**Java Email Sample Program**

Before starting on this activity, it will be extremely helpful to understand the JavaScript email example program that we discussed in class, which should refresh your memory about some things we learned regarding objects in JavaScript. If you have not taken Web Programming yet, then don’t worry. We’ll be learning objects from scratch later in this course. Here are some of the important points to remember:

1. Objects are used in programming to simulate real-world objects, and virtual objects. An example of a virtual object is an email – no one has ever held an email in his/her hand because they are not physical objects, but we all know what they are. Dates are virtual objects as well.
2. Objects are made up of **properties** and **methods**. Properties simulate the attributes of real-world objects, while methods simulate the behaviors of real-world objects.
3. You can create a **constructor function** in JavaScript that acts as a template for creating many instances of a specific type of object.

Also before starting, make sure that you have the JDK installed, and the system PATH variable set properly. You’ll be writing the code for this assignment in Sublime Text, so you’ll need to have that installed on your computer as well.

Java makes heavy use of objects. Throughout this course we will be creating our own objects, and we’ll be using objects that are created by other programmers (many of which have been built into the Java language).

In order to design and create your own object in Java, you must define a **class**. A class can be thought of as a blue-print, or a recipe, for creating objects. Before a house is built, a blue-print must be given to the contractors so that they understand all the details of the house (an object) that they must build. Before you can create a batch of cookies (a bunch of objects), you need to have the recipe so that you know exactly how to make them.

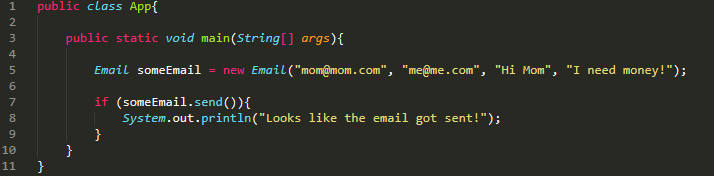
Traditionally, JavaScript did not have classes (but that has recently changed). Instead, you could define a constructor function that acts as the ‘recipe’ for creating a certain type of object. Remember from Web Programming that a constructor functions is used to create/return objects that have specific properties and methods. Java also has constructor functions, but they must be defined inside a class. By the time you finish this activity you will see some similarities, and some differences, in how objects are created in Java compared to JavaScript.

**Coding Assignment (make sure that you have cloned the intro-to-java repository from BitBucket**

1. Create a folder inside your intro-to-java folder (the one that you cloned from BitBucket)
2. Name the new folder **email-sample-app**.
3. Create a file called **Email.java** inside the email-sample-app folder.
4. Enter the code from the screen shot below into Email.java. Make sure to pay extremely close attention to the following as you type in the code:
   1. **Formatting and indentation** (if you don’t know how to format your code, then don’t tell anyone that I was your teacher!)
   2. **Capitalization** – Java is case sensitive!
   3. Semi-colons – Java requires that each statement ends with a semi-colon (JavaScript would allow you to omit the semi-colon in most cases)
   4. **Comments** – Make sure to include the comments when you type the code. They have important information that you’ll need to understand



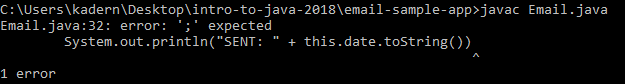
1. Create another file inside the email-sample-app folder, and call it **App.java**. Java programs usually require you to define and use many classes. A Java program requires that one of the classes has a special method called **main**. When you launch a Java program, the main method will automatically be invoked. So you put all your set up code inside the body of the main method (this is where you assemble many of the classes that are used by your program). In our case, App will be the class that we use to launch our program, we’ll put some code in it that makes use of the Email class that we created earlier. Put this code inside App.java:



1. Note that on line 5 of the screenshot, we are invoking the constructor function of the Email class. When you invoke a constructor function of a class, it will return an object, and in this case we are assigning an Email object to the variable someEmail.
2. Now open the command line terminal, then navigate into the email-sample-app folder (by using the **cd** command) and type this command to compile the Email.java class:

**javac Email.java**

1. Note that this command will compile the source code for the Email class into java **bytecode**. The name of the program that compiles your source code is called **javac.exe**, but you do not need to enter the .exe in order to run it. If you don’t have the PATH system variable set up correctly, then you’ll see an error saying that the command line doesn’t know what javac is. If you have an error in your source code (the code that you wrote in the .java files), you might see an error message when you try to run the command. For example, the error below indicates that on line 32 I forgot to put a semi-colon at the end of the line. It takes a little bit of getting used to, but it’s extremely important to be able to make some sense of the compiler error messages. They will indicate a line number where the error occurred. And there will be a carat (^) at the specific place within the line that’s causing problems. But note that if you don’t have any errors, then you won’t get any message from the compiler telling you that you did a good job!



1. Now go ahead and enter this command into the command line terminal to compile the App class:

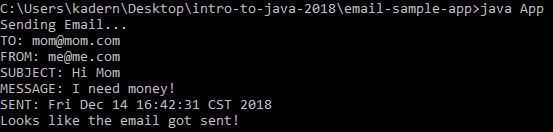
**javac App.java**

1. Now that we have compiled both classes into bytecode, we can go ahead and run our sample app. But before we do, look inside the email-sample-app folder, and notice that there are two new .class files. These are the compiled classes (bytecode) that we created. To run the app, enter this command:

**java App**

Note that you don’t need to add .class to the end. This command tells the java.exe program to launch the App class. In order to run Java bytecode, your computer must have java.exe installed on it. The java.exe program is also known as the **JVM (Java Virtual Machine)**.

The JVM will invoke the main() method when it launches your app. We’ll be seeing main() a lot throughout this course. If everything went well, this is what you should see in the command line:



**Questions**

Do some research on the internet to help you answer the following question. When doing research, keep in mind that there are differences in various programming languages, so be sure to look for answers as they apply to Java.

1. What is a class?
2. What is an object?
3. What is the purpose of a constructor function?
4. What is source code?
5. What is bytecode?
6. What program do you use to compile source code into bytecode?
7. What program do you use to run Java bytecode?
8. What is the name of method that is invoked by the JVM when it runs a Java program?