Terminal V/s File Manager & The CD command

The most frequent tasks that you perform on your PC is creating, moving or deleting Files. Let's look at various options for File Management.

To manage your files , you can either use

1. Terminal (Command Line Interface - CLI)
2. File manager (Graphical User Interface -GUI)

In the course, we will focus on the CLI , which brings us to out next question

Why learn Command Line Interface ?

Even though the world is moving to GUI based systems, CLI has its specific uses and is widely used in scripting and server administration. Let's look at it some compelling uses -

- Comparatively , Commands offer more options & are flexible .Piping and stdin/stdout are immensely powerful are not available in GUI
- Some configurations in GUI are up to 5 screens deep while in a CLI it's just a single command
- Moving, renaming 1000's of file in GUI will be time consuming (Using Control /Shift to select multiple files) ,while in CLI , using regular expressions so can do the same task in a single command.
- CLI load fast and do not consume RAM compared to GUI. In crunch scenarios this matters.

Both GUI and CLI have their specific uses. For example, in **GUI , performance monitoring graphs** give**instant visual feedback** on system health , while seeing hundreds of lines of logs in CLI is an eyesore.

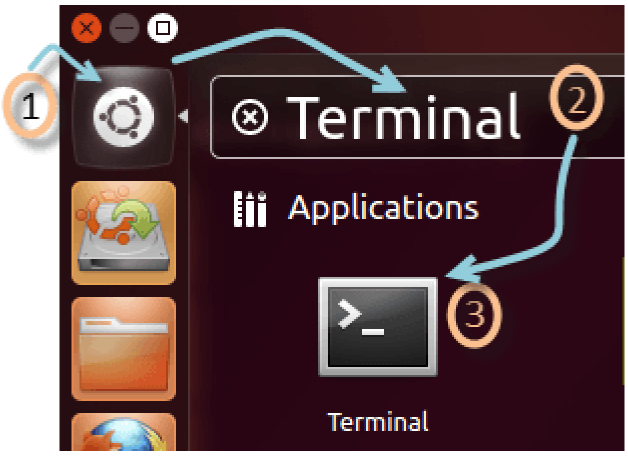
You must learn to use both GUI(File Manager) and CLI (Terminal)

GUI of a Linux based OS is similar to any other OS. Hence, we will focus on CLI and learn some useful commands.

Launching the CLI on Ubuntu

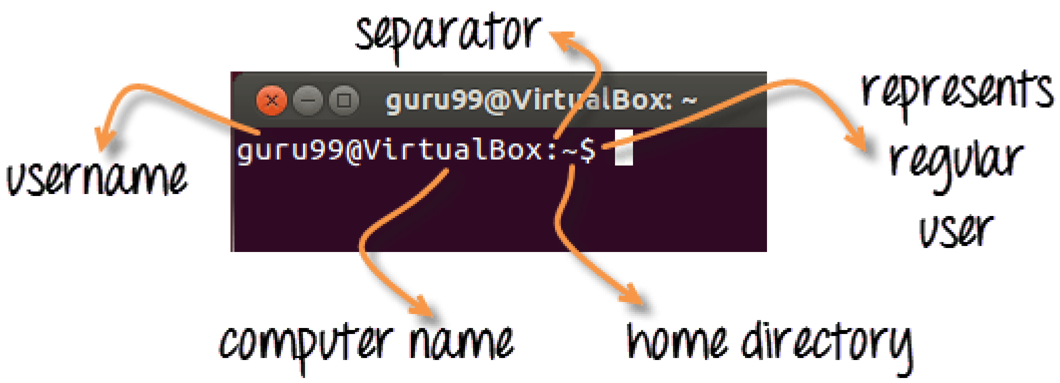
The are 2 ways to launch the terminal

1) Go to the Dash and type terminal



2) Our you can press **CTRL** + **Alt** + **T** to launch the Terminal

Once you launch the CLI (Terminal), you would find something as guru99@VirtualBox(see image) written on it.



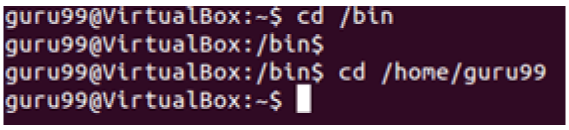
```
guru99@VirtualBox:~$
```

1) The first part of this line is the name of the **user** (bob, tom, ubuntu, home...)

2) The second part is the computer name or the host name. The hostname helps identify a computer over the network. In a server environment host-name becomes important.

3) The '**:**' is a simple separator

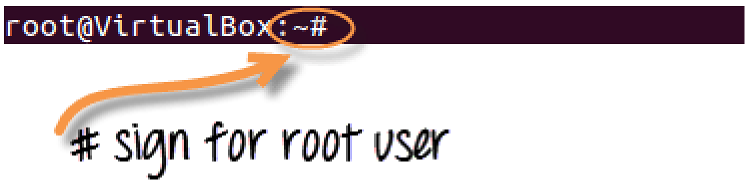
4) The tilde '~' sign shows that the user in working in the **home directory**. If you change the directory this sign will vanish



```
guru99@VirtualBox:~$ cd /bin
guru99@VirtualBox:/bin$
guru99@VirtualBox:/bin$ cd /home/guru99
guru99@VirtualBox:~$ █
```

In the above illustration we have moved from the /home directory to /bin using the **'cd'command**. The ~ sign does not display while working in /bin directory .It appears while moving back to the home directory.

5) The '$' sign suggests that you are working as a regular user in Linux. While working as a root user , '#' is displayed

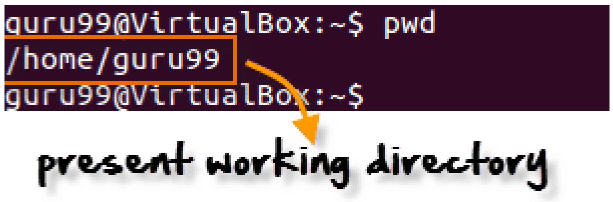


```
root@VirtualBox:~#
```

Present working Directory

The directory that you are currently browsing is called the Present working directory. You log on to the home directory when you boot your PC . If you want to determine the directory you are presently working on , use the command -

pwd



```
guru99@VirtualBox:~$ pwd
/home/guru99
guru99@VirtualBox:~$
```

pwd command stands for **p**rint **w**orking **d**irectory

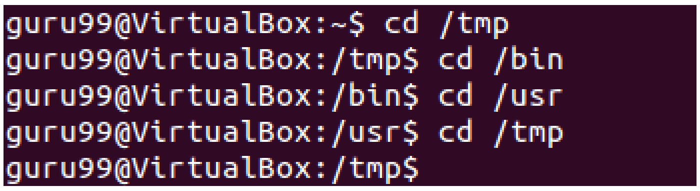
Above figure shows that /home/guru99 is the directory we are currently working on.

Changing Directories

If you want to change your current directory use the '**cd**' command .

cd

Consider the following example

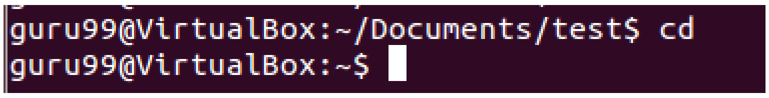


```
guru99@VirtualBox:~$ cd /tmp
guru99@VirtualBox:/tmp$ cd /bin
guru99@VirtualBox:/bin$ cd /usr
guru99@VirtualBox:/usr$ cd /tmp
guru99@VirtualBox:/tmp$
```

Here, we moved from directory /tmp to /bin to /usr and then back to /tmp.

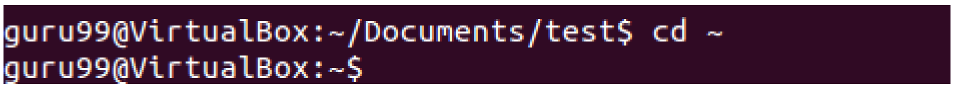
Navigating to home directory

If you want to navigate to the home directory then type **cd**.



```
guru99@VirtualBox:~/Documents/test$ cd
guru99@VirtualBox:~$
```

You can also use the **cd ~** command



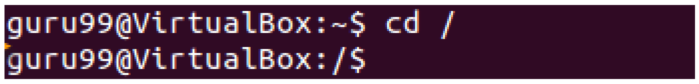
```
guru99@VirtualBox:~/Documents/test$ cd ~
guru99@VirtualBox:~$
```

Moving to root directory

The root of file system in Linux is denoted by '/'. Similar to 'c:\' in Windows.

Note: In Windows you use backward slash "\" while in UNIX/Linux , forward slash is use "/"

Type 'cd /' to move to the root directory.



```
guru99@VirtualBox:~$ cd /
guru99@VirtualBox:/$
```

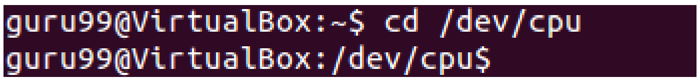
TIP: Do not forget space between **cd** and **/**. Otherwise you will get an error.

Navigating through multiple directories

You can navigate through multiple directories at the same time by specifying its complete path.

Example: If you want to move the /cpu directory under /dev, we do not need to break this operation in two parts.

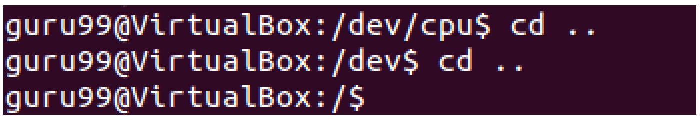
Instead, we can type '/dev/cpu' to reach the directory directly.



```
guru99@VirtualBox:~$ cd /dev/cpu
guru99@VirtualBox:/dev/cpu$
```

Moving up one directory level

For navigating up one directory level, try **'cd ..'**.



```
guru99@VirtualBox:/dev/cpu$ cd ..
guru99@VirtualBox:/dev$ cd ..
guru99@VirtualBox:/$
```

Here by using the 'cd ..' command, we have moved up one directory from '/dev/cpu' to '/dev'.

Then by again using the same command, we have jumped from '/dev' to '/' root directory.

Relative and Absolute Paths

A path in computing is the address of a file or folder.

Example -

C:\documentsandsettings\user\downloads in Windows or

/home/user/downloads in Linux

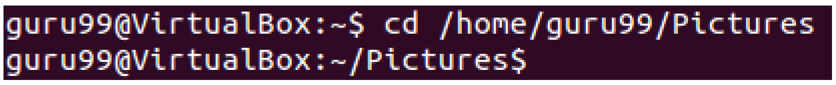
There are two kinds of paths:

Absolute Path:

Let's say you have to browse the images stored in the Pictures directory of the home folder 'guru99'.

The absolute file path of Pictures directory **/home/guru99/Pictures**

To navigate to this directory, you can use the command**cd /home/guru99/Pictures'**



```
guru99@VirtualBox:~$ cd /home/guru99/Pictures
guru99@VirtualBox:~/Pictures$
```

This is called absolute path as you are specifying the full path to reach the file.

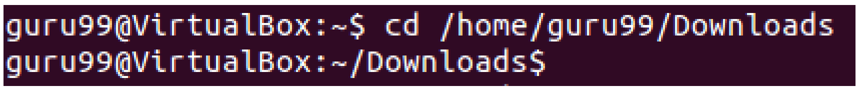
Relative Path:

Relative path comes in handy when you have to browse another subdirectory within a given directory.

It saves you from the effort to type complete paths all the time.

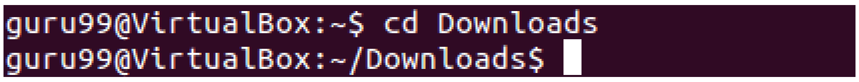
Suppose you are currently in your Home directory. You want to navigate to the Downloads directory.

