Docker Full Demo

Run and manage containers

Launch mysql 8

`docker run -p 3306:3306 --name mysql8 -e MYSQL_ROOT_PASSWORD=secreto -d mysql:8.0`

Launch MySQL 5

`docker run -p 3307:3306 --name mysql5 -e MYSQL_ROOT_PASSWORD=secreto -d mysql:5.6`

View running containers

`docker ps`

View container logs

`docker logs container_id`

`docker logs -f container_id`

Execute commands in a container

`docker exec -it mysql5 bash`

After entering in the container:

`ls`

`mysql -uroot -psecreto`

`show databases;`

```
exit
```

```
docker ps
```

Stop container

`docker stop` mysql5

`docker ps`

View all containers (even the stopped ones)

```
docker ps -a
```

Delete containers

```
docker rm mysql5
```

```
docker rm mysql8
```

Images

Show images

```
docker images
```

Dockerfiles

> Github repository: https://github.com/press-any-key-tech/001.docker-k8s-demo

> Directory: 01.starting-dockerfile

Create a folder and a file inside called Dockerfile.

```
FROM alpine:3.4
RUN apk update
RUN apk add vim
RUN apk add curl
```

Build

```
docker build -t patata:latest .
```

Run image

--rm : delete when finished

-it interactive session

After image name: commando to execute

Alpine doesn't have bash intalled.

```
docker run --rm -ti patata:latest /bin/sh
```

Inside the container `vim -v` and `curl -version`

Git not installed

Add to dockerfile

```
RUN apk add git
```

Rebuild (look at the layers)

```
docker build -t patata:latest .
```

Re-run

```
docker run --rm -ti patata:latest /bin/sh
```

Git is available

```
git
```

Build Image

> Github repository: https://github.com/press-any-key-tech/001.docker-k8s-demo

> Directory: 02.basic-web

```
docker build -t basicweb:latest .
```

Run:

```
docker run --name basicweb -d basicweb:latest
```

Map ports

```
docker run --name basicweb -p 8080:80 -d basicweb:latest
```

Push image to Docker hub

Login in docker hub

`docker login --username=yourusername --password yourpassword`

Tag image

`docker image tag basicweb:latest impalah/basicweb:latest`

Push

`docker push yourhubusername/imagename`

Docker volumes

A volume is a mapping between a host folder (or a volume object) with a folder in the container.

Files in the mapped unit in the container will be stored in a folder in the host computer.

Create a volume

`docker volume create <volume_name>`

List volumes

`docker volume ls`

Inspect volume

Get information about a volume.

`docker inspect <volume>`

Generally the folder of volumes will be under `/var/lib/docker` (see below for Windows).

Windows: WSL mapping

Docker on windows uses WSL (Linux subsystem) and maps linux directories to $wsl drive.

More generally `/var/lib/docker/` maps to `\\wsl$\docker-desktop-data\version-pack-data\community\docker\`

Delete volume

`docker volume rm <volume>`

Attach volume to container

If volume does not exists, it will be created.

`docker run -d --name devtest -v myvol2:/app nginx:latest`

Bind mounts

This operation will bind a folder in the host computer with a folder in the container.

The difference is that no volume folder will be created.

Host folder path should be absolute. If only a name is specified docker will create a volume.

`docker run -d -it --name devtest -v C:/Users/Impalah/projects:/app nginx:latest`

(test, entering nginx): `docker exec -it devtest sh`

- Docker Cheatsheet
- Docker Cheatsheet (II)
- Dockerfile Reference