What is Unix?

- Developed by AT&T, released 1969
  - Same group that developed C
- Evolved into many OS's
  - Linux, BSD, Solaris, Mac OS X, etc...
- Originally designed for mainframe use

Overview

- History
- Unix vs Linux
- Command Line
  - Usage
  - Scripting
  - Tricks
- Conclusion

History

- Developed by AT&T, released 1969
  - Same group that developed C
- Evolved into many OS's
  - Linux, BSD, Solaris, Mac OS X, etc...
- Originally designed for mainframe use

Unix vs Linux

- Proprietary
- Written for a single architecture
- Heavily tested according to standards
  - http://www.unix.org/
- Open Source
  - Packages extend OS to different architectures
  - “Peer reviewed”
  - http://www.kernel.org/

Command Line Interface

- Very quick
- Powerful
  - GUI vs CLI
- Steep Learning Curve
- Made easier with “help”
Basics

- Structure of a command
  - `<command> <arguments>`
- Arguments vary by program
- Flags / switches
  - Used to tweak the program at runtime
  - Used to redirect output to a file

Basics pt. 2

- MAN pages
  - Like a read me, helpful for learning flags
- Input/output redirection
  - `|`
  - `<`
  - `>`
  - `>>`
  - `&`

Output redirection

- Very useful to “unlock” the command line
  - Parse input
  - Link multiple commands together
  - Form complex “programs” using smaller programs as build blocks

Output redirection

- How is it done?
  - Unix and Unix-like systems' treatment of objects
  - Everything is a file
  - Redirects output from one place to another

Redirect to a file

- The `>` operator will redirect stdout to a file by default
  - A great way to capture the output of a program for later use.
- Also able to redirect stderr to files or even stdout
  - Use it to capture the error caused by compiling and view them in less
  - i.e. `g++ main.cc 2>&1 | less`

Redirecting Errors

- For example template errors
  - THE HORROR!!!!!!
Redirecting Errors
- Use head to see the first few lines

The Almighty Pipe
- The | operator allows the output of one program to be the input for another
  - Not 1 or 1 but ( shift + \ )
  - Extremely useful
- For most people the majority of redirection they will use

Fun Example
- Let's find out where our largest files are
- `du` - shows you disk use
  - The flag -h makes the file sizes human readable
- `grep` - very powerful program for searching text
- `du -h | grep M`
  - Searches the output of `du -h` for capital M's

Input redirection
- `<` uses a file as input for a program
  - I don't really use it, but I am sure there is something useful that can be done with it
- Used to check a program against known test cases

Fun Example

grep
- grcp (global / regular expression / print)
  - Searches through text files and prints every line with the search term
- Flags can vastly change the running
  - -i inverts the command, prints lines without the search term
  - -I case insensitive
  - -r recursive, follows directory structures
**less**

- Allows scrolling through output, useful for:
  - Looking at documentation
  - Quickly scanning text
  - Dealing with large amounts of text output
- “less is more”
  - more doesn’t allow scrolling up, less does
- Try “less <some random file>

**Other misc. commands**

- `sort <file>`
  - Prints the file sorted to stdout
- `uniq <file>`
  - Prints all unique lines of a file to stdout
- `cat <file>`
  - Prints the file to stdout
- `diff <file1> <file2>`
  - The differences between the two files

**Finding Out about tricks**

- Command Line Fu
  - [http://www.commandlinefu.com/](http://www.commandlinefu.com/)
- Old beards
  - Just ask people who are more advanced than you.
- Google
  - Forum threads are very helpful

**Some tricks**

- Use tab
  - Auto completes commands for you
  - Auto completes file names for you
  - Speeds up your command line usage

**cd**

- `cd`
  - Quickly return to previous directory
  - Great for moving from deep in one directory to another
- `pwd`
  - Prints the working directory to the screen

**Quickly add file extension**

- Brace expansion
  - Try “echo a{d,e,b}e”
  - Try “mkdir -p assignment/{bin,build,src}”
- Can be used with the `mv` command
  - Try “mv file1,.txt”
- Saves on typing
  - Try it with the `cp` command to make a backup
**ps**

- Many flags and syntaxes for flags
  - `-fl` prints all processes
  - `-elf` prints all processes and thread information

**kill**

- Sends signals to programs
  - `-9` stops the execution completely, cannot be blocked
  - Default signal sent is the terminate signal
  - `kill <pids>`
    - Sends the SIGTERM to all the pids

---

**Fun Scenario**

- CS 446 program out of control
  - Forking in a loop is bad
  - Have a large number of pids to kill, don’t really feel like typing them all
    - `ps -ef | grep <program name>`
    - But you still need to manually enter them
  - Is there a smarter way?

**Fun Solution pt1**

- Of course!
  - Use some of `ps`’s flags to only print pids
  - `pipe those into kill ... or can we?`
  - `ps -C cs446ass1 -o pid=` | `kill`
  - Kill requires command line arguments
    - Doesn’t like taking commands from stdin
  - What will we do?

**Fun Solution pt2**

- Back ticks!
  - Hit the tilde without shift
  - Called command substitution
  - `kill `ps -C cs446ass1 -o pid=``

**Scripting**

- Allows you to automate tasks
  - i.e. recursively traversing directories moving some kind of file some where else
  - Handled by the shell
    - aka the same command line interface you normally use
The Shebang

- Tells the shell what interpreter to use
  - Could be perl, python, ruby or ...
- Bash!
  - The shell itself
  - Other shells do exist
- #!/bin/bash

Scripting

- Can get extremely complicated
- Use the Bash reference manual
  - Super handy

cron

- Used to schedule scripts or programs to be run at certain times
- Useful for reminding yourself of special dates
  - Write a script that uses mail to email you a message about the day
  - Make cron run that script!
- Used by the OS to keep things in order

Permissions

- chmod
  - Changes permissions
    - Read the man page, the gist of it is 3 octal numbers define the permissions.
    - Learn how they are stored
sudo

- Substitute user do
  - Not super user do 😞
- Use it rather than becoming root
- Has elevated permissions
- Fun to say.

Quick trick

- Sudo !!
  - Runs the previous command with sudo
  - Useful if you forget to use sudo
    - I do all the time...

Conclusion

- Ready to take a dive off the deep end
  - Learn a command line based editor
    - I suggest vim or emacs
  - Learn a command line tool
    - sed, awk, grep, gdb, etc.
- Keep practicing to gain confidence
- Try to use GUIs as little as possible
  - Forces CLI comfortability