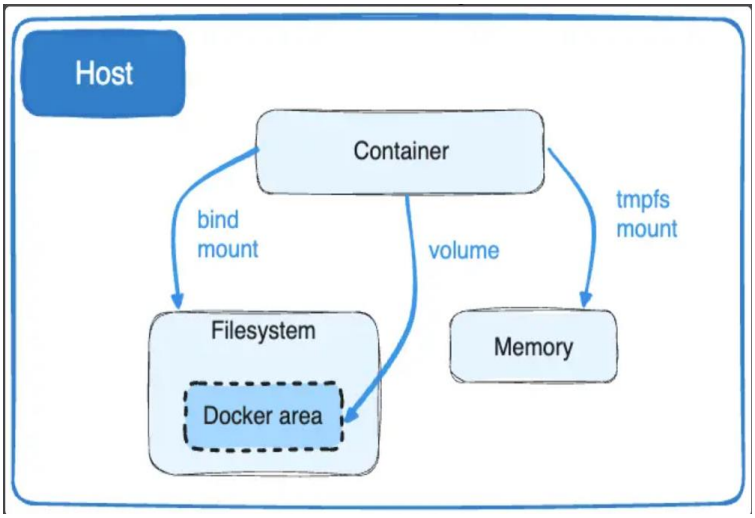
Storage

Ephemeral storage:

In the context of Containers, ephemeral storage is storage tied to the lifecycle of a Containers, so when a Container terminated that storage is cleared out.

So, in short, what is ephemeral storage? It's just temporary storage for a container that gets wiped out and lost when the container is stopped or restarted. Hence, the storage concepts came into picture.

Storage Types:



To Identify:

1) Filesystem (Docker Area) 2) Volume 3) Bind Mount 4) tmpfs mount

1) Docker Area (File system):

root@devopsubuntu:~/ # docker info | tail -10

Total Memory: 1.922GiB

Name: devopsubuntu

ID: 9f646705-da4d-40c7-81ca-1cb32f892130

Docker Root Dir: /var/lib/docker

Debug Mode: false

Experimental: false

Insecure Registries:

127.0.0.0/8

Live Restore Enabled: false

root@devopsubuntu:~/custom_images/custom_imnages2# cd /var/lib/docker

root@devopsubuntu:/var/lib/docker# ls -lrt

```
total 44
drwx------ 4 root root 4096 May 11 13:19 plugins
-rw------- 1 root root   36 May 11 13:19 engine-id
drwx------ 3 root root 4096 May 11 13:19 image
drwxr-x--- 3 root root 4096 May 11 13:19 network
drwx------ 2 root root 4096 May 11 13:19 swarm
drwx--x--x 5 root root 4096 May 13 21:27 buildkit
drwx--x--- 2 root root 4096 May 13 21:44 containers
drwx------ 2 root root 4096 May 14 04:55 runtimes
drwx-----x 2 root root 4096 May 14 04:55 volumes
drwx--x--- 3 root root 4096 May 14 04:55 overlay2
drwx------ 2 root root 4096 May 14 04:55 tmp
```

2) Volume:

If you try to create any volume then it will automatically create under /var/lib/docker/ volumes

Volumes in short:

- Volume is simply a directory inside our container
- Firstly, we have to declare the directory as a volume and then share volume
- Even if we stop container, still we can access volumes.
- Volumes work on both Linux and Windows containers.
- Volume will be created in one container.
- You can declare a directory as a volume only while creating container.
- You can't create volume from existing container.
- You can share one volume across any number of containers
- Volume will not be included when you update an image.
  - In short, if an image is created from a container and a new container is created from that image, the volume directory in the new container won't be shared or connected with the original container.
- You can map the volume in Two ways:
  - Host to Container (A host is simply a device, like a computer or a server, that connects to the internet or a network.)
  - Container to Container.

Check any data is available inside the volume directory:

root@devopsubuntu:/var/lib/docker# ls -lrt

```
total 44
drwx------ 4 root root 4096 May 11 13:19 plugins
-rw------- 1 root root   36 May 11 13:19 engine-id
drwx------ 3 root root 4096 May 11 13:19 image
drwxr-x--- 3 root root 4096 May 11 13:19 network
drwx------ 2 root root 4096 May 11 13:19 swarm
drwx--x--x 5 root root 4096 May 13 21:27 buildkit
drwx--x--- 2 root root 4096 May 13 21:44 containers
drwx------ 2 root root 4096 May 14 04:55 runtimes
drwx-----x 2 root root 4096 May 14 04:55 volumes
drwx--x--- 3 root root 4096 May 14 04:55 overlay2
drwx------ 2 root root 4096 May 14 04:55 tmp
```

root@devopsubuntu:/var/lib/docker# ls -lrt volumes

```
total 24
brw------- 1 root root 252, 0 May 14 04:55 backingFsBlockDev
-rw------- 1 root root  32768 May 14 04:55 metadata.db
```

