Vim Workshop

Presented by HackCU

Audience

- Computer Science majors?
- CSCI 1300 (intro)?
- Non-CS majors? (wooooo!)
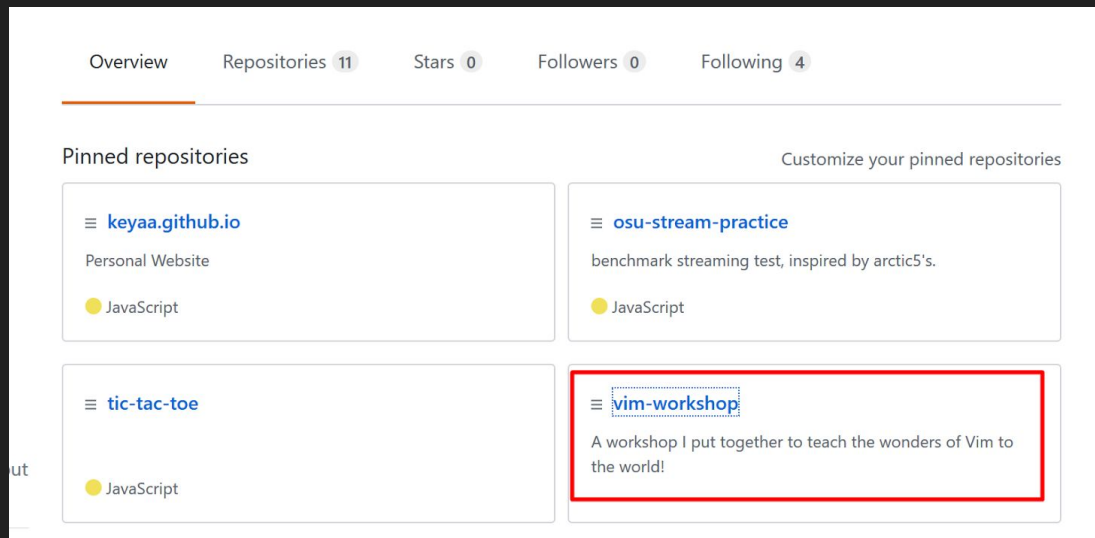- Regardless, we're happy you're here and curious!

Goals

- This workshop is not designed to give you a comprehensive understanding of everything about Vim
- Instead, it's more about motivating you to learn more about Vim, and present it as a cool text editor worth using

Setup

Google "keyaa github", first result.

Or just type: https://github.com/keyaa/vim-workshop

Setup (continued)

Download as a zip and extract files

(or git clone if you know how).

Save this "vim-workshop" folder to a

special place in your heart (or on

your desktop or something).
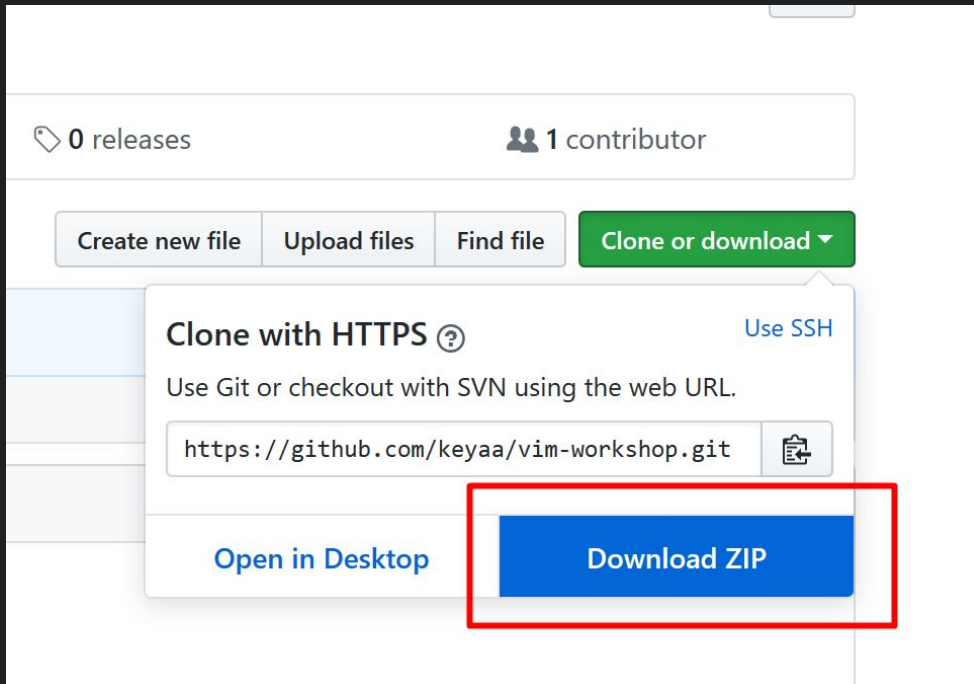
Setup (continued)

MacOS/Linux: Terminal Vim (no preparation needed here)

- It's already there, yay!

