

# Sistemas Operativos

**DEV.F**  
DESARROLLAMOS(PERSONAS);

dev

# Sistema Operativo

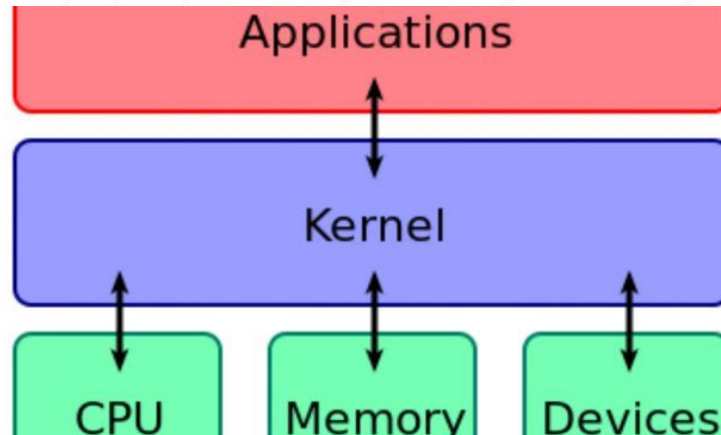
**Conjunto de programas (interfaz gráfica)** que permite manejar la memoria, disco, medios de almacenamiento de información y los diferentes periféricos o recursos de nuestra computadora, como son el teclado, el mouse, la impresora, la placa de red, entre otros.



# Kernel

También conocido como núcleo, es una **parte fundamental del sistema operativo** que **se encarga de conceder el acceso al hardware** de forma segura para todo el software que lo solicita.

- Sin el kernel el sistema operativo no podría funcionar.
- Todos los sistemas operativos tienen un Kernel.



# Servidor vs PC

## SERVIDOR

Se instala el sistema operativo **sin interfaz gráfica** (uso a base de terminal y comandos).

- **Tareas de administración del servidor.**

## PC

Se instala el sistema operativo **sistema operativo con interfaz gráfica** (uso común).

- **Tareas cotidianas.**



# ¿Por qué debo usar la terminal?



- Controlar el sistema.
- Ser programador implica configurar tu PC.
- Resolver problemas o levantar servidores.
- Agilizar algunas tareas.
- Entender más el funcionamiento de la compu.
- Entrena la capacidad de resolución de problemas.
- Me voy a dejar de bañar :)

