**Docker Volume and Binds:**

Create and manage volumes

1. Create a volume:

docker volume create my-vol

1. List volumes:

docker volume ls

1. Inspect a volume:

docker volume inspect my-vol

[

{

"Driver": "local",

"Labels": {},

"Mountpoint": "/var/lib/docker/volumes/my-vol/\_data",

"Name": "my-vol",

"Options": {},

"Scope": "local"

}

]

1. Remove a volume:

docker volume rm my-vol

**Start a container with a volume**

If you start a container with a volume that doesn't yet exist, Docker creates the volume for you. The following example mounts the volume myvol2 into /app/ in the container.

The -v and --mount examples below produce the same result. You can't run them both unless you remove the devtest container and the myvol2 volume after running the first one.

docker run -d \

--name devtest \

--mount source=myvol2,target=/app \

nginx:latest

"Mounts": [

{

"Type": "volume",

"Name": "myvol2",

"Source": "/var/lib/docker/volumes/myvol2/\_data",

"Destination": "/app",

"Driver": "local",

"Mode": "",

"RW": true,

"Propagation": ""

}

],

**Bind Mounts**

Consider a case where you have a directory source and that when you build the source code, the artifacts are saved into another directory, source/target/. You want the artifacts to be available to the container at /app/, and you want the container to get access to a new build each time you build the source on your development host. Use the following command to bind-mount the target/ directory into your container at /app/. Run the command from within the source directory. The $(pwd) sub-command expands to the current working directory on Linux or macOS hosts.

docker run -d \

-it \

--name devtest \

--mount type=bind,source="$(pwd)"/target,target=/app \

nginx:latest

Use docker inspect devtest to verify that the bind mount was created correctly. Look for the Mounts section:

"Mounts": [

{

"Type": "bind",

"Source": "/tmp/source/target",

"Destination": "/app",

"Mode": "",

"RW": true,

"Propagation": "rprivate"

}

],

This shows that the mount is a bind mount, it shows the correct source and destination, it shows that the mount is read-write, and that the propagation is set to rprivate.

Stop the container:

docker container stop devtest

docker container rm devtest