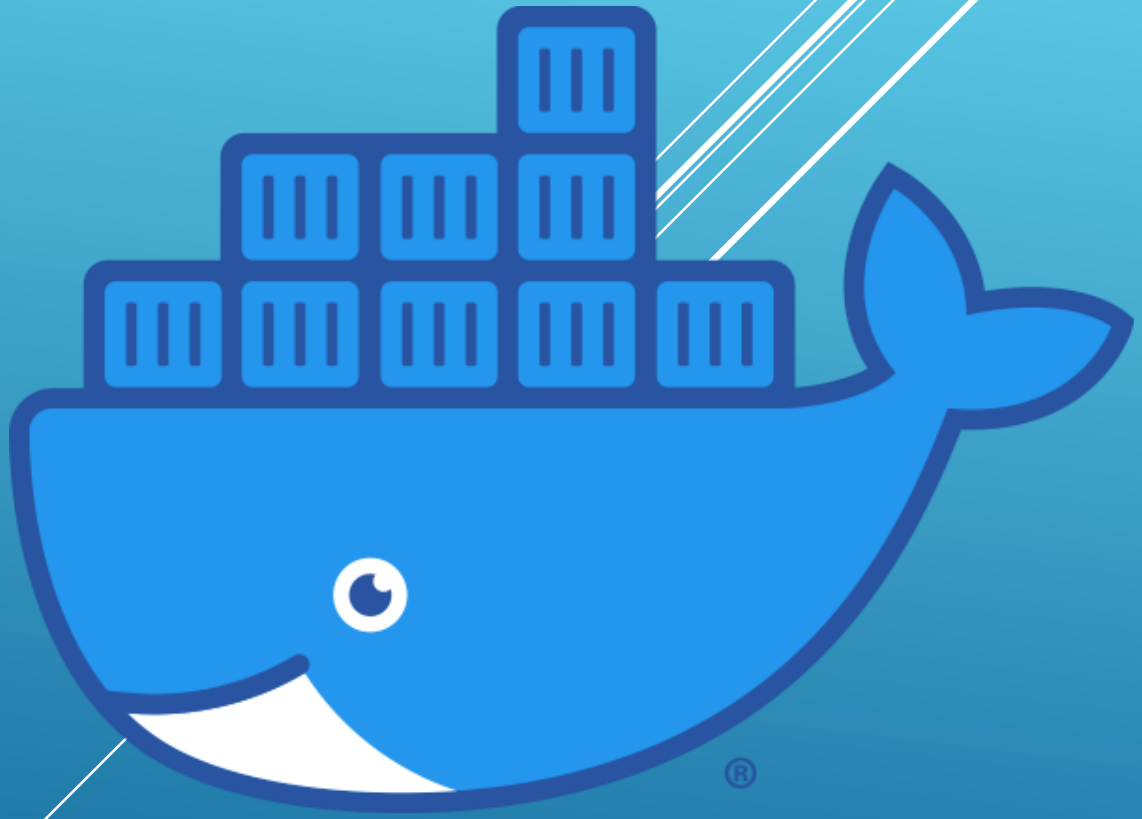
DOCKER BASICS

Lesson #1

1. Overview
2. Commands
3. Dockerfile

Lesson #2

1. Network
2. Volumes
3. Docker-compose



Open source application (PaaS) that use
OS-Level virtualization (Kernel, Hyper-V, HyperKit)
to deliver (build, run, ...) software quickly, written in Golang at 2013

