

Assignment #1b

Programming Environment Setup

In assignment #1b, you will download and install several pieces of software. In assignment #1c, you will write your first Java application. Be sure to do assignment #1c!

The software that we will use during the semester will be made available to you on our course web site. This software is free, public domain software. The software is also available for download from various world wide web sites.

1. You will need Sun Microsystem's Java Development Kit (JDK). The JDK is available on our misweb course web site, in both Windows (both 32-bit and 64-bit) and Mac versions. You can accept all default options during this installation. Java may already be installed on your computer, so you may want to delay this step to see if you encounter a problem in step 3. If step 3 (Eclipse installation) works, you already have the JDK installed on your computer.

2. (Steps 2 through 6 are optional. You do not need to install an IDE, but you may if you want to.) Some people like an Integrated Development Environment (IDE), in which they type their programs (probably getting type-ahead hints, color coding, delimiter matching, etc) and run the programs, all within that environment. If you want to do this, you can install Eclipse. You can download a free copy of Eclipse from <http://www.eclipse.org/downloads>

Download the "Eclipse IDE for Java Developers". You do not need the "EE" (Enterprise Edition) version.

3. Once downloaded, install Eclipse as you would any other Windows or Mac program. (Eclipse requires that you have a Java environment installed on your computer, which you likely already do. If you don't, you can download the Java Development Kit (JDK) from our private course web site.)

4. Run Eclipse. The first time you run it, it will ask for your desired location for your projects that you will create. You can use the default location, or customize your own location. Just be sure that you know where you are storing your files, because you'll need to find them later on. Every Eclipse project that you create will be in a separate folder, underneath the location that you create in this step.

5. If you see a Welcome window, close it.

6. By default, Eclipse will format blocks of code with the opening curly brace at the right end of a line, such as:

```
if(i==0) {  
  
}
```

Some people prefer that the opening curly brace be on the next line:

```
if(i==0)  
{  
  
}
```

If you want to configure this, in Eclipse click on Preferences (in Windows Eclipse, Preferences is under the Window menu option; in Mac Eclipse, it's under the Eclipse menu option), Java, Code Style, Formatter.

Click the "New..." button to create a new Profile. Name your new profile anything you want.

The "Edit Dialog" will open automatically. Under the Braces tab, change every setting to "Next Line".

7. An account will be created for you on the mislab server. Your mislab account name will be based on your Banner net id, such as abc123.

You can telnet to mislab.business.msstate.edu to log onto your mislab account. [Technically, telnet is disabled on the mislab server, because it is considered insecure. An alternate protocol, SSH (Secure Shell), is enabled. These instructions use the generic term telnet, but you must actually use SSH on mislab.] PuTTY, an excellent, free Windows telnet/SSH program, is available on our course web site. Installing PuTTY on your computer should be straightforward (simply copy putty.exe to your desired location). On a Mac computer, you can use the built-in ssh program.

Install PuTTY on your computer. When you open the program (after installation), you will see a screen similar to the one shown to the right. Type mislab.business.msstate.edu in the Host Name textfield.

You may want to do one more thing here, changing the default colors to black text on a white background (the default is white text on a black background). If you do want to do this, click on Colours (on the left), then change the Red, Green, and Blue values for the Foreground and Background settings (for white, change the Red, Green, and Blue values to 255, 255, and 255; for black, change them to 0, 0, and 0).

Click the Save button, which will save mislab.business.msstate.edu as your default Host Name, and save your color scheme. Then click the Open button. You will be prompted to login. Your login id is

equal to your Banner net id.

Type your login id, and press Enter. You will be prompted for your mislab password. **Your initial password is equal to the last four digits of your MSU student id number.**

Telnet note: Most telnet programs will not *echo* your password back onto the screen when you type it. That is, when you type your password, you will not see ANY CHARACTERS from your password displayed on the screen. Carefully type your password, and press *Enter*.

Hopefully, you will successfully log in. If you do, you will see a screen similar to the one shown below.

After you log in, use the *passwd* command to change your password. Your new password should be at least 6 characters long; it should not be based on a dictionary word. You might want to consider using digits and/or special characters as well. The Linux *passwd* program is very picky about the password that you select!

To change your password, after you telnet to `mislab.business.msstate.edu` and log in, type:

passwd

You will be asked for your current password (to authenticate yourself -- don't want to let just anyone change your password!), and then asked to type your new desired password -- twice.

Use the *chfn* command to specify your full name.

chfn

Type *logout* to log out of your mislab account.

logout

8. You need to set up your mislab account for web programming. If your user name is *abc123*, your web address will be:

`http://mislab.business.msstate.edu/~abc123`

Throughout the remainder of these instructions, references to *abc123* represent your actual login name.

