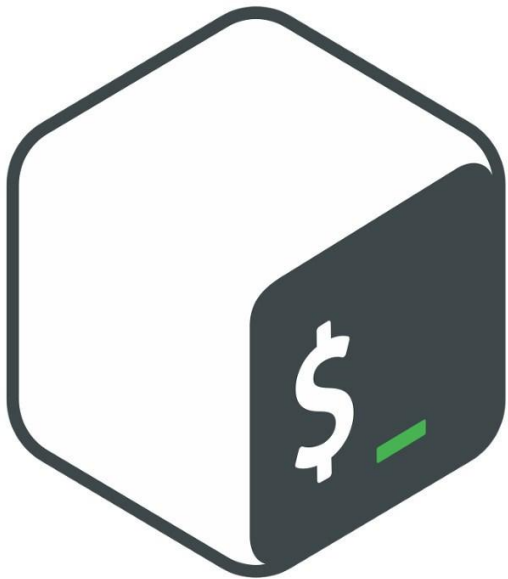


Introducción a BASH

...

Linux



BASH

THE BOURNE-AGAIN SHELL

BASH Linux

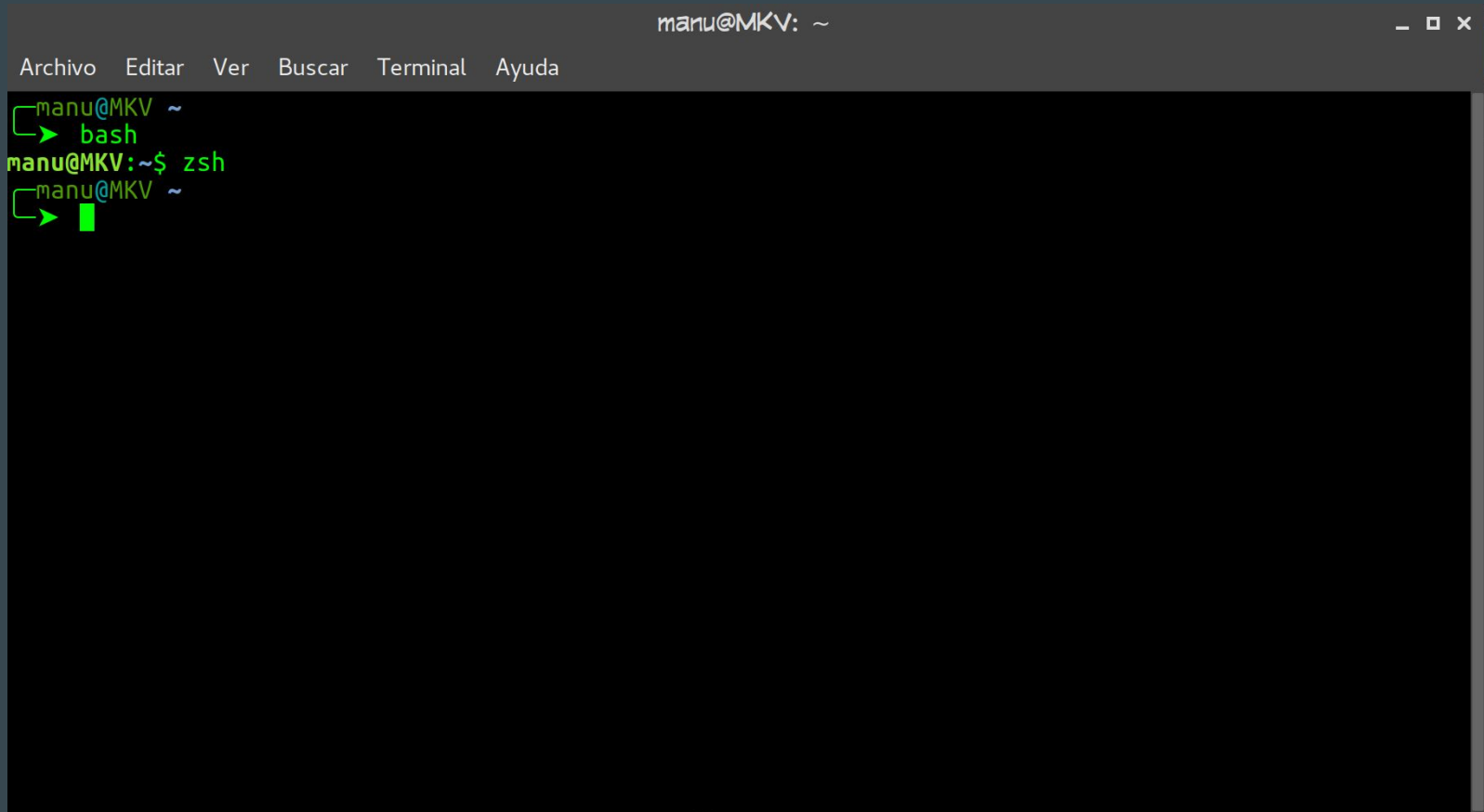
Bash es un shell de Unix. El nombre es un acrónimo de **b**ourne-**a**gain-**s**hell.