# Terminal y Comandos

**DEV.F**  
DESARROLLAMOS(PERSONAS);

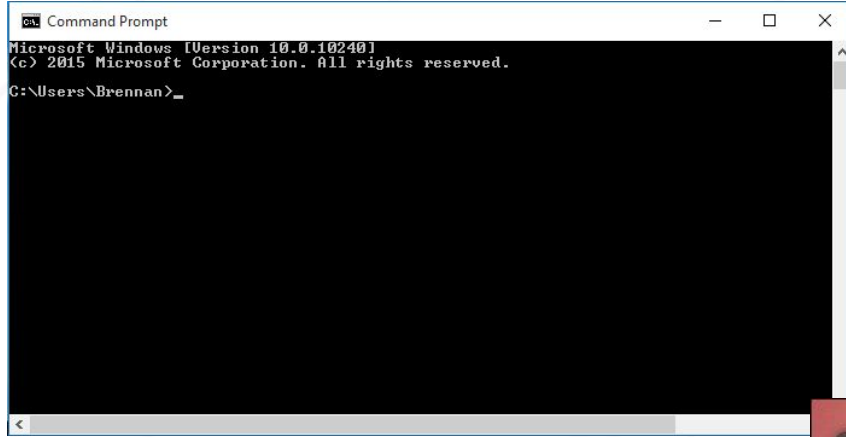
dev

# Terminal

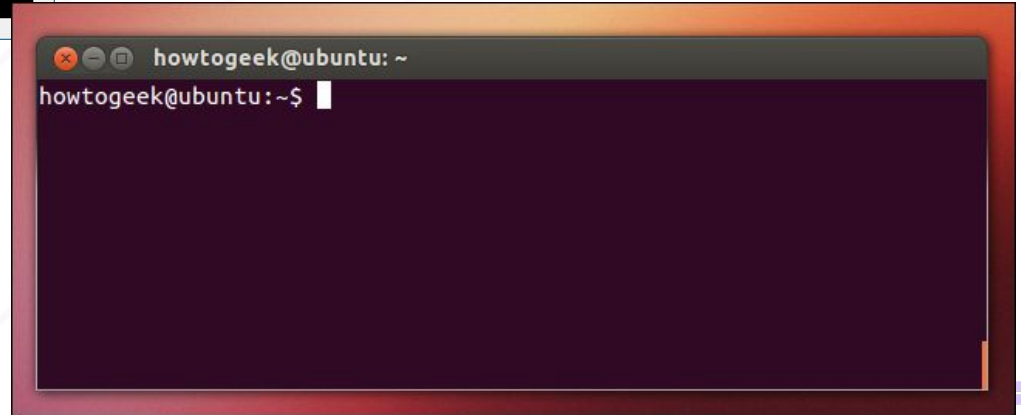
- La **terminal** o **consola** es una forma generalizada de llamar a la interfaz de **línea de comandos**: una pantalla (generalmente, de color de fondo negro sobre letras blancas).
- Al escribir **comandos** en la **CLI** con los que ordenamos al sistema realizar acciones concretas.



# Tipos de terminal



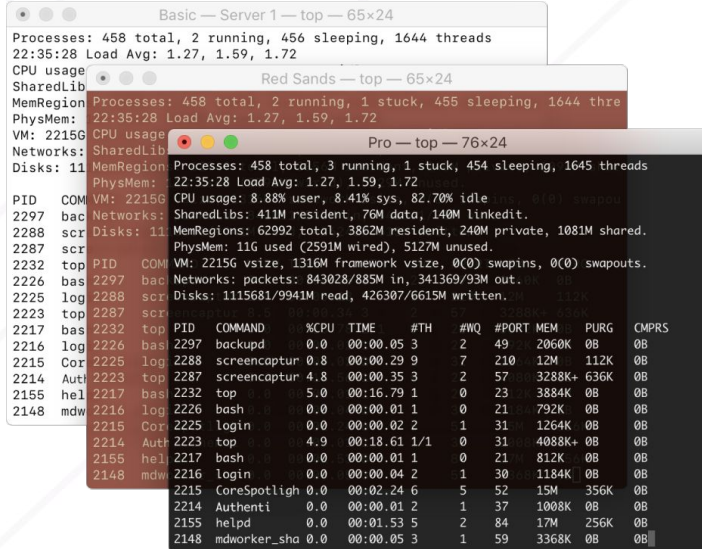
- **Windows:** La terminal se conoce como CMD.
- **Linux / OS (Mac):** La terminal se le conoce simplemente como terminal.





# Emuladores

Un **emulador de terminal** es un programa informático que **simula el funcionamiento de una terminal** de computadora en cualquier dispositivo.



