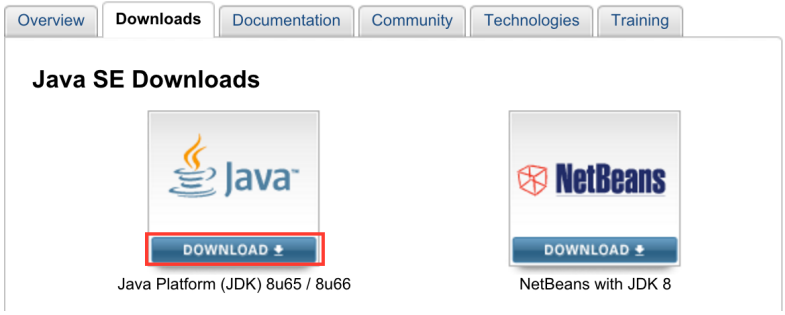
Lab 1b: Setting Up Your Computer

Installing Java

1. You'll need to install the Java 1.8 JDK in order to compile your code for this class. First, head over to the Oracle website.

2. Click the "Download" button for the JDK.



3. On the following page, find the download section entitled "Java SE Development Kit 8u65" and agree to the license. Then proceed to download the binary file

Lab 1b Navigation

Java SE Development Kit 8u65

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

Accept License Agreement Decline License Agreement

Product / File Description	File Size	Download
Linux ARM v6/v7 Hard Float ABI	77.69 MB	jdk-8u65-linux-arm32-vfp-hflt.tar.gz
Linux ARM v8 Hard Float ABI	74.66 MB	jdk-8u65-linux-arm64-vfp-hflt.tar.gz
Linux x86	154.67 MB	jdk-8u65-linux-i586.rpm
Linux x86	174.84 MB	jdk-8u65-linux-i586.tar.gz
Linux x64	152.69 MB	jdk-8u65-linux-x64.rpm
Linux x64	172.86 MB	jdk-8u65-linux-x64.tar.gz
Mac OS X x64	227.14 MB	jdk-8u65-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	139.71 MB	jdk-8u65-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.01 MB	jdk-8u65-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	140.22 MB	jdk-8u65-solaris-x64.tar.Z
Solaris x64	96.74 MB	jdk-8u65-solaris-x64.tar.gz
Windows x86	181.24 MB	jdk-8u65-windows-i586.exe
Windows x64	186.57 MB	jdk-8u65-windows-x64.exe

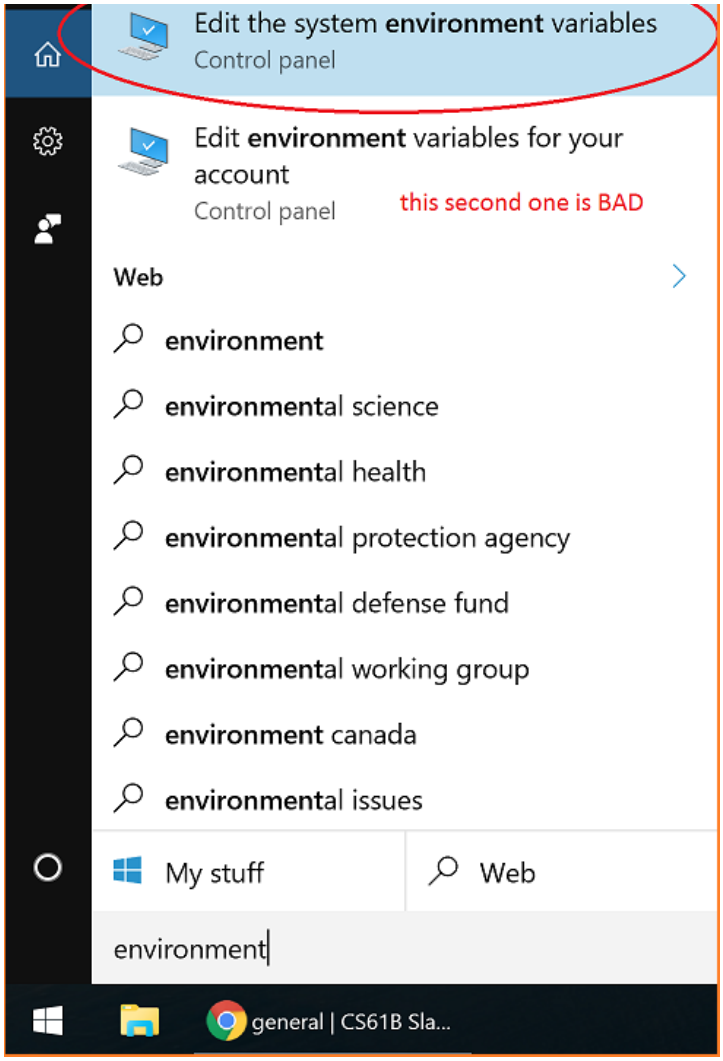
Note (1/19): Java just released "Java SE Development Kit 8u71/8u72." Either of these downloads will also work for this class!

