

# Manual de Primeros Pasos en Bash.

## Comandos Básicos y Avanzados.

---

### Inducción: Terminal, Consola, Bash, CMD y PowerShell ¿Es lo mismo?

Aunque suelen usarse como sinónimos, existen diferencias importantes entre estos términos:

Término	Descripción
Terminal	Interfaz que permite al usuario interactuar con el sistema mediante texto (puede ser física o virtual).
Consola	Término histórico similar a terminal, usado a veces como sinónimo.
Bash	Intérprete de comandos para sistemas Unix/Linux. Acrónimo de "Bourne Again SHell".
CMD	Intérprete de comandos de Windows (Command Prompt), limitado en funciones frente a Bash.
PowerShell	Intérprete de comandos avanzado de Windows, orientado a objetos.

**Resumen:** Terminal es el entorno, Bash/CMD/PowerShell son los lenguajes o programas que corren dentro.

---

### Comandos Básicos

Acción	Bash (Linux)	CMD (Windows)	PowerShell	Ejemplo (Bash)
Ver contenido de carpeta	ls	dir	Get-ChildItem	ls /home/usuario
Cambiar de carpeta	cd	cd	Set-Location	cd Documentos
Crear carpeta	mkdir	mkdir	New-Item - ItemType Directory	mkdir nueva_carpeta
Borrar archivo	rm archivo	del archivo	Remove-Item	rm notas.txt
Limpiar pantalla	clear	cls	Clear-Host	clear



Acción	Bash (Linux)	CMD (Windows)	PowerShell	Ejemplo (Bash)
Ver dirección actual	pwd	cd	Get-Location	pwd

## Comandos Avanzados

Acción	Bash	CM D	PowerShell	Ejemplo (Bash)
Buscar en archivos	grep	findstr	Select-String	grep "error" archivo.log
Redirigir salida	> y >>	> y >>	Out-File	ls > listado.txt
Ejecutar en segundo plano	comando &	No aplica	No aplica directamente	sleep 10 &
Ver procesos	ps o top	tasklist	Get-Process	ps aux
Matar procesos	kill	taskkill	Stop-Process	kill -9 1234
Permisos de archivo	chmod	No aplica	No aplica	chmod +x script.sh
Copiar archivos	cp archivo destino/	copy	Copy-Item	cp img.png /home/imgs/
Mover archivos	mv archivo destino/	move	Move-Item	mv viejo.txt nuevo.txt

## Guía Práctica y Advertencias de Uso

A continuación se detallan ejemplos con recomendaciones de uso seguro para cada comando:

- `rm archivo.txt` –  **Advertencia:** elimina archivos sin pasar por la papelera. Usa `-i` para confirmar: `rm -i archivo.txt`
- `chmod +x script.sh` – Cambia permisos para hacerlo ejecutable. Verifica con `ls -l` antes y después.
- `kill -9 PID` – Finaliza un proceso.  Usar solo si el proceso no responde.
- `> y >>` – Redirigen salida a un archivo. `>` sobrescribe, `>>` añade. Ej: `echo "log" > salida.txt`
- `grep "palabra" archivo` – Busca coincidencias. Usa `-i` para ignorar mayúsculas.
- `cd carpeta` – Navega entre directorios. Usa `cd ..` para ir al anterior.

- **mkdir nueva** – Crea carpetas. Usa -p para crear subdirectorios. Ej: `mkdir -p dir1/dir2`
- **cp archivo destino/** – Copia archivos. Usa -r para copiar carpetas. Ej: `cp -r carpeta/ respaldo/`
- **mv archivo destino/** – Mueve archivos o cambia nombre. Ej: `mv viejo.txt nuevo.txt`
- **alias nombre='comando'** – Crea atajos. Ej: `alias ll='ls -l'`
- **crontab -e** – Programa tareas. Usa con precaución para no sobrecargar el sistema.

**Consejo general:** Siempre prueba los comandos en un entorno de prueba o usa la opción `--help` o `man comando` antes de ejecutar comandos potencialmente destructivos.

---

### Estructura de un Script en Bash

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

```
# Comentario
```

```
nombre="Juan"
```

```
echo "Hola, $nombre"
```

Ejecutar un script

```
chmod +x mi_script.sh
```

```
./mi_script.sh
```

---

### Temas Complementarios

- **Variables de entorno:** `export PATH=$PATH:/nuevo/camino`
  - **Pipes (|):** Permiten encadenar comandos. Ej: `cat archivo | grep "dato"`
  - **Alias:** Crear comandos personalizados. Ej: `alias ll='ls -l'`
  - **Cron jobs:** Tareas programadas. Editar con `crontab -e`
  - **Redirecciones:** `2>` para errores, `&>` para todo
  - **Comodines:** `*` para varios caracteres, `?` para uno solo
- 

### Recursos de Interés

- [Explainshell.com](https://explainshell.com/): Descompone comandos Bash explicando cada parte.
  - `man comando`: Muestra el manual de un comando.
  - [SS64.com](https://ss64.com/): Comparativas entre comandos Bash, CMD y PowerShell.
-

**Conclusión:** Aprender Bash no solo permite automatizar tareas sino también entender mejor el funcionamiento interno de sistemas Unix/Linux. Compararlo con CMD o PowerShell ayuda a usuarios de Windows a transicionar hacia entornos más potentes y flexibles.