

# Comandos del Sistema de Ficheros Linux

Aprender a manejar correctamente archivos desde Linux es realmente importante en este sistema operativo porque Linux trata todos los recursos hardware como si fueran archivos, veremos algún ejemplo a lo largo del tutorial.

Linux y otros sistemas Unix como el de MacOS cumplen con el estándar POSIX (Portable Operating System Interface for Unix). Este estándar definido por la IEEE busca que los distintos sistemas operativos compartan las mismas interfaces de comandos de manera que los programas escritos para ejecutarse sobre estos sistemas operativos sean compatibles sin necesidad de hacer modificaciones significativas.

El sistema operativos MacOS y Ubuntu son 100% compatibles en todos los comandos de esta guía porque ambos sistemas operativos cumplen (o son) POSIX. En cambio, Microsoft Windows no es POSIX porque no cumple con la interfaz que marca la normal, en Windows se utiliza una interfaz propia definida en la WinAPI.

Desde Windows podemos usar “Cygwin” para simular un entorno POSIX desde nos funcionarán los comandos Linux. En las versiones actuales de Windows también podemos instalar WSL (Windows Subsystem Linux), una capa de compatibilidad que permite integrar POSIX dentro del terminal de Windows.



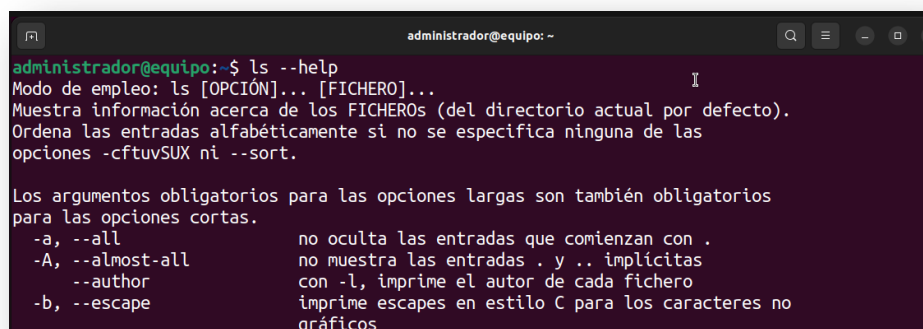
En Linux tenemos terminales que ejecutan lo que se llama shell. Para entenderlo tenemos que volver a los primeros tiempos de la informática donde los ordenadores ocupaban un gran tamaño. Entonces, la forma de trabajar habitualmente era separar el hardware de “proceso” del hardware de “interfaz de usuario” por ello lo habitual era disponer de un servidor y de múltiples terminales que no eran más que una pantalla y teclado conectado al servidor. Una vez teníamos acceso físico al terminal interactuábamos con una shell, una interfaz del sistema operativo para ejecutar comandos.

En los sistemas operativos encontramos varios shells. En Windows tenemos powershell, el símbolo de sistema y WSL. En Linux y Mac tenemos a bash, zsh, fish o dash entre otros.

Todos los shells pueden interactuar con el sistema, es decir con todos podremos copiar archivos y crear carpetas. Sus diferencias vienen dadas por aspectos visuales y de herramientas como el autocompletado y las sugerencias pero también y especialmente si cumplen o no con el estándar POSIX.

Aquellos shells que cumplan con POSIX serán compatibles y un script que funcione en uno lo hará en otro. Por ejemplo, POSIX marca que los parámetros de los comandos se deben pasar con guion o que las variables se deben declarar de una manera concreta. Aquellos shells que no cumplan con POSIX requieren de una adaptación del código para que funcione.

