Listing files (ls): If you want to see the list of files on your UNIX or Linux system, use the '**ls**' command. It shows the files /directories in your current directory.

You can use **'ls -R' to shows all the files not only in directories but also subdirectories**

'ls -al' gives detailed information of the files. Also, to view hidden files, use the command**.**

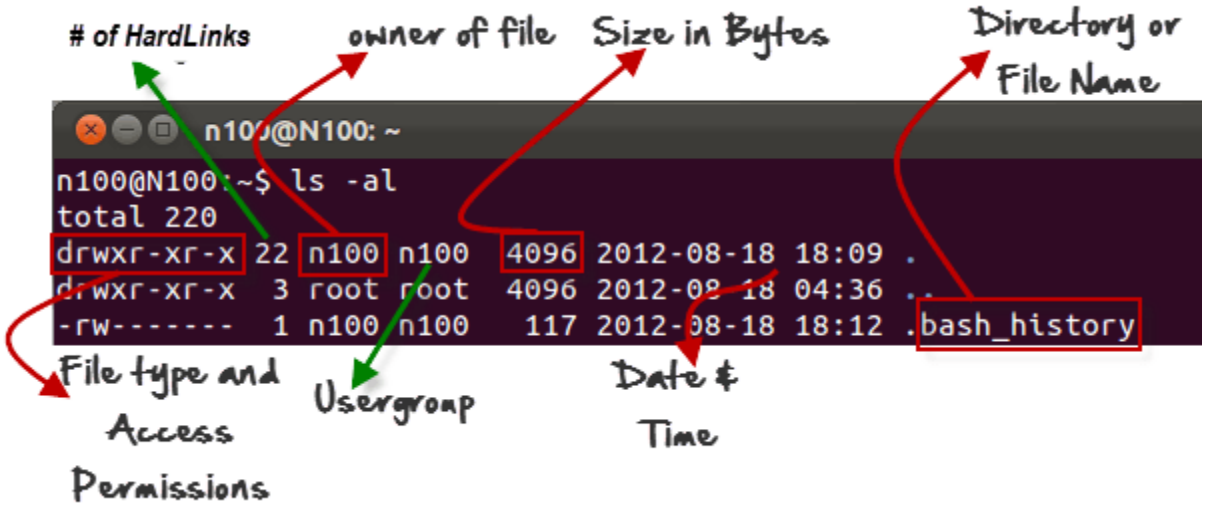
pwd — When you first open the terminal, you are in the home directory of your user. To know which directory you are in, you can use the **"pwd"** command. It gives us the absolute path, which means the path that starts from the root. The root is the base of the Linux file system. It is denoted by a forward slash(/). The user directory is usually something like "/home/username".

cat: The cat command (short for "concatenate") lists the contents of files to the terminal window**.**

Example: cat filename

cd: The cd command changes your current directory. In other words, it moves you to a new place in the filesystem. Example: cd directoryname

chmod: The chmod command sets the file permissions flags on a file or folder. The flags define who can read, write to or execute the file. When you list files with the -l (long format) option you'll see a string of characters



If the first character is a `-` the item is a file, if it is a `d` the item is a directory. The rest of the string is three sets of three characters. From the left, the first three represent the file permissions of the *owner*, the middle three represent the file permissions of the *group* and the rightmost three characters represent the permissions for *others*. In each set, an `r` stands for read, a `w` stands for write, and an `x` stands for execute.

If the `r`, `w`, or `x` character is present that file permission is granted. If the letter is not present and a `-` appears instead, that file permission is not granted.

One way to use `chmod` is to provide the permissions you wish to give to the owner, group, and others as a 3 digit number. The leftmost digit represents the owner. The middle digit represents the group. The rightmost digit represents the others. The digits you can use and what they represent are listed here:

- **0:** No permission
- **1:** Execute permission
- **2:** Write permission
- **3:** Write and execute permissions
- **4:** Read permission
- **5:** Read and execute permissions
- **6:** Read and write permissions
- **7:** Read, write and execute permissions

To set the permission to be read, write, and execute (7 from our list) for the *owner;* read and write (6 from our list) for the *group;* and read and execute (5 from our list) for the *others* we'd need to use the digits 765 with the `chmod` command:

Example: chmod -R 765 test.txt

chown: Use the chown command to change file owner and group information.

Example: chown Mahesh test.txt to change the ownership of the file

chown Mahesh:Mahesh test.txt to change the ownership and group of the file

rm - Use the **rm** command to delete files and directories. Use "**rm -r**" to delete just the directory. It deletes both the folder and the files it contains when using only the **rm** command**.**

Example: rm test.txt

touch - The **touch** command is used to create a file. It can be anything, from an empty txt file to an empty zip file.

