Introduction

Learning vim is fun, intuitive. It is well worth it in the long run.
Once you figure out how to use vim's built-in help effectively, you'll learn vim much more comfortably.

Vim is mostly preinstalled in every major linux distribution.

You can always “sudo apt-get install vim” through the terminal, with an internet connection

Attributes of vim

- The vim editor is:
  - very powerful
  - and at the same time is cryptic
  - Shifting from windows editors may be difficult but it is the best decision you would think you made once you get to see its power
- The best way to learn vim commands is to use them
- So Practice…

- The Vim help provides almost description of every command used in it

Starting vi

- Type vi <filename> at the shell prompt
- After pressing enter the command prompt disappears and you see tilde(~) characters on all the lines
- These tilde characters indicate that the line is blank

Vi modes

- There are two modes in vi
  - Command mode
  - Input mode
- When you start vi by default it is in command mode
- You enter the input mode through various commands
- You exit the input mode by pressing the Esc key to get back to the command mode

How to exit from vi

- First go to command mode
  - press Esc There is no harm in pressing Esc even if you are in command mode. Your terminal will just beep and/or or flash if you press Esc in command mode
- There are different ways to exit when you are in the command mode

How to exit from vi (comand mode)

- :q <enter> is to exit, if you have not made any changes to the file
- :q! <enter> is the forced quit, it will discard the changes and quit
- :wq <enter> is for save and Exit
- :x <enter> is same as above command
- ZZ is for save and Exit (Note this command is uppercase)
- The ! Character forces over writes, etc. :wq!

Moving Around

- You can move around only when you are in the command mode
- Arrow keys usually works(but may not)
- The standard keys for moving cursor are:
  - **h** - for left
  - **l** - for right
  - **j** - for down
  - **k** - for up

Moving Around

- w – to move one word forward
- b – to move one word backward
- $ – takes you to the end of line
- <enter> takes the cursor the the beginning of next line

Moving Around

- – – (minus) moves the cursor to the first character in the current line
- H – takes the cursor to the beginning of the current screen(Home position)
- L – moves to the Lower last line
- M – moves to the middle line on the current screen

Moving Around

- f – (find) is used to move cursor to a particular character on the current line
  - For example, fa moves the cursor from the current position to next occurrence of 'a'
- F – finds in the reverse direction

Moving Around

-) – moves cursor to the next sentence
- } – move the cursor to the beginning of next paragraph
- (– moves the cursor backward to the beginning of the current sentence
- { – moves the cursor backward to the beginning of the current paragraph
- % – moves the cursor to the matching parentheses

Moving Around

- Control-d scrolls the screen down (half screen)
- Control-u scrolls the screen up (half screen)
- Control-f scrolls the screen forward (full screen)
- Control-b scrolls the screen backward (full screen).

Entering text

- To enter the text in vi you should first switch to input mode
  - To switch to input mode there are several different commands
  - a – Append mode places the insertion point after the current character
  - i – Insert mode places the insertion point before the current character

Entering text

- I – places the insertion point at the beginning of current line
- o – is for open mode and places the insertion point after the current line
- O – places the insertion point before the current line
- R – starts the replace(overwrite) mode

Editing text

- x – deletes the current character
- d – is the delete command but pressing only d will not delete anything you need to press a second key
  - dw – deletes to end of word
  - dd – deletes the current line
  - d0 – deletes to beginning of line
- There are many more keys to be used with delete command

The change command

- c – this command deletes the text specified and changes the vi to input mode. Once finished typing you should press <Esc> to go back to command mode
- cw – Change to end of word
- cc – Change the current line
- There are many more options

Structure of vi command

- The vi commands can be used followed by a number such as

  n<command key(s)>

  - For example dd deletes a line 5dd will delete five lines.
- This applies to almost all vi commands
- This how you can accidentally insert a number of characters into your document

Undo and repeat command

- u – undo the changes made by editing commands
- . (dot or period) repeats the last edit command

Copy, cut and paste in vi

- yy – (yank) copy current line to buffer
- nyy – Where n is number of lines
- p – Paste the yanked lines from buffer to the line below
- P – Paste the yanked lines from buffer to the line above

(the paste commands will also work after the dd or ndd command)

Stupid vi Tricks

- Indent four lines: 4>>
- Will delete the character under the cursor, and put it afterwards. In other words, it swaps the location of two characters: xp
- Similar to xp, but swapping lines: ddp
- Remove all lines that start with #
  - :g/^#/d
- Remove all empty lines, assumes no spaces or tabs:
  - :g/^$/d

Some Practice

- Using a command line utility called wget let's pull down a copy of the Gettysburg Address

  wget –U " " http://wildbill.org/rose/gettysburg.txt
- Now we will run a few commands against the text (note: I placed several blank lines and lines starting with the #)

Gettysburg.txt

- Run the following commands:
  - vi gettysburg.txt
  - :g/^#/d
  - :g/^$/d

Gettysburg.txt

- Invoke vi's edit mode by pressing the Esc key, then a colon (:), and enter:
  1,$s/oldstring/newstring/g
- This will change oldstring into newstring wherever it occurs throughout the entire text. The 1 (the number one) in the above command means "start the search on the first line". The $ means "end the search on the last line". The g at the end executes the change globally on each line. If you omit the g , the search will stop after finding the first occurrence of oldstring.

Creating a shell script using vi

- Create a directory call class
- Change into class
- vi myscript.sh
- inside the file enter following commands

```
clear
echo "==========="
echo "Hello World"
echo "==========="
sleep 3
clear
echo Host is $HOSTNAME
echo User is $USER
```

Creating a shell script using vi

- Save the file
- Change the permissions on myscript.sh

  chmod 700 myscript.sh <enter>
- Now execute myscript.sh

  myscript.sh <enter>
- Did the script run?
- Why not?
  - Hint, think about absolute vs relative path
  - Type echo $PATH to see your PATH variable
  - Try this ./myscript.sh <enter>
  - The ./ mean right here in this directory!