1. Run the install file and follow the prompts to install Java onto your computer.

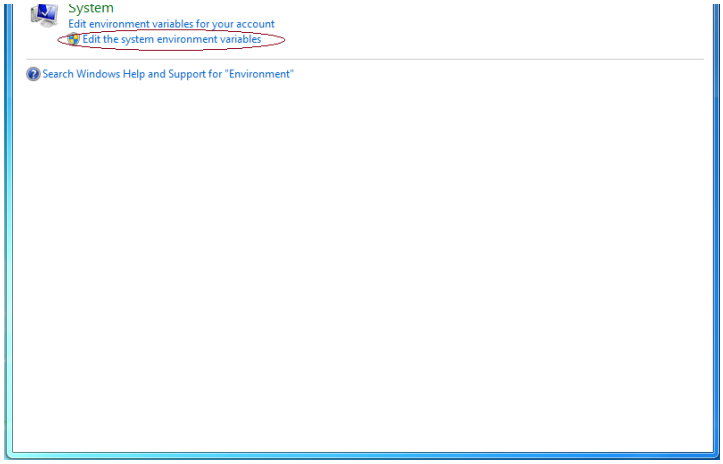
A. Windows Setup

1. First, install Java (instructions provided under the previous Installing Java section).
2. Install python3. We'll be using it to compile more complicated projects later on.
3. Update your environment variables to include java and python. The fine-grain details of this will depend on your OS, but the first step is to open up your **system** (not user) environment variables...
    1. **Windows 8/8.1/10:** Press Windows and type `Environment Variables`. Select "Edit the system environment variables". **Windows 7 and earlier:** Search the control panel for the same thing.