Los sistemas operativos Linux tienen varios tipos de ayudas para los comandos. En primer lugar tenemos la ayuda incorporada en el propio comando que podremos invocar con el parámetro “-- help”



```

administrador@equipo: ~
administrador@equipo:~$ ls --help
Modo de empleo: ls [OPCIÓN]... [FICHERO]...
Muestra información acerca de los FICHEROs (del directorio actual por defecto).
Ordena las entradas alfabéticamente si no se especifica ninguna de las
opciones -cftuvSUX ni --sort.

Los argumentos obligatorios para las opciones largas son también obligatorios
para las opciones cortas.
-a, --all                no oculta las entradas que comienzan con .
-A, --almost-all        no muestra las entradas . y .. implícitas
--author                 con -l, imprime el autor de cada fichero
-b, --escape             imprime escapes en estilo C para los caracteres no
                        gráficos

```

En la captura anterior podemos ver como los parámetros pueden ser invocados con un guion y una letra o con dos guiones y una cadena. En general, tendemos a usar guion y letra porque se pueden concatenar dando lugar a nemotécnicos fáciles de recordar y a la vez muy potentes y personalizados.

Otro sistema de ayuda consiste en el uso del programa “man”. En general todos los Linux incluyen de serie este programa al que se le pasa como argumento el nombre del comando del que necesitamos consultar su ayuda. Esta ayuda incluye un pequeño resumen, una descripción larga, opciones y ejemplos.

Para avanzar en el programa “man” usamos las flechas para subir/bajar líneas, la barra espaciadora para pasar a la siguiente pantalla y la letra “q” para salir.

```

administrador@equipo: ~
LS(1)                                User Commands

NAME
    ls - list directory contents

SYNOPSIS
    ls [OPTION]... [FILE]...

DESCRIPTION
    List information about the FILES (the current directory by default). Sort
    alphabetically if none of -cftuvSUX nor --sort is specified.

    Mandatory arguments to long options are mandatory for short options too.

    -a, --all
        do not ignore entries starting with .

    -A, --almost-all
        do not list implied . and ..

    --author
        with -l, print the author of each file

    -b, --escape
        print C-style escapes for nongraphic characters

Manual page ls(1) line 1 (press h for help or q to quit)

```

## LS

Este comando lista los archivos contenidos en una ruta. El resultado de escribir el comando “ls” sin parámetros es el de la siguiente captura.

```

administrador@equipo: ~
administrador@equipo:~$ ls
Descargas Documentos Escritorio Imágenes Música Plantillas Público snap Vid
administrador@equipo:~$

```

El comando “ls” admite una lista de opciones que podemos consultar en la ayuda y también admite una ruta de la que mostrar el contenido.

Con respecto a los parámetros podemos consultarlos en la propia ayuda del comando (con --help) o con el comando “man ls”

Como los parámetros con solo una letra se pueden concatenar es normal que tengamos memorizados algunos comandos con parámetros como “ls -la” que nos mostrará una lista larga (un archivo o carpeta por línea) incluyendo los archivos y carpetas ocultas. El resultado lo podemos ver en la siguiente captura.

```
administrador@equipo: ~
See "man sudo_root" for details.

administrador@equipo:~$ ls -la
total 84
drwxr-x--- 16 administrador administrador 4096 oct 25 12:47 .
drwxr-xr-x  3 root          root          4096 nov 28  2023 ..
-rw-----  1 administrador administrador  353 oct 25 15:14 .bash_history
-rw-r--r--  1 administrador administrador  220 nov 28  2023 .bash_logout
-rw-r--r--  1 administrador administrador 3771 nov 28  2023 .bashrc
drwx----- 11 administrador administrador 4096 nov 28  2023 .cache
drwx----- 11 administrador administrador 4096 nov 28  2023 .config
drwxr-xr-x  2 administrador administrador 4096 nov 28  2023 Descargas
drwxr-xr-x  2 administrador administrador 4096 nov 28  2023 Documentos
drwxr-xr-x  2 administrador administrador 4096 nov 28  2023 Escritorio
drwx-----  2 administrador administrador 4096 oct 28 13:36 .gnupg
drwxr-xr-x  2 administrador administrador 4096 nov 28  2023 Imágenes
-rw-----  1 administrador administrador   20 oct 25 12:47 .lessht
drwx-----  3 administrador administrador 4096 nov 28  2023 .local
drwxr-xr-x  2 administrador administrador 4096 nov 28  2023 Música
drwxr-xr-x  2 administrador administrador 4096 nov 28  2023 Plantillas
-rw-r--r--  1 administrador administrador  807 nov 28  2023 .profile
drwxr-xr-x  2 administrador administrador 4096 nov 28  2023 Público
drwx-----  3 administrador administrador 4096 nov 28  2023 snap
drwx-----  2 administrador administrador 4096 nov 28  2023 .ssh
drwxr-xr-x  2 administrador administrador 4096 nov 28  2023 Vídeos
administrador@equipo:~$
```