Example: touch new.txt

man & --help — To know more about a command and how to use it, use the **man** command. It shows the manual pages of the command. For example, "**man cd**" shows the manual pages of the **cd** command. Typing in the command name and the argument helps it show which ways the command can be used

Example: cd -help

cp – Use the **cp** command to copy files through the command line. It takes two arguments: The first is the location of the file to be copied, the second is where to copy.

mv – Use the **mv** command to move files through the command line. We can also use the **mv** command to rename a file. For example, if we want to rename the file "**text**" to "**new**", we can use "**mv text new**". It takes the two arguments, just like the **cp** command.

Example: mv new.txt example.txt

echo – The "**echo**" command helps us move some data, usually text into a file. For example, if you want to create a new text file or add to an already made text file, you just need to type in

Example: echo "hello, my name is alok" >> new.txt

sudo – A widely used command in the Linux command line, **sudo** stands for "SuperUser Do". So, if you want any command to be done with administrative or root privileges, you can use the **sudo** command.

Example: sudo vi temp.txt

df – Use the **df** command to see the available disk space in each of the partitions in your system. You can just type in **df** in the command line and you can see each mounted partition and their used/available space in % and in KBs. If you want it shown in megabytes, you can use the command "**df -m**".

du – Use **du** to know the disk usage of a file in your system. If you want to know the disk usage for a particular folder or file in Linux, you can type in the command **df** and the name of the folder or file.

Example: du test.txt

ps - The ps command lists running processes. Using ps without any options causes it to list the processes running in the current shell.

Kill - The kill command allows you to terminate a process from the command line. You do this by providing the process ID (PID) of the process to kill.

tar – Use **tar** to work with tarballs (or files compressed in a tarball archive) in the Linux command line. It has a long list of uses. It can be used to compress and uncompress different types of tar archives like **.tar, .tar.gz, .tar.bz2**,etc. It works on the basis of the arguments given to it. For example, "**tar -cvf**" for creating a **.tar** archive, -**xvf** to untar a tar archive, -**tvf** to list the contents of the archive

zip, unzip – Use **zip** to compress files into a zip archive, and **unzip** to extract files from a zip archive.

uname – Use **uname** to show the information about the system your Linux distro is running. Using the command "**uname -a**" prints most of the information about the system. This prints the kernel release date, version, processor type, etc.

apt-get – Use **apt** to work with packages in the Linux command line. Use **apt-get** to install packages.

This requires root privileges, so use the **sudo** command with it.

hostname – Use **hostname** to know your name in your host or network. Basically, it displays your hostname and IP address. Just typing "**hostname**" gives the output. Typing in "**hostname -I**" gives you your IP address in your network.

ping – Use **ping** to check your connection to a server. Example: ping google.com

chown - The chown command allows you to change the owner and group owner of a file.

find - Use the find command to track down files that you know exist if you can't remember where you put them. You must tell find where to start searching from and what it is looking for.

Example: find . -name java

free - The free command gives you a summary of the memory usage with your computer. It does this for both the main Random Access Memory (RAM) and swap memory. The -h (human) option is used to provide human-friendly numbers and units. Example: free -h

grep - The grep utility searches for lines which contain a search pattern.

groups - The groups command tells you which groups a user is a member of.

head - The head command gives you a listing of the first 10 lines of a file. If you want to see fewer or more lines, use the -n (number) option. Example: head filename

tail - The tail command gives you a listing of the last 10 lines of a file. If you want to see fewer or more lines, use the -n (number) option.

less - The less command allows you to view files without opening an editor. It's faster to use, and there's no chance of you inadvertently modifying the file. With less you can scroll forward and backward through the file using the Up and Down Arrow keys, the PgUp and PgDn keys and the Home and End keys. Press the Q key to quit from less. Example: less filename

man - The man command displays the "man pages" for a command in less . The man pages are the user manual for that command. Because man uses less to display the man pages, you can use the search capabilities of less. Example: man grep

w - The **command w** on many Unix-like operating systems provides a quick summary of every user logged into a computer

mkdir - The mkdir command allows you to create new directories in the filesystem. You must provide the name of the new directory to mkdir. Example: mkdir test

ssh - Use the ssh command to make a connection to a remote Linux computer and log into your account. To make a connection, you must provide your user name and the IP address or domain name of the remote computer. Example: ssh user@192.168.32.43

whoami - Use whoami to find out who you are logged in as or who is logged into an unmanned Linux terminal.

adduser - In Linux, a 'useradd' command is a low-level utility that is used for adding/creating user accounts in Linux and other Unix-like operating systems.

userdel - Use the *userdel* command to remove the old user

passwd - Used to change the user account passwords

top - The top command shows you a real-time display of the data relating to your Linux machine. The top of the screen is a status summary.