You do no need to type the absolute path c**d /home/guru99/Downloads'.**



```
guru99@VirtualBox:~$ cd /home/guru99/Downloads
guru99@VirtualBox:~/Downloads$
```

Instead, you can simply type **'cd Downloads'** and you would navigate to the Downloads directory as you are already present within the **'/home/guru99'** directory.



```
guru99@VirtualBox:~$ cd Downloads
guru99@VirtualBox:~/Downloads$
```

This way you do not have to specify the complete path to reach a specific location within the same directory in the file system.

Summary:

- To manage your files, you can use either the GUI(File manager) or the CLI(Terminal) in Linux. Both have its relative advantages. In the tutorial series we will focus on the CLI aka the Terminal
- You can launch the terminal from the dashboard or using the shortcut key **Cntrl + Alt + T**
- The pwd command gives the present working directory.
- You can use the cd command to change directories
- Absolute path is complete address of a file or directory
- Relative path is relative location of a file of directory with respect to current directory
- Relative path help avoid typing complete paths all the time.

• Command	Description
cd or cd ~	Navigate to HOME directory

cd ..	Move one level up
cd	To change to a particular directory
cd /	Move to the root directory

Must Know Linux/Unix Commands

File Management becomes easy if you know the right commands.

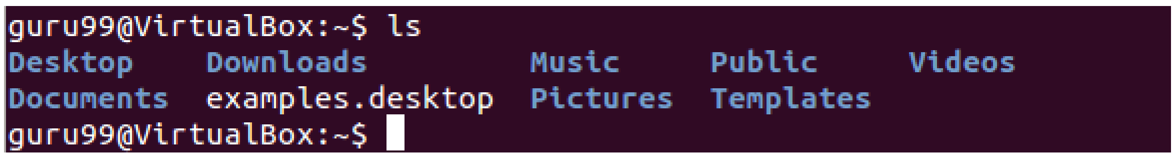
Sometimes, commands are also referred as "programs" since whenever you run a command, actually, it's the corresponding program code, written for the command, which is being executed.

Let's learn the must know Linux commands.

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.

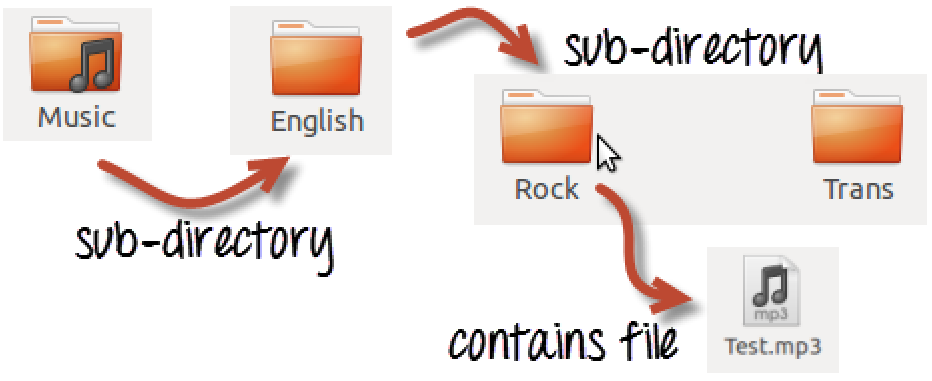


```
guru99@VirtualBox:~$ ls
Desktop    Downloads         Music     Public     Videos
Documents  examples.desktop  Pictures  Templates
guru99@VirtualBox:~$
```

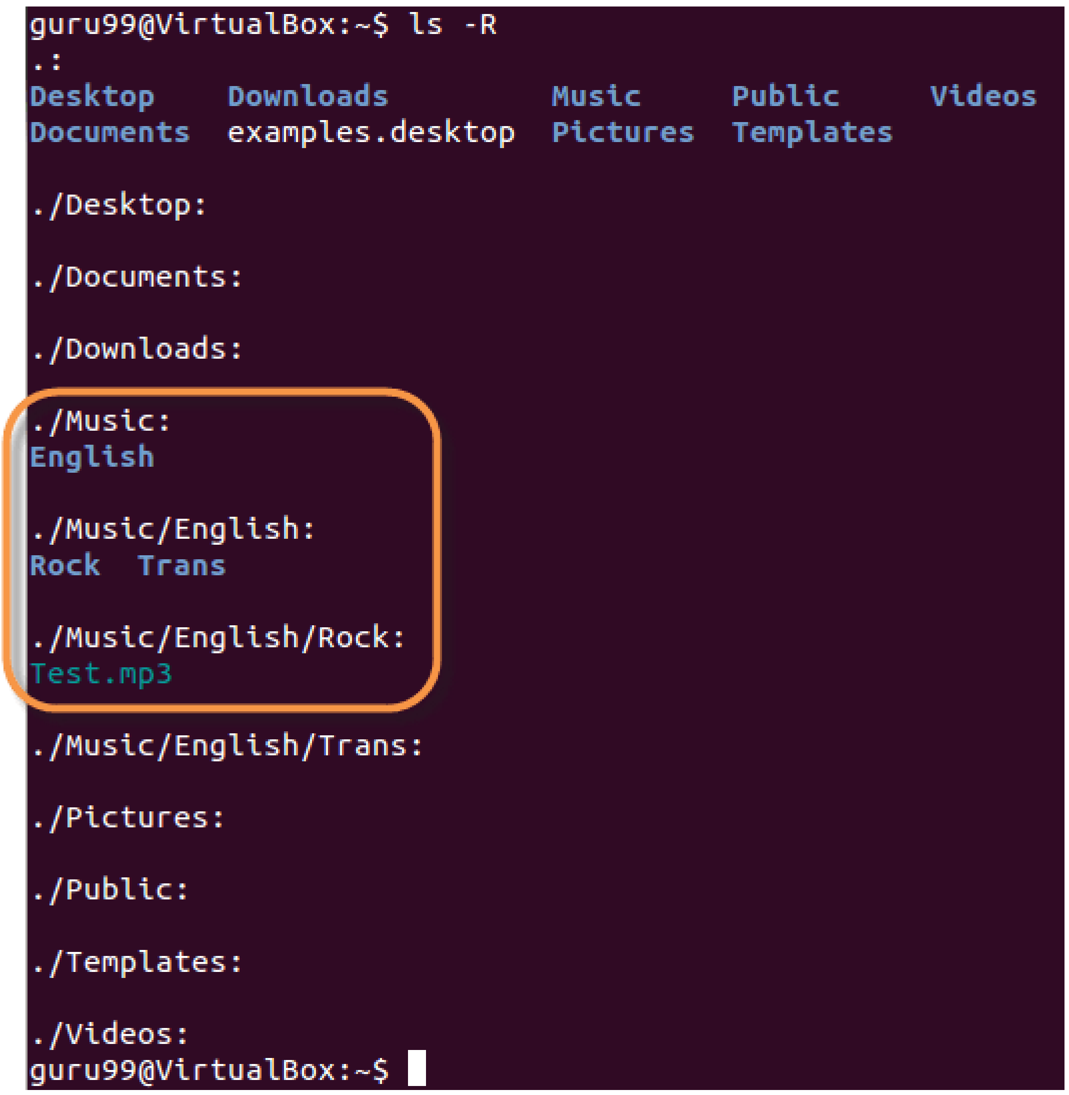
Note:

- Directories are denoted in blue color.
- Files are denoted in white.
- You will find similar color schemes in different flavors of Linux.

Suppose, your "Music" folder has following sub-directories and files.



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



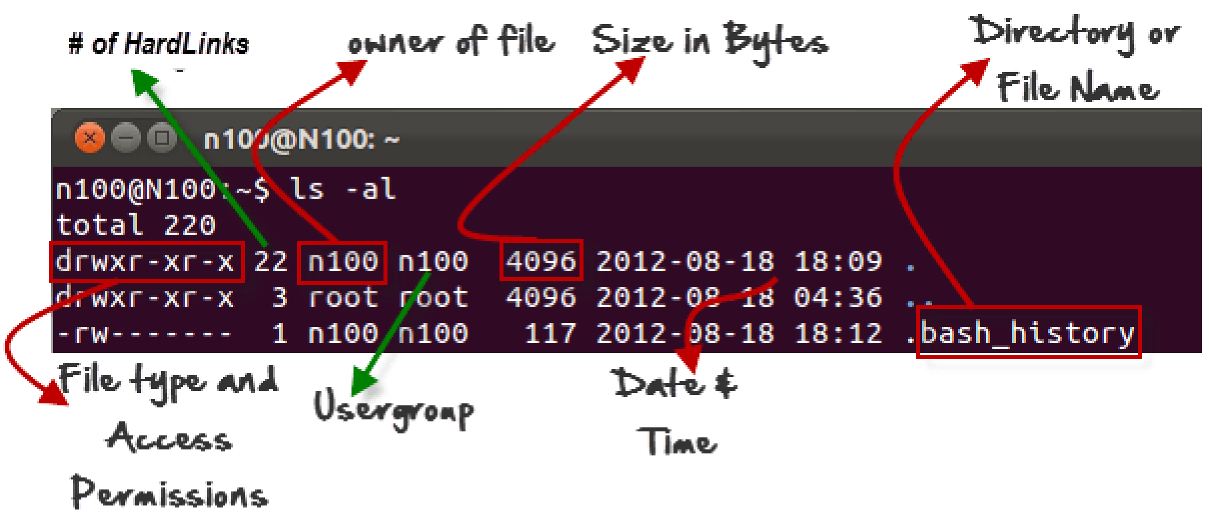
```
guru99@VirtualBox:~$ ls -R
.:
Desktop     Downloads          Music     Public     Videos
Documents   examples.desktop   Pictures  Templates

./Desktop:

./Documents:

./Downloads:

./Music:
English

./Music/English:
Rock   Trans

./Music/English/Rock:
Test.mp3

./Music/English/Trans:

./Pictures:

./Public:

./Templates:

./Videos:
guru99@VirtualBox:~$
```

NOTE: The command is case-sensitive. If you enter, "**ls - r**" you will get an error

'ls -al' gives detailed information of the files. The command provides information in a columnar format. The columns provide the following information:

1st Column	**File type and access permissions**
2nd Column	# of HardLinks to the File
3rd Column	Owner and the creator of the file
4th Column	Group of the owner
5th Column	File size in Bytes
6th Column	Date and Time
7th Column	Directory or File name

Let's see an example -



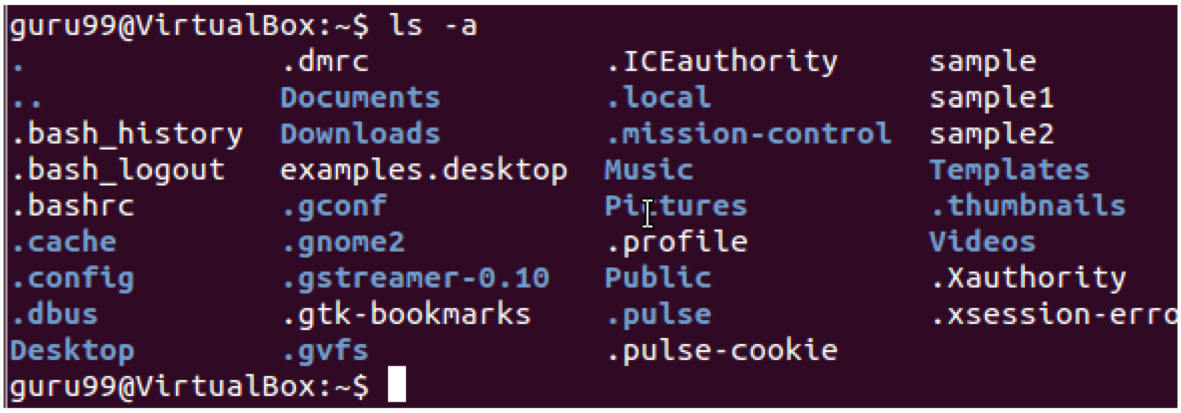
```
n100@N100:~$ ls -al
total 220
drwxr-xr-x 22 n100 n100   4096 2012-08-18 18:09 .
drwxr-xr-x  3 root root   4096 2012-08-18 04:36 ..
-rw-------  1 n100 n100    117 2012-08-18 18:12 .bash_history
```

Listing Hidden Files

Hidden items in UNIX/Linux begin with - . "period" symbol at the start, of the file or directory.

