* Linux is an operating system (Collection of software that manages a hardware system)
* Its also a kernel
* It’s (The kernel) the intermediary between software and hardware. It’s the heart of the operating system
* Free open source software
* Its not a unix derivative
* Linux distribution is a Linux kernel plus additional software that create the operating system
* <<Screenshot (50).png>>
* Most common Linux OS are redhat and ubuntu
* You need a license to use redhat but if you don’t want to pay for a license use centos which a redhat derivative
* ubuntu is popular is startups, SaaS, social networks or cloud computing
* SSH stands for secure shell which allows you to connect from one system to another. It replaced telnet
* Putty is the most popular SSH client for windows

***COMMON DIRECTORIES***

* Root is the top of the file system hierarchy and also known as /
* /bin or binaries is where you find executable programs. They are called binaries because machine code is a string of 0s or 1s
* /etc is the configuration file which controls how the operating system behave
* /home is where user home directories work. This is where you can separate your data from another users data. You can also store anything you want here like pictures
* /opt is where third party software live. Example like google earth
* /temp is for temporary space. Typically cleared on reboot. Great place to store temporary files
* /usr is where user related files live. They can have subdirectories. There will be a subdirectory within users
* /var is for variable data. Data that change often.
* Folder is the same as directory
* /boot files needed to boot the operating system
* /cdrom mount point for CD-ROMS. The files for the cd rom gets located here
* /cgroup controls group hierarchy
* /dev device files, typically controlled by the operating system and the system administrator
* /etc system configuration files
* /srv location to place data which is served by the system
* /srv/www web server files
* /sys used to display and sometimes configure the devices known to the Linux kernel
* some applications that are bundled with the Linux structure are put in the user local
* Root can be the super user account or the beginning of the file system
* The root is all powerful as it can do anything that can be done on a Linux system
* Typically restricted to administrators
* Linux command are case sensitive
* Ls -l displays a long listing which means more detail info about the file
* Cat displays the content of a file
* Echo displays the content of variables

***THE MAN COMMAND***

* Enter moves down the page
* Lower case g move to the top of the page
* Capital G moves to the very bottom
* To quit simply type q

***ENVIRONMENT VARIABLES***

* Storage location that has a name and value
* Typically uppercase
* Access the content by executing echo $VAR\_NAME
* Man -k -- search term to find out what it does
* Man is used to display documentation
* Can use which path to find the path of the command you are looking for

***DIRECTORIES***

* Single . Represents the current directory and .. Represents the parent directory
* All directories end in /
* Echo $OLDPWD is the directory you were previously in
* You can execute a command not in your $PATH
* ./command execute command in this directory

***THE COMPLETE LINUX COURSE: BEGINNER TO POWER USER***

* So many Linux distributions( distros)
* Problem with Linux is how many distros there are
* Debian is the base of countless Linux distributions
* The base of ubuntu is debian and it has be forged countless times
* It includes its own desktop environment called unity
* Naming convention of ubuntu is year month. Newest version is ubuntu 15.10 that is released in October 2015
* To download Ubuntu go to ubuntu.com
* Unity is a desktop environment of ubuntu
* Kubuntu is ubuntu with KDE file installed
* What is swap space? This is extra RAM to access memory that’s not on the RAM but on the hard drive.
* EXT4 journaling file system is the best file system in Linux
* The / is the root and /home is for home
* Ctrl plus alt and key brings up terminal
* The kernels job is to translate between software and hardware
* Guest edition helps us run the vm more efficiently and with higher performance
* Files with .run can be run directly from the terminal
* ./ in Linux means current directory
* Pwd is print working directory
* Cd is change directory. This changes the directory you are in
* Cd / means an absolute path from the root directory
* *The ~ ( tilde) is the home directory*
* Cd ../ takes you to the home directory
* Ls is to list all the contents in the directory you are currently in
* There's many options in ls
* Ls -l (Long list) list all the contents in the directory with the permissions for each content on the left
* To configure the shared folder in ubuntu virtual box you need to configure virtual box additions before you do anything else. To do so run the following command in the terminal in the virtual box

