

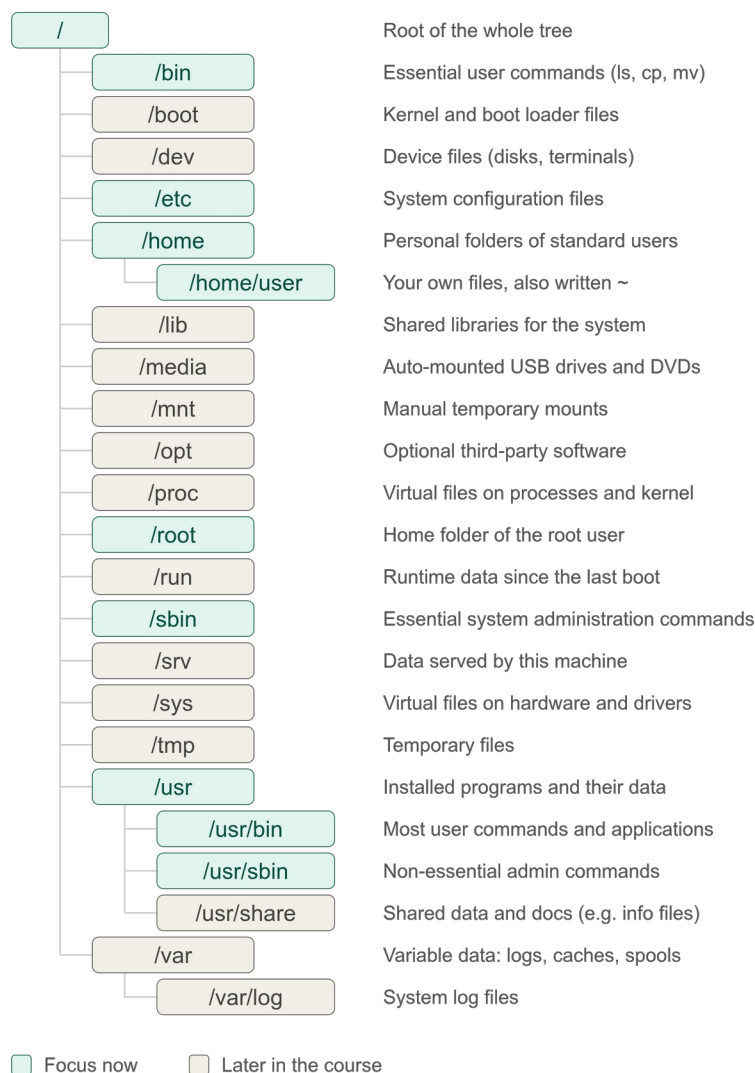
DOCUMENTATION

1- The Linux directory tree structure. Important files and paths

a) A **directory tree** is the way a file system organizes files and folders (directories) in a hierarchy, like an upside-down tree. It starts from one top-level directory, called the **root** and written **/**, and branches out into subdirectories, which can contain more subdirectories and files.

The **Filesystem Hierarchy Standard (FHS)** defines the structure of file systems on Linux and other UNIX-like operating systems.

The following image shows a typical Linux directory tree:



b) Open the following link:

<https://www.howtogeek.com/117435/htg-explains-the-linux-directory-structure-explained/>

Read what the document says about each of the following directories (folders):

a) /	d) /home	g) /usr/bin
b) /bin	e) /root	h) /usr/sbin
c) /etc	f) /sbin	

2- Basic Linux commands

2.1- Command `ls`

a) The `ls` command-line utility displays directory (folder) contents.

b) Examples:

1) `ls` → Displays the contents of the present directory

2) `ls -ls` → Displays the contents of the present directory and extra information about each file or directory (owner, size, permissions, ...)

3) `ls -ls <directory_path>` → Displays the contents of a directory and extra information about each file or directory (owner, size, permissions, ...) inside the selected directory. For instance, the following command: `ls -ls /home` displays the contents (with extra information) about each file or directory inside the `/home` directory.

4) `ls -ls -d <directory_path>` → If you want to display information about a specific directory but not about its contents. For instance `ls -ls -d /home` displays information about `/home`

5) `ls -a <directory_path>` → Displays the hidden contents of a directory. For instance, the following command: `ls -a /etc/skel` displays hidden files inside `/etc/skel`.

6) `ls -a` → Displays hidden contents inside the current directory.

