10.1(3):Intro To the Terminal

Terminal is a way of interacting with your computer through text (direct port to the functionality of the computer).

The default on a windows machine is powershell. The problem is that it accepts completely different commands, it works differently, so you need to go through some loops to get the terminal. Need to install WSL(windows subsystem for linux) to make it like the mac version. Basically it installs Linux on the computer.

It might still be worth learning powershell in your free time as it can be useful.

Basically instead of searching for a folder and double clicking, you can use the terminal. Can create files, move things around, etc faster with terminal. At first it is slower, but when it is learned, it is faster.

What is a shell? Bash/ZSH

The shell is the actual program that returns the output.

There are many popular shells like bash(the default shell on the mac). Another popular one is called ZSH(z-shell).

10.1(4): Root and home Directory

The root directory is noted by a slash(/).
Root directory is the root of your machine, the top folder that everything else is stored inside of.

With WSL, the same folder structure(unix based) will be used(same as mac).

The next important directory is the **users** folder. This folder contains all of the user accounts on the computer.

If you go into the directory of a specific user account, that is the **home directory** (has a home icon on WSL**).** This has everything on the computer, videos, games, ect…

Command(home directory): cd ~ then enter then 1s, will show the folders of the home directory.

command(root directory): cd / then enter then 1s, will change to root directory. This will show the folders of the root including the ones that were hidden in the finder.

All of this is important to jump around the folders and the computer fast.

/users/Hassan there is nothing before the root directory. The short-cut to refer to /users/Hassan is just ~

10.1(5):Introducing WSL

The default for windows (power shell) has different commands that bash(which is what most developers use).

It runs Linux on windows.

The next few vids are just an installation guide.

10.1(5):Introducing WSL

Command (pwd): will show where in the computer the terminal is.

When you make a folder on wsl on a windows machine, it will not show up in the file explorer(you will not see it, though it is saved on the machine just not visible). It can be found, but it is in a completely different location than normal desktop location, that is where all the Linux files are.

It used to be the case with wsl 1 that you could not work with the files outside of the terminal, with wsl 2 there is a command to do so. The command is:

command(explorer.exe .): This will open the Linux files in file explorer. It opens the file in the network. If you type the **ls** command, it will show the files in the Linux files

Use up arrow in terminal to use re use lines.

****On mac there is no difference between files, on windows there are the normal files and the linux files, so mac is better here.****

10.1(11):Windows - pwd and ls

pwd(print working directory): show the path to the current folder that you are on.

By default it shows **~$** which references this folder: /home/hassan, pwd will display this. Everything in the course will be in this folder.

If you are in another folder, pwd will show exactly that.

command(ls): will list the files of the current directory. On windows it will only show the files on the Linux files, on mac it will show all the files on the mac(mac is better here). It will not show hidden files tho. If you type explorer.exe . you can see the hidden files, they all start with a dot(.). You can make a hidden file by starting with a dot.

10.1(13):cd

Command(cd:change directory): This is how you move, how you navigate into and out of folders from the terminal.

If you have a space in a file name, it will show with quotes when you type ls, so use something like file-2 or something.

The file name is First, to get to it type **cd First** this will change the ~$ to ~/First$ to show that you are in the file called First, to get back to home directory type **cd ~**

When in the First file you can do the same ls to show the files in this file and go into them and so on and go back and so on… cd nested-folder will give ~/First/nested-folder$.

Remember ~ represents /home/hassan on this account.

To go back a folder type: **cd ..**

You can in one line go into a nest folder by doing **cd First/nested-folder** and so on if there are more.

If you want to cd back into the first folder in the home directory instead of doing a bunch of cd .., you could do cd /home/hassan/First or cd ~/First and if you want to go home right away do cd ~

If you do explorer.exe . , it will navigate you to the file you are on.

10.1(16):Windows - Flags

An example of a flag is ls -a, or ls -l, these mean different things.

There are many flags to remember.

ls -a: lists all files **including** hidden files.
Ls -l: Will give more info about each file like when they were created

You can **combine** flags, **ls -al or ls -la** will list all flags and give more info about all the hidden flags.

10.1(17):Windows - Accessing Windows Files From Linux

To access windows files from linux, requires you to remember the location of the file.

If you back out all the way using cd .. or just type cd /, you will get to the root directory, and when you type ls, it will display a file called mnt, and inside of that you can find the c drive.

Type cd mnt then cd c (or the whole path) to get the c drive. Then do cd Users, then cd hassa(this is the user name on my pc), then ls on this will show the files on the computer like desktop, pictures, downloads etc and go from there and ls.

The whole path to desktop is **/mnt/c/Users/hassa/Desktop **

The suggestion for this entire course, is to just store all the projects in the home directory on the Linux side and access them using Ubuntu. **

10.1(19):man pages

man(command): stands for manual, you can see the manual for ls by typing man ls. Man ls shows a bunch of flags(all of the available flags) for ls.

ls -S: this flag will sort the list based on file size. Remember can combine multiple flags ls -Sa will sort by file size including hidden files.

