

Week 4

Announcements

- ADV1 due by 11:59 PM **tonight**
- HW2, ADV2 due by 11:59 PM on Feb 3 (Monday)
- HW3, ADV3 due by 11:59 PM on Feb 10

Lecture 4: (Text) Editors

`vi` != `vim`

Overview

- What is a text editor
- Examples of text editors
- Looking at text editors
 - Featuring a large section on **vim** because it's the one I know the best

What is a text editor?

- Tool that modifies plain-text data in files
- The best ones conform to your needs and further enable your productivity

What is a text editor?

- Tool that modifies plain-text data in files
- The best ones conform to your needs and further enable your productivity

Q: Who has used features beyond moving around with arrow keys, using the mouse to select/move the cursor, copy and paste in their preferred text editor/development environment?

The goal of today's lecture is to expose you to text editors can be and how powerful they can be

The goal of today's lecture is to expose you to text editors can be and how powerful they can be

- Ultimately editor choice is a highly personalized decision
 - No, we're not fanning the flames of the Editor Wars

The goal of today's lecture is to expose you to text editors can be and how powerful they can be

- Ultimately editor choice is a highly personalized decision
 - No, we're not fanning the flames of the Editor Wars
- One editor is not *inherently* better than another: it depends on whether or not it works for *you*

The goal of today's lecture is to expose you to text editors can be and how powerful they can be

- Ultimately editor choice is a highly personalized decision
 - No, we're not fanning the flames of the Editor Wars
- One editor is not *inherently* better than another: it depends on whether or not it works for *you*
 - I'll make an exception for Microsoft Notepad: literally anything is better

The goal of today's lecture is to expose you to text editors can be and how powerful they can be

- Ultimately editor choice is a highly personalized decision
 - No, we're not fanning the flames of the Editor Wars
- One editor is not *inherently* better than another: it depends on whether or not it works for *you*
 - I'll make an exception for Microsoft Notepad: literally anything is better
- A relatively vanilla **vim** just so happens to be what works for how I work
 - Don't take this as a guideline for you to follow: I just happen to be highly productive with it
 - You can use whatever text editor you want, with as many or few customizations and plugins as you want

Q: What are some text editors?

Terminal text editors

Q: Why learn them in **\$CURRENT_YEAR**?

- Yes, nearly all of us will be working in a GUI environment

Terminal text editors

Q: Why learn them in **\$CURRENT_YEAR**?

- Yes, nearly all of us will be working in a GUI environment
- In some cases you may need to SSH into an environment that has no GUI; some base level competency in terminal text editors will come in handy

ed (1969)

The OG

(Oh god why would you use this in `$CURRENT_YEAR`)

- The original UNIX editor
- Part of the POSIX spec!
- Developed back when we had *teleprinters*, not even video terminals
 - The root of some design decisions and quirks of UNIX, such as short commands and lack of output
- Known as a "line editor" where you specified lines you wanted to edit
- Provides very little feedback

ed (1969)

The OG

(Oh god why would you use this in `$CURRENT_YEAR`)

- The original UNIX editor
- Part of the POSIX spec!
- Developed back when we had *teleprinters*, not even video terminals
 - The root of some design decisions and quirks of UNIX, such as short commands and lack of output
- Known as a "line editor" where you specified lines you wanted to edit
- Provides very little feedback
- "The most user-hostile editor ever created"

ed summary

Quit	ed q

Save	w w <file name>

Append text	a (text) .

Print all	, p
Print line	<n>p for line <i>n</i>

Delete line	<n>d for line <i>n</i>

vi (1976)

We've got these fancy "screen" things now

- Part of the POSIX spec!
- Born out of another line editor **ex** (and ultimately **ed**); the "colon" commands are actually **ex** commands
- *Modal* text editor
 - "Command" mode for commands and navigation
 - "Insert" mode for writing text
 - "Command-line"/"ex" mode for **ex** commands
- ESC brings you Command mode
 - **:** enters "Command-line"/"ex" mode and allows you to enter **ex** commands (which allow you to save and quit)
- Certain commands (e.g. **i**, **a**) bring you into Insert mode

`vim` (1991)

`vi` but better (but not in the POSIX spec 😞)

- Plain ole `vi` kinda sucks for today's use
- Many distros don't even provide OG `vi`, opting to alias it to a minimal version of `vim` or even just normal `vim`
- `vim`'s features is a superset of `vi`'s

vim (1991)

vi but better (but not in the POSIX spec 😞)