Any Directory/file starting with a '.' will not be seen unless you request for it. To view hidden files, use the command

ls - a



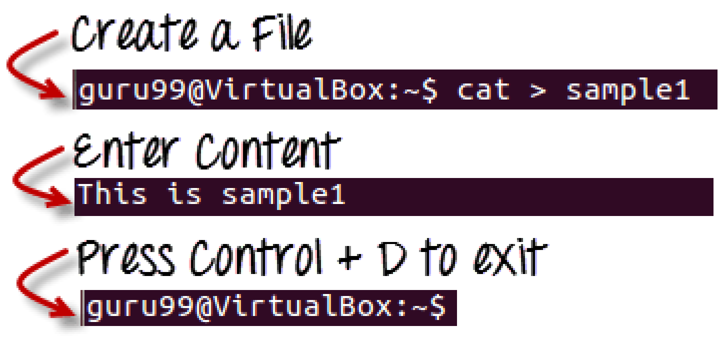
```
guru99@VirtualBox:~$ ls -a
.               .dmrc             .ICEauthority     sample
..              Documents         .local            sample1
.bash_history   Downloads         .mission-control  sample2
.bash_logout    examples.desktop  Music             Templates
.bashrc         .gconf            Pictures          .thumbnails
.cache          .gnome2           .profile          Videos
.config         .gstreamer-0.10   Public            .Xauthority
.dbus           .gtk-bookmarks    .pulse            .xsession-erro
Desktop         .gvfs             .pulse-cookie
guru99@VirtualBox:~$
```

Creating & Viewing Files

The 'cat' command is used to display text files. It can also be used for copying, combining and creating new text files. Let's see how it works

To create a new file, use the command

1. cat > filename
2. Add content
3. Press 'ctrl + d' to return to command prompt.

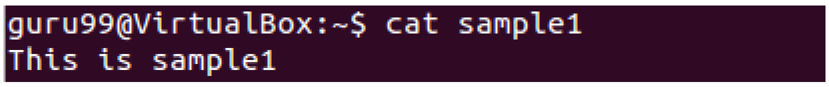


```
guru99@VirtualBox:~$ cat > sample1
This is sample1
guru99@VirtualBox:~$
```

To view a file, use the command -

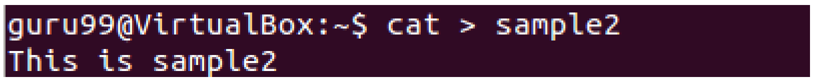
cat

Let's see the file we just created -



```
guru99@VirtualBox:~$ cat sample1
This is sample1
```

Let's see another file sample2

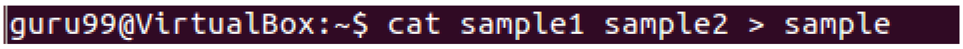


```
guru99@VirtualBox:~$ cat > sample2
This is sample2
```

The syntax to combine 2 files is -

cat file1 file2 > newfilename

Let's combine sample 1 and sample 2.

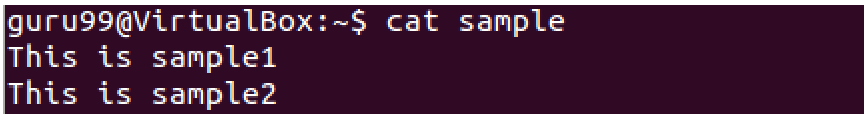


```
guru99@VirtualBox:~$ cat sample1 sample2 > sample
```

As soon as you insert this command and hit enter, the files are concatenated, but you do not see a result. This is because **Bash Shell (Terminal) is silent type**. It will never give you a confirmation message like "OK" or "Command Successfully Executed". It will only show a message when something goes wrong or when an error has occurred.

In order to view the new combo file "sample" use the command

cat sample



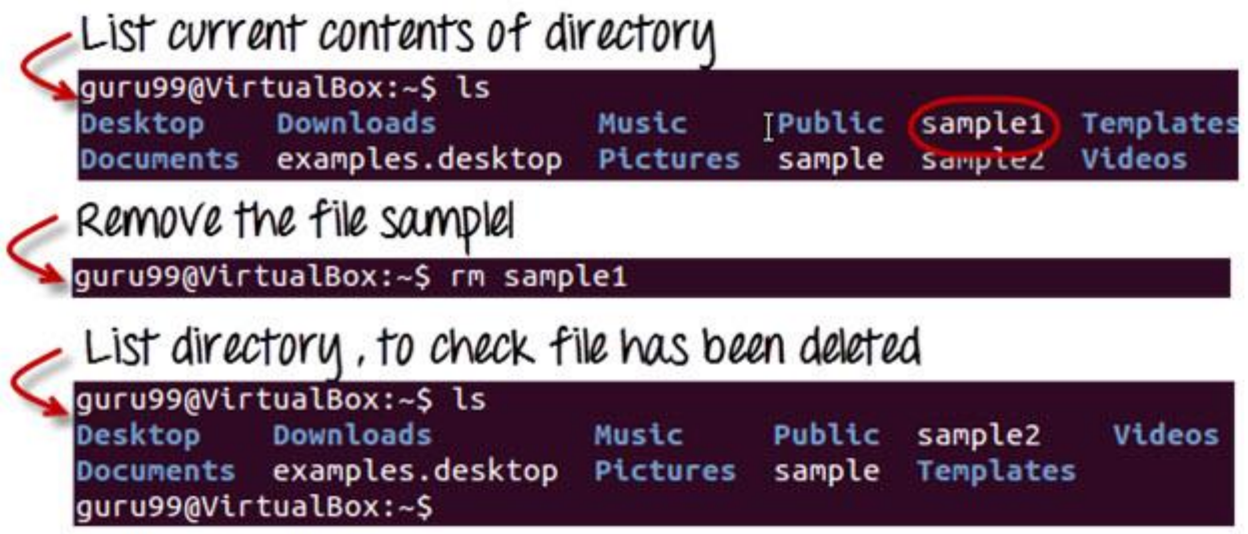
```
guru99@VirtualBox:~$ cat sample
This is sample1
This is sample2
```

Note: Only text files can be displayed and combined using this command.

Deleting Files

The 'rm' command removes files from the system without confirmation. To delete a file use syntax -

rm


List current contents of directory
Remove the file sample1
List directory , to check file has been deleted

```
guru99@VirtualBox:~$ ls
Desktop     Downloads         Music     Public  sample1  Templates
Documents   examples.desktop  Pictures  sample  sample2  Videos
```

```
guru99@VirtualBox:~$ rm sample1
```

```
guru99@VirtualBox:~$ ls
Desktop     Downloads         Music     Public  sample2   Videos
Documents   examples.desktop  Pictures  sample  Templates
guru99@VirtualBox:~$
```

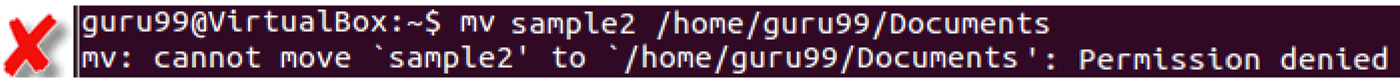
Moving and Re-naming files

In order to move a file, use the command

mv

Suppose we want to move the file "sample2" to location /home/guru99/Documents. Executing the command

mv sample2 /home/guru99/Documents



```
guru99@VirtualBox:~$ mv sample2 /home/guru99/Documents
mv: cannot move `sample2' to `/home/guru99/Documents': Permission denied
```

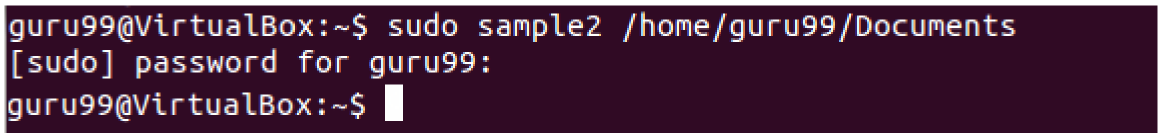
mv command needs super user permission. Currently, we are executing the command as a standard user. Hence we get the above error. To overcome the error use command

sudo

Sudo program allows regular users to run programs with the security privileges of the superuser or root.

Sudo command will ask for password authentication. Though, you do not need to know the root password. You can supply your own password. After authentication, the system will invoke the requested command.

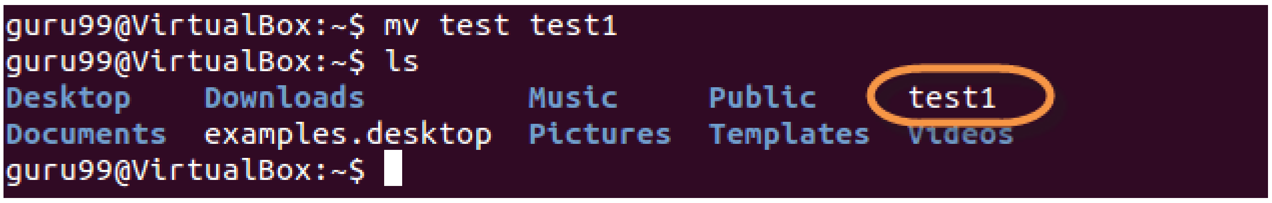
Sudo maintains a log of each command run. System administrators can trackback the person responsible for undesirable changes in the system



```
guru99@VirtualBox:~$ sudo sample2 /home/guru99/Documents
[sudo] password for guru99:
guru99@VirtualBox:~$
```

For renaming file:

mv filename newfilename



```
guru99@VirtualBox:~$ mv test test1
guru99@VirtualBox:~$ ls
Desktop     Downloads          Music     Public      test1
Documents   examples.desktop   Pictures  Templates   Videos
guru99@VirtualBox:~$
```

NOTE: By default, the password you entered for sudo is retained for 15 minutes per terminal. This eliminates the need of entering the password time and again.

You only need root/sudo privileges, only if the command involves files or directories not owned by the user or group running the comman

Directory Manipulations

Enough with File manipulations! Let's learn some directory commands

Creating Directories

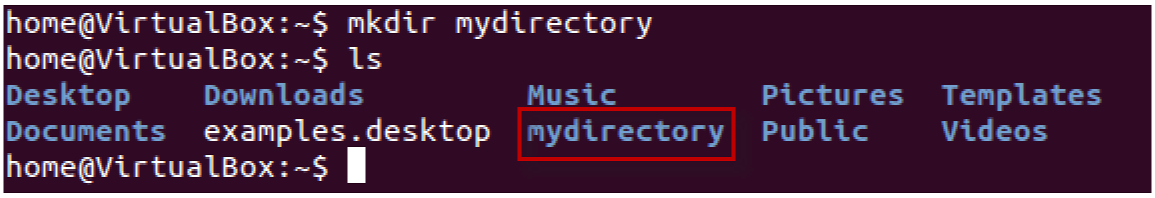
Directories can be created on a Linux operating system using the following command

mkdir

This command will create a subdirectory in your present working directory, which is usually your "Home Directory".