Windows 10: Bash on Ubuntu on Windows

- Google "howtogeek linux bash", first result.
- https://www.howtogeek.com/249966/how-to-install-and-use-the-linux-bash-shell-on-windows-10/

Setup (continued)

Windows (not 10):

- Or download from Vim website
- (Google Vim Windows, first result)
- https://vim.sourceforge.io/download.php#pc
- Click on "PC: MS-DOS and MS-Windows"
- Download and run gvim80.exe
- Right click .txt files and press Edit with Vim
- (or Open with > Vi Improved - A Text Editor)
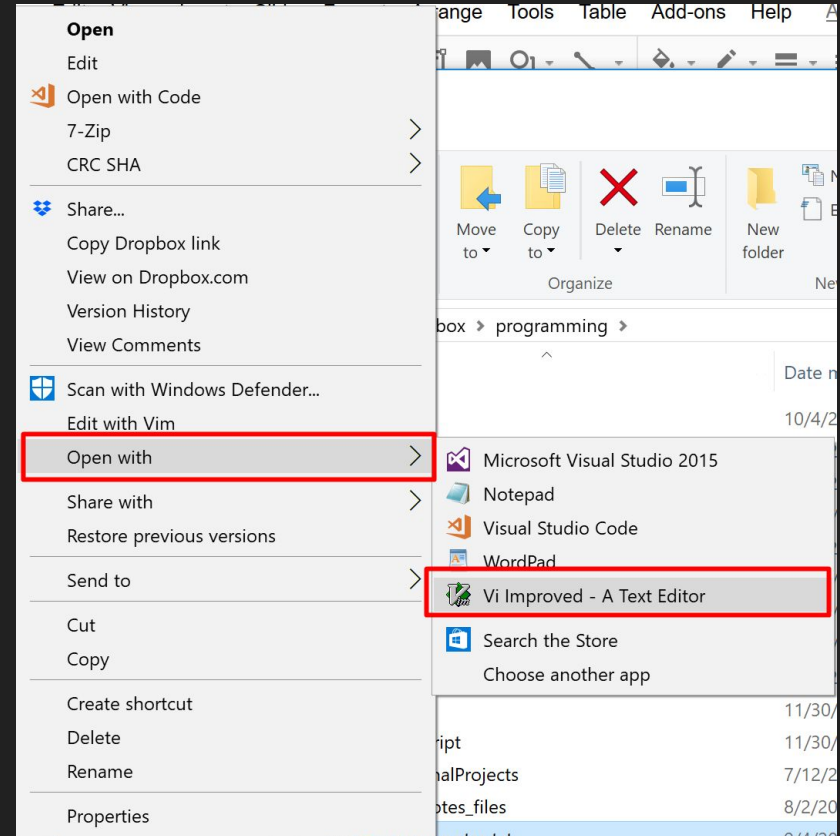
What is Vim, and why should I learn it?

- Text editor released in 1991, stands for "vi-improved" (vi stands for visual editor, and was made by Bill Joy)
- Made by Bram Moolenaar and contributed to by many, many people. Virtually all donations for Vim go to ICCF Holland (helping needy children in Uganda)
- Available pretty much everywhere (basically anywhere you have a terminal)
- Great for ssh (remotely accessing another computer), need to edit and save something quickly with only a terminal interface
- Light, powerful, and fast. Like, REALLY fast.
- Using Vim feels like a game, it's fun to use (subjective opinion)

What is Vim, and why should I learn it? (continued)

- Follows Unix philosophy: many programs to do one thing each, and to do it really well (Vim is primarily a text editor, not an IDE! Though you can configure it to have many IDE features)
- Takes a loooong time to learn to master, but is a worthwhile investment
- Plugins for literally everything (even Spotify), fully customizable to your style
- Can impress your friends who don't know what vim is, be called a nerd
- No need for a mouse (what?)

Core design
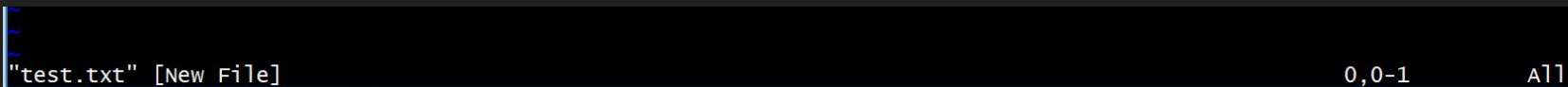
h = left, j = down, k = up, l = right

- Really weird arrow movement:
    1. Keeps hands on home row
    2. Old computers were like this
    3. Feels natural after a while
- Vim is a modal editor:
    1. Normal mode (movement, major edits, most of the time in normal mode)
    2. Insert mode (for typing in short bursts before going back to normal)
    3. Visual mode (for selecting things, highlights)
    4. And a few others, but these are the most important ones

Let's make a file!

