

# docker 1

Marcos Miquel Lisarde

## Índice

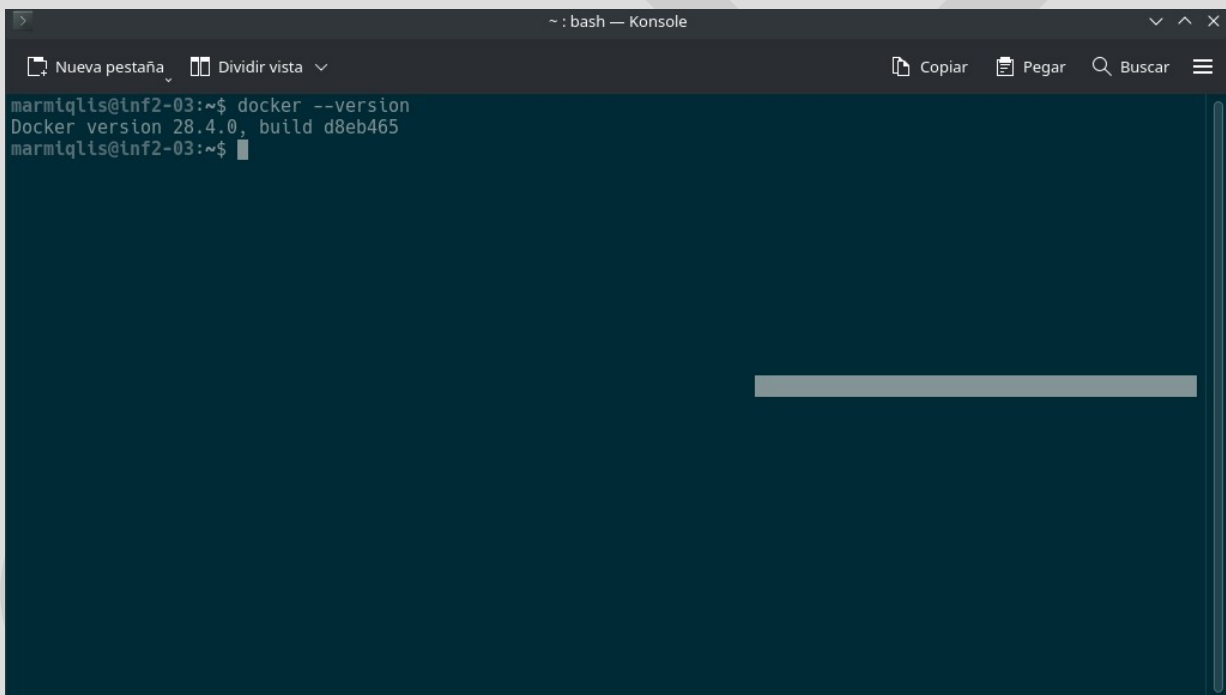
### Sumario

Ejercicio 1: Verificar la instalación de Docker.....	3
Ejercicio 2: Ejecutar el contenedor hello-world con un nombre personalizado.....	3
Ejercicio 3: Listar los contenedores (incluso los detenidos).....	4

## Ejercicio 1: Verificar la instalación de Docker

1. **Objetivo:** Comprobar si Docker está instalado y funcionando correctamente en tu sistema.
2. **Pista:** Busca el comando que te permita verificar la versión instalada de Docker.
3. **Comprobación:** Debes ver la versión actual de Docker en la salida del comando.

### Solución:

A screenshot of a terminal window titled '~ : bash — Konsole'. The terminal shows the command 'docker --version' being executed, resulting in the output 'Docker version 28.4.0, build d8eb465'. The terminal interface includes a top bar with window controls and a menu bar with options like 'Nueva pestaña', 'Dividir vista', 'Copiar', 'Pegar', and 'Buscar'. The background of the terminal is dark blue with a light blue horizontal bar at the bottom.

```
marmiqlis@inf2-03:~$ docker --version
Docker version 28.4.0, build d8eb465
marmiqlis@inf2-03:~$
```

## Ejercicio 2: Ejecutar el contenedor hello-world con un nombre personalizado

1. **Objetivo:** Ejecutar el contenedor de "hello-world" y asignarle un nombre para identificarlo fácilmente.
2. **Pista:** Usa el parámetro que te permita nombrar el contenedor al ejecutarlo.
3. **Comprobación:** Al ejecutar el comando, deberás ver un mensaje de bienvenida desde Docker.

