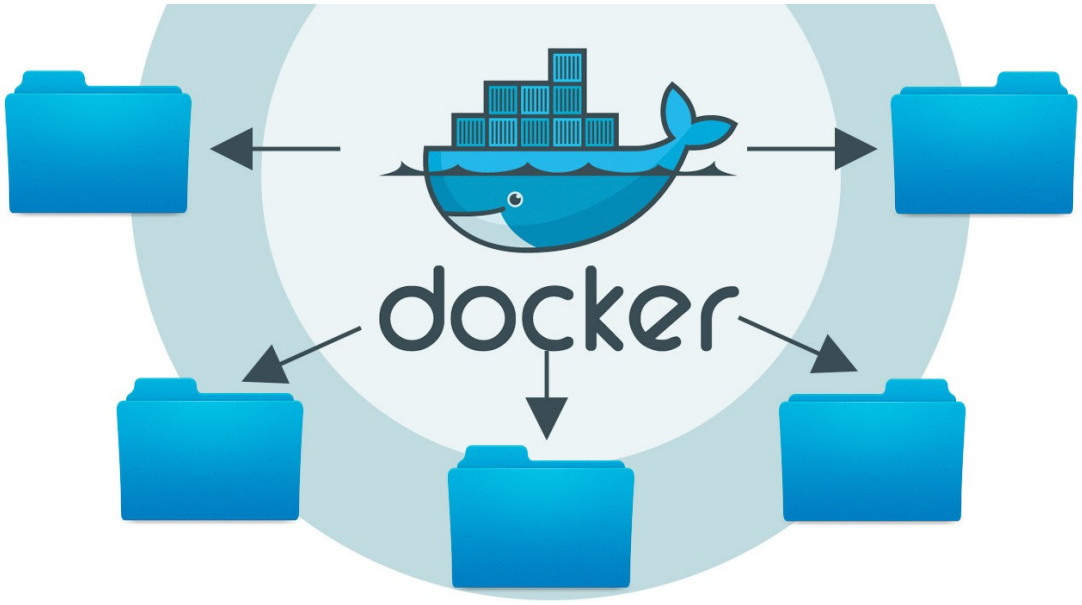
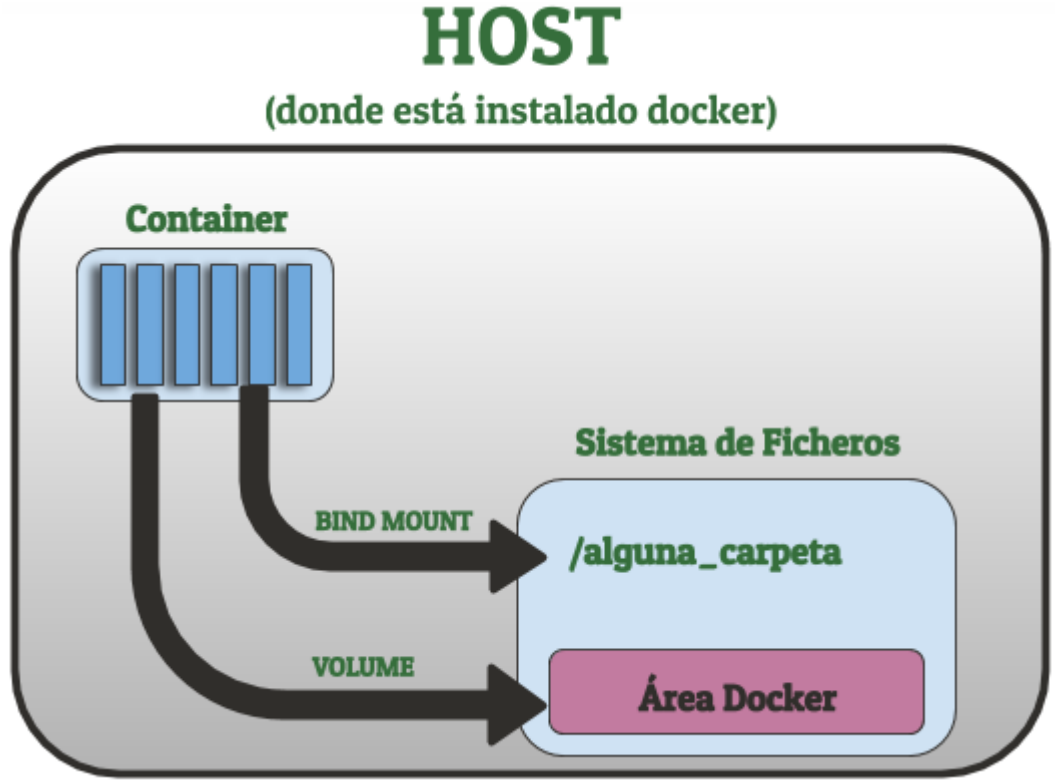