The first line shows you the time and how long your computer has been running for, how many users are logged into it, and what the load average has been over the past one, five, and fifteen minutes.The second line shows the number of tasks and their states: running, stopped, sleeping and zombie.The third line shows CPU information. Here's what the fields mean:

- us: value is the CPU time the CPU spends executing processes for users, in "user space"
- sy: value is the CPU time spent on running system "kernel space" processes
- ni: value is the CPU time spent on executing processes with a manually set nice value
- id: is the amount of CPU idle time
- wa: value is the time the CPU spends waiting for I/O to complete
- hi: The CPU time spent servicing hardware interrupts
- si: The CPU time spent servicing software interrupts
- st: The CPU time lost due to running virtual machines ("steal time")

The fourth line shows the total amount of physical memory, and how much is free, used and buffered or cached.The fifth line shows the total amount of swap memory, and how much is free, used and available (taking into account memory that is expected to be recoverable from caches).The columns in the main display are made up of:

- PID: Process ID
- USER: Name of the owner of the process
- PR: Process priority
- NI: The nice value of the process
- VIRT: Virtual memory used by the process
- RES: Resident memory used by the process
- SHR: Shared memory used by the process
- S: Status of the process. See the list below of the values this field can take
- %CPU: the share of CPU time used by the process since last update
- %MEM: share of physical memory used

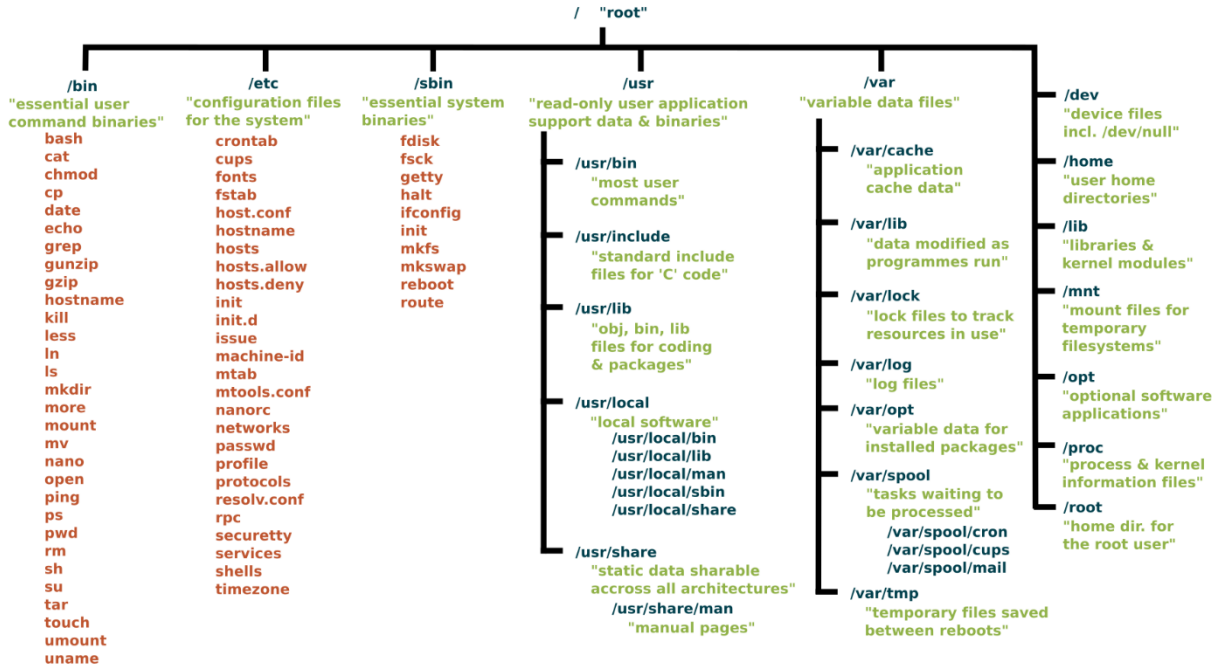
- TIME+: total CPU time used by the task in hundredths of a second
- COMMAND: command name or command line (name + options)

The status of the process can be one of:

- D: Uninterruptible sleep
- R: Running
- S: Sleeping
- T: Traced (stopped)
- Z: Zombie

Press the Q key to exit from top.

Linux File System:



Run Levels in Linux:

In Linux Kernel, there are 7 runlevels exists, starting from 0 to 6. The system can be booted into only one runlevel at a time. By default, a system boots either to runlevel 3 or to runlevel 5. Runlevel 3 is CLI, and 5 is GUI. The default runlevel is specified in **/etc/inittab** file in most Linux operating systems. Using runlevel, we can easily find out whether X is running, or network is operational, and so on.

- 0 – Halt
- 1 – Single-user text mode
- 2 – Not used (user-definable)
- 3 – Full multi-user text mode
- 4 – Not used (user-definable)

- 5 – Full multi-user graphical mode (with an X-based login screen)
- 6 – Reboot