Use Volumes

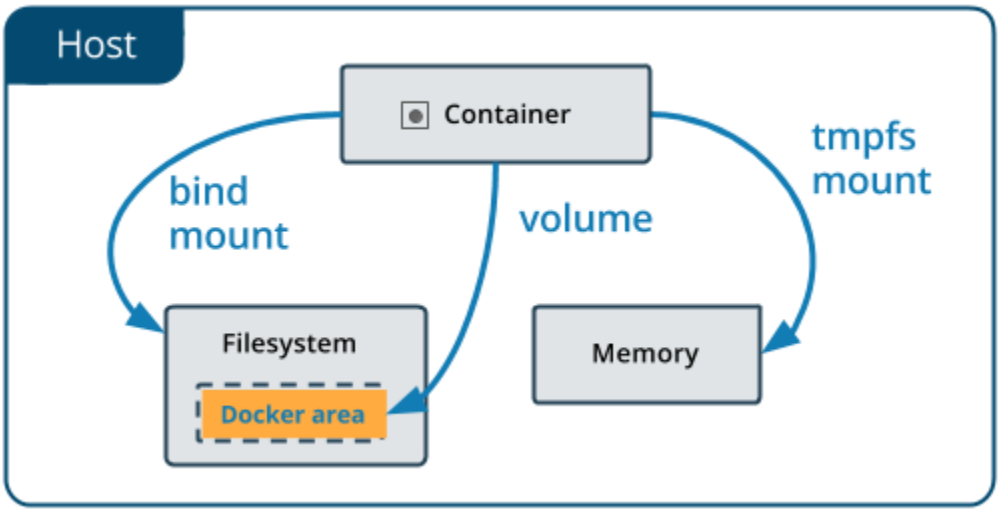
Volumes are the preferred mechanism for persisting data generated by and used by Docker containers.

While bind mounts are dependent on the directory structure and OS of the host machine, volumes are completely managed by Docker.

Volumes have several advantages over bind mounts:

- Volumes are easier to back up or migrate than bind mounts.
- You can manage volumes using Docker CLI commands or the Docker API.
- Volumes work on both Linux and Windows containers.
- Volumes can be more safely shared among multiple containers.
- Volume drivers let you store volumes on remote hosts or cloud providers, to encrypt the contents of volumes, or to add other functionality.
- New volumes can have their content pre-populated by a container.
- Volumes on Docker Desktop have much higher performance than bind mounts from Mac and Windows hosts.

In addition, volumes are often a better choice than persisting data in a container's writable layer, because a volume does not increase the size of the containers using it, and the volume's contents exist outside the lifecycle of a given container.



If your container **generates non-persistent state data, consider using a tmpfs mount** to avoid storing the **data anywhere permanently**, and to increase the container's performance by avoiding **writing into the container's writable layer**.

Volumes use **rprivate bind propagation**, and bind propagation is not configurable for volumes.

Use bind mounts

Bind mounts have been around since the early days of Docker. **Bind mounts** have limited functionality compared to volumes.

When you use a bind mount, a file or directory on the *host machine* is mounted into a container. The file or directory is referenced by its absolute path on the host machine.

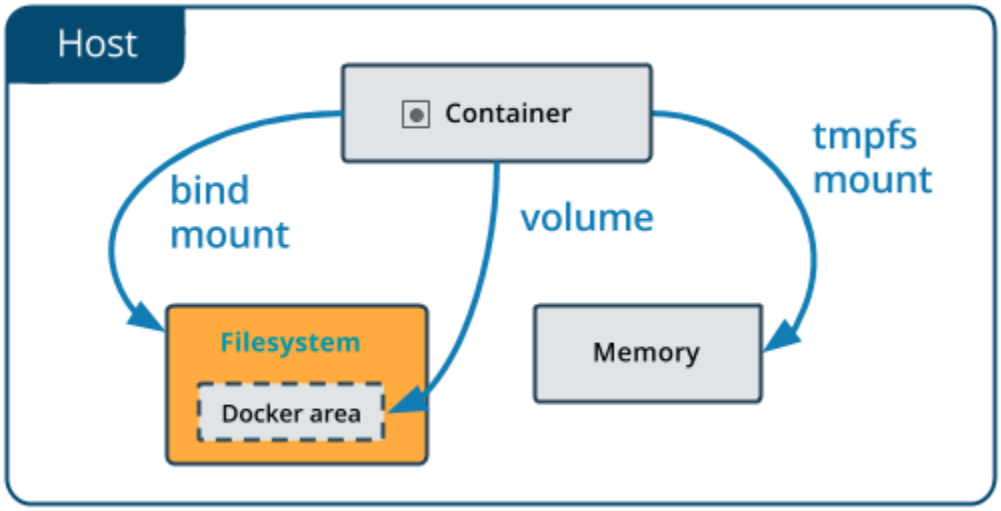
By contrast, **when you use a volume**, a new **directory is created within Docker's storage directory on the host machine, and Docker manages that directory's contents.**

The file or directory does not need to exist on the Docker host already.

It is created on demand if it does not yet exist.

Bind mounts are very performant, but they rely on the host machine's filesystem having a specific directory structure available.

If you are developing new Docker applications, consider using named volumes instead. You cannot use Docker CLI commands to directly manage bind mounts.



More information we will use in this course will be available at this link.

https://docs.docker.com/storage/volumes/#choose-the--v-or---mount-flag