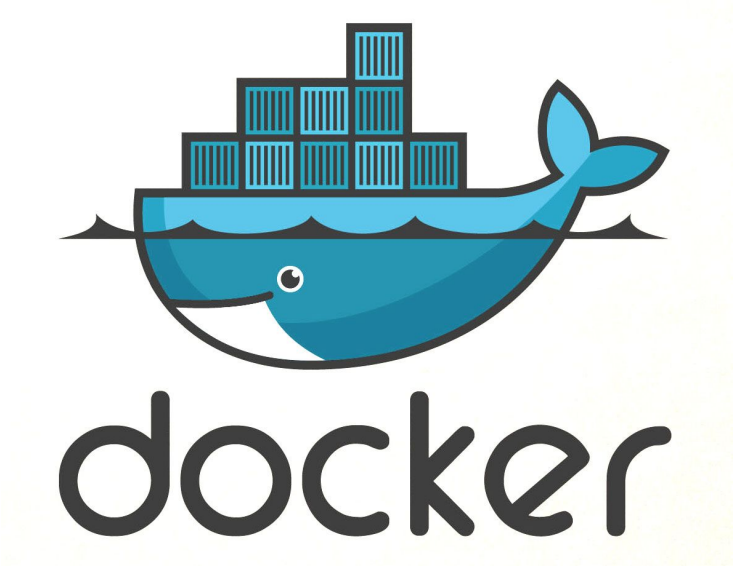
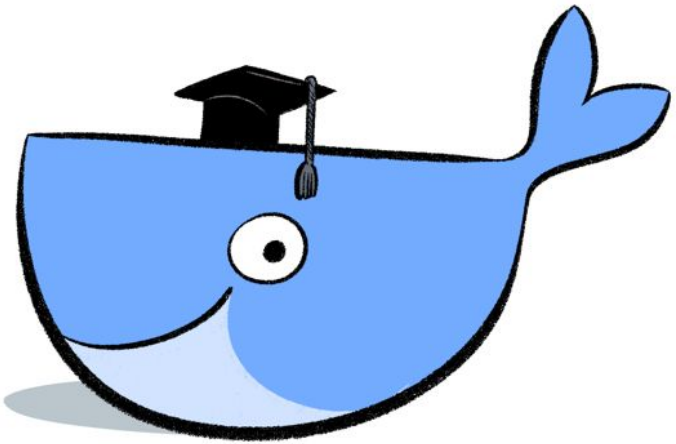
Almacenamiento



Contenido

1. Persistencia de datos
2. *Bind mount*
3. Volúmenes

Persistencia de datos



Persistencia de datos

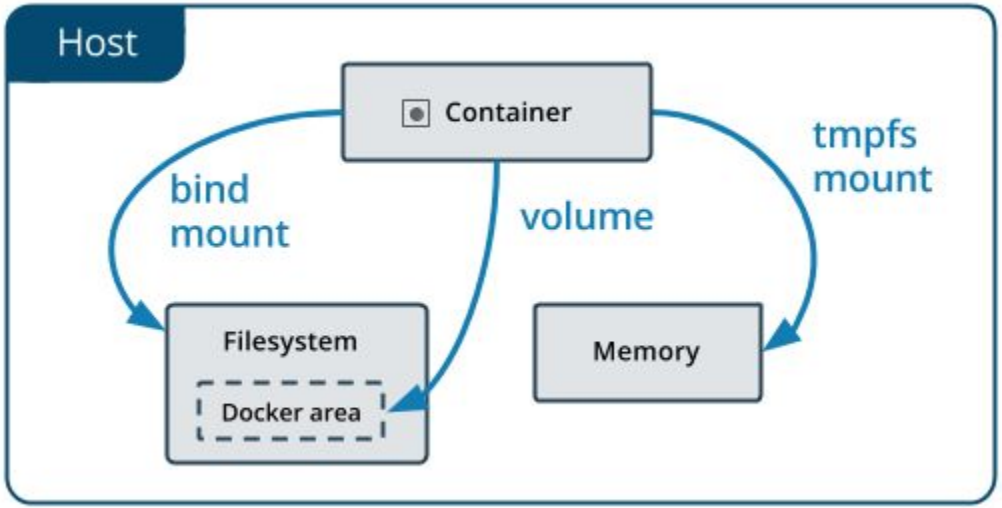
Los contenedores son **efímeros**; es decir, su configuración y sus datos sobreviven a paradas, pero se borran cuando se elimina el contenedor.

