Comandos útiles en la terminal de Linux (Parte 1)

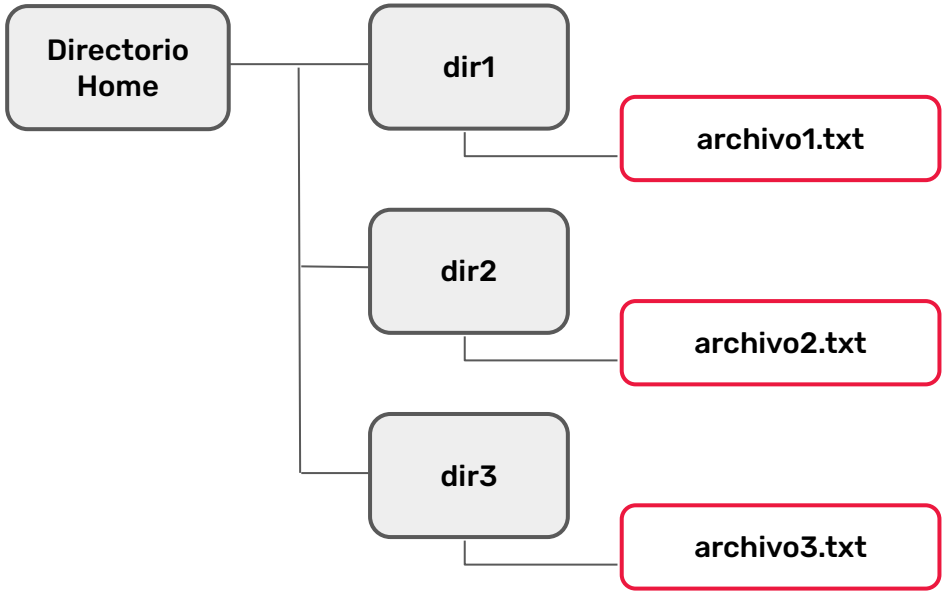
DigitalHouse>

Certified Tech Developer
The Ultimate Degree

Índice

1 Preparación del ambiente

Consolidando nuestro ambiente



Para poder seguir correctamente los ejemplos posteriores, es deseable que en tu ambiente (máquina virtual o WSL) tengas replicada la siguiente estructura de carpetas y archivos.

Consolidando nuestro ambiente

Para ello debemos ejecutar los siguientes comandos, en el orden dado (solo el texto que está luego del prompt o sea, luego del “$”, en color blanco):

{}

```
edorio@DESKTOP-W10:~$ mkdir dir1 dir2 dir3
edorio@DESKTOP-W10:~$ touch dir1/archivo1.txt
dir2/archivo2.txt dir3/archivo3.txt
```

Verificando nuestro ambiente

Listamos los directorios con la instrucción “ls -R”. Deberíamos obtener lo siguiente:

```
edorio@DESKTOP-W10:~$ ls -R
.:
dir1  dir2  dir3
./dir1:
archivo1.txt
./dir2:
archivo2.txt
./dir3:
archivo3.txt
```

2 | Comandos para el manejo de archivos

ls

Con el comando ls podrás listar los diferentes archivos y directorios de la carpeta de trabajo en la que te encuentres. El comando acepta multitud de opciones, algunas de las cuales veremos a continuación.

A continuación, podemos observar el uso más simple del comando ls. Si no le indicamos ninguna opción, enumerará todos los archivos y directorios que se encuentran en la carpeta de trabajo actual, sin tener en cuenta archivos ocultos.

```
edorio@DESKTOP-W10:~$ ls
dir1  dir2  dir3
```

ls -a

Con esta opción, el comando te mostrará —en forma de lista— todo el contenido que se encuentre dentro del directorio de trabajo, incluyendo archivos y carpetas ocultos. Dependiendo del shell, algunos tipos de archivos se mostrarán con colores diferentes.

```
{}
```

```
edorio@DESKTOP-W10:~$ ls -a
.          .aws            .bash_logout   .config    .landscape
.profile                    .vagrant.d  dir3 ..         .azure
.bashrc         .docker     .local         .ssh    dir1 .ansible
.bash_history   .cache          .fastlane   .motd_shown
.sudo_as_admin_successful   dir2
```

