**Linux**

**Introduction to Linux :-**

* Linux is developed by “Linus tordvalds” in 1991
* Linux doesn’t belongs to any company
* Linux os is a type of os that is similar to UNIX and it is built upon the Linux kernel
* Linux kernel like brain of the os because it is going to manage how the computer interacts with its hardware,resources and mainly peripherals
* Multiple users can access this os
* High security then other os
* Linux is distributed os (600+ Linux distributions) like red hat,Ubuntu,fedora,amazon Linux,Kali Linux,SUSE Linux------------
* Linux is an community based
* It is a user and network frequently
* It is maintain the privacy of the user
* Linux has high stability it is rare to freeze or lag
* Linux provides high flexibility there is no need of restart the computer it will automatically rectify the issues
* Linux is an operating system like macos or windows
* Linux is an open source operating system
* It is an interface between user and computer in order to provide environment
* It acts as a mediating element
* Provides a perfect environment to create,run,access many applications

**Commands**

**Man**

This is a man (from manual) page. Man pages are an essential tool to learn, as a developer. They contain so much information that sometimes it's almost too much.

**ls**

Inside a folder we can list all the files that the folder contains using te ls command

**ls /bin**

If we add a folder name or path, it will print the folder contents

**ls -al /bin**

Using above command we can get much more information about folder

**cd**

the cd command is used to change the directory or we can go specified folder

Example:-

mkdir fruits

cd fruits

now we are in fruits folder

We can use .. special path to idicate parent folder

cd .. #back to the home folder

You can use it to form a path:

mkdir fruits

mkdir cars

cd fruits

cd ../cars

You can also use absolute paths, which start from the root folder / :

cd /etc

**Pwd :- (present working directory)**

**mkdir**

You create folders using the “mkdir” command

mkdir fruits

You can create multiple folders with one command:

mkdir dogs cars

You can also create multiple nested folders by adding the -p option: mkdir -p fruits/apples

Options in UNIX commands commonly take this form. You add them right after the command name, and they change how the command behaves. You can often combine multiple options, too.

**rmdir**

We can delete the folder which we created ny using mkdir.

mkdir fruits

rmdir fruits

We can delete multiple folders usig rmdir

mkdir fruits cars

rmdir fruits cars

The folder you delete must be empty. To delete folders with files in them, we'll use the more generic rm command which deletes files and folders, using the -rf options: rm -rf fruits cars

Be careful as this command does not ask for confirmation and it will immediately remove anything you ask it to remove. There is no bin when removing files from the command line, and recovering lost files can be hard

**mv**

The “mv”To move the file to current path to its new path

touch test

mv pear new\_pear

The pear file is now moved to new\_pear . This is how you rename files and folders.

If the last parameter is a folder, the file located at the first parameter path is going to be moved into that folder. In this case, you can specify a list of files and they will all be moved in the folder path identified by the last parameter:

touch pear

touch apple

mkdir fruits

mv pear apple fruits #pear and apple moved to the fruits

**Cp**

We can copy the file using cp command

touch test

cp apple another\_apple

To copy folders you need to add the -r option to recursively copy the whole folder contents:

mkdir fruits

cp -r fruits cars

**open**

The open command lets you open a file using this syntax:

Open <filename>

You can also open a directory, which on macOS opens the Finder app with the current directory open:

open <directory name >

I use it all the time to open the current directory:

open .

The special . symbol points to the current directory, as .. points to the parent directory

The same command can also be be used to run an application

Open <application name>

**Touch**

We can create an empty file using touch command

touch apple

If the file already exists, it opens the file in write mode, and the timestamp of the file is updated.

**Find**

The find command is used to find files or folders matching a particular search pattern.it searches recursively.

Example:-

find . -name ‘\*.js’

Its important to use quotes around special characters like \* to avoid shell intersepting them

Find directories user the current tree matching the name “src”

find . -type d -name src

Use -type f to search only files, or -type l to only search symbolic links.

-name is case sensitive. use -iname to perform a case-insensitive search.

**ln**

The

ln

command is part of the Linux file system commands.

It's used to create links. What is a link? It's like a pointer to another file. A file that points to another file. You might be familiar with Windows shortcuts. They're similar.

We have 2 types of links: **hard links** and **soft links**.

