CSCE 156 – Lab: Java Introduction

0. Prior to the Laboratory

- Review the user documentation at Eclipse's website http://help.eclipse.org/helios/index.jsp

1. Objectives

Following the lab, you should be able to:

- Receive and activate your CSE account and log into the network using a Windows machine and your CSE account.
- Open, compile, and execute a given Java program in Eclipse.
- Write a Hello World Program in the selected IDE, compile, and execute that program.

2. Topics Covered

- Lab introduction
- CSE Account
- IDE Editor
- Variables
- Numerical Data Types
- Console I/O
- Mathematical Expressions

3. Experience Gained

1. Creating class and lab folders.
2. Creating and editing a project.
3. Compiling and executing a program.
4. Getting help with Java.
5. More work with the IDE.
6. Compiling and executing a simple Java program in your selected IDE.
7. Writing, compiling, and executing a Hello World program in Java.

Activity 1: Administrative Stuff

1. **Get Account:** Receive account sheet from Lab instructor or from system management office.
2. **Login:** Log into the network using a Windows machine by entering your CSE account name (as indicated on account sheet) and by entering the password exactly as typed.
3. **Change Password:** Click on the "options" button. Click on "Change password." Follow instructions to create new password. Note: New passwords must be at least 6 characters long and must contain at least three of the following type of characters:
    a. upper case letters
    c. numbers

b. lower case letters
d. Special symbols

4. **Fill Consent Form:** Using the web browser of your choice accept the electronic consent form at http://cse.unl.edu/consent. This form covers the ethical use of the resources made available to you by the department. You have 1 week to electronically sign this after which time, if still unsigned, your account will expire. Your account information may be accessed from http://cse.unl.edu/check.

Activity 2: Editing, Compiling, and Running a Simple Java Program in an IDE

Eclipse is a popular, industry-standard Integrated Development Environment (IDE). IDEs make development easier by providing code mark-up, code completion, and other useful features in a GUI environment. This activity will familiarize you with the Eclipse environment. Eclipse is free and open source so you can/should download it (from http://eclipse.org) and install it on your own machine(s).

Instructions

1. Download all the .java files from Blackboard to your desktop.
2. Start Eclipse and select a directory to use as your workspace on your CSE Z: drive
3. Create a new project:
   - Right-click the Project Explorer area at the left
   - Select New -> Project… -> Java -> Java Project -> Next
   - Name your project Lab01, click Finish
4. Create a new package. Java classes are stored in a hierarchy of packages. This is done for better logical organization and for package visibility reasons (to be discussed later).
   - Right-click the src (source) folder and select New -> Package
   - Name your package `unl.cse.labs.lab01`
   - Click Finish
5. Create a new class. All Java code must be contained in a class. This is in contrast to other paradigms that may allow global variables or treat functions as "first-class citizens" (functions can exist without an object or a class).
   - Right-click your new package and select New -> Class
   - Name your class `Statistics`
6. Open the downloaded `Statistics.java` file and cut and paste its contents into your new `Statistics` class.
7. We can also directly import an external Java file. Drag and drop the StatisticsDemo.java file from your desktop to the unl.cse.labs.lab01 package. Make sure that "copy files" is selected and click Ok.
8. The `StatisticsDemo` class contains a main method,
   `public static void main(String args[])`
   In Java, classes can only be executed if the main method is defined. Classes without a main method can be used by other classes, but they cannot be run by themselves as an *entry point* for the Java Virtual Machine. Run the `StatisticsDemo` as follows.

- Open The `StatisticsDemo` class file and then click on the "play" button (a green circle with a "play" arrow on it).
- The output for this program will appear in the lower part of Eclipse in the "console" tab.
- Click on the console tab and enter the input as specified.

Activity 2A: Completing the Statistics Program

Though the program runs, it does not output correct answers. You will need to modify these classes to complete the program.

1. Implement the `getMax` method in the Statistics class. Use the `getMin` method for directions on syntax.
2. Implement the `getSum` method in the Statistics class. Use the other methods for direction on syntax.

Activity 2B: Modifying the Statistics Program

The program you've completed is *interactive* in that it prompts the user for input. You will now change the script to instead use *command line arguments* to read in the list of numbers directly from the command line.

Command line arguments are available to your main method through the `args` array of Strings. The size of this array can be obtained by using `args.length` which returns an integer. Modify your code to iterate through this array and convert the arguments to integers using the following snippet of code:

```
for(...) {
  array[i] = Integer.parseInt(args[i]);
}
```

The "command line" may not be apparent as you are using an IDE. However, it is still available to you. Instead of clicking the "Play" button to run your program, click the down arrow right next to it. Then select "Run Configurations". This brings up a dialog box with which you can run custom configurations of program and JVM. Click the Arguments tab and enter a space-delimited list of numbers under "Program Arguments". And click Run.

Demonstrate your working program to a lab instructor and have them sign off on your worksheet.

Activity 3: Birthday Program & Libraries

No man is an island. Good code depends on selecting and (re)using standard libraries of other code when possible so that you are not continually reinventing the wheel. It is what separates man from beast. This activity will familiarize you with how to import and use a Java library. Java packages library code in to JAR (Java ARchive) files.

Instructions

1. Import (drag and drop) the Birthday.java file from Blackboard as described previously.

2. You will notice there are syntax errors because this class uses other classes that are not available in the standard JDK (Java Developer Kit). It instead uses classes from the Joda-Time library; a library of useful classes an utilities for dealing with dates, times, and durations.
3. Download the `joda-time-2.0.jar` file from Blackboard (save to your desktop).
4. Create a library folder in your project:
    - Right-click the project and select New -> Folder
    - Name the folder "lib" (short for library)
5. Import the Joda-Time library by dragging and dropping the jar file into this new lib folder as before
6. Right-click the new JAR file and select Build Path -> Add to Build Path
7. The library is now imported and the syntax errors should go away.

Finishing the program

Though the program should have no syntax errors, if you run it, no output will be displayed. You need to complete the program as follows.

1. For the variables, `name`, `month`, `date`, and `year`, enter your own information (your name and your birthday).
2. Add appropriate code (using `System.out.println` which prints to the standard output a full line) a greeting like:
   Greetings, NAME. Today you are XX years, XX months, and XX days old.
   Of course, XX should be replaced with variable values. Note: In Java, variable values can be concatenated with strings using the + (plus) operator.
3. Add a conditional statement that, if today is the user's birthday will output "Happy Birthday". If it is not the user's birthday, output "Your friends have XX shopping days until your next birthday" where XX is replaced with the appropriate variable value.

Complete your worksheet and demonstrate your working code to a lab instructor.

Advanced Activity (Optional)

Explore the Joda-Time library (its API is available at: http://joda-time.sourceforge.net/api-release/index.html) and Java itself by implementing the following program. The program will read in (or hard code) two birthdays and compute the number of years, months, and days between them.