sudo apt install linux-headers-$(uname -r) build-essential dkms

* Ls -r list everything in reverse alphabetical order
* Directories are typically blue in color
* Ls -p is to define file types
* Ls -s is to list based on file size
* To list files in a certain directory without leaving the root directory just pass the command while in the root directory
* To list commands in reverse order while in the root directory run ls /first directory/second directory/ -r
* *To jump from one directory to another simply type cd ~/Name of directory. For instance if you are in documents and want to go to Downloads type cd ~/Downloads and youll automatically be taken to Download without having to go back to the home directory*

***Administrator privileges in Linux***

* Sudo in linux means super user do
* Some commands would not run unless you are in sudo
* **Su** or switch user changes you to the user you want
* Etc is the etcetera directory
* In nano ctrl + o is to save and ctrl + x is to exit the file
* Sudo !! Means to run the previous command
* Sudo su - gets you to the /root directory
* Sudo su gets you to the /home/user in this case the user is bkargong.

Package management in Ubuntu

* Done through *apt-get*
* Package management for ubuntu is aptitude hence the word apt-get
* To install packages you always start with sudo apt-get then what package you want installed
* This is how to install programs
* To remove packages you type sudo apt-get remove plus program name
* Sudo apt-get upgrade to upgrade packages that need to be upgraded
* Dpkg to install local .deb files. Lets us install packages that are not in the official ubuntu repository
* Rpm installer files are for red hat and not ubuntu
* To remove a program simply type *sudo apt-get remove* then package name
* To search in the repository something you don’t remember the name type *sudo apt-cache search package name\* ex sudo apt-cache search bluefish\**
* You can also type *sudo apt-cache policy then name of program*
* To find out what you have installed you can type sudo apt-cache policy then name of program
* We can use dpkg to install local .deb files. This lets us install packages that aren't in the official ubuntu repository
* To upgrade packages you type *sudo apt-get upgrade plus program name*
* To make a directory you type *sudo mkdir plus directory name*
* nmx

***File permissions and ownership in Linux***

* Rw r r means the owner can read and write, the group can read and others can read
* Chown means change ownership of the file. This helps with the file permissions
* To change ownership you do sudo *chown user:group* and the file you want to be changed to. So for example *sudo chown bkargong:bkargong file.txt*
* 6 means it is readable and writable, 4 means its readable only and 7 means it’s a directory which is not used often
* The user is the first column and the group is the second column
* To change the permission you can do chmod plus number for permission
* So chmod 664 means user and group can read and write and others can only read
* To exit out of nano you type ctrl + x
* If I want to won a file I simply use the sudo chown bkargong:bkargong plus file name
* To remove a file you type *rm + file name*
* To create a file in a directory you type *sudo nano ./directory name/file name* ex sudo nano ./mydir/file.txt
* You can also do ls ./directory name to see what's in that directory
* The files created above will be owned by root because they were created using the super user
* To change them to a new user you do *sudo chown -R user:group ./directory name* ex sudo chown -R bkargong:bkargong ./mydir where -R means its recursive meaning the command applies to all the files in that directory. We are running this command as sudo cause the directory belongs to root and we don’t have permission
* Using sudo chown bk:bk ./mydir will only change the ownership of the directory not the files in it
* Never make any directory or file 777 or 666. 664 is the most used as it gives the user read and write and the group and public only read.
* You can also do chmod -R plus permissions ex chmod -R 664
* You can also share files to different user by running the command *chown -R user:group /usr/share*. Ex *sudo chown -R BK:BK /usr/share Download*s. Be careful when using this command as some files need to be owned by root in order to work
* To create a new file run touch plus file name ex *touch file1*
* To remove all files with a certain extension run the command rm ./\*plus extension ex *rm ./\*.txt*
* To remove a directory you run the command rm -rf plus directory name ex *rm -rf mydir*
* If you want to remove all but one directory you run rm directory name/\* ex rm mydir/\*. In this case all other directories except mydir will be removed
* Cp is copy and the first argument is the file you are trying to copy and the next file is the file which you are trying to copy to ex *cp file1 ./mydir/file2*
* To remove a file in a directory just type the path leading to the file in the directory
* To remove all files in a directory run *rm directory name/\*.txt ex rm mydir/\**.txt. In this case it removes all the text files in the mydir directory
* To remove every file in a directory run *rm dir/\**. This removes everything within the directory and leaves the directory empty.
* Moving a file is like copying but it does leave the file in the source. In this case you run *mv file1 ./dir/file2*
* To make an absolute directory you can run *mkdir /usr/bin + dir* which creates a directory in that path
* You can also create a directory one level or 2 levels up by running either *mkdir ../* + dir or *mkdir ../dir*
* Mkdir stands for make directory
* To go one level up you run ../ or 2 levels up you run ../../
* To create a directory in the current directory you run mkdir ./
* To move a file from one directory to another you run *mv file newdir/new file name*
* To copy a file you run cp ./dir/filename ./dir/filename2. you can either keep the original name of the file that you copied in the new directory or you can change the name when you copy it