Ls -1 will list it in a column rather than a row.

10.1(20 and 21,23):mkdir, touch, rm and rmdir

- Make files and folders using **mkdir**(for folders) and **touch(**for files)
- Move files and folder using **mv**
- Copy files and folders using **cp**
- Remove files and folders using **rm** and **rmdir**
- Will see 4 other commands: **cat, echo, less, open**
- Can mkdir multiple folders: mkdir folder1 folder 2
- Can remove by doing rmdir folder1
- Can make a file by doing **touch file.js**
- Can remove file by doing rm file.js
- You can ls when you are in a different location, if you are in the home directory and want to see what is inside of the folder called First that is inside of the home directory, you can do: **ls First** inside of the home directory and see what is inside of the folder called First.
- To open a file on vs code, go to that file location and type code .
- hassan@DESKTOP-RS4N0FC:~/springboard/8.3$ **code .**
- If you do code . on the springboard file(that i made to store all the course content), it will open all the files on vs code.

- Can do **touch index.html app.css app.js about.txt** This is faster than doing it manually.
- rm delete the files outright, not putting it in the trash, so be careful.
- you cannot rmdir a folder with content inside.
- you can delete a folder with things in it by using the command: **rm -rf (**rf are two flags that you pass in). Be careful with this one as you could end up removing something incredibly important.
- you can touch multiple files in. if you want to make a file inside of the folder fish and you are in a location right outside of the fish folder, you can do that by using the correct path: touch fish/goldfish.js. this will create the file goldfish.js inside of the folder fish. like other times, you can delete multiple or create multiple files by separating them with a space.

10.1(22):cat and echo

Cat: shows the content of a file, if you go into ~/springboard/8.1, and type
cat 'Unit 8.1.HTML' (this is an html file inside of 8.1), this will show the content of the html file completely.

echo: will repeat back whatever you type, echo "hello" will repeat hello. Seems useless, but there are other ways of using it.

- With echo you cannot use !
- You can combine echo with a **redirect**(more detail later). With this combo, you can echo something and redirect it to a file(place it in the file). If you do:
  touch echoExample.txt, then echo 'redirected message' > echoExample.txt, this will place the 'redirected message' inside of the txt file. Can be useful if you want to put a one liner in a file.
- Redirect **overwrites** the existing content of the file, do not do it with a file that has things in it.
- Redirect is used when you need to make a very simple file for an application.
- You can also make a file using redirect, if you redirect to a file that does not exist, it will make that file in the location that you are in.
- to open a specific file in vs code use **code filename.js/html…**
- to open all the folders just do code . on the springboard folder or any specific folder.

10.1(28): mv and cp

mv(move)**:** used to move files.

- you need to specify what file you are moving and where you are moving it to.
- **mv test.txt ../** this will move the file text.txt back one folder. This also works on folders, mv folder1 ../ will move the folder called folder1 back one folder.
- if you want to move it into a folder that is ahead do mv file.txt ahead/,
  mv echoExample.txt springboard/
- you can also move multiple locations by giving the entire location
  ~/springboard/8.1
- can use mv to rename, if you want to rename the springboard file to springboard2 you have to do: mv springboard/ springboard2, since there is no springboard 2 file, it will change it to that name.
- you can rename it and move it at the same time by doing:
  mv hello.txt ../rename.txt , this will move hello.txt back one path, and rename it to hello.txt assuming there isn't already a file named that.

cp(copy): is used to copy files

- cp path to original file path to copied file
- ex. i created a file in my home directory called test, here is how to copy:
  cp test.txt springboard/test_copy.txt, this will make a copy of test.txt called text_copy.txt and move it to the springboard file.
- cannot copy folders by default, you have to use a flag (r) **cp -r**
- you can make a copy in the same folder without mentioning
- **cp -r springboard springboard_copy** will make a copy of the file springboard in the same destination(in this case the home directory).

10.1(29):Terminal Shortcuts(keyboard shortcuts)

- **control + a:** will move to the beginning of the line.
- **control + e:** will move to the end of the line.
- **control + w:** will delete the word that your cursor is on except the letter that the cursor is on. going to the end of the word and using this shortcut will delete the word(including .txt)

- **control + u:** will delete the entire line. if you are in the middle of the line, it will only delete up to that point.
- **ctr + (left arrow/right arrow):** will jump left and right across words.
- many many more, these were just the ones shown in the video.

10.1(30):stick with it

terminal isn't just to make and remove files, it is also needed for certain programming languages like python.

you have to run databases from the terminal ect, so get used to it.

need the terminal to make a server.

tip: get a sticky note that has common commands, and refer back to it when needed.