For example,

mkdir mydirectory



```
home@VirtualBox:~$ mkdir mydirectory
home@VirtualBox:~$ ls
Desktop     Downloads          Music         Pictures  Templates
Documents   examples.desktop   mydirectory   Public    Videos
home@VirtualBox:~$
```

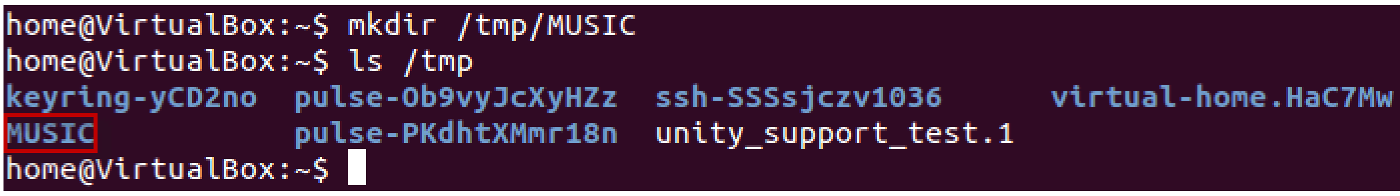
If you want to create a directory in a different location other than 'Home directory', you could use the following command -

mkdir

For example:

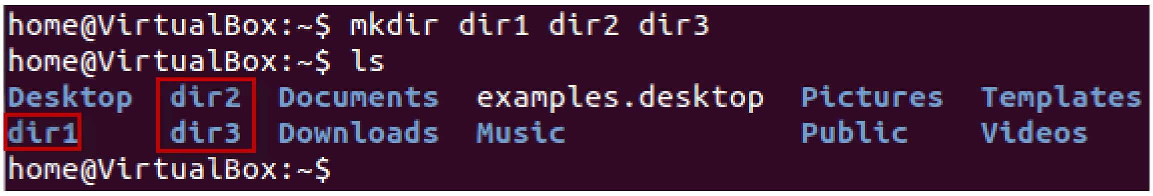
mkdir /tmp/MUSIC

will create a directory 'Music' under '/tmp' directory



```
home@VirtualBox:~$ mkdir /tmp/MUSIC
home@VirtualBox:~$ ls /tmp
keyring-yCD2no  pulse-Ob9vyJcXyHZz  ssh-SSSsjczv1036      virtual-home.HaC7Mw
MUSIC           pulse-PKdhtXMmr18n  unity_support_test.1
home@VirtualBox:~$
```

You can also create more than one directory at a time.



```
home@VirtualBox:~$ mkdir dir1 dir2 dir3
home@VirtualBox:~$ ls
Desktop  dir2  Documents  examples.desktop  Pictures  Templates
dir1     dir3  Downloads  Music             Public    Videos
home@VirtualBox:~$
```

Removing Directories

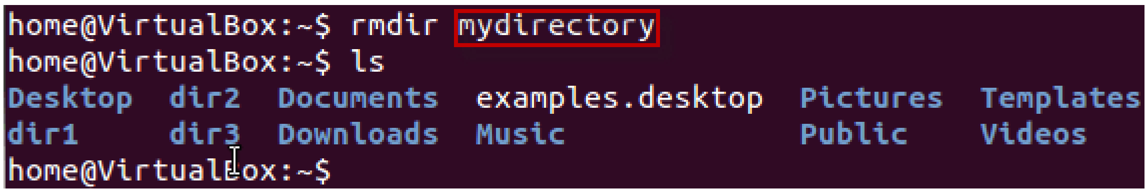
In order to remove a directory, use the command -

rmdir

Example

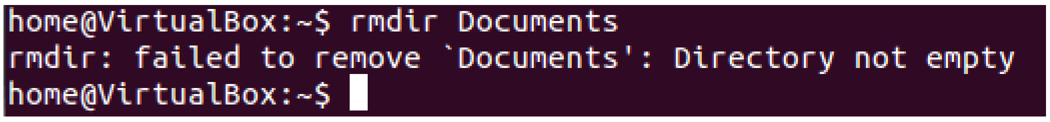
rmdir mydirectory

will delete the directory mydirectory



```
home@VirtualBox:~$ rmdir mydirectory
home@VirtualBox:~$ ls
Desktop  dir2  Documents  examples.desktop  Pictures  Templates
dir1     dir3  Downloads  Music             Public    Videos
home@VirtualBox:~$
```

Tip: Ensure that there is no file / sub-directory under the directory that you want to delete. Delete the files/sub-directory first before deleting the parent directory.



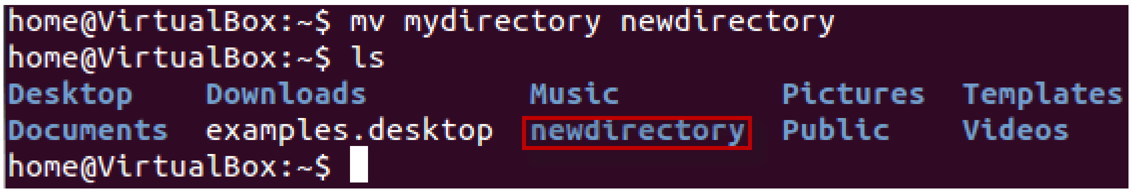
```
home@VirtualBox:~$ rmdir Documents
rmdir: failed to remove `Documents': Directory not empty
home@VirtualBox:~$
```

Renaming Directory

The 'mv' (move) command (covered earlier) can also be used for renaming directories. Use the below given format:

mv directoryname newdirectoryname

Let us try it:



```
home@VirtualBox:~$ mv mydirectory newdirectory
home@VirtualBox:~$ ls
Desktop    Downloads         Music         Pictures  Templates
Documents  examples.desktop  newdirectory  Public    Videos
home@VirtualBox:~$
```

Other Important Commands

The 'Man' command

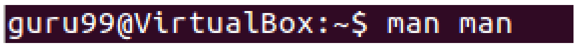
Man stands for manual which is a reference book of a Linux operating system. It is similar to HELP file found in popular softwares.

To get help on any command that you do not understand, you can type

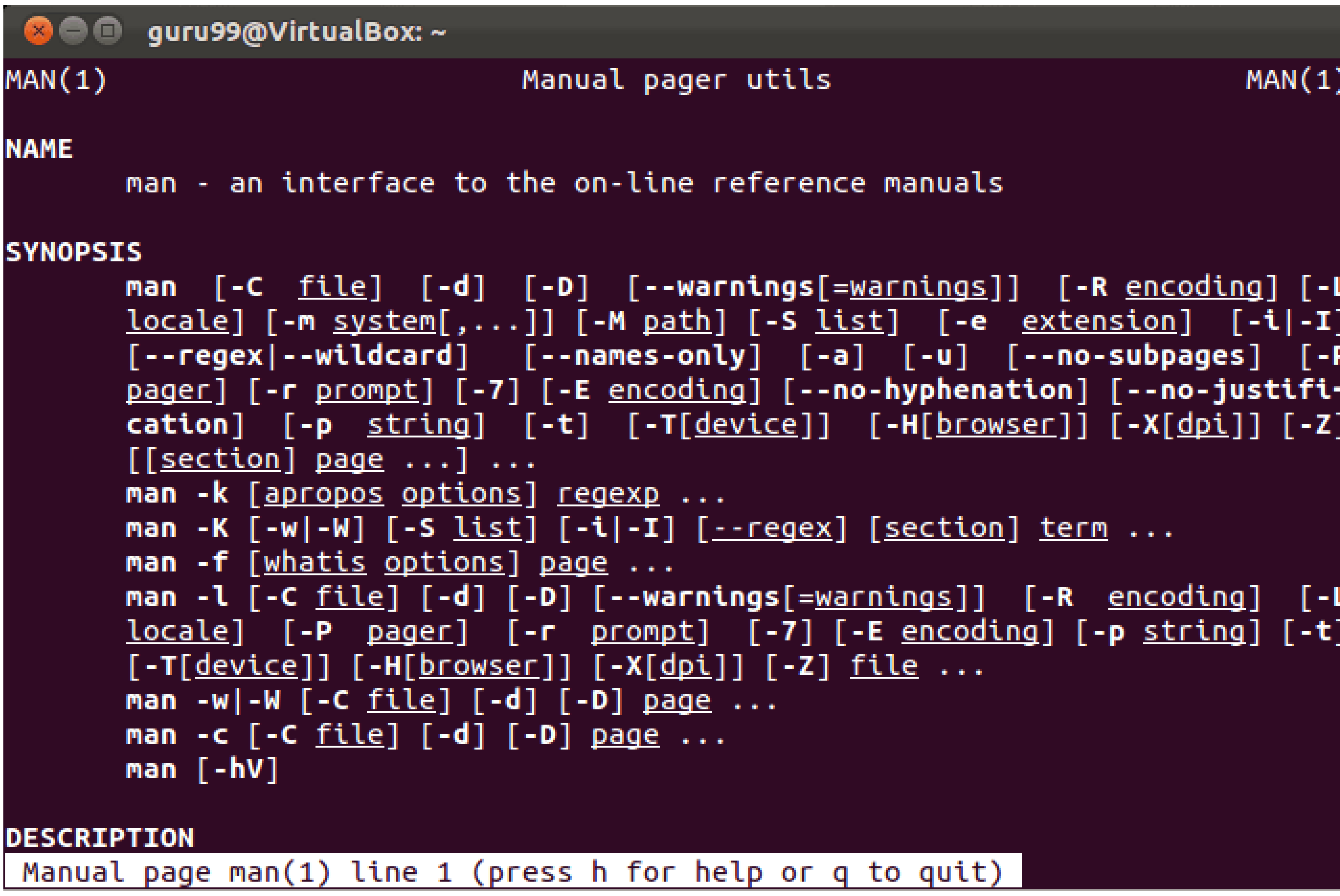
man

The terminal would open the manual page for that command.

