Lecture 27 - Editing with VI / VIM

Videos

https://youtu.be/AYa1WNgevNI - Lect-27-2150-pt1-vim-editor.mp4
https://youtu.be/VY1P0ERl2sU - Lect-27-2150-vim.mp4
https://youtu.be/pkTHyctg_uQ - Lect-27-2150-vim2.mp4
https://youtu.be/9lAbZx44qaU - Lect-27-2150-vim3.mp4
https://youtu.be/QHvdJbbL4fk - Lect-27-2150-pt2.mp4
https://youtu.be/ndmtnOSzeT8 - Lect-27-2150-pt3-vim.mp4

From Amazon S3 - for download (same as youtube videos)

http://uw-s20-2015.s3.amazonaws.com/Lect-27-2150-pt1-vim-editor.mp4
http://uw-s20-2015.s3.amazonaws.com/Lect-27-2150-vim.mp4
http://uw-s20-2015.s3.amazonaws.com/Lect-27-2150-vim2.mp4
http://uw-s20-2015.s3.amazonaws.com/Lect-27-2150-vim3.mp4
http://uw-s20-2015.s3.amazonaws.com/Lect-27-2150-pt2.mp4
http://uw-s20-2015.s3.amazonaws.com/Lect-27-2150-pt3-vim.mp4

On Vi / Vim

First there is an on-line tutorial - it is actually very good.

https://www.openvim.com/

Go and do the tutorial.

Vi is a modal editor. There is an “insert” mode and an “edit” mode. Many commands for editing only work in “edit” mode. Also there is an “ex” command line mode.

Vi Has a newer more powerful cousin, Vim - that stands for VI Improved.

Why Vim - part 1

First it is a super powerful editor. It can be found on virtually all computers. I have used it on IBM OS/380 Host, on Supercomputers, on Mac / Windows / Linux, on my router running a MIPS processor, on my thermostat on the wall. It works all the time. You can edit files that are huge and deal with them.

Second - People that use it make more money.

Third - it will make you more productive. Every time you edit and touch the mouse you are loosing your hand position on the keyboard. VIM is designed to allow you to edit and touch type. Yes you can use the mouse for some stuff - but most stuff is based on you keeping your hands in a touch-type position. VIM is an editor superficially bayed out for touch typing. A question you should ask yourself is if you have ever seen a concert pianist hunt-ant peck for the keys? No... And you should not be hunting and pecking either. Most if not all the other editors (with the exception of Emacs) are mousey-hunt-and-peck editors.

Vim Installation

Installation of VIM - Windows systems won't come with VI - you will need to install it. There is a good guide for installing it at: https://www.instructables.com/id/Install-and-Write-First-Text-File-Using-Vim-on-Win/

On the Mac a version of "vi" is pre-installed. Usually I upgrade this to MacVim. MacVim is the GUI version of VIM. There is also the command line version of vim. MacVim first. https://macvim-dev.github.io/macvim/. For the command line I use "brew" the package manager. You can install brew at: https://brew.sh/. Then to install the command line version of vim - at the Terminal (You can find the terminal in Applications/Utilities):

```
$ brew install vim
$ brew link vim
```

On Linux a minimal version of vim is pre-installed on most distributions. A full version of vim may require removing the minimal venison and installing the full version. A good example of instructions for doing this is https://www.simplified.guide/ubuntu/install-vim

On Ubuntu:

```
$ sudo apt remove --assume-yes vim-tiny
$ sudo apt update
$ sudo apt install --assume-yes vim
```

You need to go and install VIM on your system.

Why Vim - part 2

Example 1

Let's say I have a list of tables in a database that I need to turn into just a list of the table names in a JSON input file. The example file is in ex1.orig.txt and the final is in ex1.json. Both of these files are in the Lect-27 directory.

Let me demo how to use vim to transform this into the final form.

Example 2

Editing big files.

Also note that the "big" file was on a remote server that is actually in Texas. I logged in with SSH to the server and it works - to edit stuff. The server has no GUI at all.

Example 3

It works on "tiny" hardware. Let's login to my router.

Edit a file

```
$ vi file1 file2
```

To save the file after modification either ":wq" or ZZ.

To abandon changes - ":q!"

Modal Editing.

VIM is a modal editor. It is not WYSIWYG - what you see is what you get. GUI's try to be as WYSIWYG as possible. VIM is not a use-it without learning it tool. You have to learn to use VIM. It takes time and effort. After 30+ years I am still learning new commands in VIM - and I already know thousands of commands.

When you start you are in command mode. To insert you use an 'i' for insert before or an 'a' to append after the current character. When you are done inserting you use a ESC (escape) key to get from insert mode to edit mode.

If you want to use a "ex" command you type ':' (colon) to get to the EX editor prompt mode. This appears at the bottom left. To search in command mode you type a '/' and the regular expression pattern that you want to search for. You can search UP in the document with a '?' and the pattern. To get to the next occurrence of the pattern you use the 'n' key.