Jose Antonio Castillejo Lobato

En este módulo del curso vamos a adentrarnos en la persistencia de los datos de los contenedores docker. Trataremos lo siguientes aspectos:

- **Necesidad** de persistir los datos de los contenedores.
- **Formas de gestionar** esa persistencia (Volúmenes y Bind Mounts).
- **Operaciones** para la **gestión** de volúmenes y para la **obtención de información** de los mismos.
- Cómo **asociar volúmenes o bind mounts a nuestros contenedores**.
- Uso de la persistencia de los datos como **copia de seguridad**.
- **Compartición de datos** entre distintos contenedores.
- **Depuración de aplicaciones usando bind mounts**.

- Los **datos de un contenedor mueren con él**.
- Los datos de los contenedores **no se mueven fácilmente** ya que están fuertemente acoplados con el host en el que el contenedor está ejecutándose.
- **Escribir** en los contenedores **es más lento que** escribir en el **host** ya que tenemos una capa adicional.




# VOLÚMENES DOCKER

Esa "ZONA RESERVADA" de docker cambia de un sistema operativo a otro y también puede cambiar dependiendo de la forma de instalación, pero de manera general podemos decir que es:

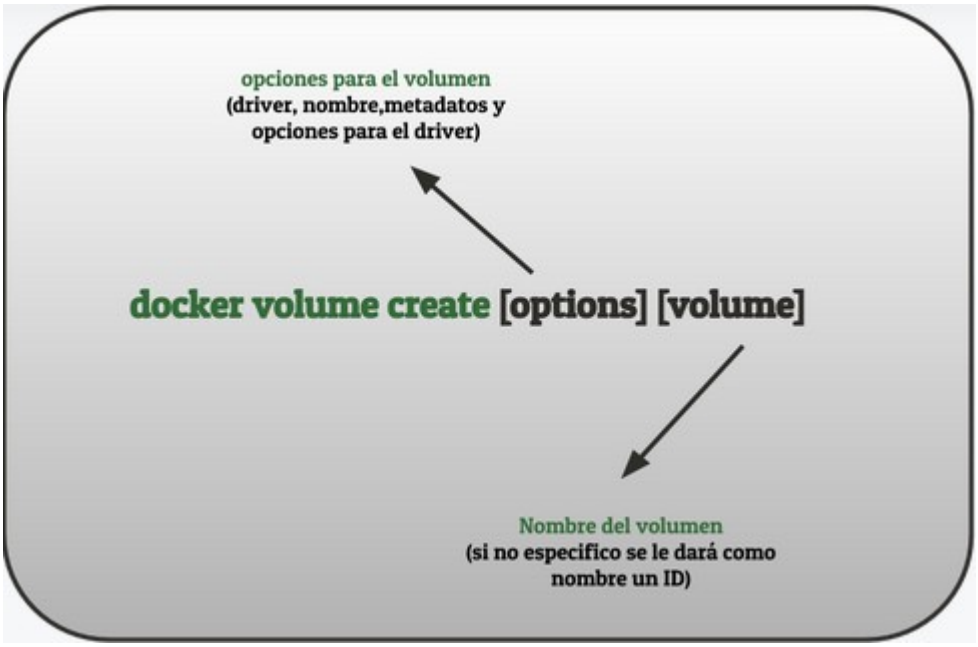
- */var/lib/docker/volumes* en las distribuciones de Linux si lo hemos instalado desde paquetes estándar.
- /var/snap/docker/common/var-lib-docker/volumes en Linux si hemos instalado docker mediante snap (no lo recomiendo).
- *C:|ProgramData|docker|volumes* en las instalaciones de Windows.
- */var/lib/docker/volumes* también en Mac aunque se requiere que haya una conexión previa a la máquina virtual que se crea.