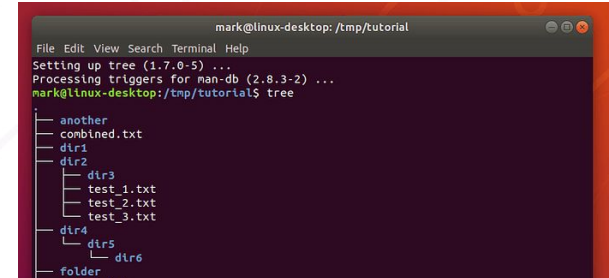
The image shows three overlapping terminal windows. The top window, titled 'Basic — Server 1 — top — 65x24', displays system statistics: 'Processes: 458 total, 2 running, 456 sleeping, 1644 threads' and '22:35:28 Load Avg: 1.27, 1.59, 1.72'. The middle window, titled 'Red Sands — top — 65x24', shows similar statistics. The bottom window, titled 'Pro — top — 76x24', displays a detailed process list with columns for PID, COMMAND, %CPU, TIME, #TH, #WQ, #PORT, MEM, PURG, and CMPRS. The list includes processes like 'backupd', 'screencaptur', 'top', 'login', and 'mdworker\_sha'.

```
Basic — Server 1 — top — 65x24
Processes: 458 total, 2 running, 456 sleeping, 1644 threads
22:35:28 Load Avg: 1.27, 1.59, 1.72
CPU usage:
SharedLib:
MemRegion:
PhysMem:
VM: 2215G
Networks:
Disks: 11

Red Sands — top — 65x24
Processes: 458 total, 2 running, 1 stuck, 455 sleeping, 1644 thre
22:35:28 Load Avg: 1.27, 1.59, 1.72
CPU usage:
SharedLib:
MemRegion:
PhysMem:
VM: 2215G
Networks:
Disks: 11

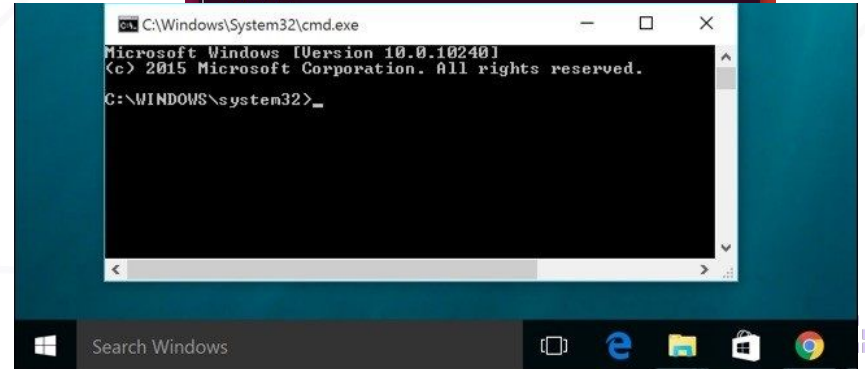
Pro — top — 76x24
Processes: 458 total, 3 running, 1 stuck, 454 sleeping, 1645 threads
22:35:28 Load Avg: 1.27, 1.59, 1.72
CPU usage: 8.88% user, 8.41% sys, 82.70% idle
SharedLibs: 411M resident, 76M data, 140M linkedit.
MemRegions: 62992 total, 3862M resident, 240M private, 1081M shared.
PhysMem: 11G used (2591M wired), 5127M unused.
VM: 2215G vszize, 1316M framework vszize, 0(0) swapis, 0(0) swapouts.
Networks: packets: 843028/885M in, 341369/93M out.
Disks: 1115681/9941M read, 426307/6615M written.

PID  COM  VM: 2215G
2297  bac   Networks:
2288  scr   Disks: 11
2287  scr   PhysMem:
2232  top   VM: 2215G vszize, 1316M framework vszize, 0(0) swapis, 0(0) swapouts.
2297  bac   Networks: packets: 843028/885M in, 341369/93M out.
2288  scr   Disks: 1115681/9941M read, 426307/6615M written.
2287  scr
2232  top
2232  top  PID  COMMAND  %CPU  TIME  #TH  #WQ  #PORT  MEM  PURG  CMPRS
2226  bas  2297  backupd  0.0   00:00.05  3    2    49    2060K  0B   0B
2225  Cor  2288  screencaptur 0.8   00:00.29  9    7    210    12M   112K  0B
2214  Aut  2223  top      4.8   00:00.35  3    2    57    3288K+ 636K  0B
2155  hel  2217  bas  2232  top      5.0   00:16.79  1    0    23    3884K  0B   0B
2148  mdw  2216  log  2226  bash     0.0   00:00.01  1    0    21    792K   0B   0B
2215  Cor  2225  login    0.0   00:00.02  2    1    31    1264K  0B   0B
2214  Aut  2223  top      4.9   00:18.61  1/1  0    31    4088K+ 0B   0B
2155  hel  2217  bash     0.0   00:00.01  1    0    21    812K   0B   0B
2148  mdw  2216  login    0.0   00:00.04  2    1    30    1184K  0B   0B
2215  CoreSpotligh 0.0   00:02.24  6    5    52    15M   356K  0B
2214  Authenti 0.0   00:00.01  2    1    37    1008K  0B   0B
2155  helpd 0.0   00:01.53  5    2    84    17M   256K  0B
2148  mdworker_sha 0.0   00:00.05  3    1    59    3368K  0B   0B
```



The image shows a terminal window titled 'mark@linux-desktop: /tmp/tutorial'. It displays the output of the 'tree' command, showing a directory structure with files and subdirectories.

```
mark@linux-desktop: /tmp/tutorial
File Edit View Search Terminal Help
Setting up tree (1.7.0-5) ...
Processing triggers for man-db (2.8.3-2) ...
mark@linux-desktop:/tmp/tutorial$ tree
.
├── another
├── combined.txt
├── dir1
├── dir2
│   ├── dir3
│   │   ├── test_1.txt
│   │   ├── test_2.txt
│   │   └── test_3.txt
│   └── dir4
│       ├── dir5
│       └── dir6
└── folder
```



