layers and layers

walk through going to github or looking at files in *excruciating* detail.

Lessons:

- there are different levels of abstraction.
    - One needs to be able to move between them, especially when things go wrong or when programming.
    - One needs to know when to focus on one level and ignore the others.
- There are layers of software and hardware
    - No need to deal with higher levels, just use the command line (why?)

OS

Operating system controls computer resources: hardware, software, file system, Input/output

hardware control

- graphic I/O
- keyboard/mouse
- printers
- communications (ethernet and wifi)

Show processes with *ps -ef*

Graphical Interface control

- OS monitors mouse, keyboard, etc.
- Responds to changes: Execute a program, re-draw the screen

Why not just execute programs without the GUI?

executing programs

Describe steps in executing a program

- find the program file
- do what the program says to do

Questions you should ask:

- how does OS know where programs are?
- how does OS know what kind of file a file is?

File system

- hierarchical: directories/folders and files (directories ARE files!)
- some directories have special names

(more below in *Terminal* section)

Terminal

Describe operating system.

terminal program is a way to speak directly to the OS

Bash Stuff

What is a shell?

commands

- navigate directories (cd, ., .., ~, pwd)

Compare directory information in *terminal* versus *finder*

- create directory (mkdir)
- create files (touch, cp, mv)
- rmdir, rm