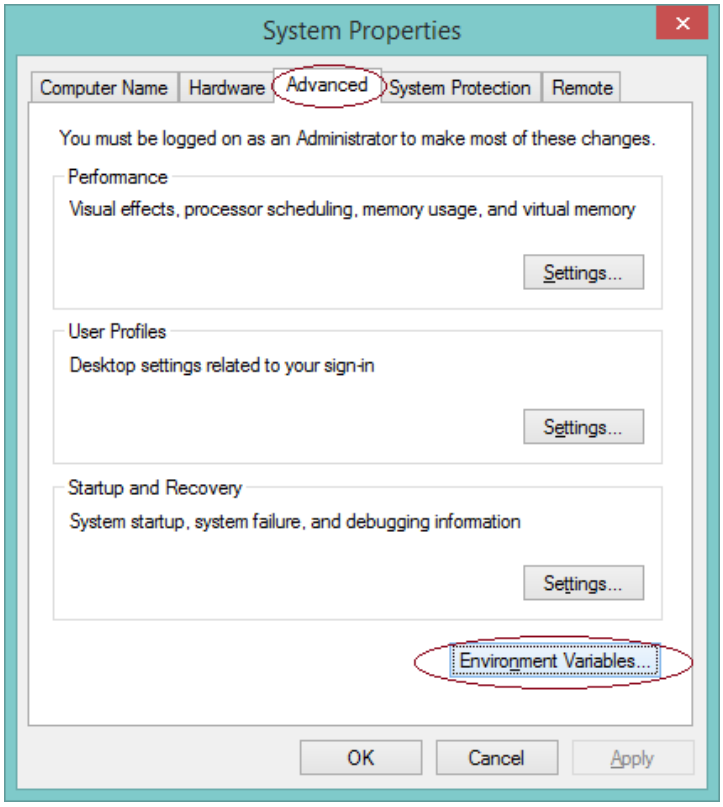
Lab 1b Navigation



Lab 1b Navigation

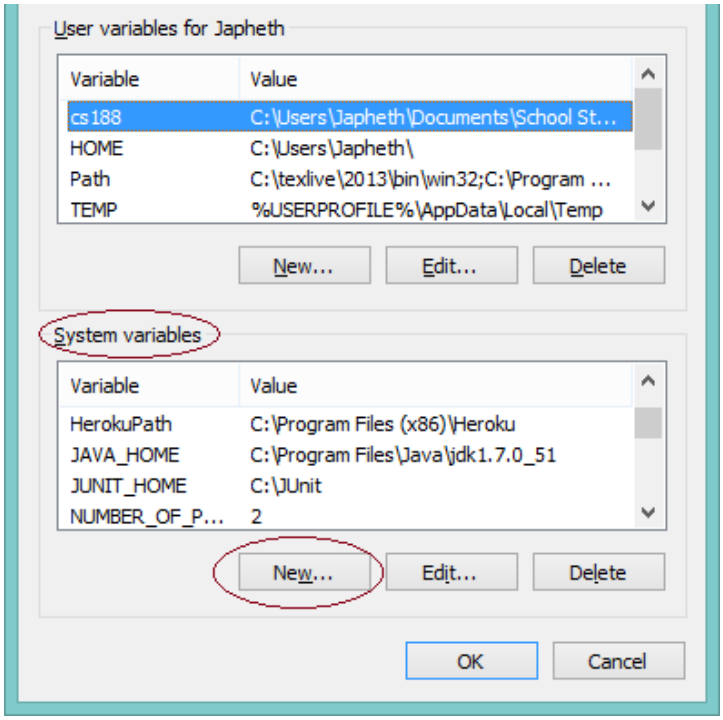


2. Navigate to the "Advanced" tab, and click "Environment Variables...".



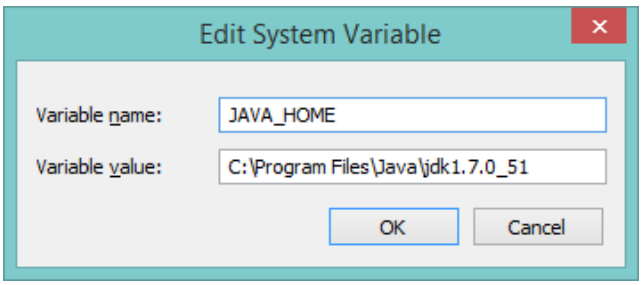
3. Under "System variables" (this section will be unavailable if you are editing *account* or *user* variables), click "New..."

Lab 1b Navigation



4. Define the following variables — click "New..." and use the values specified below as the value of the variable. If the variable already exists, select the variable and click "Edit...", then add the value specified below to the front of the value, followed by a semicolon.

    - **JAVA_HOME:** Set this to the location which you installed Java JDK (for example, `C:\Program Files\Java\jdk1.8.0_65`). Here's an old screenshot for Java 7 (remember, you're installing Java 8!):



    - **PYTHON_HOME:** Set this to the location where you installed Python, for instance, `C:\Python35` or `C:\Program Files\Python35`.