Para persistir los datos, Docker proporciona 2 sistemas:

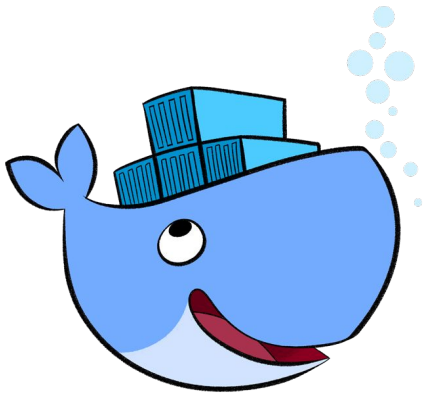
- ***Bind mount***
- **Volúmenes**

Persistencia de datos

También existe el **tmpfs mount**, para montar un sistema de ficheros en memoria, pero no lo vamos a ver.



Bind mount



Bind mount

Consiste en **montar directorios o ficheros** del host en el contenedor, de modo que ambos tienen acceso. Si el destino tenía datos, dejarán de ser accesibles.

Se establece **al crear el contenedor**, con opciones de docker run o create.

Require **rutas absolutas**.

Tienen 2 sintaxis:

- Simple: usando la opción **-v** o **--volume**
- Compleja: usando la opción **--mount**

Bind mount - Sintaxis simple

-v *ruta_host:ruta_contenedor:opciones*

Si **ruta_host** es un directorio y no existe, se crea.

opciones es opcional. La única que vamos a ver es **ro**, que sgnifica **read-only**.

Bind mount - Sintaxis simple - Ejemplo

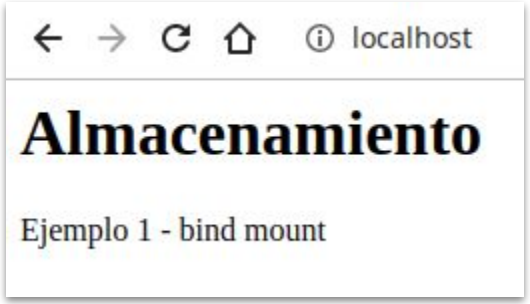
Vamos a lanzar un contenedor Nginx que muestre una web que está en un directorio del host.

```
~/Documentos/docker-examples/06-ejemplo01
➜  tree
.
└── html
    └── index.html
```

```
~/Documentos/docker-examples/06-ejemplo01
➜  \cat html/index.html
<h1>Almacenamiento</h1>
<p>Ejemplo 1 - bind mount</p>
```

```
~/Documentos/docker-examples/06-ejemplo01
➜  docker run -d -p 80:80 -v ~/Documentos/docker-examples/06-ejemplo01/html:/usr/share/nginx/html nginx
8b49c8abc731f22a99f76b6505dffbbcc3ff6c029015af43e6fc4fe086e9d0d6
```

Bind mount - Sintaxis simple - Ejemplo



Si cambiamos el archivo index.html desde el host, el contenedor lo ve.

```
~/Documentos/docker-examples/06-ejemplo01
→  \cat html/index.html
<h1>Almacenamiento</h1>
<p>Ejemplo 1 - bind mount</p>
<p>Si el archivo cambia, se ve</p>
```