The image shows a screenshot of a Windows command prompt window. The title bar reads 'C:\Windows\System32\cmd.exe'. The window content shows the Microsoft Windows [Version 10.0.10240] (c) 2015 Microsoft Corporation. All rights reserved. prompt, followed by the command 'C:\WINDOWS\system32\_'.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32_
```

# Otros emuladores de terminal

Existen otros emuladores de terminal tanto para linux como para windows, como por ejemplo:

- [Windows.](#)
- [Linux.](#)

# ¿Qué podemos hacer con la terminal?

**DEV.F**  
DESARROLLAMOS(PERSONAS);

dev

# ¿Qué podemos hacer con la terminal?

- No da un mayor **comprensión** de la **funcionalidad** del **sistema**.
- **Aprovechamiento de recursos**.
- Mayor grado de **customización del sistema** que en el entorno gráfico.
- **Seguridad** de operaciones.
- Posibilidad de **automatizar** tareas.
- Mayor capacidad de resolución de problemas técnicos del equipo.
- Nos permite interactuar con archivos de configuración, servidores y base de datos



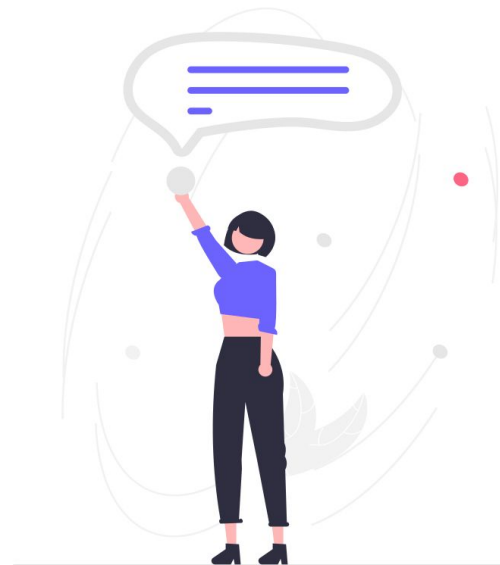
# ¿Qué es un comando?

**DEV.F**  
DESARROLLAMOS(PERSONAS);

dev

# Comandos

Un **comando**, orden o instrucción es una **indicación** que el **usuario proporciona a un sistema** informático **mediante una terminal** con la finalidad de **ejecutar una tarea**. Por ejemplo, crear un archivo, una carpeta, renombrar un archivo/carpeta, verificar la conexión a internet, etc.





**DEV.F**

# Comandos Básicos

*En Linux, MacOS y Windows*

# ¿Comandos diferentes linux vs windows?

Si la terminal tiene acceso directo al sistema y el kernel como su raíz es diferente entre sistema y sistema, es comprensible que sean diferentes los comandos que se usan para hacer una tarea en uno y otro.

## NOTA

- **MS-DOS** es el núcleo de **windows**.
- **Unix** es el núcleo de los sistemas **Linux** y **Mac OS**.





Linux / Mac OS

## **pwd** (*print working directory*)

Este comando imprime la ubicación de tu directorio de trabajo actual

Es importante saber dónde te encuentras antes de ir a un directorio principal o secundario.

`$ pwd`

Windows

## **cd** (*current directory*)

Sin pasarle ningún parámetro adicional, cd te muestra tu ubicación actual

`> cd`

También es posible usar  
`> echo %cd%`

Linux / Mac OS

## **ls** *(list)*

Imprime el contenido de un directorio.

Como casi todos los comandos, tiene variaciones como

- ls -a
- ls -l
- ls -h

**\$ ls**

Windows

## **dir** *(directory)*

Muestra todos los directorios y archivos contenidos dentro de un directorio

**> dir**

Linux / Mac OS

## **cd /directorio** *(change directory)*

Permite moverte a otro directorio

```
$ cd Documents/DEVF
```

Windows

## **cd /directorio** *(change directory)*

Si pasamos un parámetro adicional a solo escribir *cd*, podemos movernos a otro directorio

```
> cd Documents/DEVF
```

Linux / Mac OS

**clear**  
(*clear*)

Limpiado de la terminal

\$ *clear*

Windows

**cls**  
(*clear screen*)

Limpiado de la terminal

> *cls*

Linux / Mac OS

## **mkdir** (*make directory*)

El comando *mkdir* se utiliza para crear un nuevo directorio

```
$ mkdir mi_directorio
```

Windows

## **mkdir** (*make directory*)

El comando *mkdir* se utiliza para crear un nuevo directorio

```
> mkdir mi_directorio
```

Linux / Mac OS

## **rmdir** (*remove directory*)

El comando *rmdir* se utiliza para eliminar un directorio

```
$ rmdir mi_directorio
```

```
$ rm -r mi_directorio
```

Windows

## **rmdir** (*remove directory*)

El comando *rmdir* se utiliza para eliminar un directorio

```
> rd mi_directorio
```

```
> rd /s mi_directorio
```

Linux / Mac OS

## **touch**

Crear un archivo.

```
$ touch archivo.txt
```

Windows

## **notepad**

Crear un archivo.

