Globbing, tmux

CS14 - S26

Globbing and wild cards

- Most shells do *filename expansion* a.k.a. Globbing for certain special characters a.k.a. wild cards.
- Allows most basic utilities to operate on multiple files/directories whose names match a certain pattern with a single command
- Globbing is distinct from, but related to, Regular Expressions (which we'll cover next week).
- NOTE: only works on file/directory arguments, not text, flags, etc..

* - match any number of characters

- `ls *`
    - List a directory’s contents and all of its subdirectories’ contents.
- `mv *.h ../lib`
    - Move every file ending with .h to lib
- `grep “var_name” *.*`
    - Search for var_name in every file that has a period in its name
- `wc */*`
    - Call wc on every file in every subdirectory of this directory

{} - match list of comma separated patterns

- `nvim my_lib.{h,cpp}`
    - Open my_lib.h and my_lib.cpp in nvim
- `ls lab{04,05,06}`
    - List contents of lab04, lab05, and lab06.
- `rm *.{o,aux,log}`
    - Remove every file ending with .o, .aux, or .log
- `grep “var_name” {circle,rectangle}.{h,cpp}`
    - Search for var_name in circle.h, circle.cpp, rectangle.h, and rectangle.cpp

? - match any single character

- `cat ???`
    - Print contents of every file with a name that is exactly three characters

Exercises

Copy the following directory:

```
/home/ckazer/public/cs14/inclass/globbing_exercises
```

1. Write a command that saves the name of every file in a directory ending with .h to a file named “headers”.
2. I’ve broken “movies.tsv” into multiple files. Write a command that sorts all of the files together by release year, and save the result to “movies_by_year.tsv”.
3. Recall that lab write-ups are put in the `labs/` directory as adoc files, but some are contained within a single level subdirectory. Write a command that saves the last line of each write-up to a file named “last_lines.txt”

tmux - Terminal Multiplexer

tmux starts a “server” program that manages terminal login sessions separate from the default one (typically X11 on CS machines).

tmux supports:

- Multiple sessions
    - Each session is like a separate login
- Multiple windows in a session
    - Each window is like a different terminal program in a session
- Multiple panes in a window
    - Each pane is a separate command line within a window
- **Detaching and reattaching sessions**

tmux example

- One session for each project/major task
- One window for each problem/sub task
- Multiple panes when combining editing/testing/using man pages

Most useful on a server that runs 24/7 where you can detach at any point and reattach the next time you are working.

tmux configuration – .tmux.conf

Configuration goes in

`~/.tmux.conf`

Example configuration provided in

`/home/ckazer/public/cs14/inclass/example_.tmux.conf`

Contains some modifications to make tmux behave better when using (n)vim.

Using tmux

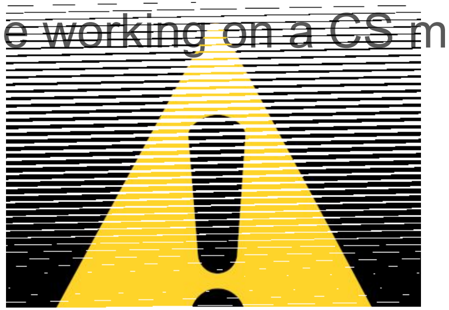
`$ tmux` to start a session

- Each session starts with a single window with a single pane.
- To issue tmux commands, press and release the prefix key (default **Ctrl+b**), then enter the particular command key
- <prefix> c – create a window
- <prefix> n – go to next window <prefix> p – go to previous window
- <prefix> s – split pane horizontally <prefix> v – split pane vertically
- <prefix> →←↓↑ – change pane
- <prefix> [– enter copy mode, which allows scrolling. **q** to quit copy mode.
- <prefix> d – detach session
- Ctrl-d – exit window

Check cheatsheet on course website for more commands

A note about tmux on CS Network Machines

CS network uses a network file system to let you see your files on any machine. However, **each tmux session is bound to a single machine**. So, make sure to exit tmux once you are done working on a CS machine to avoid dangling login sessions all over the place!



But you can (and should!) leave a tmux session running on your cs14vm!