2.2- Command `cat`

a) The `cat` command-line utility displays the contents of a text file. For instance:

```
cat /etc/resolv.conf
```

Displays the contents of `resolv.conf`.

b) If a text is longer than one screen, you can use the special character `|` and the command `more`, which displays as much as can fit on the current screen and waits for user input (Enter for a new line or space bar for a new screen) to advance. For instance:

```
cat /var/log/messages | more
```

2.3- Command `cd`

a) Description: The `cd` moves the user from the current working directory to a new working directory.

b) Synopsis: `cd directory_name`, where `directory_name` is the name of the new working directory.

c) Examples: `cd /home`, `cd /var/log`, ...

d) Special cases and special characters:

Command	Meaning
<code>cd .</code>	Changes to the current working directory. The "." character means "the current working directory".
<code>cd ..</code>	Changes to the parent directory of the current working directory. The ".." characters mean "the parent directory".
<code>cd ~</code>	Changes to the user's personal directory, which is <code>/home/username</code> . The "~" indicates the user's personal directory.
<code>cd -</code>	Changes to the previous directory. The "-" indicates the previous directory.

2.4- Special character * (asterisk)

a) The asterisk `*` is a special character that causes the shell to generate filenames. It matches any number of characters in a filename. For instance, if you want a list of files in a folder that begin with the string "prov", you can run the command: `ls -ls prov*`.

b) Example:

A working directory contains these files: `amemo mem memalx memo memo.061 memoa memoalx.062 memoalx.keep memorandum memosally sallymemo typescript user.memo`). If we run:

Command	Result	What it shows
<code>ls</code>	<code>amemo mem memalx memo memo.0612 memoa memoalx.0620 memoalx.keep memorandum memosally sallymemo typescript user.memo</code>	All files in the working directory.
<code>ls memo*</code>	<code>memo memo.0612 memoa memoalx.0620 memoalx.keep memorandum memosally</code>	Files whose name begins with the string "memo".
<code>ls *mo</code>	<code>amemo memo sallymemo user.memo</code>	Files whose name ends with the string "mo".
<code>ls *alx*</code>	<code>memalx memoalx.0620 memoalx.keep</code>	Files whose name contains the string "alx".
<code>ls *.*</code>	<code>memo.0612 memoalx.0620 memoalx.keep user.memo</code>	Files whose name contains one or more extensions.
<code>ls !(*.*)</code>	<code>amemo mem memalx memo memoa memorandum memosally sallymemo typescript</code>	Files whose name does not contain any extension. The character <code>!</code> means "the opposite of": negate, logical NOT.

2.5- Special character ~ (tilde)

a) The `~` character means "personal folder". For instance, if your user is called `dacomo`, then `~` means `/home/dacomo`.

b) You can combine `~` with a lot of commands on Linux. For instance, if your user is called `dacomo`:

```
cd ~    =>    cd /home/dacomo
ls ~    =>    ls /home/dacomo
```

2.6- Command cp

a) **Description:** The `cp` command-line utility copies one or more files and directories to a destination directory.

b) **Synopsis 1:** `cp source_file destination_file`, where `source_file` is the name of the file that `cp` will copy, and `destination_file` is the name that `cp` assigns to the copy of the file. By default, the destination file name will be equal to the source file name.

c) **Synopsis 2:** `cp -r source_directory destination_directory`, where `source_directory` is the name of the directory that `cp` will copy, and `destination_directory` is the name that `cp` assigns to the copy of the directory. By default, the destination directory name will be equal to the source directory name.

d) **Note:** you can use special characters `.` `..` `-` `*` `~` with command `cp`

e) Examples:

- 1) `cp /var/log/messages /media/usb` → The file `messages` in the directory `/var/log` is copied to the directory `/media/usb`. The new file is also called `messages`.
- 2) `cp /var/log/messages /media/usb/messages.20131110` → `/var/log/messages` is copied to `/media/usb`. The new file is called `messages.20131110`.
- 3) `cp /var/log/* /media/usb` → All files in `/var/log` are copied to the folder `/media/usb`.
- 4) `cp -r /var/log /media/usb` → The directory `log` inside `/var` is copied to the directory `/media/usb`. The new directory is also called `log`.
- 5) `cp -r /var/log /mnt/hd/logfile` → The directory `log` inside `/var` is copied to the directory `/mnt/hd`. The new directory is called `logfile`.

2.7- Command mv

a) Description: The `mv` command-line utility moves one or more source files and directories to a destination directory. The original files are removed. Refer to `cp` for synopsis and examples.

b) Note 1: you can use the special characters `.` `..` `-` `*` `~` **with command** `mv`

c) Note 2: you can use `mv` to rename a file: `mv old_name new_name`

d) Note 3: you can use `mv` to move a directory: `mv existing_directory new_directory`

2.8- Command mkdir

a) Description: The `mkdir` command-line utility creates a new directory.

b) Synopsis: `mkdir directory_name`, where `directory_name` is the name of the new directory.

c) Examples:

- 1) `mkdir test00` → Creates a new directory called `test00` as a child of the current working directory.
- 2) `mkdir /home/etpclot/test01` → Creates a new directory called `test01` as a child of the `/home/etpclot` directory.