### Solución:

## docker\_1

```
marmiqlls@inf2-03:~$ docker run --name mi_hola_docker hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

## Ejercicio 3: Listar los contenedores (incluso los detenidos)

1. **Objetivo:** Mostrar una lista de todos los contenedores, incluyendo los que ya se han detenido.
2. **Pista:** Usa el comando adecuado con el parámetro que permita mostrar contenedores detenidos.
3. **Comprobación:** El contenedor mi\_hola\_mundo debe aparecer como "Exited" en la lista.

### Solución:

```
marmiqlls@inf2-03:~$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	N
AMES						
5d5b40f21645	hello-world	"/hello"	54 seconds ago	Exited (0) 53 seconds ago		m
mi_hola_docker						
85fd9dac7855	hello-world	"/hello"	2 days ago	Exited (0) 2 days ago		z
en_yonath						
ea499eb36f03	mysql:8.0	"docker-entrypoint.s..."	8 days ago	Exited (0) 23 hours ago		a
dd-dbms						

## Ejercicio 4: Ejecutar un contenedor en segundo plano con -d

1. **Objetivo:** Ejecutar un contenedor de "nginx" en segundo plano (sin mostrar la salida en el terminal).
2. **Pista:** Busca el parámetro que permita ejecutar el contenedor en modo "detached".
3. **Comprobación:** Usa un comando para listar los contenedores en ejecución y asegúrate de que mi\_nginx está en la lista.

### Solución:

```
marmiqlls@inf2-03:~$ docker run -d --name mi_nginx nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
38513bd72563: Pull complete
10d18f46ee87: Pull complete
a8d825a0683a: Pull complete
a131bc1d4bd5: Pull complete
3818929ac19f: Pull complete
1498b1cfda15: Pull complete
c50c84d0ed4d: Pull complete
Digest: sha256:029d4461bd98f124e531380505ceea2072418fdf28752aa73b7b273ba3048903
Status: Downloaded newer image for nginx:latest
47645cbd37b670bc743987c27069a0b0efa60edd8699076f1768fae9b843eba9
marmiqlls@inf2-03:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
47645cbd37b6	nginx	"/docker-entrypoint..."	16 seconds ago	Up 16 seconds	80/tcp	mi_nginx

## Ejercicio 5: Ejecutar un contenedor interactivo con -it

1. **Objetivo:** Ejecutar un contenedor de "ubuntu" de forma interactiva, accediendo a una consola dentro del contenedor.
2. **Pista:** Busca los parámetros que te permitan interactuar con el contenedor en tiempo real.
3. **Comprobación:** Deberás estar dentro de la consola del contenedor de Ubuntu.

### Solución:

```
marmiqlls@inf2-03:~$ docker run -it --name mi_ubuntu ubuntu /bin/bash
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
4b3ffd8ccb52: Pull complete
Digest: sha256:66460d557b25769b102175144d538d88219c077c678a49af4afca6fbfc1b5252
Status: Downloaded newer image for ubuntu:latest
root@bd0f9a7e628a:/#
```

## Ejercicio 6: Descargar una imagen sin ejecutarla

1. **Objetivo:** Descargar una imagen del repositorio Docker Hub sin ejecutarla directamente.
2. **Pista:** Busca el comando que permite descargar imágenes sin ejecutar el contenedor.
3. **Comprobación:** Verifica que la imagen está en el sistema con un comando que muestre las imágenes descargadas.

**Solución:**

```
marmiqlis@inf2-03:~$ docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
Digest: sha256:66460d557b25769b102175144d538d88219c077c678a49af4afca6fbfc1b5252
Status: Image is up to date for ubuntu:latest
docker.io/library/ubuntu:latest
marmiqlis@inf2-03:~$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
nginx	latest	657fdcd1c365	2 weeks ago	152MB
ubuntu	latest	97bed23a3497	3 weeks ago	78.1MB
mysql	8.0	94753e67a0a9	4 weeks ago	780MB
hello-world	latest	1b44b5a3e06a	2 months ago	10.1kB

## Ejercicio 7: Verificar los logs de un contenedor

1. **Objetivo:** Ver los logs generados por el contenedor mi\_nginx.
2. **Pista:** Usa el comando que permite visualizar los logs de un contenedor en ejecución o ya detenido.
3. **Comprobación:** Deberás ver los logs del servidor nginx.

