# Week 01
# INTRODUCTION

Melvyn Ian Drag

January 20, 2021

**Abstract**

Today we'll discuss what lies ahead of us this semester. We'll look at some motivating examples, peek at the syllabus, and then jump into it.

## 1 Why Learn Linux?

Here's one cool thing you can do:

```
melvyn@machine$ telnet towel.blinkenlights.nl
(exit with CTRL + ], then quit)
```

Linux distros like Fedora, Debian, Redhat, Ubuntu, Slackware, Puppy Linux are all operating systems, and you might use them instead of Windows or macOS.

This class is based around Debian 10 because it's my favorite.

The stuff you'll learn in this class will generally help you with *nix operating systems, including the BSDs like dragonflyBSD, FreeBSD, etc. as well as using the terminal in macOS.

I like Linux because it's free, you can do alot of cool things with it, etc..

Demos:

1. Website running linux. Most webservers in the world run some type of Linux.
2. Beaglebone Black
3. Raspberry Pi
4. Cars run Linux `https://events19.linuxfoundation.org/events/automotive-linux-summit-2018/`
5. Laptop runs linux ( show T series laptops ).
6. You can save old hardware - show 2006 MacBook.
7. Playstation4 runs a Unix operating system
8. There is alot of money to be made ( can't demo money, but if you look at salaries on indeed for Linux pros you'll see there is a nice amount of cash out there for you if you aren't a bonehead.

You can do anything you want with Linux! Whatever you're interested in. So lets study hard so you learn all the basic stuff you need to know to use this wonderful family of operating systems.

## 2 About this class

$$runthroughthesyllabus$$

## 3 Tools we'll use in this class

Show digital ocean

Show Linode

Point out the free credits

Make sure you only use digital ocean for now. We'll activate Linode later. Theres a timer on the free credits, so dont register for both now.

## 4 Getting started with the terminal in Linux

1. ls
2. ls -l
3. ls -ltr
4. ls -a
5. ls -la
6. cd
7. cd
8. cd .
9. cd ..
10. cd ../../
11. cp
12. cp -r
13. mv
14. mkdir
15. mkdir -p
16. touch
17. rm
18. rm -r
19. which

## 5 Bash commands part 2

1. cat $file1
2. cat $file2 $file3
3. echo “hello”
4. echo “my path $PATH”
5. man
6. wc
7. wc -c $fileName
8. wc -l
9. wc -w
10. md5sum
11. cut -b
12. cut -c
13. tree
14. wget

Note about cut: cut -c is just like cut -b on every machine I've seen. In the future -b ( byte) and -c ( character) will be different. Ill give some insight on that later in the class, and if you know Java + C you will likely have a bit of insight on the definitions of byte and character, where they are the same, and where they are different. There is no time for that now, and if you already have insight, just keep it to yourself for now haha.

## vim

### 5.1 Why learn vim?

Vim is a text editor. You use it to edit files. We'll use Vim instead of Notepad++, WordPad, Eclipse, PyCharm, etc.. On our Linux servers, whenever we need to edit a file we'll use vim.

Here's your first dose of linux/unix/bsd culture! There is alot of playful fighting in the nix world about the best way to edit files on the command line. Mainly the war is around two editors, `https://en.wikipedia.org/wiki/Editor_war,emacsandvim`. I like vim. Some folks like emacs. I started on emacs a decade ago, but when I learned vim I found that I liked it more and I haven't looked back in about 10 years. I'm going to teach you vim, and if you want to learn Emacs you'll have to do it on your own time. If you get a job and have to edit a file on a server, and you fire up vim and edit it smoothly you'll instantaneously become the celebrity of the office. You'll be one of the cool cats. Emacs would also get you cool points.

But vim is featured in every standard linux distro, while emacs is not! Whatever, too much said already. We're using vim for this class. There are simpler editors like pico and nano, but I know vim by heart so that's what I use and teach!

If you happen to install Linux on a Laptop or in a Virtual Machine on your computer, you can use graphical editors like VSCode or Sublime or Geany. I used Geany during my first year or so on Linux, Anyway, we're on the command line so we can't use graphical editors.

### 8:35 - 8:40 Basic Vim workflow

Look at the stupid vim memes. They are dumb and frustrating. Memes in the foloder here. Memes about not knowing how to exit vim. So many 'hackers' want to complain that they can't even remember the following:

To open a file with vim type

```
melvyn@computer$ ls
filename otherfilename dirname otherstuff
melvyn@computer$ vim filename
# vim will open a file with name 'filename'
# if the file exists, it will create the file.
```

Vim is a 'mode based' editor. Hit i to enter insert mode. Start typing as normal. Hit esc to get out of insert mode. Type :wq to save and quit. Thats 99% of what you'll do in this class.

Here's a youtuber making fun of people who complain about getting out of vim. `https://youtu.be/8bGiiLW_ss4i`

### More vim commands to make your work better.

- h, j, k, l
- I prefer to just use the arrow keys
- modes i, esc
- quit with :q. To save and quit you use :wq or :x. If you want to know the difference, google the diff between :wq and :x. there is a slight difference, but it doesnt matter.
- dd to delete a line
- y is copy.
- p is paste after the cursor
- shift p is paste before the cursor
- u is undo
- CTRL + r is to redo.

Here is some more functionality that is so powerful and unique to vim that it will make you feel scared. You've never had this power before and this is going to make you think vim is hard. It's not. You can ignore the following for now, but within 2 weeks when you've mastered the above you'll be ready to appreciate how great the below commands are. I'm just showing you now to plan the seed so that it can start to develop in your subconscious.

- w to go forward a word
- e to go to the end of the next word
- b to go to the previous word beginning.
- to go forward 5 words, 5w
- To go back 5 words, 5b.

## 9:05 - 9:10 Bash Part 3

Return value of last command is : $?. All these linux commands I've shown you return information to the terminal. They tell the terminal if the command was successful or not. To check the return code you type 'echo $?'.

```
melvyn@machine$ echo "hello world"
hello world
melvyn@machine$? echo $?
0
melvyn@machine$ ehco "aksljdg"
# error
melvyn@machine$ echo $?
127
#the 127 indicates an error. There was an error because ehco is not a command,
echo is.
```

## 6 Homework

Show homework and end lecture.

## 7 9:10 - 9:30 More vim and the inevitable questions

TODO move this to another lecture! This is too much too soon, these folks will be confused about the i / esc stuff. END LECTURE NOW AND SAY GOOD NIGHT

The coupe de gras. Registers in vim. Vim remembers the things you've yanked in the past. Type :reg See that vim remembers the stuff you've put on your clipboard in the past. Exercise Type: Hello World Foo Bar Baz dd Hello dd World highlight and yank the other three Then look at clip board. Paste world Paste Baz Foo Bar Baz Look at :reg and see what it remembers. I think it remembers the dd stuff but only remembers the latest yanked stuff. Paste the things from your registers with ¡reg id¿p e.g. ( when in normal mode, not insert mode ) ”1p, see what happens.