Command Line

<https://youtu.be/dQ8JgDUS8DA>

<https://www.puttygen.com/linux-commands>

* **cd**  *for changing directory to home folder*
  + cd / for changing directory to root
  + cd ~ for changing directory to home directory
  + cd .. for changing one directory back
  + cd ../.. for changing two directory back
  + cd <name> for changing directory to the given name directory
  + cd <path> for changing directory to the given path
  + cd <long name> for directories with long names
    - cd my\ long\ name
    - cd “my long name”
    - cd ‘my long name’

* **ls** *lists contents of current file*
  + ls <directory name> lists the contents of given file
  + ls ~ lists the contents of your home directory
  + ls .. lists the contents of one folder back
  + ls ../.. lists the contents of two folders back
  + ls -l lists the contents in long format i.e more details
  + ls -a lists all the contents in current, hidden files also
  + ls -lS lists the content in sorted order w.r.t file size
  + ls \*.html lists the content with html extension
  + ls \*.\* lists all the files with all the extensions
  + ls \*.\* > out.txt save list of sorted file as txt
  + ls -d lists all the directories
* **cat**  (*displaying, combining copies, creating new) text files* 
  + cat <textFile> for printing out file content
  + cat <textFile>... for printing out all files content together
  + cat -b <textFile> for printing content with line numbers

Does Not count blank lines

* + cat -n <textFile> for printing content with line numbers

Counts the blank lines

* + cat -s <textFile> prints with squeezing multiple blank lines to one
  + cat -E <textFile> prints with adding ‘$’ at the end of each line
* **Redirection[> , >>]** I/O stream transfer
  + <command> > out.txt sends command’soutput to a txt file
  + <command> >> out.txt appends command’s output to a txt file
* cat <textFile> textFile> > ab.txt new file with merge content
* **mkdir & rmdir** creating and deleting directories
  + mkdir <name> creates a directory with given name
  + mkdir <path/name> creates with *name* at given existant path
  + mkdir -p <path/name> creates directory and given path structure also
    - mkdir --parent <path/name> longer parameter version
  + mkdir -p <name/{list}> creates *name* and subdirectories with names in the list separated by commas[no spaces around commas].
    - mkdir -p subjects/{english, math, science}
* **cp\*** copying files
  + cp <options> <source> <destination> basic syntax
  + cp <filenameToCopy> <newFilename> create new file out of a file
  + cp <filenameToCopy> <directory> copies file into a directory
  + cp <filename>..<filename> <directory> copies multiple files to a directory. Two of above Overrides if files exists already.
  + Cp -i <filenameToCopy> <directory> copies but if file already there asks before overriding.
  + Cp <filenameToCopy> . copies file into same file
  + Cp -R <dir1> <dir2> Recursive copy directory
  + Cp -vR <filenameToCopy> <newFilename> create new file out of a file notifies at every step
* **mv\*** moving files
  + mv <options> <source> <destination> basic syntax
  + mv <filename1> <filename2> change file name
  + mv <filename> <directory path> move file to a path
  + mv -i <filename> <directory path> asks in the case of overriding
  + mv <directory1> <directory2> moving directory content
  + mv -v <filename> <directory> explanation whats was done
* **less\*** read the files or search in the file
  + less <filename> it will show some content, inside the text...
    - Proceed one line down/up with [down/up arrow key]
    - Page by page Downward movement with [space key]
    - Page by page Upwards movement with [‘B’ key]
    - End of the file with [‘G’]
    - At the start of file [‘g’]
    - Searching [‘/<word you are searching>’]
      * Press ‘n’ to go to next match
* **touch\*** creating new empty files and change time stamps on existing files
  + touch <filename> to create a file with given name [syntax]
  + touch <filename of existing file> time stamp is changed [data stays same]
* **nano\*** just a handy text editor cli
  + nano <filename> to open given file (creates if file isn't there)
* **sudo** to tell linux that you are administrator user while running a command
  + sudo <command> just identifying that you have admin privileges.

It would ask for the admin password and when you type the password it wont show that you are typing, so just type and press enter while assuming that it's happening.

* + sudo !! !! means insert here the last command used

If i used ‘ls’

sudo !! would be same as ‘sudo ls’.

