UNIX/Linux Tutorial

Based on https://users.cs.duke.edu/~alvy/courses/unixtut

FORDHAM

CISC 5004 - Programming C++
Sam Kamens
Fall, 2022

Introduction

- What is UNIX?
  - Operating System
    - Set of programs that make a computer work (see also CISC 5595)
    - Stable, multi-user, multi-tasking system
    - Supports GUIs, but we will focus here on shell access
- Types of UNIX
  - Gnu/Linux
    - Open Source Operating System
    - Layered with Gnu tools
  - MacOS X
    - Based on a different variant of UNIX (FreeBSD)

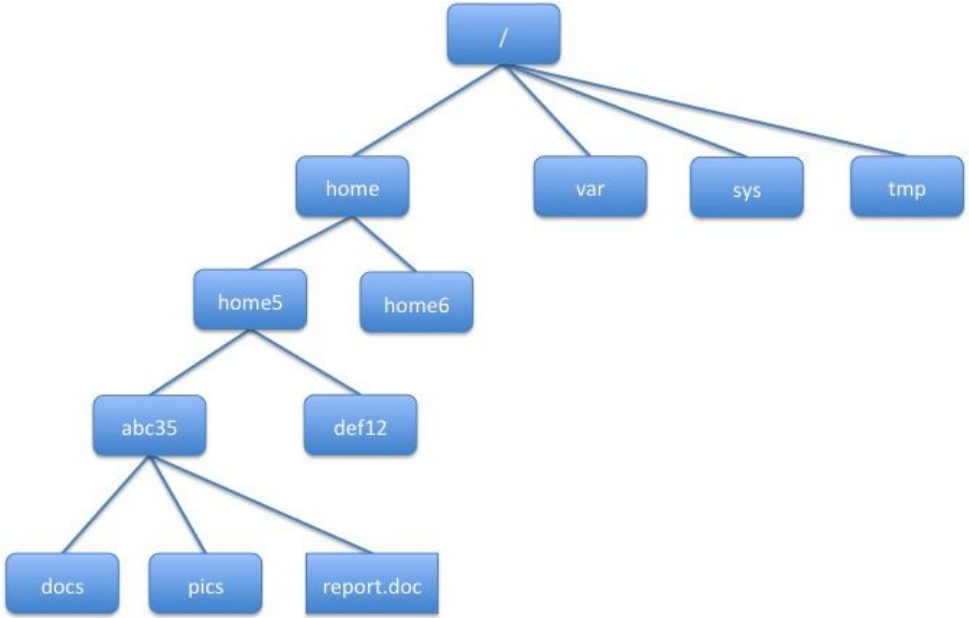
Parts of the Operating System

- Kernel
  - Core software that makes the operating system work
  - Privileged execution
- Shell
  - Interface between the user and the kernel
  - Command line interpreter
  - Shells have helpful features like
    - Filename completion
    - History
  - Most Linux systems use bash (Bourne Again Shell)
    - Tutorial

Files and Processes

- Process
  - An executing program identified by a unique process ID (PID)
- File
  - A collection of data
  - Directories are also files

Directory Structure



Starting a Terminal

- erdos
  - Shared Linux system available to Fordham students and faculty
  - Obtain an account (let me know if you need one)
  - Connect via:
    - Linux Login in the computer lab (LL612)
    - Remote desktop
    - Secure Shell (SSH)
- MacOS X
  - Open a Terminal
- Windows
  - Install and use WSL

File System Commands

- `ls`
- `mkdir`
- `cd`
- `pwd`
- Special directory shortcuts
  - Current Directory: `.`
  - Parent directory: `..`
  - Home directory: `~`
- Argument options
  - aka “command line arguments”

Modifying the filesystem

- `cp`
- `mv`
- `rm`
- `rmdir`

Viewing Files on the Screen

- `clear`
- `cat`
  - (catenate)
- `less`
- `head`
- `tail`

Searching file contents

- Using `less`
- `grep`
- `wc`

Redirection

- Most UNIX commands write to “standard output” (usually the screen) and read from “standard input” (aka the keyboard)
- cat
  - With an argument, it takes input from a file
  - Without an argument, takes input from the keyboard
- Redirect output
  - >
  - >>
- Redirect Input
  - <
- Can redirect both input and output

Pipes

- Connect the output from one command to the input of another

Wildcards

- *
  - Matches 0 or more characters in a file or directory name
- ?
  - Matches a single character

File and Directory Naming conventions

- Do not use special characters, such as `/ * & % ( )`
- Generally avoid spaces
- Usually lowercase (although the file system is case sensitive)
- Use an appropriate suffix
  - `.txt`
  - `.h`
  - `.cpp`

Getting help

- man
  - Also available online
- apropos

Processes and jobs

- `ps`
- Running a job in the background
  - `&`
- Placing a foreground job in the background
  - `Ctrl-Z (^Z)`
  - `bg`
- `jobs`
- `kill`
  - `kill -9`

Other useful commands

- `file`
- `diff, sdiff`
- `find`
- `history`

Environment Variables

- Variables set by the system and/or the shell
- You can customize your settings in the file `.bashrc`
- Key variables:
  - $HOME - your home directory
  - $PWD - current directory (usually the same as the result of the pwd command)
  - $PATH - Set of directores the shell searches to find commands that are not specified with a full path name
    - `$ echo $PATH`
      `/bin:/usr/bin:/home/skamens/bin:/usr/local/bin`

Editors

- vi/vim
  - Interactive tutorial
- Emacs
  - Tutorial: Run emacs, and type Ctrl-h t