Bind mount - Sintaxis simple - Ejemplo

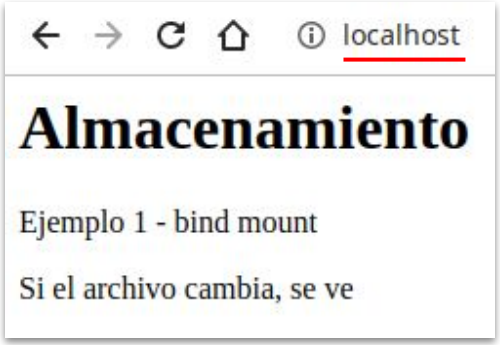
En este caso, como Nginx solo lee el archivo, sería más seguro montarlo con **ro**:

```
~/Documentos/docker-examples/06-ejemplo01
➜  docker run -d -p 80:80 -v ~/Documentos/docker-examples/06-ejemplo01/html:/usr/share/nginx/html:ro nginx
bca1e32d2f72f7e88bc5ca36afafc40ed3e4e31c0bc66ea99c6e4cbeb0911507
```

También se podría crear otro contenedor con el mismo directorio montado:

```
~/Documentos/docker-examples/06-ejemplo01
➜  docker run -d -p 8080:80 -v ~/Documentos/docker-examples/06-ejemplo01/html:/usr/share/nginx/html nginx
420e325750924df2aaa7ac0a0f166a072b72215f7ae0e0b6632311c11469bb13
```

Bind mount - Sintaxis simple - Ejemplo




```
~/Documentos/docker-examples/06-ejemplo01
➜  docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS                                   NAMES
420e32575092   nginx     "/docker-entrypoint.…"   3 minutes ago   Up 3 minutes   0.0.0.0:8080->80/tcp, :::8080->80/tcp   happy_franklin
bca1e32d2f72   nginx     "/docker-entrypoint.…"   6 minutes ago   Up 6 minutes   0.0.0.0:80->80/tcp, :::80->80/tcp       busy_zhukovsky
```

Bind mount - Sintaxis compleja

Usa la opción **--mount** de docker run o create.

No la vamos a ver en detalle, sino en comparación con el ejemplo de la simple.

Simple:

```
-v ~/Documentos/docker-examples/06-ejemplo01/html:/usr/share/nginx/html
```

Complejo:

```
--mount type=bind,src=/home/pepe/Documentos/docker-examples/06-ejemplo01/html,dst=/usr/share/nginx/html
```

Bind mount - Sintaxis compleja

Simple solo lectura:

```
-v ~/Documentos/docker-examples/06-ejemplo01/html:/usr/share/nginx/html:ro
```

Complejo solo lectura:

```
--mount type=bind,src=/home/pepe/Documentos/docker-examples/06-ejemplo01/html,dst=/usr/share/nginx/html,readonly
```

Ejercicio 1

Ejercicio 1

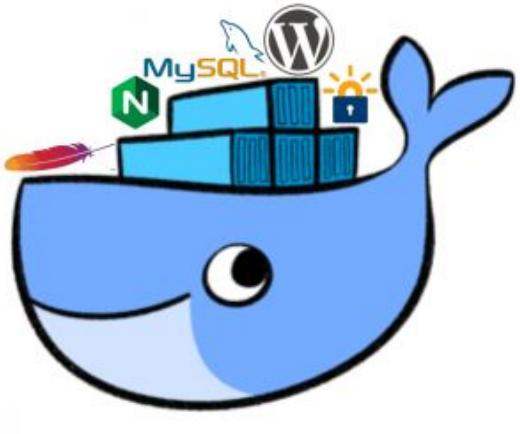
1. Crea un contenedor a partir de la imagen oficial de NextCloud usando *bind mount* para persistir los datos en una carpeta llamado “datos”. Selecciona como base de datos SQLite durante la instalación.
2. Sube algún archivo.
3. Comprueba desde el host que el archivo está dentro de la carpeta “datos”.
4. Luego para el contenedor y bórralo (el contenedor, no el archivo).
5. Finalmente crea otro contenedor usando el mismo directorio “datos” que has usado en el primero y comprueba que el archivo que has subido sigue ahí en la interfaz de NextCloud.