* sudo -s changes to super user mode
* **top** to display processes consuming your memory [refreshes every 3 secs]
  + PID : process id
  + USER : which user using process
  + %CPU : cpu usage
  + %MEM : memory usage
  + TIME+: total time its been running
* If you press ‘i’ , it will only show running process
  + Pressing ‘i’ again will show all processes including idle ones
* If you press ‘k’ , it will ask for *PID* number to kill. By entering *PID* and pressing enter to agree, the processes with that *PID* will be killed[deleted].
* **kill -<flags> *PID***  kills all processes with given PID
  + ***xkill*** is for having cursor as killer
* ***Pidof <name of process>*** is for getting pid of the processes
* **kill -KILL <*PID*>**force kill
* **Kill -9 <*PID*>** extra force kill
* **ps -ux** gives list of processes.
* Eg:
  + Use ps -ux and get command for process you want to kill
  + User pidof to get its PID
  + Use kill <PID> to kill it
* **echo** Used to echo back text
  + echo “Text” It will echo back “Text” or anything on the place of it
  + Eg:
    - num=123
    - var=”abc $num”
    - echo $var would print “abc”
* echo -e ‘here \text’ it will enable backslash interpretation

In the text ‘\t’ would work as a tab instead

of \text

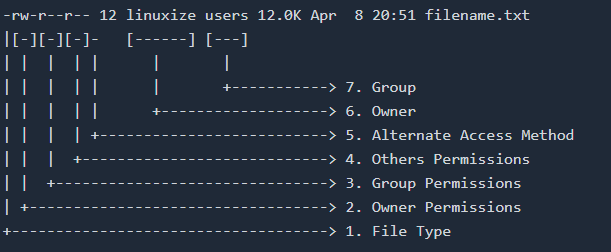
Note the single quotations.

* echo -e ‘here \new’ would give..

*here*

*ew*

* **File permissions**
  + Type ls -l to view permissions on files
    - It would look like *-rw-rw-r--*
    - r : read permission
    - w : write permission
    - x : execute permission
* chmod +x <filename> allowing everyone to write
* chmod <permissions> <filename> used to ‘change mode’ / change permissions
  + chmod o+w <filename> : allowing others to write
  + chmod o+x <filename> : allowing others to execute
  + chmod g+r <filename> : allowing group to read
  + In general:
    - chmod o/g/u +- w/r/x <filename>
      * o/g/u : others or group or you
      * +- : add (+) or remove (-) permission
      * w/r/x : write or read or execute permission
* chmod a-xrx <filename> no permission to anyone
* chmod ug=rwx <filename> all o/g/upermissions to everyone
* **Directory permissions**
  + Type ls -ld dir permissions
    - It will look like *drwxrwxr-x*
      * d : to indicate it is directory
      * r : read permission
      * w : write permission
      * x : execute permission



* **Octal and numerical permissions** give permissions in numbers
* https://ss64.com/bash/chmod.html

|  |  | **USER** |  |  | **GROUP** |  |  | **OTHERS** |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | r[read] | w[write] | x[execute] | r[read] | w[write] | x[execute] | r[read] | w[write] | x[execute] |
| 0 | 1 | 2 | 4 | 1 | 2 | 4 | 1 | 2 | 4 |

* Choose one of the three options for each of the permissions i.e user,group and others
  + Eg: chmod 214 <file>
    - Will be write for user, read only for group and execute for others
* Multiple permissions for each kind of users
  + Add up numbers under the permissions to form a sum
    - Eg: if you want to give user permission to write and execute

2[write]+4[execute] = 6

Chmod 641 <filename>

* Eg 2: if you want to give all kinds of permissions to all users
  + User 1+2+4=7
  + Group 1+2+4=7
  + Others 1+2+4=7

Chmod 777 <filename>

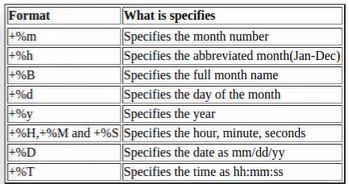
* Sum of numbers can be adjusted as per need
* **Bash Scripting** 
  + Extension of bash script files: ‘.sh’
  + Shabang[should be at the start]: #!/bin//bash
  + Execute:
    - Chmod +x <filename>.sh
    - ./<filename>.sh
  + Eg:

| #!/bin/bash  echo “HELLO WORLD” |
| --- |

* **which**
  + which <command/file> shows location of the command/file to identify
* **whatis**
  + whatis <installed cmd tool> explains what that tool is
* **Users** Multiple users management [it will require root permission]
  + useradd <username> -m[homeFolder] -s /bin/bash [shell] -g [userGroup] …

for creating a user with a home folder and a shell

* userdel <username> -r for deleting a user dir
* **Bashrc file**
  + It is executed whenever a new terminal is started
  + You can add code at the end of file to run whenever a terminal is opened
* **Resources**
  + du : used disk space available
    - du -h : for making it human readable form
    - du -s : for summary
  + df : free disk space available
    - df -h : for making it human readable form
    - du -s : for summary
  + free : gives used and free space
    - free -b: for viewing it byte unit
    - free -k: for viewing it kilo byte unit
    - free -m: for viewing it mega byte unit
    - free -g: for viewing it giga byte unit
    - free -h: for making it simple human readable form
* **watch** runs scripts/commands at a regular interval
  + watch <command/script name> runs & displays details every 2 secs initially
  + watch -n <seconds> <command/script name> runs & displays details for every given number of seconds
* **head/tail** shows first/last ten lines of the file
  + head <filename> shows first ten lines of the file
  + tail <filename> shows last ten lines of the file
  + head -<no of lines> <filename> shows first given number lines of the file
  + tail -<no of lines> <filename> shows last given number lines of the file
  + head -<no of lines> <filename> shows first given number lines of the file
  + tail -f <filename> shows last ten lines and follows/waits for changes
  + head/tail <filename**s**> shows first/last ten lines of all the files
* **find** is used to search of files in dir hierarchy
  + find <directory> -name <filename> finds file by name
  + 
  + 
  + find <directory> -mtime -<days ago> finds file by made time
* **wc** It is word count command
  + wc <filename> for lines,words and characters count respectively
  + wc -c <filename> for characters count
  + wc -l <filename> for lines count
  + wc -w <filename> for words count
* **cal** Conventionally formatted calendar on Command line
  + cal shows calender with current data highlighted
  + ncal shows calender with week days style
  + cal <year> shows calender of that year
  + cal <months number> <year> shows given month
  + cal -<number> shows given number months around the current one
* **date** gives current date and time [till current sec]
  + date -s “<date> <time>” sets date and time
  + 
  + Flags



* **Multiple commands at once** 
  + <command>; <command>;..
  + <command> <arithmetical op> <command.
* **apt-get** to manage apt[advanced packaging tool] packages [needs sudo]
  + apt-get update syncs local and global files of all packages
    - apt-get update <package name> updates given package
  + nano /etc/apt/sources.list file with list of packages
  + <package name> --version shows version of java package
  + apt-get install <pkg name> install package with given name
  + apt-get remove <pkg name> removes package with given name
    - apt-get remove --purge <pkg name> removes all files
    - apt-get autoremove remove useless dependent packages
* **Ifconfig** interface configuration [view change network interface]
  + ifconfig shows internet configuration of the computer
  + ifconfig eth0 down/up enable[up] or disable[down] internet connection
* **tar** to extract tar files compressed in tar format
  + tar -c to create an archive file
  + tar -v to display the progress while creating
  + tar -f to be able to specify name of the file
  + tar -cvf <folder name> <output name> to create one
  + tar -xvf <name of tar file> <output name> to extract the file
  + tar -z to create or extracting tar.gz format archive [must be after ‘c’ flag for creating and after ‘x’ flag for extracting]
* **grep** global regular expression print
  + grep “<pattern/word>” <filenames> finds the pattern/words in a file
  + grep -i “<pattern/word>” <filenames> finds the pattern/words of lower and upper case [it is case insensitiveness]
  + grep -n “<pattern/word>” <filenames> prints lines numbers also
  + grep “<pattern/word>” \* searches in all the files
  + grep -v “<pattern/word>” <filenames> it greps everything that doesn't match
* **netstats** network statistics
  + netstats -a shows all connections available
  + netstats -a | less shows less connections available
  + netstats -t shows all TCP connections
  + netstats -u shows all UDP connections
  + netstats -l shows all ports listening
  + netstats -s shows all statistics
  + netstats -p shows PID of all the connections
  + netstats ie shows interface extended
* **xdg**
  + xdg-open <filenames> opens files with user preferred app