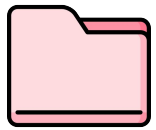
ls -l

Esta opción es similar a la anterior, pero muestra el contenido en forma de lista e incluye información referente a cada elemento. Es de las más utilizadas, siendo especialmente útil a la hora de conocer el propietario y los permisos de cada fichero.

```
edorio@DESKTOP-W10:~$ ls -l
total 12
drwxr-xr-x 2 edorio edorio 4096 May 21 01:59 dir1
drwxr-xr-x 2 edorio edorio 4096 May 21 01:59 dir2
drwxr-xr-x 2 edorio edorio 4096 May 21 01:59 dir3
```

mkdir

Te permitirá crear un directorio con el nombre y la ruta que especifiques. Si no le indicás ninguna ruta, por defecto, te creará la carpeta dentro del directorio de trabajo en el que te encuentres.



```
{}  edorio@DESKTOP-W10:~$ mkdir dir4
```

Caso contrario, le podés indicar que cree un directorio con un path definido dentro de **dir1**.

```
{}  edorio@DESKTOP-W10:~$ mkdir dir1/subdir1
```

rmdir

Te permite **eliminar el directorio** que le especifiques. Un detalle importante es que para poder utilizar este comando, el directorio a borrar debe estar vacío.

```
{}  edorio@DESKTOP-W10:~$ rmdir dir4
```

El de arriba es el uso más simple del comando, sin indicar ruta.

Podemos también borrar un directorio con un path definido.

```
{}  edorio@DESKTOP-W10:~$ rmdir dir1/subdir1
```

rm

Este comando permite eliminar archivos sueltos y directorios que no se encuentren vacíos.

```
{}  edorio@DESKTOP-W10:~$ rm dir1/archivo1.txt
```

El de arriba es el uso más simple del comando, sin indicar ruta.

Eliminamos 1 archivo específico dentro de **dir1**.

```
{}  edorio@DESKTOP-W10:~$ rm -r dir2
```

Con el modificador **-r** eliminamos el directorio **dir2** y, recursivamente, todo su contenido. Es un comando a utilizar con mucha precaución.

cp

Usando este comando serás capaz de copiar archivos y directorios. Así como ubicarlos en otras rutas, definiendo origen primero y luego destino.

```
{}  edorio@DESKTOP-W10:~$ cp dir3/archivo3.txt dir1/archivo1.txt
```

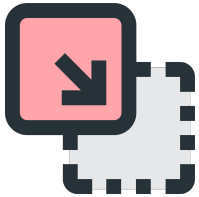
El de arriba es el uso más simple del comando, sin indicar ruta. Copiamos en este caso el **archivo3.txt**, hacia **dir1** y lo nombramos **archivo1.txt**.

En este caso, con el modificador -r copiamos el directorio **dir3** en uno llamado **dir2**, que el mismo comando creo.

```
{}  edorio@DESKTOP-W10:~$ cp -r dir3 dir2
```

mv

Este comando te servirá para mover archivos desde la consola. La sintaxis es muy sencilla, solamente deberás especificar la ubicación de inicio —incluyendo el nombre del archivo— y la ubicación de destino.



```
{}  edorio@DESKTOP-W10:~$ mv dir1/archivo1.txt dir3/archivo1.txt
```

Movimos un archivo de **dir1** a **dir3**, conservando su nombre original.

En el siguiente caso usamos el comando **mv** para renombrar un archivo, ya que las rutas definidas son las mismas.

```
{}  edorio@DESKTOP-W10:~$ mv dir3/archivo1.txt dir3/archivo3bis.txt
```

3 | Comandos para leer archivos de texto

cat

Este es uno de los comandos más utilizados cuando se trata de manejar archivos de texto (en formato .txt) desde la terminal. Entre sus múltiples opciones, está la posibilidad de crear un archivo e imprimir por pantalla su contenido.

```
{}  edorio@DESKTOP-W10:~$ cat >dir1/archivo1.txt
```

Esto nos abrirá el **archivo1.txt**, permitiendo editarlo. Con la combinación **CTRL+D** terminaremos la edición y se guardará el contenido.

```
{}  edorio@DESKTOP-W10:~$ cat dir1/archivo1.txt
    Hola Digital House, esto es CAT
```

