Today's Agenda

> To make sure everyone set up their git repo properly

> To look at/discuss your processing sketches

> To talk about text editors

> To look at some more Unix commands

> To talk about .bash_profile and how to customize it

> To compete in a game I made up

> To talk about object-oriented programming concepts

> To write a program in Processing together

> To show you your second assignment

Unix - Common Commands

cd = change directory

```
cd ..                    move up to parent directory
cd ~                     change to home directory
cd /                     change to top level "root" directory
cd /Users/angus/Music    change to a directory by providing the absolute path
cd src                   move into a child folder by providing a relative path
```

pwd = print working directory

```
cd ~                     change to home directory
pwd                      prints out "/Users/angus"
```

ls = list files and directories

```
ls                       print out all files in current directory
ls -alh                  print out all files include hidden files in a long format
ls -R mydir              recursively list all files in mydir
```

cat = print out file

```
cat myText.txt           prints out contents of "myText.txt"
```

echo = print out from command line

```
echo hello there!        prints out "hello there!"
```

Unix - Common Commands

cp = copy file

```
cp fileA fileB      copies the contents of "fileA" into a new file named "fileB"
cp src/* ../dest/   copies all files in "src" into "dest" inside the parent directory
cp –r src/ dest/    copies all files and directories in "src" recursively into "dest"
```

mv = move file

```
mv fileA fileB      renames "fileA" to "fileB"
mv dirA ~/dirZ      moves "dirA" into the home directory, renaming it "dirZ"
```

mkdir = make a new directory

```
mkdir newdir        makes a new directory "newdir" under the current directory
```

rmdir = delete an empty directory

```
rmdir emptydir      empty dir is deleted
```

rm = permanently removes a file or directory

```
rm fileA            deletes "fileA"
rm –rf dirA         VERY DANGEROUS! recursively deletes everything in "dirA"
```

Unix - Common Commands

more = page through text

```
more bigfile.txt        pages through bigfile.txt (by pressing space)
```

head / tail = print out first or last lines of a file

```
head -n 5 file.txt      prints out first 5 lines of file.txt
tail -n 5 file.txt      prints out last 5 lines of file.txt
```

find = find files

```
find .                  recursively list all files from the current directory
find ~/logs/*2010*      list all files in the logs dir that contain "2010"
```

grep = print lines matching a pattern

```
grep "A" cell*          print lines containing "A" from files starting with "cell"
grep -i "a" file.txt    print lines containing "a" or "A" from file.txt
grep -v "a" file.txt    print lines that don't contain "a" from file.txt
```

man = get the manual for a command

```
man grep                page through documentation about the grep command
```

diff = compare two files

```
diff f1.txt f2.txt      output the lines where the files don't match up
```

Unix - I/O Redirection

pipe | = funnel the output of one program into the input of another one

```
    cat bigfile | more           pages through bigfile
    ls mydir | grep a*           only prints out filenames that start with an "a"
    cat afile | grep a* | more   pages through lines that start with "a"
```

redirect output > = funnel the output of a program into a file

```
    cat myfile > text.txt        write the output of "cat myfile" into the file text.txt
    cat myfile >> text.txt       append the output to the end of text.txt
```

Unix - Connect to Other Machines

ssh = open a secure shell to a remote server

```
ssh my.name@my.server        logs you into your home directory on "my.server"
```

sftp = secure file transfer protocol (using ssh)

```
sftp my.name@my.server       browse through and transfer files from "my.server"
```

Unix - Misc.

There are hundreds of other more specialized helper programs bundled with the various Unix distributions (FreeBSD for OSX).

You can see what these are by typing

> echo $PATH

from the command line, which will list all of the places the terminal will look for a command

you can list out the contents of one these directories, e.g.

> ls /usr/bin

most of these are little C programs and you can find out how they work by looking at their *man* page.

If you want to know where a program you use lives, you can use the *which* command, e.g.

> which ls *//prints out /bin/ls*
> which which *//prints out /usr/bin/which*

Unix - bash profile

The actual terminal you use is handled by a program called a "shell." Most Unix shells are quite similar, at least in their basic functionality. Cygwin and OSX both default to a shell called "bash", which is by far the most common shell in use.

You can customize your shell by editing your "profile" which lives at ~/.bash_profile

The most common things to do are to define **environment variables**, which includes the default search path to Unix programs, and to set **aliases** for common commands.

To set an environment variable in bash you use the "export" command. For example:

```
> export MYNAME='angus'     //defines your variable
> echo $MYNAME              //prints 'angus' to the console
```

(show example of editing PATH variable)

To set up an alias in bash, you use the "alias" command. For example:

```
> alias cd201b='cd /Users/angus/teaching/201b/website'  //creates the alias
> cd201b                                                //moves to my 201b directory
```

Unix - archiving/compressing files

There are various methods to bundle and then compress a file. We'll look at a common archiving program called "tar", which is short for "tape archive" but has outgrown its original format...

You create an archive by specifying the files and directories you want to store:

> **tar -cvf myArchive.tar file1.txt dir1 file2.txt** *//as many as you want*

The option "c" = create, "f" = file, "v" = verbose

The first argument ("myArchive.tar") = the name of the archive file we're creating

The other arguments indicate what we are going to put into that archive.

so our command means:

create an archive named "myArchive.tar" with the files file1.txt, file2.txt, and all the files in dir1, and give me verbose output as you do it.

We extract an archive by using the "x" = extract option instead of the "c" option

> **tar -xvf myArchive.tar** *//unpacks the archive into your current directory*

Unix - archiving/compressing files

You can also just check to see what's in an archive using the "t" option

> **tar -tvf myArchive.tar** *//prints out what's in the archive without actually extracting*

There's an archiving program called "zip" (at least on OSX) which does the same thing as tar, except that it compresses the files as it goes (use "unzip" to extract and uncompress)

There's also a "compress" command which will compress a .tar file (or any file). Compressed files have a .Z at the end of them. For instance, "myArchive.tar.Z". You use "uncompress" to uncompress the file.

Editors

from the **terminal**:

```
cat > filename.txt      = write from stdin into filename.txt
cat >> filename.txt     = write from stdin into filename.txt, appending to it if there's
                          already stuff in there
pico or nano            = simple terminal editor
vi / vim                = an editor with different modes (this is my main editor)
emacs                   = a more fully featured terminal editor
```

from a **GUI**:

```
Notepad, Wordpad, TextEdit       = super basic gui editors
TextMate, Text Wrangler, BBEdit  = simple editors with nice stuff for programming
gvim (or macvim)                 = graphical version of vim
```

via an **Inegrated Development Environment** (we'll start using IDEs later on...):

```
XCode              = OS X, for C/C++/Objective-C/iPhone/iPad
Eclipse            = cross platform, mainly for Java and JVM languages, web languages
Visual Studio      = Windows, .NET languages
Netbeans           = cross platform, mainly Java and JVM, gnu C/C++, web languages
```