

What is a Container?

- Layers of images



So we have layers of stacked images on top of each other.

110 / 946 14. Will 01:10:21

Docker is multi layer image file.

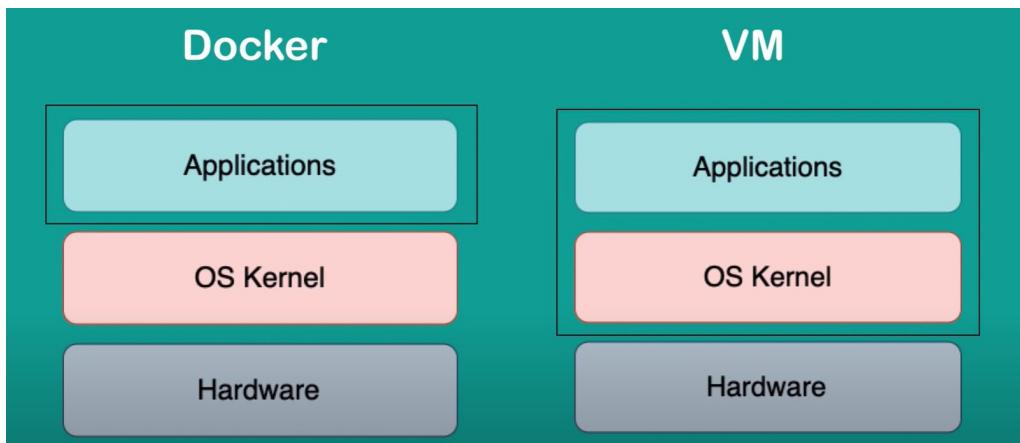
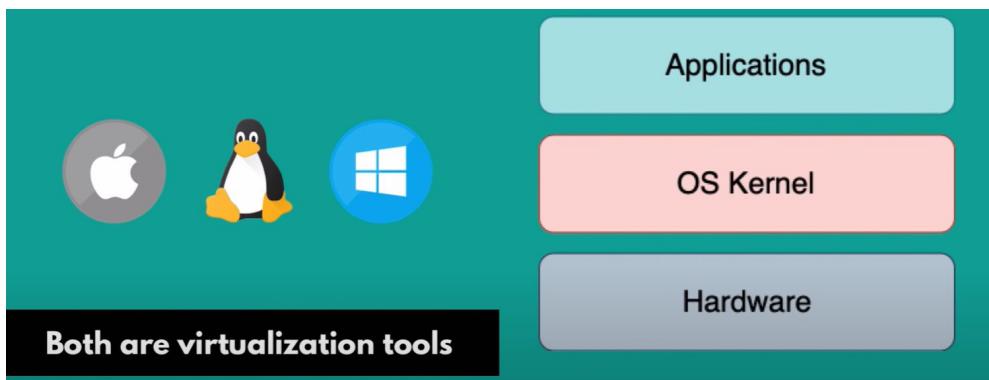
When you want upgrade version of software at that time it NOT download whole docker , it just download only upgraded layer , base layer kept as it.

```
Last login: Sat Sep 28 10:57:20 on ttys001
[Nanans-MBP:~ nanabiz$ docker ps
CONTAINER ID        IMAGE               COMMAND       CREATED          STATUS          PORTS
S
fad0f8456ca7      postgres:9.6      "docker-entrypoint..."   45 seconds ago   Up 47 seconds   5432/tcp
eless_habit
Nanans-MBP:~ nanabiz$
```

```
Nanans-MBP:~ nanabiz$ docker run postgres:10.10
Unable to find image 'postgres:10.10' locally
10.10: Pulling from library/postgres
8f91359f1fff: Already exists
c6115f5efcde: Already exists
28a9c19d8188: Already exists
2da4beb7be31: Already exists
fb9ca792da89: Already exists
cedc20991511: Already exists
b866c2f2559e: Already exists
5d459cf6645c: Already exists
6de9d066d892: Downloading [=====]
401fc8e29c4: Download complete
9b130e26214a: Download complete
1c048e77610c: Download complete
431b5e6c27b3: Download complete
```