Create Volume:

root@devopsubuntu:/var/lib/docker# docker volume create nginx-data

```
nginx-data
```

root@devopsubuntu:/var/lib/docker# docker volume ls

```
DRIVER    VOLUME NAME
local     nginx-data
```

root@devopsubuntu:/var/lib/docker# ls -lrt /var/lib/docker/volumes/

```
total 28
brw------- 1 root root 252, 0 May 14 04:55 backingFsBlockDev
drwx-----x 3 root root   4096 May 14 08:07 nginx-data
-rw------- 1 root root  32768 May 14 08:07 metadata.db
```

Now link any data to the volume / create a hardlink (or) bind the data from container usr/share path to manually created volume:

root@devopsubuntu:/var/lib/docker# docker container run -d --name nginx --volume nginx-data:/usr/share/nginx/html -p 2121:80 nginx:latest

```
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
b0a0cf830b12: Pull complete
4d84de5fb9b2: Pull complete
2818b7b6a9db: Pull complete
1e5314d67f16: Pull complete
8066e07ce4f2: Pull complete
05f7109fea9e: Pull complete
Digest: sha256:32e76d4f34f80e479964a0fbd4c5b4f6967b5322c8d004e9cf0cb81c93510766
Status: Downloaded newer image for nginx:latest
93dd42855f5cbd0d015b75c850dc42917a1f54ba74df06841baebce44ef8836a
```

root@devopsubuntu:/var/lib/docker# docker container ps -a

```
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS                                   NAMES
93dd42855f5c   nginx:latest   "/docker-entrypoint.…"   15 seconds ago   Up 14 seconds   0.0.0.0:2121->80/tcp, :::2121->80/tcp   nginx
```

root@devopsubuntu:/var/lib/docker# ls -lrt /var/lib/docker/volumes/nginx-data/

```
total 4
drwxr-xr-x 2 root root 4096 May 14 08:18 _data
```

root@devopsubuntu:/var/lib/docker# tree /var/lib/docker/volumes/

```
/var/lib/docker/volumes/
├── backingFsBlockDev
├── metadata.db
└── nginx-data
    └── _data
        ├── 50x.html
        └── index.html
```

root@devopsubuntu:/var/lib/docker# cat /var/lib/docker/volumes/nginx-data/_data/index.html

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

Inside the container

root@devopsubuntu:/var/lib/docker# docker container exec -t -i nginx bash

```
root@93dd42855f5c:/# cat /usr/share/nginx/html
cat: /usr/share/nginx/html: Is a directory
```

root@93dd42855f5c:/# cat /usr/share/nginx/html/index.html | tail -5

```
<a href="http://nginx.com/">nginx.com</a>.</p>
<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@93dd42855f5c:/# cat /usr/share/nginx/html/index.html | tail -10
working. Further configuration is required.</p>
<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</
<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

Scenario 1:

Manual Volume is created and added to the container as read and write permission in the above scenario so if you write anything the host machine under /var/lib/docker/volumes it will reflect in the created containers

Host Machine:

root@devopsubuntu:/var/lib/docker# vi /var/lib/docker/volumes/nginx-data/_data/index.html

root@devopsubuntu:/var/lib/docker# cat /var/lib/docker/volumes/nginx-data/_data/index.html

```
<h1> Welcome to Dhileep's nginx file </h1>
```

Inside the container

root@devopsubuntu:/var/lib/docker# docker container exec -t -i nginx bash

root@93dd42855f5c:/# cat /usr/share/nginx/html/index.html | tail -5

```
<h1> Welcome to Dhileep's nginx file </h1>
```

Scenario 2:

Manual Volume is created and added to the container as read permission in the below scenario so if you tend to write anything it will throw an error saying you have only read permission.

Host Machine:

root@devopsubuntu:/var/lib/docker# docker container run -d --name nginx_4 -v nginx-data:/usr/share/nginx/html:ro -p 2323:80 nginx:latest

```
599340eb9d853bf5b6c7cd31ea42b2b7f17dc2b067a917186df0859223fdb9d2
```

Inside the container

root@devopsubuntu:/var/lib/docker# docker container exec -t -i nginx_4 bash

root@599340eb9d85:/# cd /usr/share/nginx

root@599340eb9d85:/usr/share/nginx# echo "<h1> Welcome to Dhileep's nginx file 2</h1>" > index.html

```
bash: index.html: Read-only file system
```