- Navigate to the "first" folder using terminal (cd, ls).
- In the "first" folder, type "vim test.txt", press enter.
    - If you're using Lu Wang's web-vim, type ":e test.txt", then press enter.
- You should get a big, black screen with this at the bottom.

```
~
~
~
~
"test.txt" [New File]                                          0,0-1          All
```

- We are in normal mode by default, let's change to insert mode (press i).
- Type "Vim is cool!", then press Esc (Esc goes back to normal mode).
- After pressing Esc, type ":w", then press enter. :w stands for "write" (save).
- Now type ":q", then press enter. This is how you quit Vim.
- :w and :q are probably the 2 most important commands in Vim.

Now let's edit a file!

- There will be MANY commands used here, you don't need to memorize all of them, just try to remember a few that you like.
- This is simply to demonstrate the power of Vim, you can always refer back to documentation later.
- Go to the second folder (from first folder, type "cd ..", then "cd second").
- Type "vim example.txt".
- Type ":set nu" (adds line numbers)
- Time to follow along!

Your screen should look something like this:

```
 1 gg
 2 after deleting, go back to line 28
 3
 4 this line needs another line below
 5 Supercalifragilisticexpialidocious there is an unecessary word at the start of this sentence
 6
 7 This sentence needs an exclamation point
 8 this line needs another line above it
 9
10 this line needs to be deleted
11
12 so does this one
13 and this one
14 and this one
15 and this one
16
17 move "delete the underscore" line to below this one
18
19 this line needs to go later (skip for now)
20
21        delete the underscore: 2138094j234912j384cj_18394891234jc123j49c123
22
23 these two lines
24 need to be indented to the right
25
26        put the line "this line needs to be deleted (skip for now)" below this one
27
28 delete gg at the top
29
30 (change hate to "love") I hate Vim
31
32 delete everything after here @213890jsadfklaklsdjf,123m,asdl;
33
34 iojqwe023r1jcklsjaf,1j2kjro12r;' delete everything before here
35
36 while (true)
37 { // we are going to jump down
38
39        // meaningless garbage
40        // meaningless garbage
41        // meaningless garbage
42        // meaningless garbage
43        // meaningless garbage
44        // meaningless garbage
45        // meaningless garbage
46        // meaningless garbage
47        // meaningless garbage
48
49 // "recenter this text"
:set nu                                                      1,1         Top
```

Let's edit multiple files!

- Go to the third folder (from second folder, type "cd ..", then "cd third").
- Type "vim destination.txt", press enter.
- Type ":sp source.txt", press enter.
- Go down 2 lines, copy text in brackets, go back to destination and paste.
- Type ":Sex" (Split horizontal file explorer), use hjkl and enter to go to second folder, open example.txt.
- Go to bottom, copy text in brackets, go back to destination and paste.
- Copy + Paste all of destination.txt into destination2.txt

Let's try out some more powerful commands.

- "cd ..", "cd fourth", "vim macros.txt"
- Press "q", then press "a". Insert -> "Type out this sentence. " -> Esc, press "q".
- o -> (ctrl + a), inserts what you last typed in insert mode
- Delete the sentence, then press "." ("." repeats your last action).
- Type "@a", and "Type out this sentence." should appear.
- Press "o" for a new line, then press Esc.
- Press "q", then press "z".
- Type "@a", then insert "please" before the period. Esc, then press "q".
- Press "o", then Esc. Then type "@z". What is this doing?
- Vim macros can actually use other macros!

Miscellaneous Things

- :pwd to find where your current file is
- Vim can use external shell commands.
- ":!" used for these external commands
- Example, :!ls to see what else is in the same directory
- :%TOhtml can convert current file to an html rendering in a split
- Vim can run a subshell :sh, then can be returned to by ctrl+D or "exit"
- :%s/foo/bar/g finds all occurrences of "foo" and replaces them with "bar"
- "gf" on an #include "header.h" will open that header file
- q: to view command history
- …and many, many other commands

Whoa, that was a lot of stuff to remember!

- It's a lot of work to memorize these commands, but it definitely pays off!
- There are many ways to do the same thing, find your own style
- Practice a lot (take class notes in Vim to help you get used to it)
- I have some documentation here, you can refer back to these slides to access these resources online

Resources

- Type "vimtutor" in tutorial for the basic commands again.
- https://vim-adventures.com/ = online flash game, helps you to get used to using hjkl in place of arrow keys
- https://vim.rtorr.com/ = nice, quick reference for commands
- https://danielmiessler.com/study/vim/ = great article which covers the general useful commands, as well as describes clearly how to use them effectively
- http://www.angelwatt.com/coding/notes/vim-commands.html = much more comprehensive reference of commands
- http://vim.wikia.com/wiki/Category:Getting_started = wiki page, has some great starter categories

Thanks for Listening!