some layers already exist ✓

110 / 946 14. Will 01:10:21



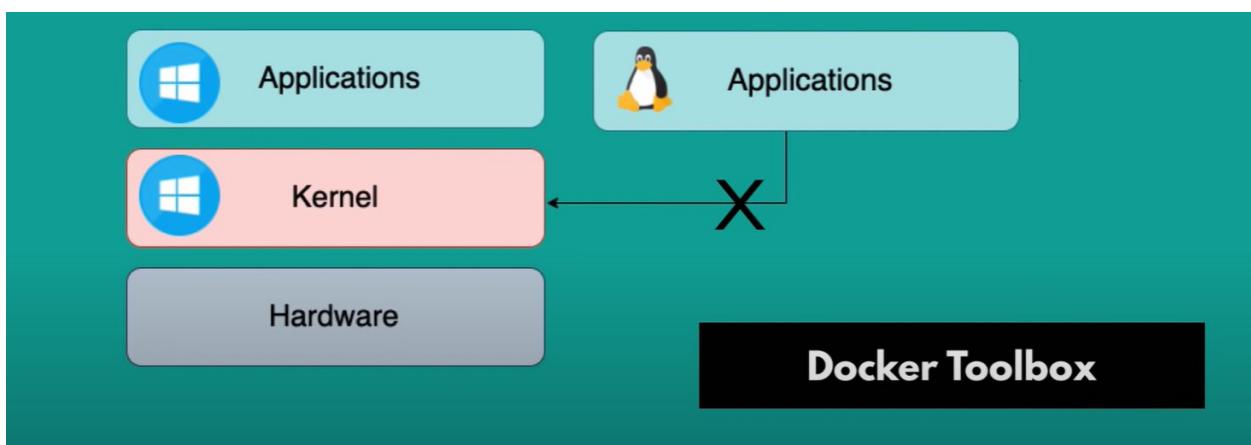
Docker virtualize at application layered where VM Virtualize at OS layered .

SIZE:- That why Docker is small in SIZE than VM

SPEED:- Docker is much faster then VM

OS Kernel Dependency (Compatibility) :- Docker run on any linux but if it window it extra Docker tool , When VM doen't required , VM can run any host .

FOR Windows



Workaround for Windows & Mac

Overview

- ✓ Before installing Docker - pre-requisites
- ✓ Install Docker on Mac
- ✓ Install Docker on Windows
- ✓ Install Docker on Linux



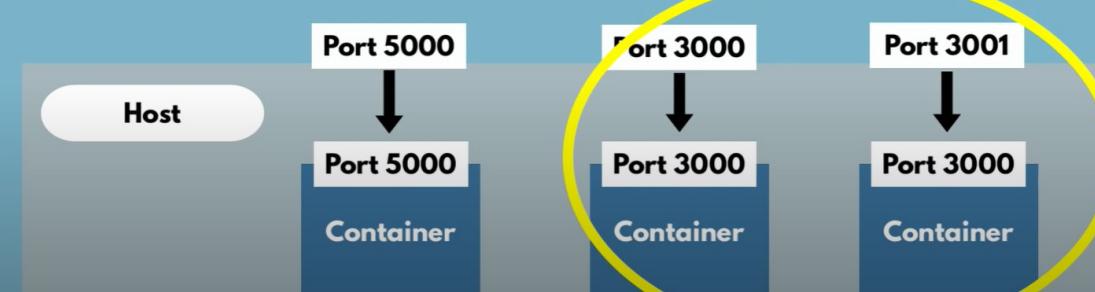
Install Docker



CONTAINER Port vs HOST Port



- ▶ **Multiple containers** can run on your host machine
- ▶ Your laptop has only certain ports available
- ▶ Conflict when **same port** on host machine



Port Mapping

What is port binding ?

=> connecting a **port on the host machine** to a **port inside the container**.
A process of linking a **port on your host machine** (your computer or server) to a **port inside a Docker container**. Service inside container can access out of container is port mapping.

I remember that when you run a container, **services inside it aren't accessible** from outside unless you map the ports. For example, if a container is running a web server on port 80, you need to map the host's port 8080 to the container's port 80 to access it via localhost:8080.

Why port mapping is necessary?

=> Containers have their **own isolated network**, so without mapping, the ports aren't exposed to the host or the outside world.

If a service (e.g., a web server) runs on port 80 inside a container, it's **not accessible from outside the container** unless you explicitly map the container's port to a host port.

Port Mapping Ex. Docker run command syntax. It's usually **-p host_port:container_port**.

What's the difference between -p and -P?

=>-P flag publishes **all exposed ports** to random ports **on the host**. That's useful when you don't want to specify each port manually.

How It Works

- **Host Port:** Port on your machine (e.g., 8080).
- **Container Port:** Port the service uses inside the container (e.g., 80).

When you map 8080:80, traffic sent to `http://localhost:8080` on the host is forwarded to port 80 in the container.

ex. `docker run -p <HOST_PORT>:<CONTAINER_PORT> <IMAGE Name>`

1. `docker run -p 8080:80 nginx`
2. `docker run -p 192.168.1.100:8080:80 nginx`
3. # Docker picks a **random host port** (e.g., 32768) and maps it to container port 80

`docker run -p 80 n nginx`
`# Check assigned port with `docker ps``

What is docker-compose, how port mapping is handled

=> It define the configuration of all your application's services (e.g., databases, web servers, etc.) in a single YAML file. It used link containers, share volumes, or set environment variables.

