Docker Volumes

- How to persist and save data using Docker volumes.
- How to make data available to a container in read-only mode.
- Share asme data with multiple containers
- How to use ephemereal volumes and how to quickly remove unused volumes

Managing Volumes

- Images are bult to keep containers based on the small, portable and disposable
- Images will typically contain only the packages required to provide the intented service

It's best to discard a container without worrying about losing important data. If possible, you don't want important data only exist inside the container.

- TIP: Don;t want important data to only exist inside a container.
    - If you want to persist or save data generated or used by a container -> use a volume
    - A volume is ideal choice to share the same data between multiple containers

Command Options

- The `-v` or `--volume` option in a `docker run` command declare that we wanted to use a volume with a container
- The preferred way to mount volumes is `--mount` option, insthead of `-v` because is easier to use

Creating Volumes

```
docker volume create testdata
>> testdata
docker volume ls
>> ...
>> local testdata
docker volume rm testadata
>> testdata

docker volume create mydata1
>> mydata1
docker volume ls
>> local mydata1
docker volume # Inspect information about a volume
```

Inspect Command Output

- Created At
- Driver
- Labels
- Mountpoint

- Name
- Options
- Scope

Attaching Volumes

```
# Startup a container
# Attaching a volume to it
# => Container has access to the dta in that volume

docker run -d --name withvolume --mount
source=mydata1,destination=/root/volume nginx
docker ps

docker inspect withvolume | grep Mounts -A10
```

Mounts section shows that /root/volume is the volume's destination inside the container, and it is set to read and wirte mode, as indicated by "RW: true" line.

- SOURCE/SRC path: (host) /var/lib/docker/volumes/mydata1/_data/index.html
- DESTINATION/TARGET/DST: (inside the container) /root/volume

```
echo "Hello world from mydata1 volume" >
/var/lib/docker/volumes/mydata1/_data/index.html
cat /var/lib/docker/volumes/mydata1/_data/index.html

docker exec -it withvolume /bin/bash
cd /root/volume
cat intex.html
exit
```

- Docker allows to mount the same volume to multiple containers

Read Write Options

```
docker run -d --name readonlyvolume --mount
src=volume_name,dst=/usr/share/nginx/html,readonly nginx
# readonly and ro are equivalent
```

Ephemereal Volumes

- Ephemereal or Short-Lived volumes are auto-destroyed when the container that started them is stopped.

```
docker run -dit --name ephemereal_volume --mount
type=tmpfs,dst=/root/volume nginx
docker inspect ephemereal_volume | grep Mounts -A10
```

```
#output
Type:: tmpfs,
```

Space limit

```
docker run -dit --name ephe_vol --mount type=tmpfs,tmpfs-
size=256M,dst=/root/volume nginx
# No more than 256M of data can be stored in /root/volume
```

Ephemereal volumes are a powerful way of cleaning up after containers that need extra storage space, but only temporaly.

Limitations: You cannot share tmpfs volumes between containers. These types of volumes disappear after the container stops, they can make it more difficult to diagnose issues.

Volume Types

- `-v` option syntax

```
docker run -p 8080:80 --name nginx-with-vol -v
${PWD}/webpages:/usr/share/nginx/html:ro -d nginx
```

Volumes are less flexible because the mount target is a directory or just a file

Delete Volumes

```
docker rm vol1 vol2 vol3 volN
docker ps -a
docker volume ls
docker volume prune
```

Recap

```
docker volume create localvolume
docker volume inspect localvolume
echo "this file exist" > /var/lib/docker/volumes/localvolume/_data/file.txt
```

```
docker -d --name mountvolume --mount src=localvolume,dst=/data httpd
docker exec -it mountvolume /bin/bash

# inside the container shell
df -h
cat /data/file.txt

echo "Created from inside the container" > /data/from-container.txt
cat /data/from-container.txt
exit

# Outside the container
cat /var/lib/docker/volumes/localvolume/_data/from-container.txt
```

Temporal Volume

```
docker run -d --name tempvolume --mount type=tmpfs,dst=tempdata httpd
docker inspect tempvolume | grep Mounts -A10
```