***USING THE FIND COMMAND***

* To use the find command
* You can use the find command to navigate within some directories
* To find a file you run *find . -type f -name "\*.php"* to find all php files. Here the . Is for the current directory -type is for what we are finding, f is for file and -name is for the file name
* In case you want to ignore all case sensitive file extensions run the same command as above but for name you type iname in which case the I stands for ignore case sensitive extensions. So its going to be *find . -type f -iname "\*.php"*
* F finds files and d find directories
* To find files with a certain permission you can run *find . -type f -perm 0664* for rw- rw- r
* *Find . -type f -not -iname "\*.php"* finds all files that are not php
* *Find . Maxdepth 1 -type f -iname "\*.conf"*

***USING THE GREP COMMAND TO FIND FILES***

* To use the grep you run *grep "function" ./\** to find all files in the current directory
* Grep -n -I "function" ./\*
* To use bot the find and grep command you run *find . -type f -iname "\*.txt" -exec grep -I -n "function" {} +*. The exec command works after the find command. So after the system finds the files with the extension, the second part of the command if then executed hence the word exec. The open and closed curly braces and the plus sign help close the exec command.
* To redirect the output of a command you run *ls > outfile.txt* or whatever you want to redirect the command to
* In the grep command the word in the parenthesis is what you are trying to find so in the example above we are looking for function but if I wanted something else ill just input it in the parenthesis and run the command
* So the full command for find, grep and output redirect is find . -type f -size -10k -iname "\*.php" exec grep "option" {} + > filename to be redirected to ex *find . -type f -size -10k -iname "\*.php" exec grep -I -n "name" {} + > file.txt* . The first part of the command fins all the files with the .php extensions that are less than 10k and then the second part finds the files with the word name in them and then redirects their output to file.txt
* The I and n in the grep command ignores case sensitivity and numbering lines respectively
* If you want to redirect the output of your command and also see them on the screen you run the tee command. This will look like this *find . -type f -size -10k -iname "\*.php" exec grep -I -n "name" {} + | tee file.txt*

***WHAT ARE PROCESSES AND HOW YOU CAN MANAGE THEM***

* A process is a running application basically
* You can find the running application by running the top function. So its going to be *top. This command run in real time*
* *PID* is the process ID. You can use this to manage the process
* The user is who the application is running as. You can see how long the application has been open
* You can use *ctrl + c* to escape
* To see the entire list you run *ps aux* which is not real time. It captures it at the time you ran the command.
* To find a process specifically you can run *ps aux | grep name of process*
* To get the process id of a running process you run *pgrep name of process*. The order of the process id is chronological so the top one is the oldest process and the last is the newest one that was ran.
* To kill a process you run *kill -9 PID*
* You can also kill apps by runing *kill -9 xxxx yyyy* where xxxx and yyyy are the PID
* You can run *killall plus browser name* to kill all the processes of that application

***SERVICES***

* They are a type of Linux process
* We start a service by typing *sudo service plus service name start*
* This starts the service
* To stop the service we type *sudo service name stop*
* *Sudo service plus service name restart* to restart the service
* *Sudo systemctl (system control) start service name* is another way to manage services.