MI Terminal

Archivo Editar Ver Buscar Terminal Ayuda

```
manu@MKV ~  
└─> bash  
manu@MKV:~$ █
```

Sí, lo sé, mi terminal es ZSH e invoco a Bash



The image shows a terminal window titled "manu@MKV: ~". The window has a menu bar with the following options: Archivo, Editar, Ver, Buscar, Terminal, and Ayuda. The terminal content shows the following sequence of events:

```
manu@MKV ~  
➤ bash  
manu@MKV:~$ zsh  
manu@MKV ~  
➤ █
```

The prompt changes from "manu@MKV ~" to "manu@MKV:~\$" after running "bash", and back to "manu@MKV ~" after running "zsh". A red cursor is visible on the line following the second "manu@MKV ~" prompt.

BASH

Nos sirve para interpretar órdenes en un terminal, que mola más.

Nautilus de Ubuntu

Nemo

Mi primer Hola Mundo en Bash debe de comenzar con la primera línea o shebang dando la ruta del propio bash, que por lo general se ubica en **/bin/bash**. Por ejemplo: **#!/bin/bash** . Y el comando **echo** para imprimir en pantalla.

```
#!/bin/bash  
# Script de Hola Mundo  
echo "Hola Mundo"
```

Salida:

Hola Mundo

Asignado Variables en Bash

En BASH, las variables se asignan simplemente dando el nombre de la variable y su valor:

```
#!/bin/bash
```

```
#Asignando variables
```

```
hola=1
```

Como podemos leer aquí, este código asigna el valor 1 a la variable "**hola**". En BASH las variables son "CASE SENSITIVE" (sensibles), es decir, la variable *Hola* no es lo mismo que *HOLA* ni que *holA*. Tampoco hay que asignarles un tipo, es decir podremos darle cualquier valor sin decirle si es numérico o letras.

Invocando Variables

En BASH, las variables las invocamos simplemente anteponiendo un símbolo de dólar '\$' antes del nombre de la variable. Ejemplo:

```
#!/bin/bash
```

```
#Asignando variables
```

```
hola=1
```

```
#Llamando a la variable $hola
```

```
$hola
```

```
#Mostrando el contenido de la variable
```

```
echo $hola
```

Explicación

Si ponemos atención al código, en BASH, las variables simplemente se reemplazan por su valor al llamarlas, de modo que en ellas podemos almacenar X texto ó número, ya sean comandos o lo que sea.

Al llamar a la variable, observamos que simplemente da el valor y lo pasa como una orden al intérprete, pero si lo ponemos siguiendo un comando como echo entonces este mostrará el contenido de la variable.