For an example, if we type *man man* and hit enter; terminal would give us information on man command



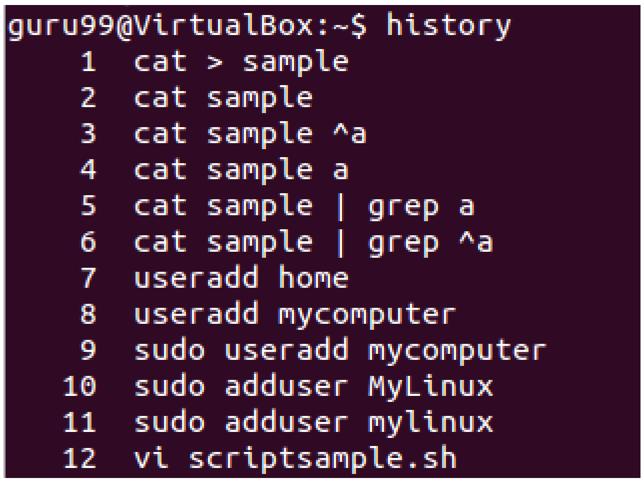
```
guru99@VirtualBox:~$ man man
```



```
MAN(1)                        Manual pager utils                        MAN(1

NAME
       man - an interface to the on-line reference manuals

SYNOPSIS
       man  [-C  file]  [-d]  [-D]  [--warnings[=warnings]]  [-R encoding] [-L
       locale] [-m system[,...]] [-M path] [-S list]  [-e  extension]  [-i|-I]
       [--regex|--wildcard]  [--names-only]  [-a]  [-u]  [--no-subpages]  [-P
       pager] [-r prompt] [-7] [-E encoding] [--no-hyphenation] [--no-justifi-
       cation]  [-p  string]  [-t]  [-T[device]]  [-H[browser]] [-X[dpi]] [-Z]
       [[section] page ...] ...
       man -k [apropos options] regexp ...
       man -K [-w|-W] [-S list] [-i|-I] [--regex] [section] term ...
       man -f [whatis options] page ...
       man -l [-C file] [-d] [-D] [--warnings[=warnings]]  [-R  encoding]  [-L
       locale]  [-P  pager]  [-r  prompt]  [-7] [-E encoding] [-p string] [-t]
       [-T[device]] [-H[browser]] [-X[dpi]] [-Z] file ...
       man -w|-W [-C file] [-d] [-D] page ...
       man -c [-C file] [-d] [-D] page ...
       man [-hV]

DESCRIPTION
 Manual page man(1) line 1 (press h for help or q to quit)
```

The History Command

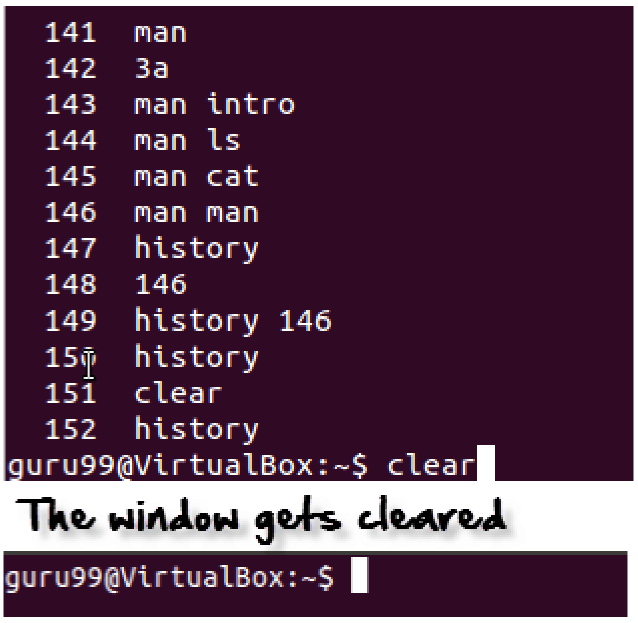
History command shows all the commands that you have used in the past for the current terminal session. This can help you refer to the old commands you have entered and re-use them in your operations again.



```
guru99@VirtualBox:~$ history
    1  cat > sample
    2  cat sample
    3  cat sample ^a
    4  cat sample a
    5  cat sample | grep a
    6  cat sample | grep ^a
    7  useradd home
    8  useradd mycomputer
    9  sudo useradd mycomputer
   10  sudo adduser MyLinux
   11  sudo adduser mylinux
   12  vi scriptsample.sh
```

The clear command

This command clears all the clutter on the terminal and gives you a clean window to work on, just like when you launch the terminal.



```
  141  man
  142  3a
  143  man intro
  144  man ls
  145  man cat
  146  man man
  147  history
  148  146
  149  history 146
  150  history
  151  clear
  152  history
guru99@VirtualBox:~$ clear
```

```
guru99@VirtualBox:~$
```

Pasting commands into the terminal

Many a times you would have to type in long commands on the Terminal. Well, it can be annoying at times, and if you want to avoid such a situation then copy, pasting the commands can come to rescue.

For copying, the text from a source, you would use **Ctrl + c,** but for pasting it on the Terminal you need to use**Ctrl + Shift + v**. You can also try **Shift + Insert or select Edit>Paste on the menu**

Summary

Below is a summary of commands we have learned in this tutorial

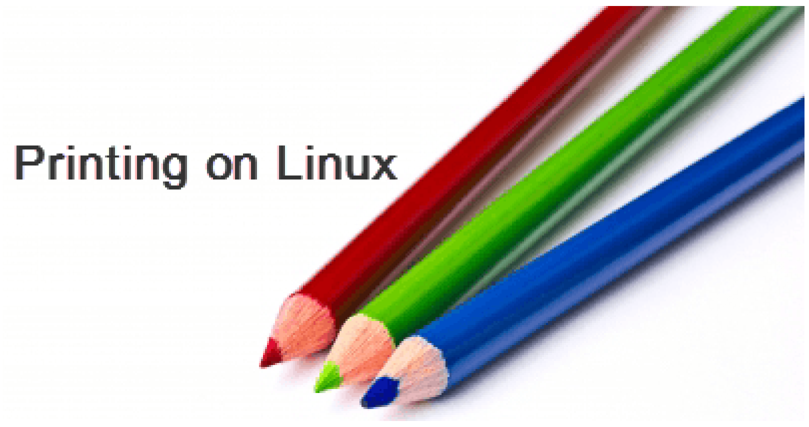
Command	Description
Ls	Lists all files and directories in the present working directory
ls – R	Lists files in sub-directories as well
ls – a	Lists hidden files as well
ls – al	Lists files and directories with detailed information like permissions , size , owner etc.
cat > filename	Creates a new file
cat filename	Displays the file content
cat file file2 > file3	Joins two files (file1, file2) and stores the output in a new file (file3)
mv file "new file path"	Moves the files to the new location

mv filename new_file_name	Renames the file to a new filename
Sudo	Allows regular users to run programs with the security privileges of the superuser or root
Rm	Deletes a file
Man	Gives help information on a command
History	Gives a list of all past commands typed in the current terminal session
Clear	Clears the terminal
Mkdir	Creates a new directory in the present working directory
Mkdir	Create a new directory at the specified path
rmdir	Deletes a directory
Mv	Renames a directory

Unix/Linux - Print , Email , Install New Software

Lets try out some easy commands which **can print files** in a format you want. What more, your original file does not get affected at all by the formatting that you do. Let us learn about these commands and their use.

'pr' command



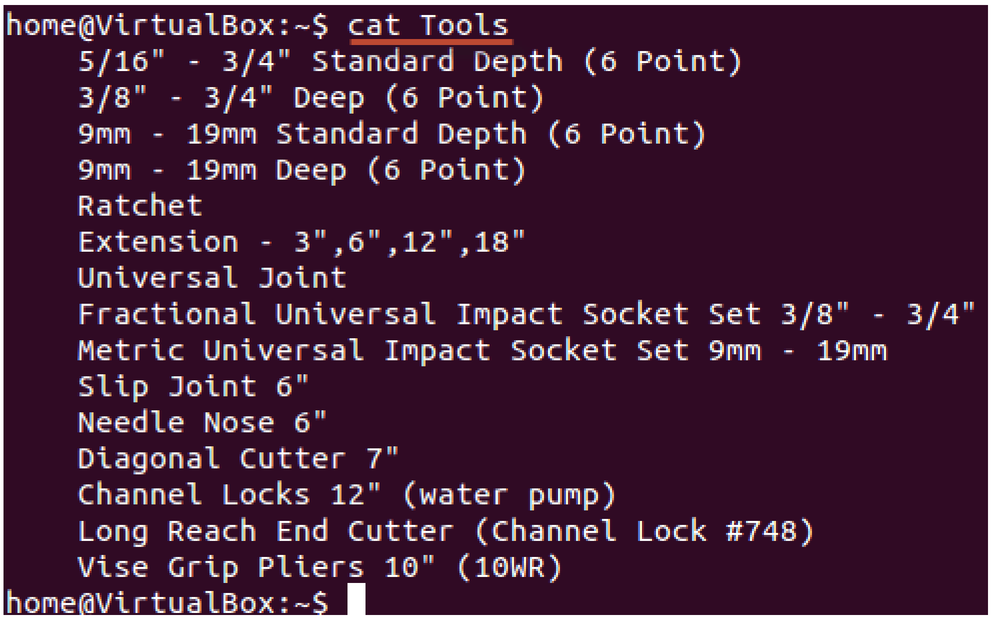
This command helps in formatting the file for printing on the terminal. There are many options available with this command which help in making desired format changes on the file. The most used '**pr**' options are listed below.

Option	**Function**
-x	Divides the data into 'x' columns
-h "header"	Assigns "header" value as the report header
-t	Does not print the header and top/bottom margins
-d	Double spaces the output file
-n	Denotes all line with numbers
-l page length	Defines the lines (page length) in a page. Default is 56
-o margin	Formats the page in accordance with the margin number

Let us try some of the options and study their effects.

Dividing data into columns

'**Tools**' is a file (shown below) .

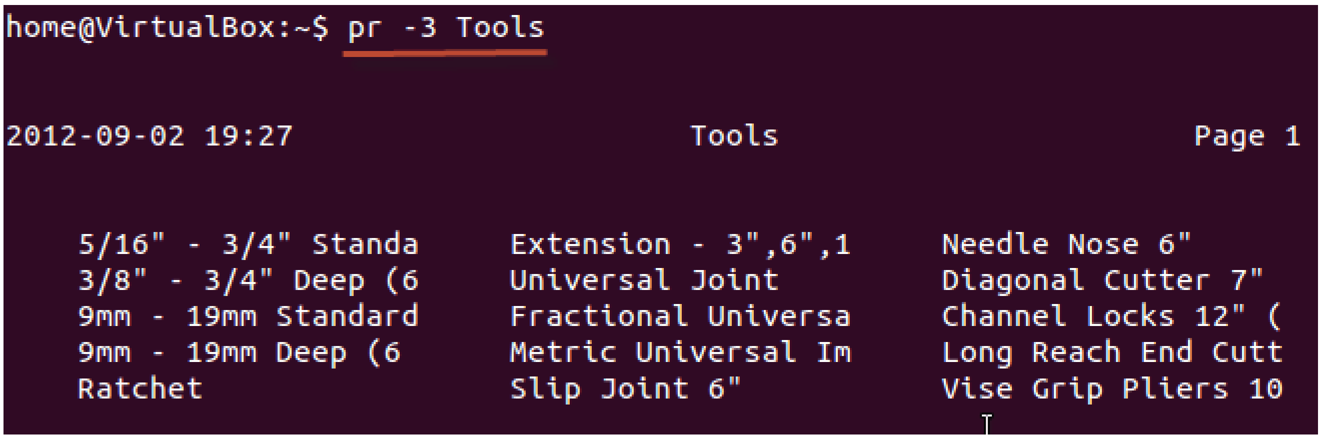


```
home@VirtualBox:~$ cat Tools
    5/16" - 3/4" Standard Depth (6 Point)
    3/8" - 3/4" Deep (6 Point)
    9mm - 19mm Standard Depth (6 Point)
    9mm - 19mm Deep (6 Point)
    Ratchet
    Extension - 3",6",12",18"
    Universal Joint
    Fractional Universal Impact Socket Set 3/8" - 3/4"
    Metric Universal Impact Socket Set 9mm - 19mm
    Slip Joint 6"
    Needle Nose 6"
    Diagonal Cutter 7"
    Channel Locks 12" (water pump)
    Long Reach End Cutter (Channel Lock #748)
    Vise Grip Pliers 10" (10WR)
home@VirtualBox:~$
```

We want its content to be arranged in three columns. The syntax for the same would be:

pr -x Filename

The '-x' option with the 'pr' command divides the data into x columns.