- Plain ole **vi** kinda sucks for today's use
- Many distros don't even provide OG **vi**, opting to alias it to a minimal version of **vim** or even just normal **vim**
- **vim**'s features is a superset of **vi**'s
- *Massively* extends the functionality of **vi**
 - Syntax highlighting!
 - Line numbers!
 - Undo history larger than 1!
 - Plugins!
 - Multiple windows!
 - ...and much more!
- New modes:
 - "Visual" mode for selecting text
 - "Command" mode renamed to "Normal" mode

vi/vim abridged cheatsheet

- **<ESC>**: Enter Command/Normal mode
- The following are for when you're in Command/Normal mode
- A neat thing is that you can put a number before a command to repeat it
 - **10j** to move down 10 lines
- You can record *macros* with **q <letter to save to> <commands> q**
 - You can invoke them with **@<letter you saved to>**
- The "register" I refer to is sort of like a copy-paste clipboard
- **^** (caret) is shorthand for the Control key serving as a modifier

Navigation

- **h, j, k, l**: move cursor left, down, up, right
 - **vi**: Arrow keys *might* be supported, and *might* work in Insert mode
 - **vim**: Arrow keys work as expected (nowadays)
- **w**: "word", go to beginning of next word
- **b**: "back", go to beginning of current word (or beginning of previous word)
- **e**: "end", go to end of current word (or end of next word)
- **0**: go to beginning of line
- **\$**: go to end of line
- **^u**: go up half a page
- **^d**: go up down a page
- **g**: go to top of document
- **G**: go to bottom of document
- **<n>G**: go to line *n*
- **/**: search for a pattern
 - **n**: next match
 - **N**: previous match

Editing

- **i**: "insert", goes into Insert mode *before* character under cursor
 - **I**: goes into Insert mode at the beginning of the line
- **a**: "append", goes into Insert mode *after* character under cursor
 - **A**: goes into Insert mode at the end of the line
- **x**: deletes character under cursor, putting character into "register"
 - **X**: deletes character *before* character under cursor, putting character into "register"
- **r**: "replace", replaces character under cursor with next entered character
- **R**: enter a "replacement" mode
- **d<w, e>**: "delete word", deletes word; **w** puts cursor on next word, **e** puts cursor at the end of the word
- **cw**: "change word", deletes word and enters Insert mode
- **u**: "undo" (in **vi**, there's only a history of 1 so undo-ing again reverts the undo)
- **^r**: "redo" (**vim**)
- **dd**: "delete", deletes line under cursor (putting line in "register")
- **yy**: "yank", copies line to "register"
- **p**: "paste", copies "register" contents *after* character under cursor
- **P**: "paste", copies "register" contents *before* character under cursor
- **v**: enter Visual mode (**vim**)

Visual mode (**vim**)

- While in Visual mode you can select text, offering some more options
- **x, d**: deletes selection, putting it into the "register"
- **y**: yanks selection, putting it into the "register"

Command-Line/**ex** mode

- **:e**: "edit", open file for editing
- **:w**: "write", save
- **:w <file name>**: "write", save to particular file
- **:q**: quit
- **:q!**: quit without saving
- **:wq**: save and quit
- **:x**: quit, write if modified
- **:s/<pattern>/<replace>**: search for pattern and replace
 - **:snomagic/<pattern>/<replace>**: non-magical pattern substitution

...and there's many many more

emacs (1976, 1984)

What's a mode?

- Powerful and fancy modeless editor
- Highly extensible
- Has an image manipulation library as a dependency (wut)
 - Can display embedded images
- Exit with **C-x C-c** where **C-** is Control
- Heavy use of modifier keys such as Control and "Meta" (Alt)

nano (2000)

- Fairly straightforward, acts like a "typical" basic text editor
- On screen legend shows you common editing shortcuts
- **^G** for more shortcuts
- Exit with **^X**

But wait, what about GUIs?

Once we get here, there's a lot more functionality

gedit (1999) and Kate (2001)

gedit: GNOME's basic editor

Kate: KDE's basic editor

- "Basic" text editors associated with desktop environments
 - Still pretty well featured text editors
- Analogous to Microsoft Notepad but way better

Sublime Text (2008)

This was the hotness when I was an undergrad

- \$\$\$
- Huge plugin ecosystem

Visual Studio Code (2015)

The new hotness

You're probably already using this

- Almost steps into IDE territory while remaining lightweight

Parting thoughts

Try out another editor and see if you like it

You may find something that you really dig

Try to learn more about the features of your preferred editor

Questions?