Also used :

- Use a single command (`docker-compose up`) to start multiple services at once.
- Easily scale services, manage networking, and configure volume mounts.

Key Concepts of Docker Compose:

1. **Services:** These are the containers defined in the `docker-compose.yml` file. Each service can represent a different container (e.g., a web app container, a database container).
2. **Networks:** Docker Compose automatically creates a network for your services to communicate with each other.
3. **Volumes:** Shared storage that can be used by containers.

Example of docker compose file:

```
version: '3.8' # Specify the version of Docker Compose

services:
  web:
    image: nginx:latest # Use the official Nginx image
    ports:
      - "8080:80" # Map port 8080 on host to port 80 on the container
  db:
    image: mysql:5.7 # Use the official MySQL image
    environment:
      MYSQL_ROOT_PASSWORD: example_password # Set environment variables
    volumes:
      - db_data:/var/lib/mysql # Use a named volume for database data

volumes:
  db_data: # Define the named volume
```

Port Mapping Format in docker compose file :

```
yaml
ports:    - "host_port:container_port"
```

Example of Port Mapping:

```
version: '3'  
services:  
  app:  
    image: your_app_image  
    ports:  
      - "5000:80" # HTTP port  
      - "8443:443" # HTTPS port
```

Starting Docker Compose:

```
docker-compose up
```

Stopping Docker Compose:

```
docker-compose down
```

docker run command

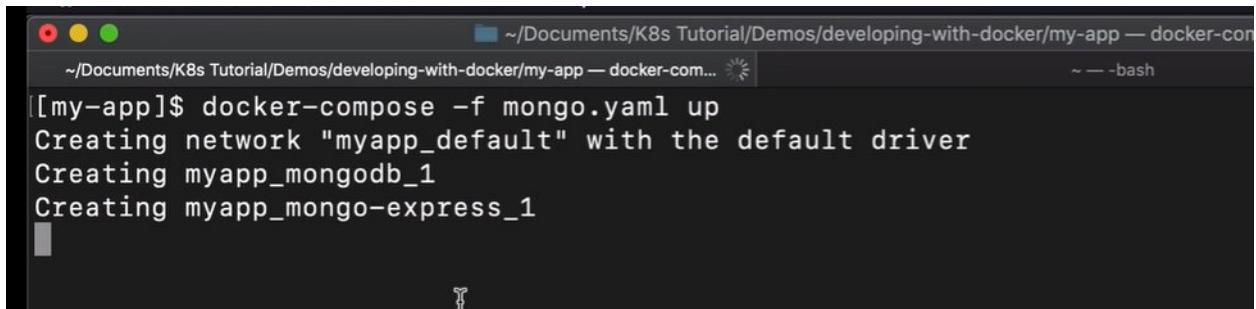
```
docker run -d \
--name mongo-express \
-p 8080:8080 \
-e ME_CONFIG_MONGODB_ \
ADMINUSERNAME=admin \
-e ME_CONFIG_MONGODB_ \
ADMINPASSWORD=password \
-e ME_CONFIG_MONGODB_ \
SERVER=mongodb \
--net mongo-network \
mongo-express
```

mongo-docker-compose.yaml

```
version: '3'
services:
  mongodb:
    image: mongo
    ...
  mongo-express:
    image: mongo-express
    ports:
      - 8080:8080
    environment:
      - ME_CONFIG_MONGODB_A....
```

it's going to be easier
for you to edit the file

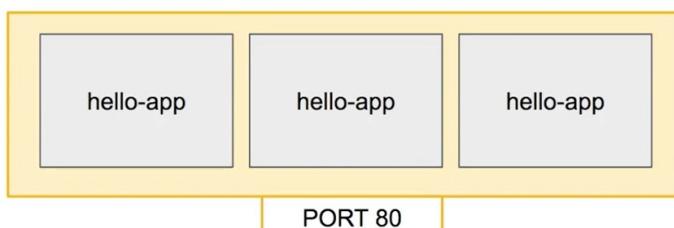
To Run docker compose file



The screenshot shows a terminal window with the following text:

```
~/Documents/K8s Tutorial/Demos/developing-with-docker/my-app — docker-com... ~ — bash
[[my-app]$ docker-compose -f mongo.yaml up
Creating network "myapp_default" with the default driver
Creating myapp_mongodb_1
Creating myapp_mongo-express_1
```

With a Service



(My simple) definition:

Service = n containers all running with the same parameters,
available at the same port