Aprovechamos esta captura anterior para explicar que los archivos cuyo nombre empieza por un punto (.) se consideran ocultos y no se mostrarán con un “ls” normal, comparemos la captura anterior y siguiente.

```

administrador@equipo: ~
administrador@equipo:~$ ls -l
total 36
drwxr-xr-x 2 administrador administrador 4096 nov 28 2023 Descargas
drwxr-xr-x 2 administrador administrador 4096 nov 28 2023 Documentos
drwxr-xr-x 2 administrador administrador 4096 nov 28 2023 Escritorio
drwxr-xr-x 2 administrador administrador 4096 nov 28 2023 Imágenes
drwxr-xr-x 2 administrador administrador 4096 nov 28 2023 Música
drwxr-xr-x 2 administrador administrador 4096 nov 28 2023 Plantillas
drwxr-xr-x 2 administrador administrador 4096 nov 28 2023 Público
drwx----- 3 administrador administrador 4096 nov 28 2023 snap
drwxr-xr-x 2 administrador administrador 4096 nov 28 2023 Videos
administrador@equipo:~$
  
```

El otro parámetro que acepta el comando “ls” es la ruta de la que mostrar el contenido. Las rutas pueden ser absolutas si indican todo el camino desde la raíz (/) o relativas si indican el camino desde la carpeta actual o su carpeta padre, el punto y los dos puntos seguidos respectivamente. En caso de que la ruta sea vacía o no cumpla con lo anterior (ni raíz ni puntos) el sistema operativo interpretará un punto (.) por defecto y mostrará el contenido de la carpeta activa.

En la ruta también podemos indicar filtros usando el comodín asterisco (\*) que se interpreta como un “nada o cualquier otra cadena”.

```

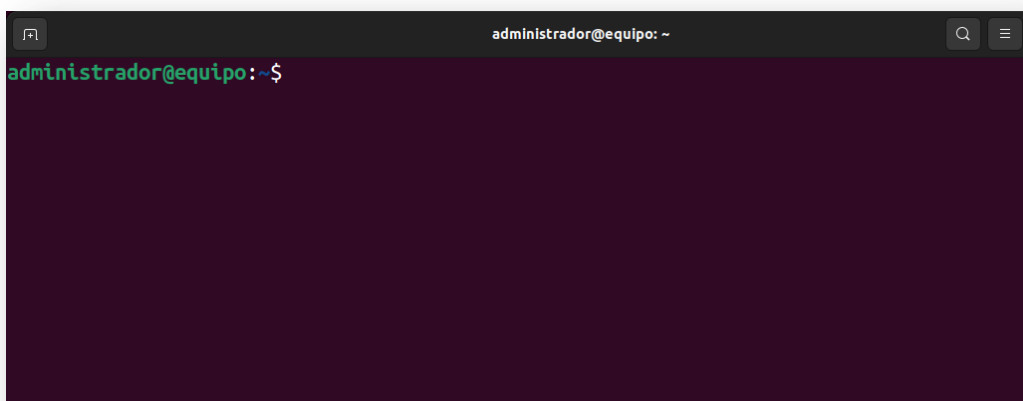
ls *.jpg
ls verano*
ls *
ls kb*.exe
ls *factura*
  