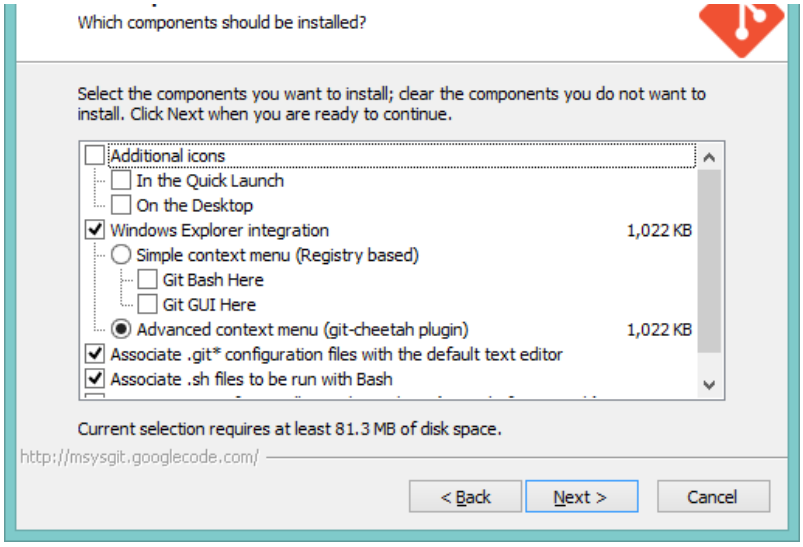
Lab 1b Navigation

`%JAVA_HOME%\bin;%PYTHON_HOME%;` to the beginning of the value of this variable. (The `%` symbols demarcate a path substitution in Windows. Note that there are NO spaces. Putting spaces in Windows path definitions can sneakily RUIN your day!)

5. Save your changes by hitting `OK` on the window. At this point, your `javac` should be working. Close and reopen your terminal (such as Git Bash or Command Prompt) and type in `javac -version` and ensure that it responds `java version "1.8....` . If it claims `javac` isn't a recognized command, something is wrong with your path setup. It should be noted that java installation and git installation are independent, and don't affect each other.

4. Lastly, we'll need to install git. Head over here and grab Git for Windows. Select `Advanced context menu` , so you can also install Git Bash. Git Bash is a bash shell with built-in git support. If you don't have a favorite bash terminal (such as Git Bash or Cygwin), use Git Bash for the meantime. Checking `Windows Explorer integration` will let you do git things upon right-clicking a file or folder. Not required, but might be handy.

Lab 1b Navigation



At this point, check one last time to make sure `javac`, `java`, and `git` are all recognized terminal commands. If so, congratulations! You defeated Windows Java Setup!

B. OS X Setup

1. First, install Java using the instructions provided under the Installing Java section. Downloading the JDK should also provide `javac`, a Java compiler on the terminal.
2. Install Homebrew, a very easy to use package manager. To install, go to your Terminal and enter the following:

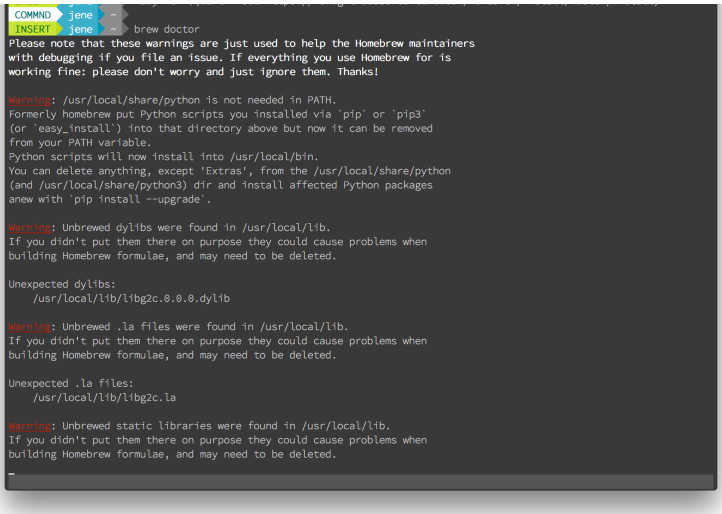
```
$ ruby -e "$(curl -fsSL https://raw.githubuser
```

3. Then, check to make sure brew is working properly on your system by typing:

```
$ brew doctor
```

You may encounter warnings like this, but you should be fine. Proceed to the next step.