Ejemplo de uso:

```
#!/bin/bash
```

```
#
```

```
# Se guarda en la variable el valor generado por $RANDOM,
```

```
# el % 5 asegura obtener un numero menor a 5 .
```

```
RNM=`expr $RANDOM % 5`
```

```
echo $RNM
```

Comandos básicos de Bash

mkdir-----	<u>Crea un directorio</u> -----	mkdir tmp
pwd-----	<u>Muestra la ruta del directorio actual</u> -----	pwd
ls-----	<u>Lista el contenido del directorio</u> -----	ls -l /usr/bin
rm fich-----	<u>Borra un fichero</u> -----	rm foo.c
rm -r dir-----	<u>Borra todo un directorio</u> -----	rm -rf prog_dir
passwd-----	<u>Cambia la contraseña</u> -----	passwd
file arch-----	<u>Muestra el tipo de un archivo</u> -----	file arc_mi_archivo

history-----**Saber el historial**

history 10-----**Ver los últimos 10 comandos**

Ctrl+R-----**Comenzamos a escribir la parte del comando que recordemos**

find-----**Busca archivos o carpetas en la ruta que le indique**

grep-----**Localiza una palabra, clave o frase en un conjunto de directorios**

sudo-----**super-user do-----Permite ejecutar acciones con privilegios**
-----**de seguridad del root de manera segura**

su-----**Cambiar de usuario sin necesidad de hacer un cierre**
o cambio de sesión

aptitude-----**Versión mejorada de apt**

Comandos básico de Bash

who / rwho-----Muestra información usuarios conectados-----who

kill [-señal] PID-----Matar un proceso-----kill 223344

echo string-----Escribe mensaje en la salida estándar-----echo `Hola mundo

mv fich1 ...fichN dir-----Mueve un archivo(s) a un directorio-----mv a.out prog1

mv fich1 fich2-----Renombra un fichero-----mv .c prog_dir

cd [dir]-----Cambia de directorio-----cd /tmp

chmod permisos fich-----Cambia los permisos de un archivo-----chmod +x miscript

glxgears-----**Para saber el rendimiento (frames x segundo)**

Comandos básico de Bash

tail -count fich-----Muestra el final de un archivo-----**tail prog1.c**

man comando-----Ayuda del comando especificado-----**man gcc, man -k printer**

tree-----**Muestra la estructura de directorios y archivos en forma gráfica--tree**

more-----**Ve el contenido de los archivos página a página**

locate-----**Localiza archivos según una lista generada**

split-----**Dividir archivos**

file-----**Muestra el tipo de archivo**

lspci | grep -i vga-----**Para saber que tipo de gráfica tengo**

Comandos básico de Bash

tar	Empaqueta archivos
gzip	Comprime archivos en formato .gz
gunzip	Descomprime archivos en formato .gz
chmod	Cambia permisos de archivos y directorios
chown	Cambia de propietario/usuario
chgrp	Cambia de grupo
nano	Abre el editor nano

Comandos para la gestión de discos y dispositivos

mount	Monta un disco/dispositivo
umount	Desmonta un disco/dispositivo
df	Muestra el espacio libre de los discos/dispositivos
mkfs	Formatea un disco/dispositivo
fsck	Estado del disco/dispositivo
du	Muestra el espacio usado por el disco/dispositivo o directorios
fdisk	Abre la aplicación para la gestión de particiones

Comandos de red

netstat-----	Muestra estado de la red
ifconfig-----	Muestra la configuración del dispositivo de red
iwconfig-----	Muestra la configuración del dispositivo de red inalámbrico
nmap-----	Escanea la red y busca los puertos que se encuentran disponibles
ping-----	Indica si hay respuesta por parte del servidor
netconfig-----	Configuración de la red
route -n-----	Muestra la tabla de rutas de la conexión de red

Introducción a Bash

Por Manu Cogolludo [@makova65](#)



Oficina de Software Libre, UGR ([OSL](#))
[GitHub](#)

Esta obra está bajo una [licencia de Creative Commons Reconocimiento 4.0 Internacional](#).