Tmux & Other Tools

Jake Zimmerman

October 22, 2016

Package Managers

Package managers make installing software easy

Package managers…

- let us install software from the command line.
- are usually specific to an operating system.
  - macOS? → `brew`
  - Ubuntu? → `apt-get`
  - Arch? → `pacman`
- usually ship along side the operating system.
  - exception: `brew` on macOS needs to be installed separately

Using your package manager[1]

Package managers all have different commands and syntax.

- macOS
  - First download & install: `http://brew.sh`
  - `brew install <package>`
- Ubuntu
  - `sudo apt-get install <package>`

For other systems, take a second to look up how to install packages using your package manager.

[1] Windows users: the rest of this talk assumes macOS or Linux.

`tmux`

`tmux` is a terminal multiplexer

From Wikipedia:

> `tmux` *is a software application that can be used to "multiplex" several virtual consoles, allowing a user to access multiple separate terminal sessions inside a single terminal window or remote terminal session. It is useful for dealing with multiple programs from a command-line interface.*

TL;DR:

- You can create tabs running different commands
- You can create splits (horizontally and vertically) running different commands

Launching `tmux` creates a “session”

```
# Create a session
$ tmux

# Create a named session
$ tmux new –s gpi

# Attach to last session you detached from
$ tmux a

# Attach to any named session you detached from
$ tmux a –t gpi
```

Note that we can “detach” from sessions. In fact, if we get disconnected: `tmux` automatically detaches us!

We use `tmux` with a "prefix"

Nearly all `tmux` commands look like:

- Press `<prefix>`, release, then press `<key>`

`<prefix>` is `Ctrl + B` by default

- You can rebind this if you want (ex: I use `Ctrl + F`)

Examples:

- `Ctrl + B, %` → create vertical split
- `Ctrl + B, "` → create horizontal split
- `Ctrl + B, c` → create new tab

Why use `tmux`, then?

- Never lose your work from a shaky network
  - Just re-login and re-attach to your session
- Split the screen
  - 80-character line length enables screen splitting
- Avoid re-typing your password
  - Creating a new tab or split doesn’t require a password
- Customize `tmux`
  - You can choose the fastest keyboard shortcuts that work for you

`tmux` config & cheat sheet

It's dangerous to go alone! Take this.

This config file makes using `tmux` easier to start.

- Starter `tmux.conf` file

This cheat sheet has everything else you want to know.

- `tmux` shortcuts & cheatsheet

`fzf`

`fzf` is a command line fuzzy finder

- TL;DR: look at this awesome demo
- To install: `https://github.com/junegunn/fzf`
    - Just go ahead and answer "yes" to everything

Using `fzf` is crazy powerful

- Press `Ctrl + T` to fuzzy-find files in the current folder.
    - Type things to filter results
    - Use arrow keys to highlight result
    - Press `Enter` to select it
- Fuzzy-patterns let you omit characters:
    - `"fzf"` matches `"fuzzy-find.txt"`
    - `"itl"` matches `"insert_tree_in_leaf.c"`

`fzf` is great

- The previous slide is already enough to love
- But there's even more!
  - Color themes
  - Custom keybindings
  - History search
  - `my-command $(fzf)`
  - It's all documented: `https://github.com/junegunn/fzf`
- For the adventurous: my `fzf` config
  - `https://github.com/jez/dotfiles/blob/d540b50/util/fzf.zsh`

`ag`: The Silver Searcher

`ag` is like `grep` but for humans

- `ag` is similar to a tool called `ack`, but faster
- `ag` is recursive by default
  - avoid having to always type `grep -r ...`
- `ag` groups results by file
  - makes it easier to see where the results are from
- `ag` prints one line per source line
  - `grep` will duplicate a line if there is more than one match in it
- `ag` lets you search specific file types
  - `ag --html --css --js` will search only HTML, CSS, and JavaScript files
  - `ag --cc` will search only `*.c` and `*.h` file types

Installing & Learning

- Install from your package manager
  - The package isn't just called `ag`
  - See here for the name on your system
- Examples
  - `ag foo`
    - searches recursively for "foo"
  - `ag --python main`
    - searches recursively for "main" in Python files
- Learn more about `ag`
  - `ag --help`