***CRONTABS OR CRON JOBS***

* This is how you schedule a task or job to run at a certain time
* You can do so by typing *crontab -e*
* M in the crontab is minutes, h is hours, dom is day of month, dow is day of week and command is the command you are trying to run
* So to run a command at 15 minutes and at 14 hours regardless of day of the month, month, and day of week the command is going to look like this 15 14 \* \* \* ls > /home/bkargong/Adeck.txt
* Notice that the command is going to be outputted in the home/bkargong directory in the Adeck.txt file
* In the crontab Sunday is 0, Monday is 1 and so on
* So in the above command if I wanted the command to run every Monday at noon it will look like this *00 12 \* \* 2 ls > /home/bkargong/Adeck.txt*
* To run the crontab command as a super user all you need to do is do *sudo crontab -e*. This is a great tool when trying to running maintenance commands for the system that require the sudo command.
* To see the running cron jobs you run *crontab -l* or *sudo crontab -l* for super users

***INTEGRATED DEVELOPMENT ENVIRONEMNTS (IDE)***

* There's a difference between a code editor and an IDE
* A code editor is used to edit either single or multiple files and they usually provide a decent syntax highlighting ,smart indentation and sometimes code completion
* However, if you are working in a large project or need a more complete set of tools for development
* IDE offer the same code features as code editors but also more advanced features like version control support, tool chains and ways to run the application from right in the IDE.
* Some IDE support multiple languages while others only support 1. ex Eclipse is a widely used IDE that provides version control support and support for various languages. It requires JAVA to be installed to use the IDE.
* Code block is another popular IDE
* Qt is a framework

An IDE is **a software development environment that provides tools to develop complex software**. It offers various tools like an editor, debugger, and compiler needed in the development cycle. Different IDEs provide different features and configurations suitable for specific projects.

* After you downloading the IDE install Java by running *sudo apt-get install openjdk-8-jre*. The version can be 7 or 8.
* To find java version type *java -version*
* To run the installer while in the directory with the installer run installer-inst. Ex ./*eclipse-inst* in our case
* With IDE you can run commands from the IDE itself unlike with code editor that you need the CLI to run your commands
* After you are done installing the IDE (Eclipse in this case) you go to file to create a new project and there's where you will be working
* For pycharm you go into the bin directory and run sh ./pycharm.sh

GIT AND WHY YOU SHOULD BE USING VERSION CONTROL FOR YOUR PROJECTS

* Git is a great tool if you are working in a team, large projects or on open source projects. It allows people work on the same file at the same time
* Merge conflict is when the same block of code is edited or modified by more than one person
* Command to install git is sudo apt-get install git git-extras
* The reason why we are installing git extras is because it gives us more control in git
* Go to github and create a new repository
* To add git to your CLI you type git remote add origin "url for github here"
* To configure your Github you run git config --global user.name "Nick" and git config --global user.email kbrigthain@gmail.com
* After doing the pull now you can set the upstream by running the following command git branch --set-upstream-to=origin/master
* To remove a directory run git rm -r plus directory name

METEOR JS AND HOW TO USE IT

Meteor **allows you to develop in one language, JavaScript, in all environments: application server, web browser, and mobile device**. Meteor uses data on the wire, meaning the server sends data, not HTML, and the client renders it.

* In the address bar type install meteor
* To install meteor run the command curl <https://install.meteor.com/> | sh
* If you get an error saying that curl is not a command then you don’t have it install so just go ahead and install it by running sudo apt-get install curl
* The meteor project is created in the home directory and from there you can cd into the project name
* From there you can create a project running meteor create plus project name
* Now cd into the newly created project and type meteor which will launch the program you are trying to create in port 3000
* To add packages to meteor you type meteor add plus package name
* To see hidden packages in file manager hit ctrl plus h
* To run multiple commands together you simply run the first command and add && which tells Linux to run the second command after running the first command