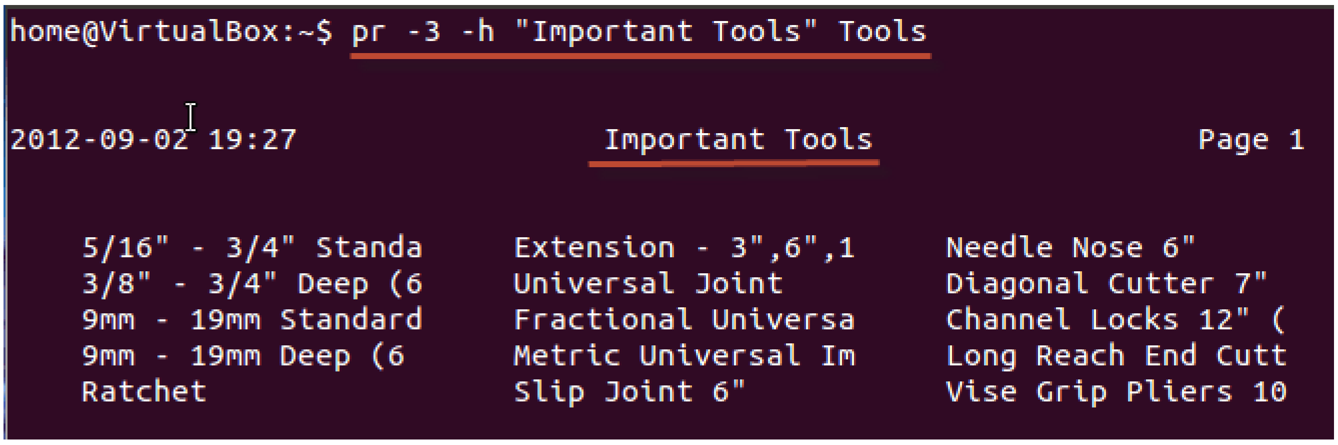
```
home@VirtualBox:~$ pr -3 Tools


2012-09-02 19:27                     Tools                      Page 1


    5/16" - 3/4" Standa     Extension - 3",6",1     Needle Nose 6"
    3/8" - 3/4" Deep (6     Universal Joint         Diagonal Cutter 7"
    9mm - 19mm Standard     Fractional Universa     Channel Locks 12" (
    9mm - 19mm Deep (6      Metric Universal Im     Long Reach End Cutt
    Ratchet                 Slip Joint 6"           Vise Grip Pliers 10
```

<u>Assigning a header</u>

The syntax is:

pr -h "Header" Filename

The '-h' options assigns "header" value as the report header



```
home@VirtualBox:~$ pr -3 -h "Important Tools" Tools


2012-09-02 19:27                Important Tools                Page 1


    5/16" - 3/4" Standa     Extension - 3",6",1     Needle Nose 6"
    3/8" - 3/4" Deep (6     Universal Joint         Diagonal Cutter 7"
    9mm - 19mm Standard     Fractional Universa     Channel Locks 12" (
    9mm - 19mm Deep (6      Metric Universal Im     Long Reach End Cutt
    Ratchet                 Slip Joint 6"           Vise Grip Pliers 10
```

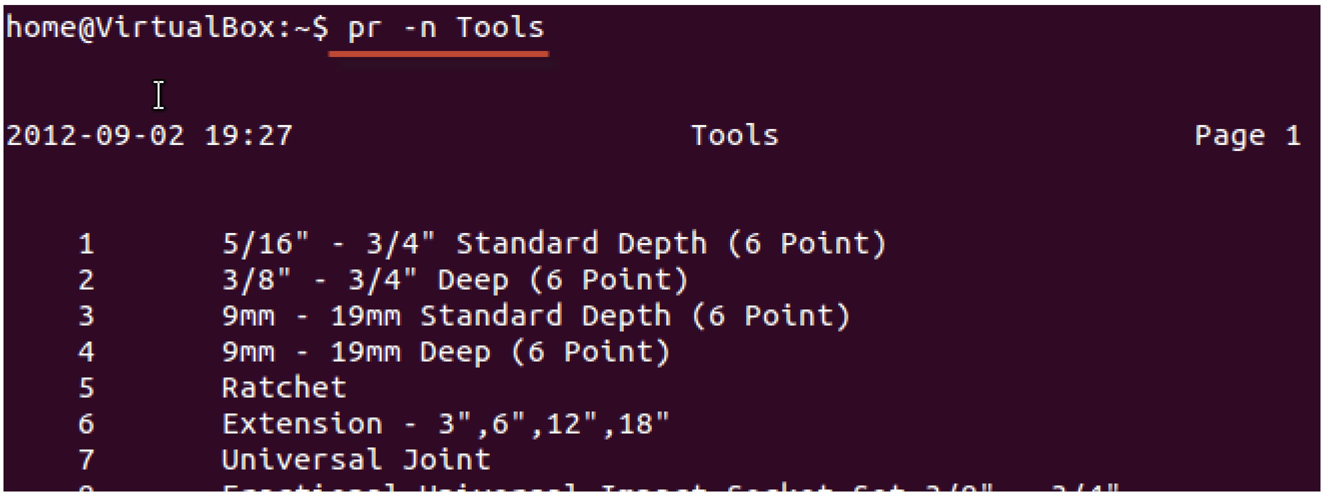
As shown above, we have arranged the file in 3 columns and assigned a header

Denoting all lines with numbers

The syntax is:

pr -n Filename

This command denotes all the lines in the file with numbers



```
home@VirtualBox:~$ pr -n Tools


2012-09-02 19:27                      Tools                      Page 1


    1       5/16" - 3/4" Standard Depth (6 Point)
    2       3/8" - 3/4" Deep (6 Point)
    3       9mm - 19mm Standard Depth (6 Point)
    4       9mm - 19mm Deep (6 Point)
    5       Ratchet
    6       Extension - 3",6",12",18"
    7       Universal Joint
```

These are some of the 'pr' command options that you can use to modify the file format.

Printing a file

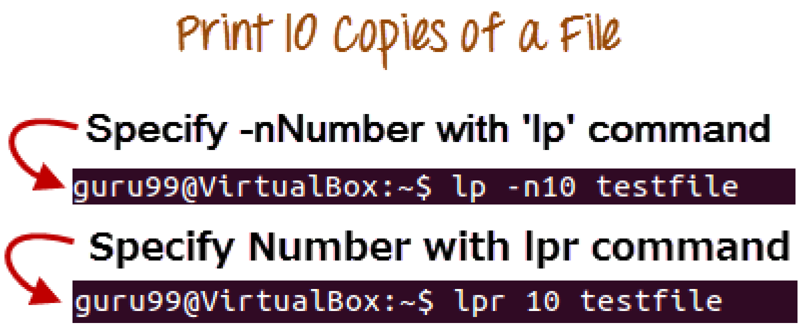
Once you are **done with the formatting** and it is time for you to get a **hard copy** of the file, you need to use the following command:

lp Filename

or

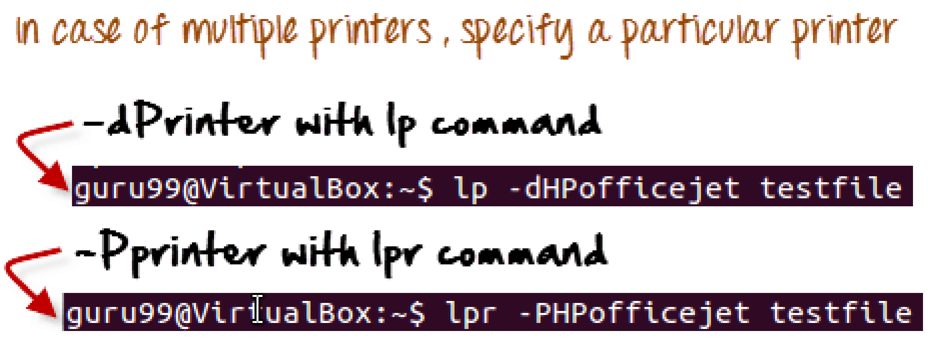
lpr Filename

In case you want to print multiple copies of the file, you can use the number modifier.


Print 10 Copies of a File
Specify -nNumber with 'lp' command
Specify Number with lpr command

```
guru99@VirtualBox:~$ lp -n10 testfile
guru99@VirtualBox:~$ lpr 10 testfile
```

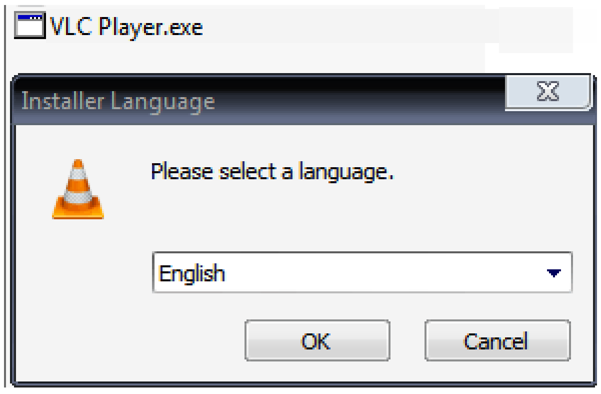
In case you have multiple printers configured, you can specify a particular printer using the Printer modifier


In case of multiple printers, specify a particular printer
-dPrinter with lp command
-Pprinter with lpr command

```
guru99@VirtualBox:~$ lp -dHPofficejet testfile
guru99@VirtualBox:~$ lpr -PHPofficejet testfile
```

Installing Software

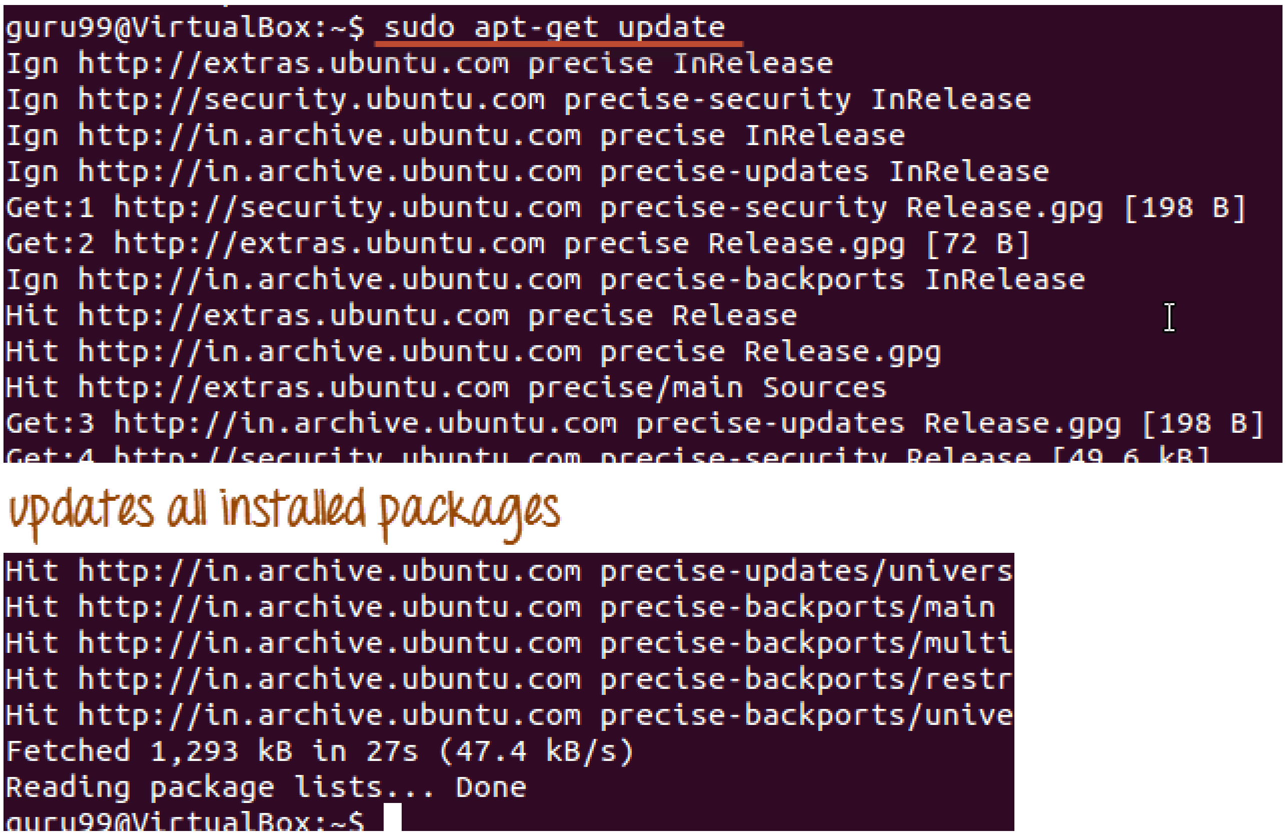
In windows the installation of a program is done by running the setup.exe file. The installation bundle contains the program as well various dependent components required to run the program correctly.