**Solución:**

```
marmiqlis@inf2-03:~$ docker logs mi_nginx
/docker-entrypoint.sh: /docker-entrypoint.d/ is not
/docker-entrypoint.sh: Looking for shell scripts in
/docker-entrypoint.sh: Launching /docker-entrypoint.
10-listen-on-ipv6-by-default.sh: info: Getting the c
10-listen-on-ipv6-by-default.sh: info: Enabled liste
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d
/docker-entrypoint.sh: Launching /docker-entrypoint.
/docker-entrypoint.sh: Launching /docker-entrypoint.
/docker-entrypoint.sh: Configuration complete, ready
```

## Ejercicio 8: Eliminar un contenedor detenido

1. **Objetivo:** Eliminar el contenedor mi\_hola\_mundo que ya está detenido.
2. **Pista:** Usa el comando que permite eliminar contenedores con el nombre o ID.
3. **Comprobación:** Verifica que el contenedor ya no aparece en la lista de contenedores, incluso con los detenidos.

### Solución:

```
marmiqlis@inf2-03:~$ docker rm mi_hola_docker
mi_hola_docker
marmiqlis@inf2-03:~$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND
ES		
f122d7c97350	nginx:latest	"/docker-entrypo
antic_williamson		
bd0f9a7e628a	ubuntu	"/bin/bash"
ubuntu		
47645cbd37b6	nginx	"/docker-entrypo
nginx		
85fd9dac7855	hello-world	"/hello"
_yonath		
ea499eb36f03	mysql:8.0	"docker-entrypoi
-dbms		

## Ejercicio 9: Crear un contenedor con puertos mapeados

1. **Objetivo:** Ejecutar un contenedor de "httpd" y mapear su puerto 80 al puerto 8080 del host.
2. **Pista:** Usa el parámetro adecuado para mapear puertos entre el host y el contenedor.
3. **Comprobación:** Accede a <http://localhost:8080> en tu navegador y verifica que el servidor está funcionando.

### Solución:

```
marmiqlis@inf2-03:~$ docker run -d --name mi_httpd -p 8080:80 httpd
Unable to find image 'httpd:latest' locally
latest: Pulling from library/httpd
38513bd72563: Already exists
a6c97c1311d6: Pull complete
4f4fb700ef54: Pull complete
37be031b3615: Pull complete
359a248d4bde: Pull complete
72ba0317f875: Pull complete
Digest: sha256:d3b88ca0822f91e2dec6eb58a2ac7cfade27880926467fc63dcd857010b083
Status: Downloaded newer image for httpd:latest
f01062ea292f016811909ed50cd4b608b3ed263076b6d0868e96361469467fa8
marmiqlis@inf2-03:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
f01062ea292f	httpd	"httpd-foreground"	4 seconds ago	Up 3 seconds	0.0.0.0:8080->80/tcp
:]:8080->80/tcp	mi_httpd				
f122d7c97350	nginx:latest	"/docker-entrypoint...."	4 minutes ago	Up 4 minutes	80/tcp
	romantic_williamson				

## Ejercicio 10: Listar solo las imágenes en el sistema

1. **Objetivo:** Mostrar todas las imágenes de Docker descargadas en el sistema.
2. **Pista:** Usa el comando que permite listar imágenes de Docker.
3. **Comprobación:** Deberás ver una lista con las imágenes, incluyendo nginx y httpd.

### Solución:

```
marmiqlis@inf2-03:~$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
nginx	latest	657fdcd1c365	2 weeks ago	152MB
ubuntu	latest	97bed23a3497	3 weeks ago	78.1MB
mysql	8.0	94753e67a0a9	4 weeks ago	780MB
hello-world	latest	1b44b5a3e06a	2 months ago	10.1kB
httpd	latest	4613a77dcb46	2 months ago	117MB



## Ejercicio 11: Detener un contenedor en ejecución

1. **Objetivo:** Detener el contenedor mi\_nginx que está en ejecución.
2. **Pista:** Usa el comando que permite detener contenedores en ejecución.
3. **Comprobación:** Verifica que mi\_nginx ya no aparece en la lista de contenedores en ejecución.

### Solución:

```
marmiglts@inf2-03:~$ docker stop mi_nginx
mi_nginx
marmiglts@inf2-03:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
f122d7c97350	nginx:latest	"/docker-entrypoint..."	12 minutes ago	Up 12 minutes	80/tcp	romantic_wil
liamson						

## Ejercicio 12: Reiniciar un contenedor detenido

1. **Objetivo:** Reiniciar el contenedor mi\_nginx que detuviste previamente.
2. **Pista:** Busca el comando que permite reiniciar un contenedor que ya ha sido detenido.
3. **Comprobación:** El contenedor mi\_nginx debe aparecer nuevamente en la lista de contenedores en ejecución.