Ejercicio 2

Ejercicio 2

1. Crea un contenedor a partir de la imagen oficial de MySQL usando *bind mount* para persistir las bases de datos en una carpeta llamado “db”.
2. Lanza un contenedor phpmyadmin que se conecte al MySQL y crea una base de datos.
3. Para y elimina los contenedores.
4. Vuele a hacer los pasos 1 y 2 y comprueba que la base de datos ha persistido.

Volúmenes



Volúmenes

Los volúmenes son similares a los bind mounts, pero los **gestiona Docker**, por lo que **no** es necesario especificar la **ruta**.

En cada sistema se almacenan en un sitio. Por ejemplo, en Linux están en /var/lib/docker/volumes.

Son entidades propias. Se pueden **crear, listar, eliminar e inspeccionar**.

Tienen las mismas 2 sintaxis que los bind mounts, pero usando el nombre del volumen:

- Simple: usando la opción **-v** o **--volume**
- Compleja: usando la opción **--mount**

Volúmenes vs *bind mounts*

Volúmenes

- Se referencian por **nombre**.
- La ruta **no** importa.
- Pensados para la **comunicación entre contenedores**.
- Si el destino en el contenedor tiene datos y el volumen está vacío, **los datos se copian al volumen** al montarse.

Bind mounts

- Se referencian por **ruta**.
- La ruta es **importante**.
- Pensados para la **comunicación entre el host y los contenedores**.
- Si el destino en el contenedor tiene datos, al montar el origen esos datos quedan **inaccesibles**.

Volúmenes - Comandos

docker volume *comando*

create *nombre*	Crea un volumen
ls	Lista todos los volúmenes
rm *nombre*	Elimina un volumen
prune	Elimina los volúmenes no usados por ningún contenedor
inspect	Proporciona información sobre el volumen

Volúmenes - Comandos

docker volume create *nombre*

```
→  docker volume create web
web
```

```
→  docker volume create database
database
```

Volúmenes - Comandos

docker volume ls

```
➜  docker volume ls
DRIVER    VOLUME NAME
local     database
local     web
```

Volúmenes - Comandos

docker volume inspect name

```
→  docker volume inspect database
[
    {
        "CreatedAt": "2022-04-11T17:54:48+02:00",
        "Driver": "local",
        "Labels": {},
        "Mountpoint": "/var/lib/docker/volumes/database/_data",
        "Name": "database",
        "Options": {},
        "Scope": "local"
    }
]
```

Volúmenes - Comandos

docker volume rm name

```
➜  docker volume rm database
database
```

```
➜  docker volume ls
DRIVER    VOLUME NAME
local     web
```

Volúmenes - Comandos

docker volume prune

```
→  docker volume prune
WARNING! This will remove all local volumes not used by at least one container.
Are you sure you want to continue? [y/N] y
Deleted Volumes:
web

Total reclaimed space: 0B
```

```
→  docker volume ls
DRIVER    VOLUME NAME
```

Volúmenes - Sintaxis compleja

```
→  docker run -d -p 80:80 --mount type=volume,src=web2,dst=/usr/share/nginx/html nginx
0d0a304dc6195886d71d5e77879a945542081d399ec8dc5a45910c0772042f99
```

Si el volumen no existe, se crea automáticamente.

```
→  docker volume ls
DRIVER    VOLUME NAME
local     web
local     web2
```

Si se omite el src, se crea un volumen anónimo.

```
→  docker volume ls
DRIVER    VOLUME NAME
local     ffbe44ea68044c29066cbcaed7b881c7ba980d68b865
local     web
local     web2
```

Ejercicio 1

Ejercicio 1

Haz el ejercicio 1 de bind mount usando ahora volúmenes.

Ejercicio 2

Ejercicio 2

Haz el ejercicio 2 de bind mount usando ahora volúmenes.

Ejercicio 3

Ejercicio 3

A partir del ejercicio anterior, en que la persistencia de la base de datos mysql se está haciendo en un volumen, se pide crear un contenedor que haga un archivo tar.gz con el contenido del volumen en un directorio del host llamado backup.

El contenedor tienen que borrarse después de hacer la copia.