In Linux/UNIX , installation files are distributed as packages. But the package contains only the program itself. Any dependent components will have to be installed separately which are usually available as packages themselves.

You can use the **apt** commands to install or remove a package. Let's updated all the installed packages in our system using command -

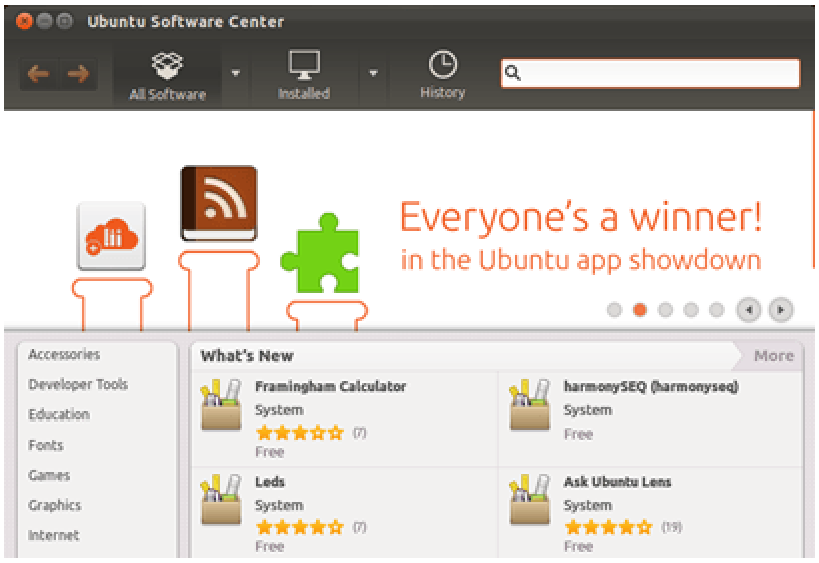
sudo apt-get update



```
guru99@VirtualBox:~$ sudo apt-get update
Ign http://extras.ubuntu.com precise InRelease
Ign http://security.ubuntu.com precise-security InRelease
Ign http://in.archive.ubuntu.com precise InRelease
Ign http://in.archive.ubuntu.com precise-updates InRelease
Get:1 http://security.ubuntu.com precise-security Release.gpg [198 B]
Get:2 http://extras.ubuntu.com precise Release.gpg [72 B]
Ign http://in.archive.ubuntu.com precise-backports InRelease
Hit http://extras.ubuntu.com precise Release
Hit http://in.archive.ubuntu.com precise Release.gpg
Hit http://extras.ubuntu.com precise/main Sources
Get:3 http://in.archive.ubuntu.com precise-updates Release.gpg [198 B]
Get:4 http://security.ubuntu.com precise-security Release [49.6 kB]
```

```
Hit http://in.archive.ubuntu.com precise-updates/univers
Hit http://in.archive.ubuntu.com precise-backports/main
Hit http://in.archive.ubuntu.com precise-backports/multi
Hit http://in.archive.ubuntu.com precise-backports/restr
Hit http://in.archive.ubuntu.com precise-backports/unive
Fetched 1,293 kB in 27s (47.4 kB/s)
Reading package lists... Done
guru99@VirtualBox:~$
```

The easy and popular way to install programs on Ubuntu is by using the Software center as most of the software packages are available on it and it is far more secure than the files downloaded from the internet.



Sending E-mails

For sending mails through terminal you will need to install packages 'mailutils'.

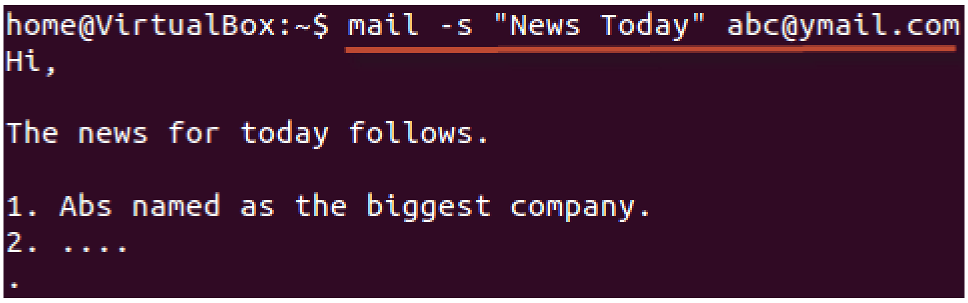
The command syntax is -

sudo apt-get install

Once done, you can then use the following syntax for sending an email.

mail -s 'subject' -c 'cc-address' -b 'bcc-address' 'to-address'

This will look like:



```
home@VirtualBox:~$ mail -s "News Today" abc@ymail.com
Hi,

The news for today follows.

1. Abs named as the biggest company.
2. ....
.
```

Press Cntrl+D you are finished writing the mail. The mail will be sent to the mentioned address.

Summary

- You can format and print a file directly from the terminal. The formatting you do on the files does not affect the file contents
- In Unix/Linux, software is installed in form of packages. A package contains the program itself. Any dependent component needs to be downloaded separately.
- You can also send e-mails from terminal using the **'mail' command**

Command	Description
pr –x	Divides the file into x columns
pr –h	Assigns a header to the file
pr –n	Denotes the file with Line Numbers
lp -nc	Prints "c" copies of the File

lpr c	
lp –d lp -P	Specifies name of the printer
apt-get	Command used to install and update packages
mail -s 'subject' -c 'cc-address' -b 'bcc-address' 'to-address'	Command to send email
Mail -s "Subject" to-address < Filename	Command to send email with attachment

Redirection in Linux/Unix - Demystified!

Most of the commands , we have learned so far, take an input and give an output.

- The standard input (stdin) device is the keyboard.
- The standard output (stdout) device is the screen.

Linux , is a very flexible system and you can change the standard input / output devices. Let's learn how this re-direction works

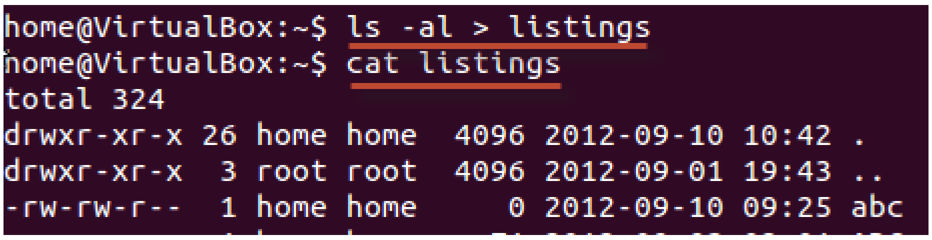
Output Redirection

The '>' symbol is used for output (STDOUT) redirection.

Example:

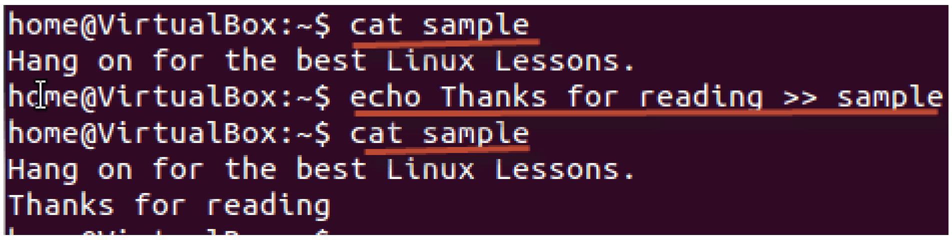
ls -al** > **listings

Here the output of command ls -al is re-directed to file "listings" instead of your screen.


```
home@VirtualBox:~$ ls -al > listings
home@VirtualBox:~$ cat listings
total 324
drwxr-xr-x 26 home home  4096 2012-09-10 10:42 .
drwxr-xr-x  3 root root  4096 2012-09-01 19:43 ..
-rw-rw-r--  1 home home     0 2012-09-10 09:25 abc
```

Note: Use the right file name while redirection. If there is an existing file with the same name, it will be overwritten.

If you do not want a file to be overwritten but want to add more content to an existing file then you should use'>>' operator.


```
home@VirtualBox:~$ cat sample
Hang on for the best Linux Lessons.
home@VirtualBox:~$ echo Thanks for reading >> sample
home@VirtualBox:~$ cat sample
Hang on for the best Linux Lessons.
Thanks for reading
```

You can re-direct standard output, to not just files, but also devices!

$ cat music.mp3 > /dev/audio

The cat command reads the file music.mp3 and sends the output to /dev/audio which is the audio device. If the sound configurations in your PC are correct, this command will play the file music.mp3

Input redirection

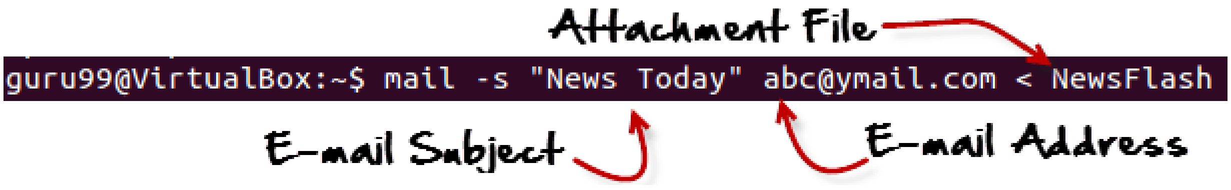
< Input Redirection

The '<' symbol is used for input(STDIN) redirection

Example: The mail program in Linux can help you send e-mails from the Terminal.

You can type the contents of the email using the standard device keyboard. But if you want to attach a File to email you can use the input re-direction operator in following format -

Mail -s "Subject" to-address < Filename



```
guru99@VirtualBox:~$ mail -s "News Today" abc@ymail.com < NewsFlash
```

This would attach the file with the mail and it would be sent to the recipient.