d) Note: you can use the special characters `.` `..` `-` `*` `~` **with command** `mkdir`

2.9- Command rm

a) Description: The `rm` command-line utility removes files or directories. By default it does not remove directories.

b) Synopsis 1: `rm file_name`, where `file_name` is the name of the file that will be removed. Example: `rm /var/share/INDEX.gz` → Remove a file named `INDEX.gz` located in the `/var/share` directory.

c) Synopsis 2: `rm -r directory_name`, where `directory_name` is the name of the directory that will be removed. Example: `rm -r /home/etpclot/test00` → Remove a directory named `test00` located in the `/home/etpclot` directory.

d) Note: you can use the special characters `.` `..` `-` `*` `~` **with command** `rm`

e) Note 2: `rm !(filename)` removes all files with the only exception of `filename`. For instance, if you run `rm !(ex01.c)`, every file in the directory is removed with the only exception of `ex01.c`. The character `!` means "the opposite of"/negate/logical NOT.

2.10- Command gzip

a) Description: The `gzip` command-line utility compresses or expands (uncompresses) files. A file compressed with `gzip` is marked by a `.gz` filename extension.

b) Synopsis 1 (compressing): `gzip file_name`, where `file_name` is the name of the file that will be compressed. Example: `gzip /home/etpcloud/INDEX`. After running the command, the original file is replaced with a new file called `INDEX.gz`.

c) Synopsis 2 (expanding): `gzip -d file_name.gz`, where `file_name.gz` is the file that will be expanded. Example: `gzip -d /home/etpcloud/INDEX.gz`. After running the command, the compressed file is replaced with a new file called `INDEX`.

d) Synopsis 3 (listing contents): `gzip -l file_name.gz`, where `file_name.gz` is the name of a compressed file. Option `-l` lists the following fields: size of the compressed file, size of the uncompressed file, compression ratio and uncompressed name.

2.11- Command tar

a) Description: The `tar` command-line utility packs multiple files or directory hierarchies into a single new file, marked by a `.tar` filename extension. It can also extract (unpack) files from a `.tar` file.

b) Synopsis 1: `tar cf filename.tar list_of_files`, where `filename.tar` is the `.tar` file that will be created and `list_of_files` is a list of files.

c) Synopsis 2: `tar xf filename.tar`, where `filename.tar` is the packed file that will be unpacked.

d) Synopsis 3: `tar cf filename.tar directory_name`, where `filename.tar` is the `.tar` file that will be created and `directory_name` is a directory that will be packed.

e) Synopsis 4: `tar tf filename.tar` shows the contents of `filename.tar`

f) Note: you can use the special characters `.` `..` `-` `*` `~` **with command** `rm`

f) Examples:

1) `tar cf test.tar test00.c test01.c test02.c asm.hex TODO.txt` → Creates a new file `test.tar` that packs `test00.c`, `test01.c`, `test02.c`, `asm.hex` and `TODO.txt`.

2) `tar cf test_c.tar test*.c` → Creates a new file `test_c.tar` that packs the `.c` files.

3) `tar xf test.tar` → Unpacks `test.tar`

4) `tar cf home.tar /home` → Creates a new file `home.tar` that packs the content of `/home`

5) `tar tf test.tar` → Shows the contents of `test.tar`

2.12- Command wget

a) Description: A command-line utility for non-interactive download of files from the Web.

b) Synopsis: `wget <URL>`, where `<URL>` is the web address of any resource on the Internet or the LAN.

c) Example:

1) `wget http://www.collados.org/asix1/sm1/sm1act00.pdf` → Downloads a file named `sm1act00.pdf` whose URL is `http://www.collados.org/asix1/sm1/sm1act00.pdf`

2.13- Command which

a) Description: It shows where the binary file (also called executable) of any command/program installed on your file system is located.

b) Synopsis: `which program_name`, where `program_name` is the name of a program or command-line utility.

c) Example:

1) `which geany` → This command shows where `geany` binary (executable) is located. In this case, the output is `/usr/bin`, which means that `geany` is stored in the `/usr/bin` directory.

2.14- Command find

a) Description: The `find` command-line utility searches for files (or directories) in a directory tree.

b) Synopsis: `find <starting-point> -name <file or directory name> 2> /dev/null`, where `<starting-point>` is the directory where the search starts.

c) Examples:

1) `find /etc -name sources.list 2> /dev/null` → It finds where a file called `sources.list` is stored. The `/etc` directory is the starting-point.

2) `find /boot/grub -name *.conf 2> /dev/null` → It finds where files with extension `.conf` are stored. The `/boot/grub` directory is the starting-point.

3) `find /usr/lib/modules -name host 2> /dev/null` → It finds where files with basename `host` are stored. The `/usr/lib/modules` directory is the starting-point.