Starting Containers

	docker images → Shows images. - react-app - ubuntu - alpine
active process	docker ps
Start Container	docker run react-app
Start Container in detached (background)	docker run -d react-app
Create a Container with name.	docker run -d --name blue-sky react-app

Viewing logs.

how to view logs

docker logs 65529e455

options

docker logs -f 123
f is follow

docker logs -n 123
n is tail

docker logs -t 123
t is timestamp.

Publishing the ports

docker ps
→ Shows ports
→ 3000/tcp

docker run -d -p
3000:3000 --name c1
react-app

In browser

localhost:3000
It opens page.

docker ps
→ shows port
→ 0.0.0.0:80 → 3000/tcp

Executing Commands in Running Containers

Command

We need to run **exec** Command.

What's difference between RUN and EXEC

RUN	EXEC
Starts a new Container & run Command	Execute a Command in running Container

list app directory content

```
docker exec c1 ls
```

—
—
—
—

Execute shell

docker exec -it c1 sh
/app $

∴ The Container will still run. It wont stop

Starting & stopping Containers

Stop Container.	docker stop C1
Start Container	docker start C1

Removing Containers

docker rm c1
→ If c1 is in stopped state -

docker rm -f c1
→ force remove.

To see stopped Containers — docker ps -a

another way to verify its removed — docker ps -a | grep c1

Remove all Stopped Containers — docker container prune

Container file system

Each docker container will have a file system invisible to other containers

To test above run 2 containers

Start shell on 1st

```
docker exec -it b55 sh
/app $ echo data > data.txt
$ exit
```

Start shell on 2nd

```
docker exec -it feb sh
ls | grep data.
```

no results

Volumes

What ?

It is storage outside Container. It is either in host or directly in cloud

Create Volume

```
docker volume create app-data.
```

Inspect

```
docker volume inspect app-data
{
    Created at
    Driver
    Mount Point
    Name
    Options
    Scope
}
```

Start volume & mount this volume

```
docker run -d -p
  4000:3000 -v
  app-data:/app/data
    react-app.
```

```
docker exec -it 71b sh
/app # echo data >
          data.txt
```

We see an error

```
sh: can't create data.txt
  : Permission denied.
```

Why?

/app/data got created by root user and so app user does not have permission to create.

```
drwxr-xr-x  root
            data.
```

To prevent this, we want to create this folder as app user.

dockerfile.

```
FROM node:14.16-0-alpine3.13
RUN addgroup app
 && adduser -S -G app app
USER app
WORKDIR /app
RUN mkdir data
COPY package*.json
RUN npm install
COPY . .
ENV API_URL = http://
EXPOSE 3000
CMD ["npm", "start"]
```

Rebuild

```
docker build -t react-app .
```

Run

```
docker run -d -p
5000:3000 -v
app-data:/app/data
react-app
```

```
docker exec -it 007 sh
/app $ cd data.
/app/data $ echo
        data > data.txt
/app/data $ exit.
```

Remove the Container now

```
docker rm -f 007
```

Start new Container with same Volume.

you will see the file data.txt.

COPYING FILES BETW HOST & CONTA

docker ps

Creating log file

docker exec -it c1c sh
/app # echo hello > log.txt

Copy cmd (Container to host)

docker cp c1c9043ea8ce:/app/log.txt .

/

ls

Copy (host to container)

echo hello > secret.txt
docker cp secret.txt c1c9043ea8ce:/app

docker exec -it
a1c9043ea8ce sh

we find here the
secret.txt file.

Sharing Code between host & Container

Lets make a change in index.html in host, but it does not reflect until build.

For production — always build

For development
- → no need to rebuild
- → dont manual copy

So what can be done?

Create a mapping / binding between directory on host and container.

Start Container.

```
docker run -d -p
5001:3000 -v
$(pwd):/app
react-app
```

docker logs -f 696

Now

localhost:5001
→ title is updated

Again change in index.html

Now back to browser the change is updated.

This feature is from react. app reloading.