### Solución:

```
marmiglts@inf2-03:~$ docker start mi_nginx
mi_nginx
marmiglts@inf2-03:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
f122d7c97350	nginx:latest	"/docker-entrypoint..."	14 minutes ago	Up 14 minutes	80/tcp	romantic_wil
liamson						
47645cbd37b6	nginx	"/docker-entrypoint..."	3 days ago	Up 1 second	80/tcp	mi_nginx

## Ejercicio 13: Ver estadísticas en tiempo real de un contenedor

1. **Objetivo:** Ver las estadísticas en tiempo real de consumo de recursos del contenedor mi\_nginx.
2. **Pista:** Busca el comando que te permite monitorear en tiempo real el uso de CPU y memoria de un contenedor.
3. **Comprobación:** Deberás ver estadísticas de uso en tiempo real en el terminal.

### Solución:

```
marmiglts@inf2-03:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	PIDS
CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O
47645cbd37b6	mi_nginx	0.00%	10.44MiB / 15.39GiB	0.07%	3.27kB / 126B	0B / 4.1kB

## Ejercicio 15: Crear un contenedor con un volumen montado

1. **Objetivo:** Crear un contenedor de "nginx" y montar un volumen local en el contenedor.
2. **Pista:** Usa el parámetro que permite montar volúmenes locales en el contenedor.
3. **Comprobación:** Modifica un archivo en el directorio del host y verifica que el cambio se refleja en el contenedor.

### Solución:

```
marmiglts@inf2-03:~$ docker run -d --name mi_nginx_volumen -v ./html:/usr/share/nginx/html nginx
cb526dc20f2dc2460da029080e58075a02d3e99151d9a0420aaa11eea4d771cd
marmiglts@inf2-03:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
cb526dc20f2d	nginx	"/docker-entrypoint..."	5 minutes ago	Up 5 minutes	80/tcp	mi_nginx_volumen
f122d7c97350	nginx:latest	"/docker-entrypoint..."	24 minutes ago	Up 24 minutes	80/tcp	romantic_williamson
47645cbd37b6	nginx	"/docker-entrypoint..."	3 days ago	Up 10 minutes	80/tcp	mi_nginx

## Ejercicio 16: Crear un contenedor con una variable de entorno

1. **Objetivo:** Pasar una variable de entorno al contenedor nginx\_env.
2. **Pista:** Busca el parámetro que permite pasar variables de entorno a los contenedores.
3. **Comprobación:** Usa un comando para inspeccionar el contenedor y verificar que la variable de entorno se ha establecido correctamente.

### Solución:

```
marmiqlis@inf2-03:~$ docker run -d --name nginx_env -e MI_VARIABLE=valor nginx
e7d14ae7626e67caa450ee0fe57821f6d1907e1abea11c6fef3302cb625a3391
marmiqlis@inf2-03:~$ docker inspect nginx_env
[
  {
    "Id": "e7d14ae7626e67caa450ee0fe57821f6d1907e1abea11c6fef3302cb625a3391",
    "Created": "2025-10-28T08:50:33.21672174Z",
    "Path": "/docker-entrypoint.sh",
    "Args": [
      "nginx",
      "-g",
      "daemon off;"
    ],
    "State": {
      "Status": "running",
      "Running": true
```

## Ejercicio 17: Crear una red personalizada para contenedores

1. **Objetivo:** Crear una red personalizada y ejecutar el contenedor nginx\_red en ella.
2. **Pista:** Busca el comando para crear una red y el parámetro que asocia contenedores a redes.
3. **Comprobación:** Verifica que el contenedor está conectado a la red personalizada usando el comando de inspección.

### Solución:

