# Docker Demo

## About Docker

Docker data

*/var/lib/docker*

- *aufs*
- *containers*
- *image*
- *volumes*

## Docker Commands

Basic Commands

```
$ docker pull nginx
$ docker run nginx
$ docker ps
$ docker ps -a
$ docker stop [CONTAINER_NAME]
$ docker rm [CONTAINER_NAME]
$ docker images
$ docker rmi [IMAGE_NAME]
$ docker run ubuntu sleep 5
$ docker exec [CONTAINER_NAME] cat /etc/hosts
$ docker run [IMAGE_NAME]   # Attach mode
$ docker run -d [IMAGE_NAME]  # Detach mode
$ docker attach [CONTAINER_NAME] # Attach it again
$ docker network ls     # list networks
```

### Run Command

Run-tag

```
$ docker run redis:4.0
```

Run-Name

```
$ docker run --name TestServer ubuntu
```

Run-STDIN: Takes an input from the user

```
$ docker run -i [IMAGE_NAME]
```

Run-STDIN: Interactive mode

```
$ docker run -it [IMAGE_NAME]
```

Run-PORT mapping

```
$ docker run -p [HOST_PORT]:[CONTAINER_PORT] [IMAGE_NAME]

$ docker run -p 8001:3306 mysql
```

Run-Volume mapping

```
$ docker run -v [HOST_VOLUME]:[CONTAINER_VOLUME] [IMAGE_NAME]

$ docker run -v /opt/datadir:/var/lib/mysql mysql

$ docker run -v data_volume:/var/lib/mysql mysql

$ docker run \
--mount type=bind,src=/data/mysql,target=/var/lib/mysql mysql

$ docker run \
--mount type=volume,src=my_volume,target=/var/lib/mysql mysql
```

Run-Environment variables

```
$ docker run -e APP_COLOR=blue [IMAGE_NAME]
```

Run-Network

```
$ docker run ubuntu --network=host
```

- *Bridge*: Containers are connected together via bridge switch.
- *none*: Isolated containers.
- *host*: the port is binded to the host

Run-Volumes

```

```

### Networks

Create New Network

```
$ docker network create \
    --driver bridge \
    --subnet 182.18.0.0/16 custom-isolated-network
```

list Networks

```
$ docker network ls
```

Inspect Network

```
$ docker inspect [CONTAINER_NAME]
```

The Containers can connect each others via names because of the buit-in DNS.

### Docker volumes

### Troubleshooting

Inspect Container

```
$ docker inspect [CONTAINER_NAME]
```

Container Logs

```
$ docker logs [CONTAINER_NAME]
```

## Dockerfile

It works as

*INSTRUCTION ARGUMENT*

```
FROM ubuntu    # BASE IMAGE

RUN apt update   # Commands to run before Containerize it

COPY . /opt/source-code  # Copy From host to image

ENTRYPOINT flask run  # Start command
```

Build the image

```
$ docker build . -t m4tt4r/myapp:1.0
```

Push the image to Docker Registry

```
$ docker push m4tt4r/mysapp:1.0
```

Run multiple commands once in Dockerfile

```
RUN \
    COMMAND 1 && \
    COMMAND 2 \
```

Define moutable directories.

```
VOLUME ["/etc/nginx/sites-enable", ""] # WHAT IS THIS
```

Define working directory

```
WORKDIR /etc/nginx
```

Starting command

```
CMD ["sleep","5"]
```

Change listening port

```
EXPOSE 3306 33060
CMD ["mysqld"]
```

Add a file

```
ADD root/.bashrc /root/.bashrc
```

Set Environment Variables

```
ENV HOME /root
```

ENTRYPOINT vs CMD

```
CMD ["sleep","5"]
$ docker run ubuntu sleep 10
Startup command: sleep 10
```

```
ENTRYPOINT ["sleep"]
$ docker run ubuntu 10
Startup command: sleep 10
```

## Docker Compose

Run Your All Apps with single Command.

*Docker Compose Versions:*

-