```
> notepad archivo.txt
```

Linux / Mac OS

**rm**

Eliminar un archivo.

*\$ rm archivo.txt*

Windows

**del**

Eliminar un archivo.

*> del archivo.txt*



Linux / Mac OS

## **nano** *(concatenate)*

Se utiliza para imprimir el contenido de un archivo en la pantalla, útil cuando deseas verlo rápidamente

```
$ nano hola.html
```

Windows

## **notepad** *(change directory)*

Si pasamos un parámetro adicional a solo escribir *cd*, podemos movernos a otro directorio

```
> notepad hola.html
```

Linux / Mac OS

## **cat** (*concatenate*)

Se utiliza para imprimir el contenido de un archivo en la pantalla, útil cuando deseas verlo rápidamente

```
$ cat main.js
```

Windows

## **type** (*change directory*)

Si pasamos un parámetro adicional a solo escribir *cd*, podemos movernos a otro directorio

```
> type main.js
```

Linux / Mac OS

## **cp** (*copy*)

Sirve para copiar archivos y directorios.

```
$ cp archivo1 archivo2
```

Windows

## **copy**

Nos permite copiar archivos y directorios.

```
> copy archivo1 archivo2
```

## Linux / Mac OS

### **mv** (*move*)

El comando *mv* se usa para mover o renombrar directorios y archivos.

```
$ mv nombre1 nombre2 // renombra
```

```
$ mv nombre1 /carpeta1/carpeta2/nombre2
```

```
$ mv nombre1 nombre2
```

## Windows

### **move**

El comando *mv* se usa para mover o renombrar directorios y archivos.

```
> move nombre1 nombre2 // renombra
```

```
> mv nombre1 /carpeta1/carpeta2/nombre2
```

Linux / Mac OS

## ping

Comprobar conexiones de red.

```
$ ping www.google.com
```

Windows

## ping

Comprobar conexiones de red.

```
> ping www.google.com
```



# Prácticas

**DEV.F.**  
DESARROLLAMOS(PERSONAS);

# Práctica 1

## EJERCICIO 1

en descargas

prueba

----- documentos

----- musica

----- reggeaton

----- rock

ejercicios

----- html

index.html

hola.txt

----- css





## EJERCICIO 2

en documentos|  
devf

```
----- intro-a-la-web
      ----- redes-internet
      ----- terminal-git
            ----- comandos.txt
            ----- dudas.txt
----- intro-a-javascript
      ----- logica-programacion
      ----- sintaxis-javascript
```

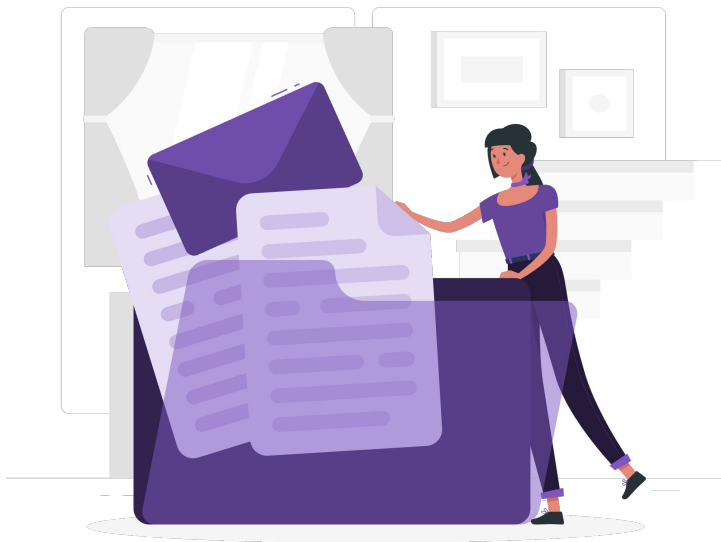


# Documentación

**DEV.F**  
DESARROLLAMOS(PERSONAS);

dev

# Documentación



- [Comandos Linux.](#)
- [Comandos Windows.](#)
- [Guia de comandos en Linux.](#)
- [Curso de la terminal.](#)
- [Porque un programador debe usar consola.](#)

# Notas

**DEV.F**  
DESARROLLAMOS(PERSONAS);

dev

- Con la tecla tab autocompleta texto.
- Con las flechas arriba y abajo vemos comandos previos.
- Se recomienda crear archivos y carpetas sin espacios (**kebab-case**).
- Los comandos suele necesitar de banderas y parámetros.
- Los comandos son case sensitive.

## Nomenclaturas de escritura sin espacios

camelCase  
snake\_case  
kebab-case  
Train-Case