Invocando el comando sin el símbolo “>”, nos mostrará por pantalla el contenido del mismo. Se puede usar con el modificador **-n**, para numerar las líneas y con el **-b**, con el propósito de no mostrar las líneas en blanco.

more

Este es otro comando útil para imprimir por pantalla el contenido de un archivo de texto. Esencialmente es igual que el comando cat, con la diferencia que este comando pagina el contenido, por lo que es más adecuado para leer archivos largos.

```
{}  edorio@DESKTOP-W10:~$ more /var/log/dpkg.log
```

Nos paginará el archivo en cuestión, de tal manera que en sus últimas líneas lo veremos así:

```
2021-02-19 23:56:28 remove linux-headers-5.4.0-65:all 5.4.0-65.73 <none>
2021-02-19 23:56:28 status half-configured linux-headers-5.4.0-65:all 5.4.0-65.73
2021-02-19 23:56:28 status half-installed linux-headers-5.4.0-65:all 5.4.0-65.73
--More--(5%)
```

nano

Nano es un editor de textos para la terminal, que más que para leer archivos sirve para modificarlos y editarlos. Aunque para esta guía también nos vale perfectamente para abrir el archivo y visualizar su contenido desde la línea de comandos.

```
edorio@DESKTOP-W10:~$ nano dir1/archivo1.txt
```

Nos mostrará:

```
 GNU nano 4.8                                    dir1/archivo1.txt
Hola Digital House, esto es CAT
```

Una vez abierto, en la parte inferior se visualizará las diferentes combinaciones de teclas que necesitarás a la hora de trabajar con archivos.

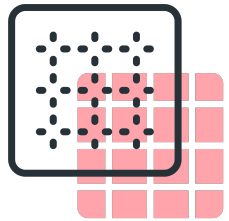
nano (cont.)

En la parte inferior se muestran las diferentes combinaciones de teclas que se necesitarán a la hora de trabajar con archivos:

- **CTRL+R**: combinación para indicarle un archivo de texto a Nano para que lo abra y muestre su contenido por la consola.
- **CTRL+V**: estando dentro de Nano y con el archivo abierto en la consola, esta combinación sirve para avanzar a la página siguiente.
- **CTRL+Y**: sirve para retroceder a la página anterior.
- **CTRL+W**: sirve para introducir un carácter o grupo de caracteres y buscar en el texto cualquier letra o palabra que coincida con el parámetro de búsqueda.
- **CTRL+X**: para cerrar el archivo una vez que lo hayas terminado de visualizar en la consola. Eso cerrará el editor de texto Nano y volverá a aparecer el prompt de Bash por consola.

grep

Este comando, perteneciente a la familia Unix, es una de las herramientas más versátiles y útiles disponibles. Se encarga de buscar un patrón que definamos en un archivo de texto. Su primer parámetro es la cadena de texto a buscar, luego el o los archivos (acepta comodines como *, pudiendo con el modificador -r recorrer recursivamente) que vamos a buscar.



```
{}  edorio@DESKTOP-W10:~$ grep "Digital House" * -r
```

En este caso, buscamos la cadena “Digital House” en todos los archivos, de manera recursiva, la ejecución nos devolvió lo siguiente:

```
edorio@AR-CARRERA-09:~$ grep "Digital House" * -r
dir1/archivo1.txt:Hola Digital House, esto es CAT
edorio@AR-CARRERA-09:~$
```

tee

Lee una entrada estándar y la escribe en la salida estándar y en uno o más archivos. De forma normal, en la redirección de salida, las líneas del comando se escriben en un archivo, pero si queremos ver dicha salida al mismo tiempo, no podemos. ¡Usando el comando tee sí es posible lograrlo!

```
{}  edorio@DESKTOP-W10:~$ ls -l | tee listado.txt
```

En este caso, además de mostrarnos el directorio, el mismo será guardado en un archivo.

```
{}  edorio@DESKTOP-W10:~$ ls -l | tee -a listado.txt
```

Utilizando el modificador -a, se agregará el contenido al archivo, sin pisar lo anterior.

DigitalHouse>