```
marmiqlis@inf2-03:~$ docker network create mi_red_personalizada
d1c2c8c82745746857cc1dc52dc162a0181b344448885fa90d3869b3c76e58d6
marmiqlis@inf2-03:~$ docker run -d --name nginx_red --network mi_red_personalizada nginx:latest
2b024de4e7e1c38ea2034264b97969b6e381d24f10df2a6ff7b4e5d7088e72a8
marmiqlis@inf2-03:~$ docker inspect nginx_red
[
  {
    "Id": "2b024de4e7e1c38ea2034264b97969b6e381d24f10df2a6ff7b4e5d7088e72a8",
    "Created": "2025-10-28T08:51:56.306315378Z",
    "Path": "/docker-entrypoint.sh",
    "Args": [
      "nginx",
      "-g",
      "daemon off;"
    ],
    "State": {
      "Status": "running",
```

## Ejercicio 18: Conectar un contenedor a una red existente

1. **Objetivo:** Conectar el contenedor mi\_nginx a la red mi\_red creada en el ejercicio anterior.
2. **Pista:** Usa el comando que permite conectar contenedores a redes ya existentes.
3. **Comprobación:** Verifica que mi\_nginx está conectado a la red mi\_red.
4. **Solución:**

```
marmiqlis@inf2-03:~$ docker network connect mi_red_personalizada mi_nginx
marmiqlis@inf2-03:~$ docker inspect mi_nginx
[
  {
    "Id": "47645cbd37b670bc743987c27069a0b0efa60edd8699076f1768fae9b843eba9",
    "Created": "2025-10-24T11:24:15.387759358Z",
    "Path": "/docker-entrypoint.sh",
    "Args": [
      "nginx",
      "-g",
      "daemon off;"
    ],
    "State": {
      "Status": "running",
      "Running": true,
      "Paused": false,
      "Restarting": false,
      "OOMKilled": false,
      "Dead": false,
      "Pid": 10201,
      "ExitCode": 0,
      "Error": "",
      "StartedAt": "2025-10-28T08:37:44.13802486Z",
      "FinishedAt": "2025-10-28T08:36:38.057266842Z"
```

## Ejercicio 19: Ejecutar un contenedor temporal con --rm

1. **Objetivo:** Ejecutar un contenedor temporal que se elimine automáticamente al finalizar.
2. **Pista:** Busca el parámetro que permite eliminar un contenedor automáticamente después de que termine.
3. **Comprobación:** Verifica que el contenedor no aparece en la lista de contenedores después de su ejecución.
4. **Solución:**

```
marmiglis@inf2-03:~$ docker run --rm --name temporal_ubuntu ubuntu echo "Contenedor temporal ejecutado"
Contenedor temporal ejecutado
marmiglis@inf2-03:~$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
2b024de4e7e1	nginx:latest	"/docker-entrypoint..."	14 minutes ago	Up 14 minutes	80/tcp
nginx_red	nginx	"/docker-entrypoint..."	15 minutes ago	Up 15 minutes	80/tcp
e7d14ae7626e	nginx	"/docker-entrypoint..."	23 minutes ago	Up 23 minutes	80/tcp
nginx_env	nginx	"/docker-entrypoint..."	38 minutes ago	Exited (0) 30 minutes ago	
cb526dc20f2d	httpd	"httpd-foreground"	42 minutes ago	Up 42 minutes	80/tcp
mi_nginx_volumen	nginx:latest	"/docker-entrypoint..."	3 days ago	Up 28 minutes	80/tcp
f01062ea292f	ubuntu	"/bin/bash"	6 days ago	Exited (0) 6 days ago	
mi_httpd	nginx	"/docker-entrypoint..."	11 days ago	Exited (0) 4 days ago	
f122d7c97350	hello-world	"/hello"			
romantic_williamson	mysql:8.0	"docker-entrypoint.s..."			
bd0f9a7e628a					
mi_ubuntu					
47645cbd37b6					
mi_nginx					
85fd9dac7855					
zen_yonath					
ea499eb36f03					
add-dbms					

## Ejercicio 20: Eliminar todas las imágenes no utilizadas

1. **Objetivo:** Eliminar todas las imágenes no utilizadas del sistema.
2. **Pista:** Busca el comando que permite eliminar imágenes "dangling" o no asociadas a contenedores en ejecución.
3. **Comprobación:** Verifica que no quedan imágenes innecesarias en el sistema.
4. **Solución:**

```
marmiglis@inf2-03:~$ docker image prune -a
WARNING! This will remove all images without at least one container associated to them.
Are you sure you want to continue? [y/N] y
Total reclaimed space: 0B
marmiglis@inf2-03:~$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
nginx	latest	657fdcd1c365	2 weeks ago	152MB
ubuntu	latest	97bed23a3497	3 weeks ago	78.1MB
mysql	8.0	94753e67a0a9	4 weeks ago	780MB
hello-world	latest	1b44b5a3e06a	2 months ago	10.1kB
httpd	latest	4613a77dcb46	2 months ago	117MB

SECRET