The above examples were simple. Let's look at some advance re-direction techniques which makes use of File Descriptors

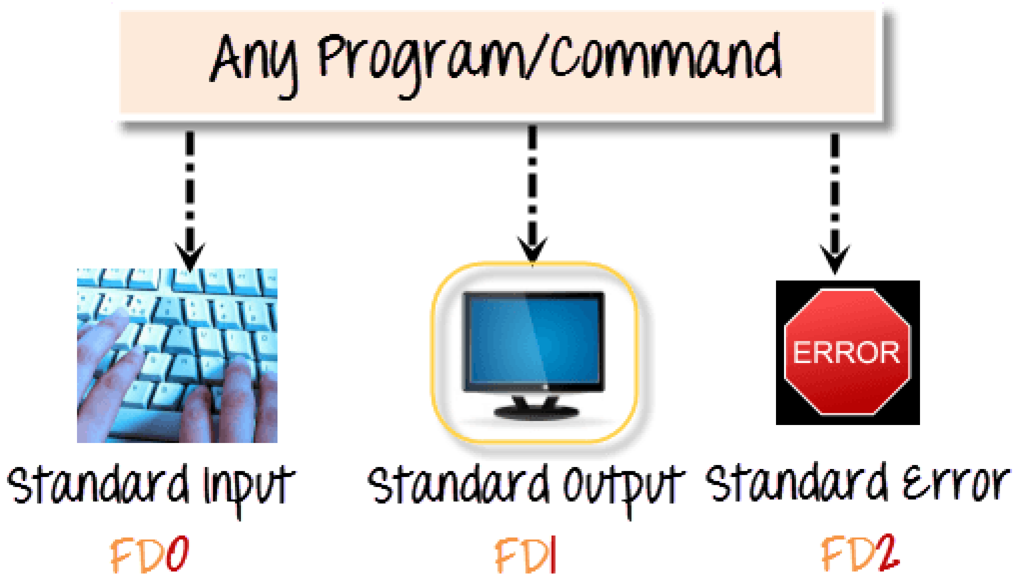
File Descriptors

In Linux/Unix everything is a file. Regular file, Directories and even Devices are files. Every File has an associated number called File Descriptor (FD).

Your screen also has a File Descriptor. When a program is executed the output is sent to File Descriptor of the screen and you see program output on your monitor. If the output is sent to File Descriptor of the printer, the program output would have been printed.

Error Redirection

Whenever you execute a program/command at the terminal, 3 files are always open, viz., standard input, standard output, standard error**.**



These files are always present whenever a program is run. As explained before a file descriptor, is associated with each of these files

File	File Descriptor
Standard Input STDIN	**FD0**
Standard Output STDOUT	**FD1**
Standard Error STDERR	**FD2**

By default, error stream is displayed on the screen. Error redirection is routing the errors to a file other than the screen.

Why Error Redirection?

Error re-direction is one of the very popular features of Unix/Linux.

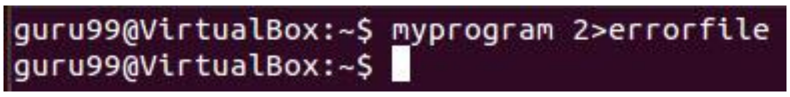
Frequent UNIX users will reckon that many commands give you massive amounts of errors.

- For instance, while searching for files, one typically gets permission denied errors. These errors usually do not help the person searching for a particular file.
- While executing shell scripts, you often do NOT want error messages cluttering up the normal program output.

The solution is to re-direct the error messages to a file.

Example 1

$ myprogram 2>errorsfile



```
guru99@VirtualBox:~$ myprogram 2>errorfile
guru99@VirtualBox:~$ 
```

Above we are executing a program names myprogram.

The file descriptor for standard error is 2.

Using "2>" we re-direct the error output to a file named "errorfile"

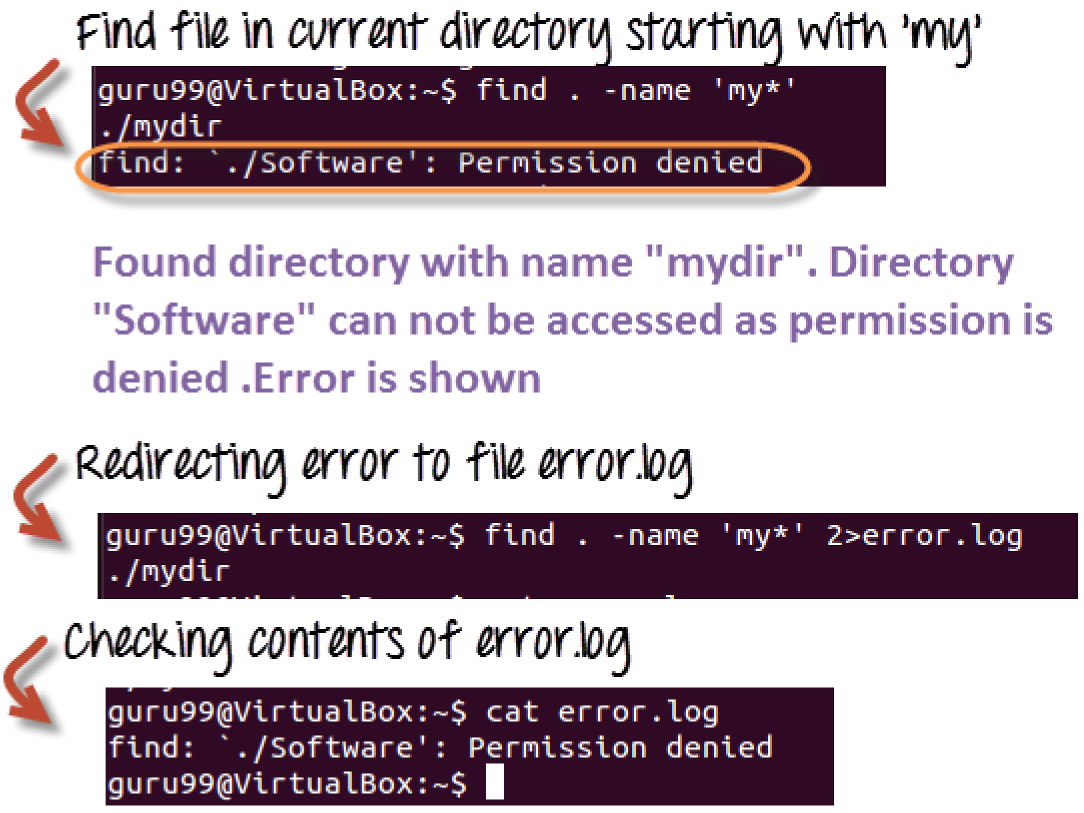
Thus, program output is not cluttered with errors.

Example 2

Here is another example which uses find statement -

find . -name 'my*' 2>error.log

Using the "find" command we are searching the "." current directory for a file with "name" starting with "my"


Find file in current directory starting with 'my'
Found directory with name "mydir". Directory "Software" can not be accessed as permission is denied .Error is shown
Redirecting error to file error.log
Checking contents of error.log

```
guru99@VirtualBox:~$ find . -name 'my*'
./mydir
find: `./Software': Permission denied
```

```
guru99@VirtualBox:~$ find . -name 'my*' 2>error.log
./mydir
```

```
guru99@VirtualBox:~$ cat error.log
find: `./Software': Permission denied
guru99@VirtualBox:~$
```

Example 3

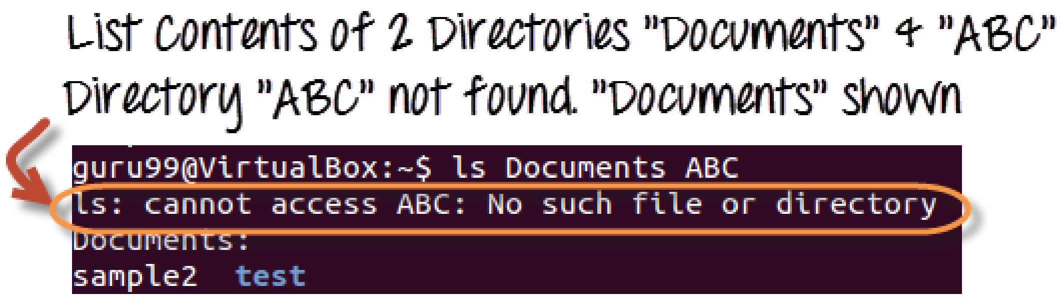
Let's see a more complex example,

Server Administrators frequently, list directories and store both error and standard output into a file, which can be processed later. Here is the command.

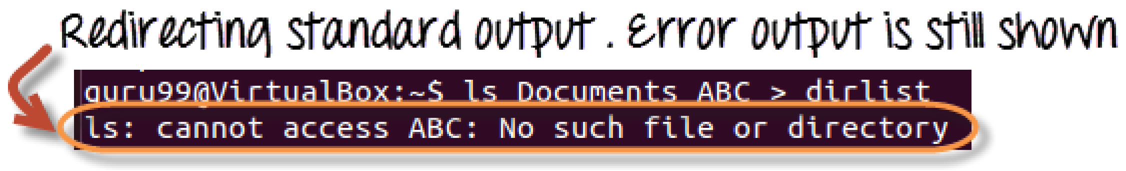
ls Documents ABC> dirlist 2>&1

Here,

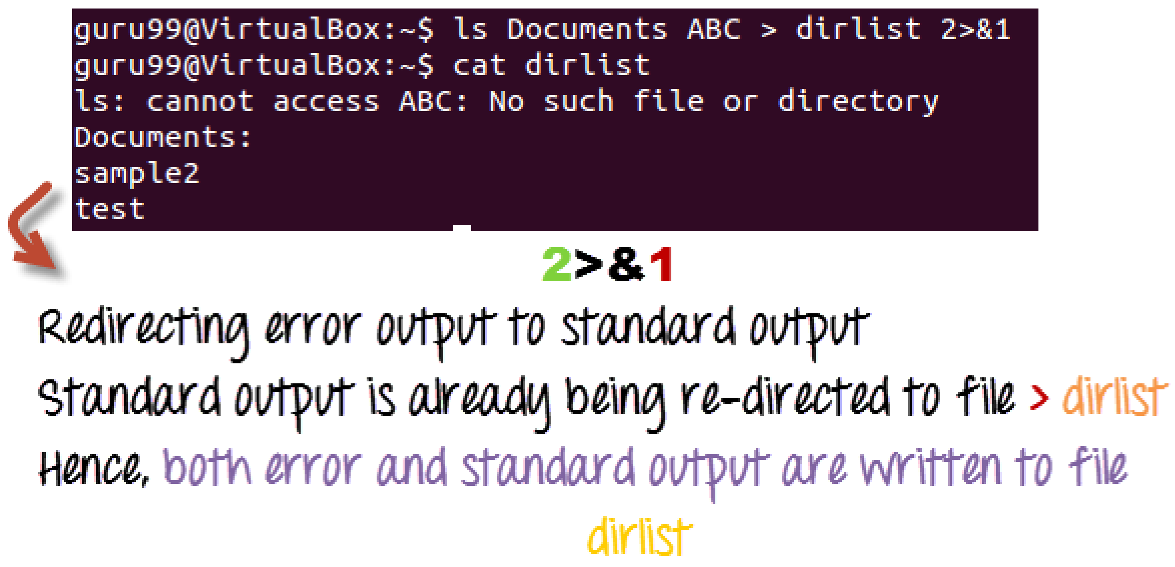
- ">&" which writes the output from one file to the input of another file.
- We are redirecting error output to standard output which in turn is being re-directed to file dirlist. Hence , both the output is written to file dirlist


List Contents of 2 Directories "Documents" + "ABC"
Directory "ABC" not found. "Documents" shown

```
guru99@VirtualBox:~$ ls Documents ABC
ls: cannot access ABC: No such file or directory
Documents:
sample2  test
```



```
guru99@VirtualBox:~$ ls Documents ABC > dirlist
ls: cannot access ABC: No such file or directory
```



```
guru99@VirtualBox:~$ ls Documents ABC > dirlist 2>&1
guru99@VirtualBox:~$ cat dirlist
ls: cannot access ABC: No such file or directory
Documents:
sample2
test
```

2>&1
Redirecting error output to standard output
Standard output is already being re-directed to file > dirlist
Hence, both error and standard output are written to file dirlist

Summary

- Each file in Linux has a corresponding File Descriptor associated with it
- The keyboard is the standard input device while your screen is the standard output device
- ">" is the output redirection operator. ">>" appends output to an existing file
- "<" is the input redirection operator
- ">&"re-directs output of one file to another.
- You can re-direct error using its corresponding File Descriptor 2.