**Hardlinks:-**

Hard links are rarely used. They have a few

limitations: you can't link to directories, and you can't

link to external filesystems (disks)

ln recipes.txt newrecipes.txt

**Softlinks:-**

Soft links are different. They are more powerful as you can link to other filesystems and to directories, but when the original is removed, the link will be broken.

You create soft links using the -s option of ln:

ln -s recipes.txt newrecipes.txt

**Ggip:-**

You can compress a file using the gzip compression protocol named LZ77 using the gzip command

gzip filename

This will compress the file, and append a .gz extension to it. The original file is deleted. To prevent this, you can use the -c option and use output redirection to write the output to the filename.gz file:

#### ****gunzip****

The gunzip command is used to decompress .gz files. It works like the gzip command but always has the -d option enabled.

**Usage:**

gunzip filename.gz

Decompresses filename.gz and removes the .gz extension.

Overwrites the original file if it already exists.

**Extract to a Different File:**

gunzip -c filename.gz > newfile

#### ****tar****

The tar command creates and extracts archives (grouping multiple files into one). It originated from "tape archive," as archives were historically stored on tapes.

**Create an Archive:**

tar -cf archive.tar file1 file2

-c: Create an archive.

-f: Specify the archive file name.

**Extract Files:**

tar -xf archive.tar

-x: Extract files.

-C directory: Extract to a specific directory.

**List Archive Contents:**

tar -tf archive.tar

**Create a Compressed Archive:**

tar -czf archive.tar.gz file1 file2

-z: Compress the archive using gzip.

**Extract a Compressed Archive:**

tar -xf archive.tar.gz

The command automatically detects and decompresses .gz archives.

#### ****alias****

An alias is a shortcut for a command with specific options.

**Example:**

alias ll='ls -al'

Creates ll as a shortcut for ls -al.

Use alias alone to list all aliases.

**Permanent Alias:**

Add the alias to your shell configuration file,

e.g., ~/.bashrc or ~/.bash\_profile.

**Cat**

Cat is similar to tail in someway,it is simplest usage cat print a file to the standard output

Cat file

Cat file1 file2 (to print multiple files)

**less**

it shows content stored inside a file

Less<file name >

窗体顶端

**Tail**

tail is used to see last lines

Tail 100

tail -n 10 <filename>

**Wc**

The wc command gives us useful information about a file or input it receives via pipes.

echo test >> test.txt

wc test.txt

1 1 5 test.txt

-l to check how many lines

-w to check how many words

-c bytes to check how many bytes

**grep**

Grep command is used to search the files

You can use grep to search in files, or combine it with pipes to filter the output of another command.

Example:

grep document.getElementById index.md

**Sort**

To sort the list files in alphabetical order or numeric asc or dsc order

Sort -r

**Uniq**

the command uniq is used to sort lines of the text

You can get those lines from a file, or using pipes from the output of another command:

uniq dogs.txt

ls | uniq

**diff**

Diff is handy command.suppose we have 2 files which contains almost the same information but u cant find the difference between two

The diff command is process the file and will tell what is the diffrence

diff dogs.txt moredogs.txt

**echo**

The echo command does one simple job: it prints to the output the argument passed to it.

**Ex:-**

echo “hello”

echo \* (we can see echo files in current folder )

echo o\*(we can see echo files starts with o)

echo $(ls -al)

**Chown**

Every file/directory in an Operating System like Linux or macOS (and every UNIX systems in general) has an **owner**. The owner of a file can do everything with it. It can decide the fate of that file. The owner (and the

root user) can change the owner to another user, too, using the chown

command:

chown jeevan jeevan.txt

**Chmod**

Chmod command uses to give the permission read write and excute

There are number of combination and will give the permission as below

1 if has execution permission

2 if has write permission

4 if has read permission

This give us 4 combinations

0 no permissions

1 can execute

2 can write

3 can write, execute

4 can read

5 can read, execute

6 can read, write

7 can read, write and execute

chmod 777 filename

chmod 755 filename

chmod 644 filename

**du**

this command is used to calculate the size of the directory as a whole

Running du \* will calculate the size of each file individually

**df**

This command is used to check the disk usage

df -h (to see the human format)

**basename**

Suppose you have a path to a file, for example

/Users/jeevan/test.txt

we will get last sigment of the path

**Kill**

we can terminate the program using kill command

Kill <pid>

**Whoami**

This command uses to to see the username of currently logged in the terminal session

**Who**

Who command the users logged into the system