

Vim presentation

Dylan Pang

7 August 2020

Why vim?

- ▶ It is **lightweight** (Compared to IDEs/Modern text editors)
- ▶ **Multi-mode text editing** means that the shortcuts are easy to hit and remember, **NO MORE CTRL!**
- ▶ Heavily extensible with **plugins** and **personalised configs**
- ▶ **Completely keyboard driven**
- ▶ **Pre-installed on all unix systems**

Modes

Look at the bottom left of your screen

1. Normal mode: Default mode for you to input commands and move around (Hit 'Esc' in the other 2 modes)
2. Insert mode: Mode for you to type (Hit 'i' in normal mode)
3. Visual mode: Used for selecting portions of text for manipulation (Hit 'v' in normal mode)

Lesson

```
cd ~
```

```
wget https://raw.githubusercontent.com/dylanpjt/unix_workshop/master/demo.txt
```

Default .vimrc

```
cd ~
```

```
wget https://raw.githubusercontent.com/dylanpjt/unix_workshop/master/.vimrc
```

Movement

In normal mode

1. gg => top of page (!!)
2. G => bottom of page (!!)
3. 0 => start of sentence
4. \$ => end of sentence
5. w => jump from **w**ord to **w**ord (!!)
6. b => jump from word to word **b** (!!)
7. e => jump from end of word to end of word
8. / or ? => forward or backward search (!!)
9. f => **f**ind first occurrence of character (!!)
10. zt, zz, zb => scroll to top, middle, bottom (!!)
11. hjkl => HIGHLY UNRECOMMENDED (AND NO ARROW KEYS AS WELL)

Editting (Stay in normal)

In normal mode

1. **y** => **y**ank (Copy), yy or Y to yank sentence
2. **d** => **d**elete (Cut), dd to cut sentence, D to cut after your cursor
3. **x** => Same as d but for one character only
4. **p** => **p**aste after cursor, P to paste before cursor
5. **r** => **r**eplace character on cursor

Editing (Jumps to insert, ie. can continue typing)

1. **a** => **append**, **A** to append at the end of the sentence (!!)
2. **s** => **substitute**, **S** to sub everything after your cursor
3. **c** => **change** (Delete and continue typing), **cc** to change sentence, **C** to change after your cursor
4. **o** => **open new line**, **O** opens a new line before the current line

How it works?

{Operator} {Count} {Motion}

Eg. `d 5 w` => delete 5 words from cursor Eg. `y 5 j` => yank 5 lines down

Exiting vim

1. `:wq` or `ZZ` => save and quit
2. `:q!` or `ZQ` => quit without saving

Advanced features

- ▶ `:tabe {file}` => open in new tab
- ▶ `:vsp {file}` => open a split vertically
- ▶ `:sp {file}` => open a split horizontally
- ▶ `:%s/{foo}/{bar}/g` => looks through the entire file and subsitutes foo for bar (Google search and replace vim)
- ▶ `:r !{cmd}` => puts output of \$cmd into the current file

Popular vim plugins (Don't use until you are comfortable with base vim)

Plugin manager: Vim plugged

- ▶ easymotion
- ▶ coc.nvim
- ▶ goyo
- ▶ indentLine
- ▶ vim surround
- ▶ autopair
- ▶ NerdTree/Ctrl-P/netrw