```

Aquí tenemos varios usos del comodín en estos 5 comandos “ls” que, como no hemos indicado una ruta que empiece por la raíz ni por la carpeta actual o padre, tomará por defecto el punto (./)

El primer comando “ls” busca en la carpeta actual todos los archivos y carpetas que acaben en “.jpg”. El segundo comando “ls” mostrará todos los archivos y carpetas que empiecen por la cadena “verano”. El tercer comando “ls” es un abuso de lenguaje ya que le estamos diciendo que nos liste todo lo que haya en la carpeta actual, es exactamente lo mismo que escribir el comando “ls” sin ningún parámetro. El cuarto comando “ls” muestra todos los archivos y carpetas que tengan el patrón de empezar por “kb” y finalizar por “.exe” sin importar la cadena que tengan en medio. Por último, el quinto comando “ls” mostrará archivos y carpetas que tengan la subcadena “facturas” en algún punto del nombre.

## CLEAR

Después de listar el contenido de nuestras carpetas es posible que tengamos la terminal llena y deseamos limpiarla para volver a tener la línea del prompt en primera posición. El comando “clear” permite hacer eso mismo y nos dejará la terminal como recién abierta. Algo a tener en cuenta es que en la mayoría de casos perderemos el “scrollback” y no podremos ver la salida de los comandos anteriores

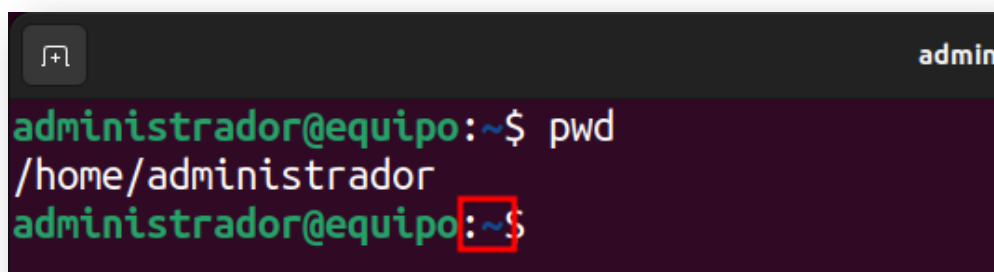


```
administrador@equipo: ~  
administrador@equipo:~$
```

## PWD

Normalmente el prompt está configurado por defecto para mostrarnos la ruta actual activa pero en ocasiones es complicado determinarlo a simple vista. El comando “pwd” nos devuelve la ruta actual activa.

Linux es un sistema multiusuario y como tal está configurado para que cada usuario disponga de una carpeta personal y privada donde dejar sus documentos con seguridad. Todas estas carpetas se crean bajo el directorio “/home” que depende directamente de la raíz (/). Al entrar en la carpeta de cada usuario se dice que estamos en su “home” o carpeta personal.



```
administrador@equipo:~$ pwd  
/home/administrador  
administrador@equipo:~$
```

En esta captura podemos ver el resultado del comando “pwd” que nos responde que la carpeta actualmente activa es “/home/administrador”, la carpeta personal o “home” del usuario. El prompt nos indica que estamos en nuestro “home” con una virgulilla, el resto de las ubicaciones nos las mostrará como ruta absoluta desde la raíz.

```
administrador@equipo:~/Descargas$  
administrador@equipo:/usr/bin$  
administrador@equipo:/home$
```

## CD

El comando “cd” lo utilizamos para cambiar de directorio o carpeta activa y para ello debemos pasarle una ruta. Recordemos que las rutas deben ser absolutas si empiezan desde la raíz (/) o relativas si empiezan por el punto o dos puntos seguidos indicando la propia carpeta activa o su carpeta padre respectivamente. Aquí aplicamos las mismas normas que anteriormente, si no hay una referencia explícita a una ruta absoluta o relativa se usará el punto por defecto.

