Terminal Basics

Getting Setup

If you are a Windows user, use Git Bash (included with Git) for all Terminal interactions. To launch, Start Menu -> Programs -> Git -> Git Bash

Install Git Bash at:
http://git-scm.com/downloads

What is the Terminal?

- An interface to a 'lower-level' of your computer than the Graphical User Interface (GUI, or 'point and click')
- Similar to older computer systems without a GUI
  - Every command must be typed in
  - The mouse is not used to interact with the terminal

Why are we learning about the terminal?

- Some programs needed for development only run from within the terminal
- More advanced programming isn't possible without using the terminal, so it's good to start when you start learning
- This way, you can practice your skills as you go before they become crucial and a stumbling block
- Your friends will think you're a `l33t h4ck3r` when they see you typing commands into a black screen!

What can you do with the Terminal?

- Quickly navigate around your computer's filesystem
- Run command-line-only programs like Git and your own web server
- Monitor the processes running on your computer and have control over them

Basic Terminal Commands

Before you learn more advanced things, you should first master how to navigate around your filesystem using the terminal.

To open the Terminal on a mac, press command space, type Terminal, then Enter.

On a Windows machine, use Git Bash (downloadable at git-scm.com)

A Note

If you see text with a dollar sign before it, it means to enter it into the Terminal.

```
$ pwd
```

means to enter the word 'pwd' into your Terminal and press enter.

If you see text directly after this, it is "sample output", meaning what you might see after you enter the command.

Don't be afraid to press enter[1], your computer won't explode!

[1] Unless someone tells you to type sudo `rm -rf /`, which will delete your entire Hard Drive.

`pwd` - print working directory

- Consider pwd the "north star" - it will always tell you what directory you're currently in.
- What directory are you in right now?

```
$ pwd
/Users/zachfeldman
```

`ls` - list files in current directory

- This is another way of figuring out where you are
- If you know what files are in the directory you're searching for, it's a confirmation of your location in the filesystem

```
$ ls
Applications     DropBox     Movies
Desktop          Library     Pictures
Documents        Music       Public
```

`cd` - change directory

- If typed with no input, will take you to your home directory

```
$ cd
$ pwd
/Users/zachfeldman
```

- If given an input directory right afterwards, `cd` will take you there if that directory is in the directory you're currently residing in

```
$ cd Desktop
$ pwd
/Users/zachfeldman/Desktop
```

`cd` - change directory

- You can also give an absolute location to get anywhere specific

```
$ cd /Users/zachfeldman/Documents
$ pwd
/Users/zachfeldman/Documents
```

- In the Terminal, a dot represents the current folder and two dots represents the folder this folder is in
- You can `cd` to the folder above the one you're in like so:

```
$ cd ..
```

`cd` - change directory

- If you want to go up multiple levels

```
$ pwd
/Users/zachfeldman/dev/project-one
$ cd ../..
$ pwd
/Users/zachfeldman
```

- The tilde (~) is a shortcut for your home directory

```
$ cd ~/Desktop
$ pwd
/Users/zachfeldman/Desktop
```

Tip: Tab autocompletion

- Try pressing tab after typing the first letter or two of the directory you're trying to go to
- Keep this trick in mind later on when entering files or folders as input to terminal commands

touch - **create a blank file**

- Touch will create a completely blank file in a directory for you
- This can be nice when 'scaffolding' a new project

```
$ touch index.html
$ ls
index.html
```

Tip: File and directory names

- If your file name contains a space, it must be escaped, otherwise the Unix interpreter will assume that you meant to enter the next command.

```
$ cd /My Awesome Folder #WRONG
$ cd /My\ Awesome\ Folder #RIGHT
```

- This is why it's better to name directories with underscores, however, you can also use tab to autocomplete filenames with spaces in them

Creating directories with `mkdir`

- To create a directory, use the `mkdir` command

```
$ mkdir directory_name
```

`rm` - remove

- **There is not a recycle bin for files removed this way, so be careful, it will delete files *permanently***
- Takes a file name as input to delete ***permanently***

```
$ rm file.txt
```

- To remove folders and everything in them ***permanently***, use a recursive `rm`

```
$ rm -r folder/
```

Tip: Command line flags

- Wondering what the `-r` meant after the `rm` command on the last slide?
- `-r` is a **command line flag**, an option sent to a command
- You could also append an `f` to the flag to force removal:

```
$ rm -rf folder/
```

Exercise: Getting to know the Terminal

- Open up the terminal and test out the commands you've just learned
- Try creating the scaffolding for a basic website inside of your `Desktop` folder using only Terminal commands:

```
# directory structure
my-project
  index.html
  styles
    style.css
```

Quickly viewing files

- less - a program included with unix-based systems that lets you easily view text files

```
$ less filename.rb
```

- To navigate, use the arrow keys. To quit out of less, use the q key

Copying files

To copy a file, use the aptly named cp command.

```
$ cp index.html index2.html
$ ls
index.html index2.html
```

Renaming or Moving files

To move a file, use the mv command:

```
$ mv index.html ~/Desktop/project_folder
```

To rename a file, mv is also used:

```
$ mv index.html about.html
```

Tip: Re-entering commands

- To re-enter a command you've entered on the command line recently, press the up arrow
- Your history of commands is preserved within the session you're currently in
- Once you restart the terminal, you'll lose this history