When you Telnet to mislab and login to your account, you will be in your home directory.

public_html

In general, subdirectories are a great way to organize your files in a meaningful structure. In this class, however, you will put all of your files in your `public_html` directory. This makes it much easier for your teacher to find your files.

Make sure that you are currently in your home directory (that's where you will be placed automatically when you telnet and log in to your account, so you should be there right now). Look at the prompt on the command line. Your prompt should look something like:

```
[abc123@mislabs abc123]$ _
```

This prompt shows (in order) your login id, the name of the server, and the *current* directory (the `_` is the cursor).

Get a directory listing: (type the letter `l`, not the digit 1).

```
ls -l
```

 get a directory listing

The directory listing will show (for example):

```
drwxrwxr-x 10 abc123 abc123 5120 Oct 4 13:41 public_html
```

This directory listing shows several things. First, the first character (`d`) indicates that this directory entry is a *directory*. If the entry related to a file, a dash would appear as the first character. The next nine characters show the *permissions* that apply to this directory. You need to understand Unix directory and file permissions.

Unix directories and files have three sets of permissions: one set for the owner of the file (sometimes called *user* permissions); a second set for accounts which are in a Unix group (an individual group will be created for you and your teacher; use the middle set of permissions to give your teacher permission to access your files); and a final set, called *other*, or *world*, for everyone else. (The acronym *UGO* is sometimes used to refer to User, Group, and Other.) Look at the directory listing again:

```
drwxrwxr-x 10 abc123 abc123 5120 Oct 4 13:41 public_html
```

The nine characters "rwxrwxr-x" indicate the user, group, and other permissions for this directory. The first three characters relate to the user permissions, the next three to the group permissions, and the final three to other permissions. Each entity (user, group, other) can have any combination of three permissions: read, write, and execute. The directory listing above shows:

user permissions: read, write, execute
group permissions: read, write, execute
other permissions: read, execute

When you create a directory or a file, you need to think about what permissions users will need to that directory or file. In general, you want to give users the minimal permissions necessary to do what they need to do. If users need to read your file, give them read permission. If they need to execute the file, give them execute permission. If they need to write onto your file (high risk! be careful!), give them write permission.

Each set of Unix directory permissions consists of three possible rights; each right has a numeric equivalent:

Read = 4 Write = 2 Execute = 1

The permissions for a file (or directory) are set with the *chmod* command, specifying a 3-digit number which relates to:

digit

#1 -- user permissions

#2 -- group permissions

#3 -- other permissions

Each of the three digits is the sum of the permission rights desired for that entity. If you want to set the permissions on your `public_html` directory to (these are common permissions for web-based files):

user permissions: read, write, execute

group permissions: read, execute

other permissions: read, execute

The numeric equivalents to these permissions are:

user permissions: 4+2+1 7

group permissions: 4+1 5

other permissions: 4+1 5

Use the `chmod` command to set the correct desired permissions on your `public_html` directory:

```
chmod 755 public_html
```

 change permissions

Now get a new directory listing:

```
ls -l
```

 get a directory listing

The directory listing will show (for example):

```
drwxr-xr-x 10 abc123 abc123 5120 Oct 4 13:41 public_html
```

his shows that `public_html` is a directory. It also shows the three sets of user/group/other permissions. Finally, it shows the group id (GID) and user id (UID) of the owners of the directory. The GID indicates the mislab group who owns this file (*abc123*); the UID indicates the user who owns the file (*abc123*).

9. You will need to create an HTML file in your `public_html` subdirectory before you will actually be able to access your web site from a web browser.

As a quick exercise, you can use nano (a Linux text editor) to create an HTML file.

Important: Put your HTML files in your `public_html` directory.

First, be sure to move into your `public_html` directory.

```
cd public_html
```

Your prompt should look something like:

```
[abc123@mislab public_html]$
```

This prompt shows your login id, the name of the server, and the current directory. Now, to create your new file, type:

```
nano index.html
```

Type the following HTML code:

```
<html>
<body>
Hello world
</body>
</html>
```

Press `<Ctrl>-x` to exit nano. Answer "y" to "Save modified buffer?", and press `<enter>` when prompted for the file name.