Lab 1b Navigation



4. If you encounter a warning asking you to download Command Line Tools, you will need to do this. Please follow the StackOverflow post here.
5. Install python3 and git. You can do this by typing:

```
$ brew install git
$ brew install python3
```

C. Unix and Linux Setup

1. If you are using a Linux/Unix distro, use your package manager (apt-get, yum, etc) to install the Java 1.8 JDK, python3, and git.

First, check to see if Java is already installed by typing:

```
$ java -version
```

If you see "The program java can be found in the following packages" or something similar, Java has not been installed yet. You can install java by typing:

```
$ sudo apt-get install oracle-java8-installer
```

To install python3:

```
$ sudo apt-get install python3
```

Lab 1b Navigation


```
$ sudo apt-get install git
```

Alternatively, follow these beautiful instructions for Ubuntu, Linux Mint, or these beautiful instructions for CentOS, Redhat, Fedora. If you're a different Debian system, you can follow the first link. If you're running a Linux distro that hasn't been mentioned, you probably already know how to install Java 8 and much more!

D. Test Run

Let's try running a Java program to try out your new setup! Just this once, we will tell you to do things without any explanation as to how or why they work. This dark magic will be explained in lab 1 and lecture 1, but for now, is just here for you to check your setup in a quick n' dirty manner.

1. First, open up your terminal (such as Git Bash) and run this magic:

```
mkdir -p ~/temp && cd ~/temp # Forcibly create
```

Then, do a platform specific action to open your file explorer in this directory:

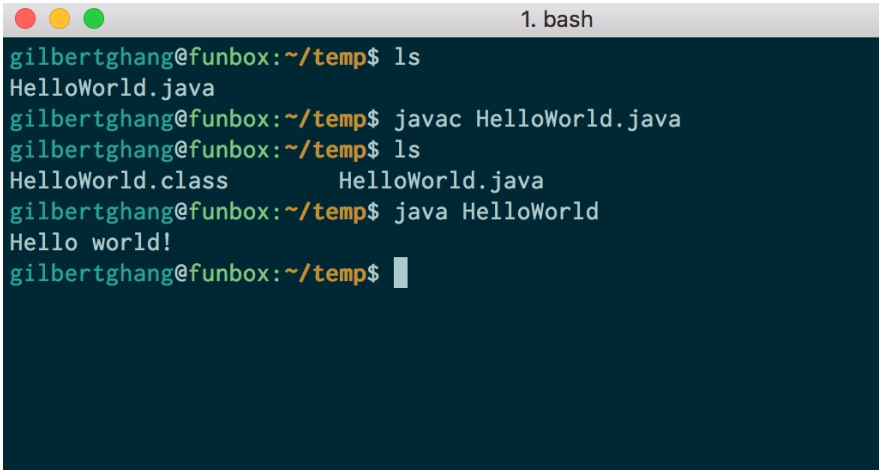
- Mac: `open .`
- Windows: `explorer .`
- Ubuntu: `gnome-open .`
- Linux Mint: `xdg-open .` or `mate .`

In this newly opened directory, create a file `HelloWorld.java` with these contents:

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
```

Lab 1b Navigation

screenshot of your dream goal, a perfect, no-error run-through that indicates your java setup is just fine:



1. In your terminal, enter `ls` (list the files/folders in this directory). You should see `HelloWorld.java` listed.
2. Run `javac HelloWorld.java`. If this produces any output, then something is wrong with your setup. Now if you `ls`, you should see both `HelloWorld.java` and a freshly created `HelloWorld.class` (the `javac` command created this file).
3. Run `java HelloWorld`. It should print out "Hello world!" for you. If it didn't, something is wrong with your setup!
4. You're done! You can also delete the "temp" folder and its contents as you please.