# Bind Mounts

Este mapeado de partes de mi sistema de ficheros con el sistema de ficheros del contenedor me va a permitir:

- **Compartir ficheros** entre el host y los containers.
- Que otras aplicaciones que no sean docker tengan acceso a esos ficheros, ya sean código, ficheros etc...

# CREACIÓN DE VOLÚMENES




- --driver o -d para especificar el driver elegido para el volumen. Si no especificamos nada el driver utilizado es el *local* que es el que nos interesa desde el punto de vista de desarrollo porque desarrollamos en nuestra máquina. Al ser Linux en mi caso ese driver local es overlay2 pero existen otras posibilidades como *aufs, btrfs, zfs, devicemapper o vfs.* Si estamos interesados en conocer al detalle cada uno de ellos aquí tenemos más información.
- --label para especificar los metadatos del volumen mediante parejas clave-valor.
- --opt o -o para especificar opciones relativas al driver elegido. Si son opciones relativas al sistema de ficheros puedo usar una sintaxis similar a las opciones de la orden mount.
- --name para especificar un nombre para el volumen. Es una alternativa a especificarlo al final que es la forma que está descrita en la imagen superior.

| | |
|---|---|
| docker volume create data | Creación de un volumen llamado datos |
| docker volume create -d local data | Creación de un volumen data especificando el driver local |
| docker volume create --label servicio=http --label server=apache Web | Creación de un volumen llamando web añadiendo varios metadatos |

| | |
|---|---|
| docker volume rm nombre_volumen | Borrar un volumen por nombre |
| docker volume rm nombre_volumen1 nombre_volumen2 | Borrar dos volúmenes de una sola vez |
| docker volume rm -f nombre_volumen | Forzar el borrado de un volumen |
| docker volume prune | Borrar todos los volúmenes que no tengan contenedores asociados |
| docker volume prune -f | Borrar todos los volúmenes que no tengan contenedores asociados sin pedir confirmación |
| ocker volume prune --filter label=valor | Borrar todos los volúmenes sin usar que contengan cierto valor |
| docker volume ls | Listar los volúmenes creados en el sistema |
| docker volume inspect nombre_volumen | Información detallada de un volumen por nombre |
| docker run --name apache -v /home/usuario/web:/usr/local/apache2/htdocs -p 80:80 httpd | La carpeta web del usuario será el directorio raíz del servidor apache. Se crea si no existe |
| docker run --name apache -p 80:80 --mount type=bind,src=/home/usuario/web,dst=/usr/local/apache2/htdocs httpd | La carpeta web del usuario será el directorio raíz del servidor apache. Se crea si no existe |
| docker run --name apache -p 80:80 --mount type=volume,src=Data,dst=/usr/local/apache2/htdocs httpd | Mapear el volumen previamente creado y que se llama Data en la carpeta raíz del servidor apache |
| docker run --name apache -p 80:80 --mount type=volume,dst=/usr/local/apache2/htdocs httpd | Igual que el anterior pero al no poner nombre de volumen se crea uno automáticamente (con un ID como nombre) |