Get a directory listing.

```
ls -l
```

Review the permissions on your `hello.html` file. You may see something like:

```
-rw-rw-r-- 10 abc123 abc123 5120 Oct 4 13:41 index.html
```

You should now be able to get into your web browser and access your sample web page at (for example):

<http://mislab.business.msstate.edu/~abc123>

10. As soon as possible, a list of all Advanced Languages students will be available at:

<http://mislab.business.msstate.edu>

As soon as you create your index.html file, be sure to test your page from the Advanced Languages student list, since that is the page from which grading will be performed.

Step 7 tells you how to install FileZilla on your own computer. If you are using a COBI Lab computer, FileZilla is already installed. You can skip to step 10.

On COBI Lab computers, FileZilla is available in the *Programming Tools* icon, right there on the desktop. Double-click the Programming Tools icon to open it.

11. You will need an FTP (File Transfer Protocol) program to upload your work to your mislab account. If you do not already have such a program, you can download **FileZilla** (Windows) from our private course web site. On a Mac computer, YummyFTP is a very good FTP program. At this time, it costs \$4.99.

If you are using a COBI Lab computer, you should store your files on a flash drive.

Be sure to upload your files into your public_html directory on your mislab account.

12. Download the Java API (Application Programmer's Interface) documentation from the course web site and unzip it on your computer.

You should create a bookmark link in your web browser to the JDK online documentation (the provided documentation gives detailed information about the JDK API -- Application Programmer's Interface -- all of the classes, methods, and properties available in the JDK).

Get into your web browser and select "File", "Open", then "Choose File". Browse your JDK directory structure to find the file a file named java\API\index.html (there are several of them, read on to make sure you open the correct one).

This page is an alphabetical list of Java classes, properties, and methods.

Save a bookmark to this page, then get used to using this online documentation when you program in Java.

13. The programs and files that are included in section 3 of your course packet are available in the /home/bis3523/resources/coursePacket directory on mislab. If you ever want a copy of a program that we go over in class, you'll know where to find it.

14. Finally... an unfortunate fact of life for web developers is that the various web browsers are somewhat incompatible. In this class, the standard is set at the current version of Firefox. That is, your assignments and programming exams will be graded with the Firefox browser. It is highly recommended that you test your programs in this browser so that you can be sure that they will work when they are graded.

15. (Windows only) If you want to use the JDK from the command-line (we will on mislab, and you can on your computer if you want to, but this is totally optional), you need to configure your environment. We will probably not need to do anything from the command line on your computer, because we will probably be able to do everything from within Eclipse, so these remaining steps are totally optional.

Modify the value of your computer's Path environment variable. The Path environment variable tells your operating system locations in which to look for executables (such as javac.exe). For this class, you should add c:\program files\java\jdk1.8.0_20\bin to your Path. You can do this with the following steps:

1. open Control Panel
2. open System
3. click on the Advanced tab
4. click on the Environment Variables button
5. Under System Variables, edit the value of Path

The value of the Path environment variable is a list of directories, separated by semi-colons. Simply add your program files\java\jdk1.8.0_20\bin directory to the end of this value, after a semi-colon. The next time you open a DOS window, the path will be updated.

When you open a new DOS window, you can type:

```
path
```

to see the value of your Path environment variable. Your java\bin directory should be part of that value, and the operating system will look there for executables.

16. Test your Java installation. Get to your command line and type:

```
javac -version
```

This should display the version of the JDK that you just installed. More importantly, it will show that the javac command works. This is the command that you will type in the future to compile your Java programs.

You should see something like:


```
javac 1.8.0_20
```

17. The Java interpreter (JVM) uses an environment variable named Classpath to specify “a list of directories to be searched for .class files”. That is, when you try to run a program, if your program uses any outside .class files, the interpreter will search for those .class files – but it will search only in the directories specified by the value of your Classpath environment variable.

You can check the value of your Classpath environment variable by going to the DOS prompt and typing:

```
set classpath
```

This will display something like:

```
CLASSPATH=C:\Program Files\Java\jre1.8.0_20\lib\ext\QTJava.zip
```

This Classpath specifies only one place that the interpreter should look for .class files – in the one listed location. You probably also want the interpreter to ALWAYS look in the current directory first, then elsewhere. To set this up, you need to edit the value of your Classpath environment variable, using steps similar to those followed to modify the value of your Path environment variable.

In directory parlance, . indicates “the current directory”. You want to insert this in your list before the other specified location, so you should change the value to something like:

```
.;C:\Program Files\Java\jre1.8.0_20\lib\ext\QTJava.zip
```

This tells the interpreter, first, search the current directory for .class files; if you don’t find them there, look in the second location.

Revision #2

Created 9 January 2019 22:52:25 by Aaron Kimbrell

Updated 10 January 2019 16:40:21 by Aaron Kimbrell