To turn on line numbers ":set nu" and enter - to turn them off ":set nonu" and enter. You can set permanent defaults in the ~/.vimrc file.

If you want all the fancy code-completion stuff - like an IDE - there is a VIM IDE for most languages.

Commands in EX are prefixed with the line range that they apply to. So with line numbers on you can say

```
:3,38
```

That would be the line numbers from 3 to 38.

```
:3,38s/aaa/bbb/g
```

Change all the aaa's to bbb - and do all of them across each line - that is the 'g' global at the end. If you want to be prompted for a yes/no for each one then add a 'c'.

```
:3,38s/aaa/bbb/gc
```

There are 2 special line numbers '.' - the current line and '$' the last line in the file. You can do computations on the line numbes.

```
1,$-2s/aaa/bbb/
```

The `$-2` says subtract 2 from the last line in the file.

The pattern `aaa` is a regular expression. So I can say, `s/^[^|]*|//` .

This is find all the strings that start at the beginning of the line, the 1st `^` . Then match any character except `|` - that is the `[^|]` and do a string of as many as possible of this, `[^|]*` followed by `|` and replace it with the empty string.

Frequently-Used Commands

If you just use the bare bones set of commands - then you miss out on the opportunity to be more productive. I like productive - that means less time typing and more time skiing.

To this end learn at least some of these other frequently-used vi commands:

Command	Description

Command	Description
h	move cursor one character to left
j	move cursor one line down
k	move cursor one line up
l	move cursor one character to right
+	move to the next line at the beginning
o	open up a new line to enter stuff on.
w	move cursor one word to right
b	move cursor one word to left
0	move cursor to beginning of line
$	move cursor to end of line
nG	move cursor to line n, 120G for example.
*	match the word you are on and move to next one.
control–f	scroll forward one screen
control–b	scroll backward one screen
control–d	scroll forward one half screen
control–u	scroll backward one half screen
i	insert to left of current cursor position
	Exit insert moded with ESC - Escape Character.
a	append to right of current cursor position
	Exit insert moded with ESC - Escape Character.
dw	delete current word (end with ESC)
cw	change current word (end with ESC)
r	change current character
~	change case (upper-, lower-) of current character
dd	delete current line
Xdd	delete X lines, so 3dd deletes 3 lines.
D	delete portion of current line to right of the cursor
x	delete current character
mA	mark current position - line - can use A..Z marks

Command	Description
`d'a`	delete everything from the marked position to here
`'a`	go back to the marked position
`p`	dump out at current place your last deletion
`fX`	move to the next F character.
`dfX`	delete from the current position to next F
`dw`	delete from the current position next word.
`u`	undo the last command
`.`	repeat the last command
`J`	combine (``join'') next line with this one
`:w`	write file to disk, stay in vi
`:q!`	quit VI, do not write file to disk,
`ZZ`	write file to disk, quit vi
`:r filename`	read in a copy of the specified file to the current
	buffer
`:A,Bw file`	write a section out to a new file. From line A to B
`/string`	search forward for string (end with Enter)
`?string`	search backward for string (end with Enter)
`n`	repeat the last search (``next search'')
`:s/s1/s2`	replace (“substitute”) (the first) s1 in line by s2
`:A,Bs/s1/s2/g`	replace all instances of s1 in the line range A,B
	by s2 (A,B is where A and B are
	either explicit line numbers, or . (current line)
	or $ (last line) or by marks that you have placed.
`%`	go to the “mate,” if one exists, of this parenthesis
	or brace or bracket (very useful for programmers!)

All of the `:' commands end with your hitting the Enter key. (By the way, these are called “ex” commands, after the name of the simpler editor from which vi is descended.)

The a command, which puts text to the right of the cursor, does put you in insert-text mode, just like the i command does.

By the way, if you need to insert a control character while in append/insert mode, hit control-v first. For example, to insert control-g into the file being edited, type control-v then control-g.

One of vi's advantages is easy cursor movement. Since the keys h,j,k,l are adjacent and easily accessible with the fingers of your right hand, you can quickly reach them to move the cursor, instead of fumbling around for the arrow keys as with many other editors (though they can be used in vi too). You will find that this use of h,j,k,l become second nature to you very quickly, very much increasing your speed, efficiency and enjoyment of text editing.

Many of the commands can be prefixed by a number. For example, 3dd means to delete (consecutive) three lines, starting with the current one. As an another example, 4cw will delete the next four words.

The p command can be used for "cut-and-paste" and copy operations. For example, to move three lines from place A to place B: 1. Move the cursor to A. 2. Type 3dd. 3. Move the cursor to B. 4. Type p.

The same steps can be used to copy text, except that p must be used twice, the first time being immediately after Step 2 (to put back the text just deleted). Note that you can do operations like cut-and-paste, cursor movement, and so on, much more easily using a mouse. This requires a GUI version of vi, which we will discuss later in this document.