El comando “cd” sin ninguna ruta tomará el punto como ruta por defecto. El punto indica la propia carpeta activa así que literalmente, el comando “cd” sin parámetros está solicitando cambiar de carpeta activa a la carpeta activa, no tiene ni efecto ni sentido.

La carpeta home del usuario tiene un acceso especial usando la virgulilla. En los teclados actuales no es habitual que aparezca la virgulilla dibujada pero se puede conseguir con la combinación de teclas Alt Gr + 4 en sistemas Windows y Linux, en MacOS se puede conseguir con la combinación Fn + opción + ñ.

También es posible llegar directamente al home del usuario actual usando la variable de entorno HOME. Para acceder al contenido de esta variable hay que usar el símbolo dólar (\$) antes del nombre. De esta manera, “cd \$HOME” nos llevará a la home del usuario activo.

## TREE

El comando “tree” muestra los archivos y carpetas de la carpeta indicada como origen y todas sus subcarpetas en forma de árbol dentro del entorno CLI de la terminal.

Por defecto, en la versión Ubuntu 20.04, el comando “tree” no viene instalado por defecto.



```
administrador@equipo: ~  
administrador@equipo:~$ tree  
No se ha encontrado la orden «tree», pero se puede instalar con:  
sudo snap install tree # version 2.1.3+pkg-5852, or  
sudo apt install tree # version 2.0.2-1  
Consulte «snap info tree» para ver más versiones.  
administrador@equipo:~$
```

En la propia salida, el sistema operativo nos propone instalarlo mediante “snap” o “apt”. Nosotros usaremos apt con el siguiente comando:

```
sudo apt install tree
```

```
administrador@equipo:~$ sudo apt install tree  
[sudo] contraseña para administrador:  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias... Hecho  
Leyendo la información de estado... Hecho  
Se instalarán los siguientes paquetes NUEVOS:  
  tree  
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 9 no actualizados.  
Se necesita descargar 47,2 kB de archivos.  
Se utilizarán 108 kB de espacio de disco adicional después de esta operación.  
Des:1 http://es.ports.ubuntu.com/ubuntu-ports jammy/universe arm64 tree arm64 2.0.2-1 [47,2 kB]  
Descargados 47,2 kB en 0s (101 kB/s)  
Seleccionando el paquete tree previamente no seleccionado.  
(Leyendo la base de datos ... 145211 ficheros o directorios instalados actualmente.)  
Preparando para desempaquetar .../tree_2.0.2-1_arm64.deb ...  
Desempaquetando tree (2.0.2-1) ...  
Configurando tree (2.0.2-1) ...  
Procesando disparadores para man-db (2.10.2-1) ...  
administrador@equipo:~$
```

Si no hay errores, como en la captura anterior, ya tendríamos instalado el comando “tree”.

Cuando indiquemos la ruta a la que hacer un “tree” debemos recordar que las rutas pueden ser absolutas sin parten desde la raíz (/) o relativas si parten de punto o dos puntos seguidos. En caso de que la ruta no coincida con estos dos inicios o bien que sea vacía, el sistema operativo entenderá que debe usar el punto y ejecutar el comando “tree” con origen en la carpeta activa.





```
administrador@equipo: ~  
administrador@equipo:~$ tree  
.  
├── Descargas  
├── Documentos  
├── Escritorio  
├── Imágenes  
├── Música  
├── Plantillas  
├── Público  
├── snap  
│   └── snapd-desktop-integration  
│       ├── 255  
│       │   ├── Descargas  
│       │   ├── Documentos  
│       │   ├── Escritorio  
│       │   ├── Imágenes  
│       │   ├── Música  
│       │   ├── Plantillas  
│       │   ├── Público  
│       │   └── Vídeos  
│       └── 85  
│           ├── Descargas  
│           └── Documentos
```

MKDR

RMDIR

TOUCH



CAT

RM

CP

MV