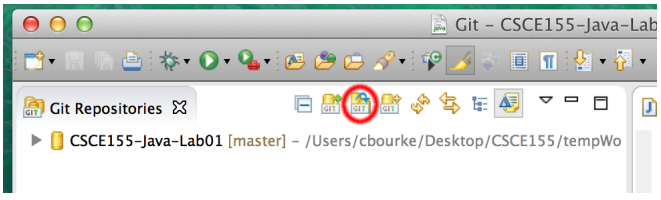
Each lab will have some starter code and other *artifacts* (data files, scripts, etc.) that will be provided for to you. However, the code is hosted on Github (https://github.com) and you must check it out. You will not need to know the details of using git nor be a registered Github user to get access to the code necessary for your labs or assignments. However, you are *highly encouraged* to learn this essential tool. You may find it very useful to keep track of your own code and to share code if you work in pairs or groups.

1. First we need a Git *perspective* (a context in the Eclipse User Interface that will allow us to work with Git). To open the Git perspective, click on the "Open Perspective" tab in the upper right:

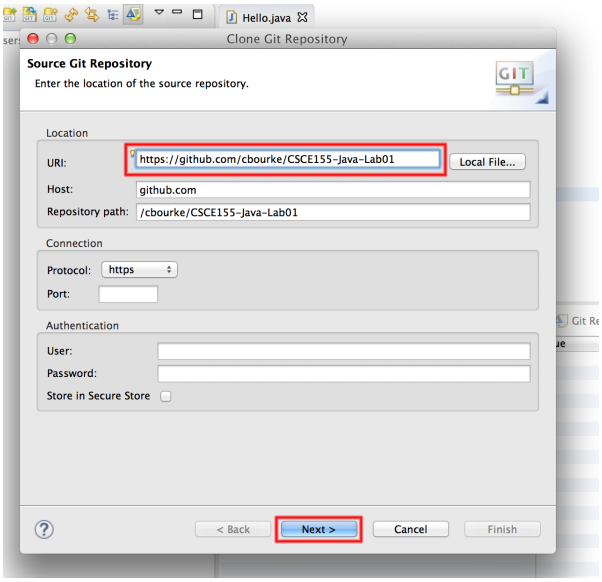


   Select "Git" from the menu and click OK

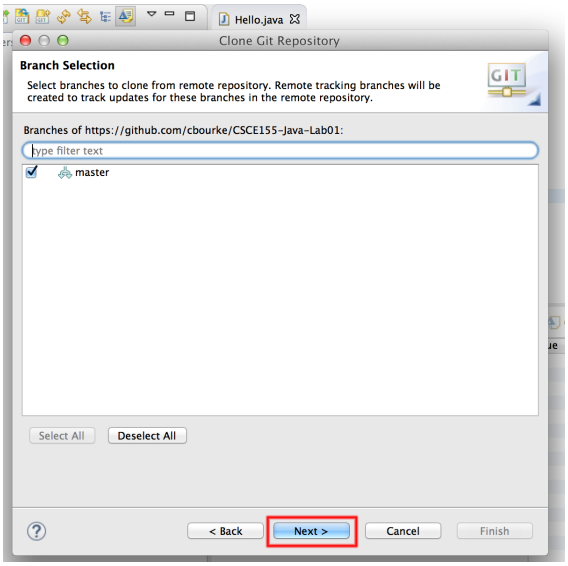
2. Click the "Clone a Git repository" in the Git Repositories navigation menu:



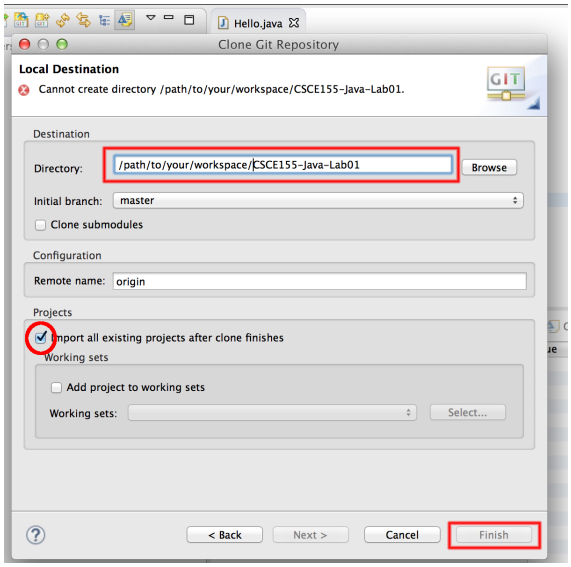
3. Copy/past or type into the URI field, the URL:
   https://github.com/cbourke/CSCE155-Java-Lab01



4. Click "Next"; once Eclipse has grabbed the project, the "master" branch should be selected (checkbox); click "Next" again.



5. Select the directory where you want your project to be saved. Caution: the default option may not correspond to your default workspace. You may want to change it to your workspace, but the choice is yours. Also mark the "Import all existing projects after clone finishes" checkbox option or you will need to manually import the cloned project into Eclipse.



6. Switch back to your Java or JavaEE perspective and you can see your cloned project.
7. Run the `Hello.java` program in this project to verify everything works.

5.1 Modify the Program

1. Modify the `Hello.java` program by changing the author to you (and your partner) and change the date.
2. Change the message that is printed by the program to instead print you (and your partner's) name(s).

6 Handing In & Grading

Many of your assignments will include a programming portion that will require you to hand in *soft-copy* source files for graders to compile and evaluate. To do this, you will use a web-based handin program. After handing your file(s) in, you can then grade them by using the web grader. To demonstrate, do the following.

1. Open a browser to https://cse-apps.unl.edu/handin
2. Login with your CSE credentials
3. Click on this course/lab 01 and handin the `Hello.java` file. You can either click the large "handin" area and select the file or you can drag-drop the file. You will be able to re-handin the same file as many times as you want up until the due date.

4. Now that the file has been handed in, you can "grade" yourself by using the webgrader; open a new tab/window and point your browser to one of the following depending on which course you are in:
   - https://cse.unl.edu/~cse155a/grade
   - https://cse.unl.edu/~cse155e/grade
   - https://cse.unl.edu/~cse155h/grade
5. Fill the form with your CSE login and password, select the appropriate assignment and click "Grade"
6. For future assignments and labs, you can compare the results of your program with the "Expected Results". If there are problems or errors with your program(s), you should fix/debug them and repeat the handin/grading process. You can do this as many times as you like up until the due date. Some programs and assignments will run test cases and may provide expected output alongside your output. Others may have more sophisticated test cases and actually provide you a percentage of test cases passed. It is your responsibility to read, understand and *address* all of the errors and/or warnings that the grader produces.