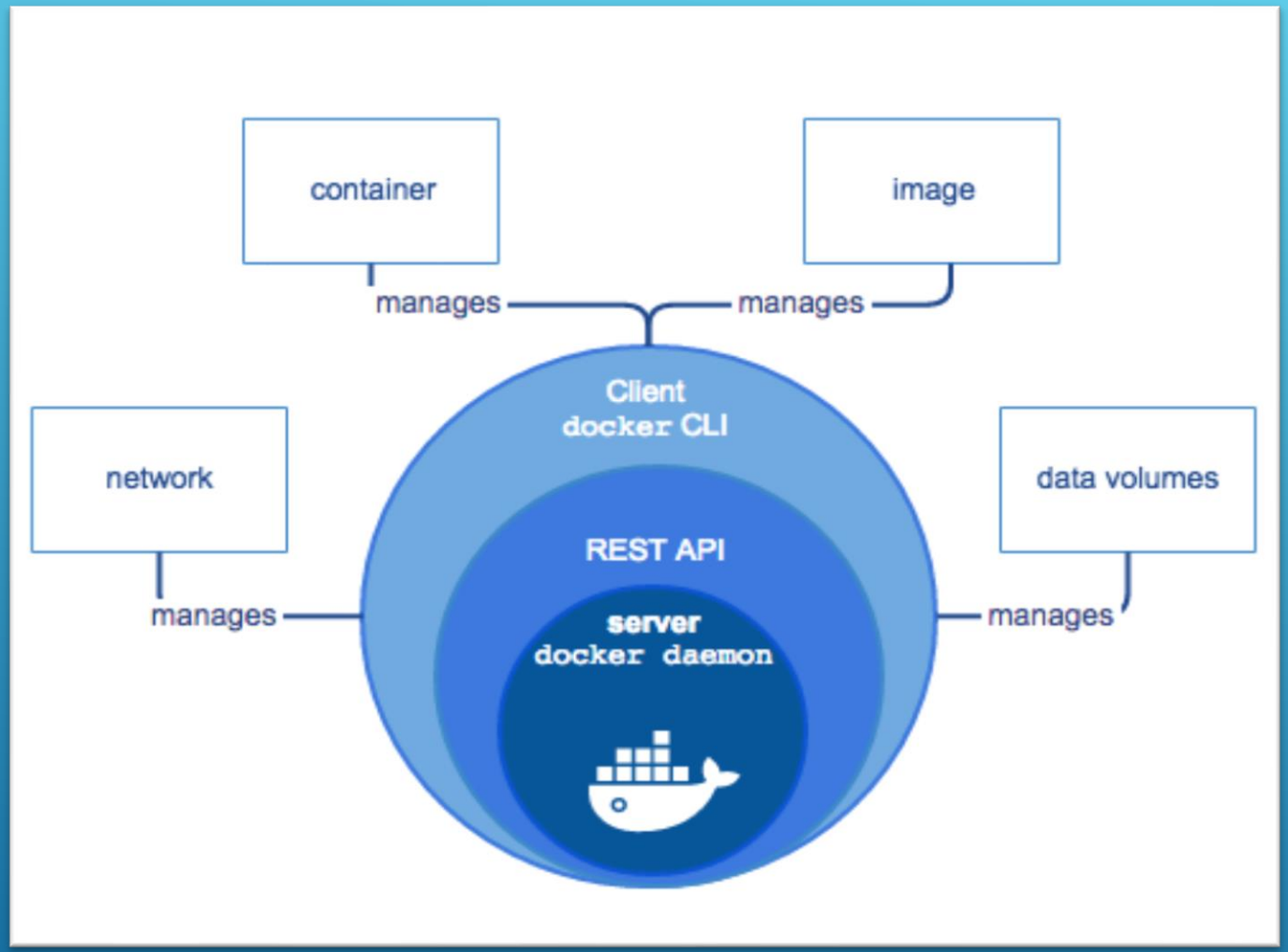
Benefits: Fast, Lightweight
Usages: Development, Deployment, Scaling

Best knowledge base:

https://hub.docker.com/

https://docs.docker.com/

ARCHITECTURE



Dockerfile => <Build> => Image => <Run> => Container

How to install Docker?

Linux

1. sudo apt install docker.io
2. sudo curl -L "https://github.com/docker/compose/releases/download/1.24.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
3. sudo chmod +x /usr/local/bin/docker-compose

Windows

1. Install Docker for Windows (Powershell)

(Optional) Pin docker to WSL

1. Expose daemon on 2375 ports in settings
2. echo "export DOCKER_HOST=tcp://localhost:2375" >> ~/.bashrc && source ~/.bashrc

Image command

Image - compiled dockerfile, used to create containers.

Command	Description
docker build <path>	build an image from the dockerfile
docker run <id or name>	**create and run image**
docker images	**show all the images**
docker rmi <id or name>	**delete image**
docker pull <name>	pull the image from dockerhub
docker push <name>	push image to the dockerhub

Container commands

Container - running instances of docker images.

Commad	Description
docker ps	**show all the containers**
docker rm <id or name>	**remove container**
docker start <id or name>	start stopped container
docker stop <id or name>	stop the container
docker history <id or name>	showing history of the image

Usefull commands

Command	Description
docker logs <id or name>	**show application logs**
docker exec –it <id or name> /bin/sh	get inside docker container
docker rm –f $(docker ps –aq)	**remove all containers**
docker rmi –f $(docker images –q)	**remove all images**
docker inspect <id or name>	show details
docker system prune	clear docker

Common flags

Flag	Description
-p	expose ports <local-port>:<docker-port>
-P	Expose on default ports
-e	environment variable
-d	detached, run in a background
--name	give name
-t	add tag to image
-v	expose volume <local-path>:<docker-path>

One line commands

Command
docker run –P -d -e POSTGRES_PASSWORD=docker postgres
docker run –p 50000:50000 –d --name=jenkins jenkins
docker run –P sonarqube
docker run -p 8443:8080 -e KEYCLOAK_USER=admin -e KEYCLOAK_PASSWORD=admin -d jboss/keycloak

Dockerfile

.dockerignore

Command	Description
FROM	**creates a layer from the Docker image**
COPY(layer)	**adds files from your Docker client's current directory**
RUN(layer)	**builds your application with make**
CMD	**specifies what command to run within the container**
ADD(layer)	like COPY + extract and download from source url(not recommended)
EXPOSE	indicates the ports on which a container listens for connections
ENV	environment variable
ENTRYPOINT	like CMD but cannot be overriden
WORKDIR	define your current workplace
VOLUME	expose storage area

BASIC IDEA: COPY & RUN

#Practice task

- Dockerize our application
  - Spring
  - Postgres
  - Angular

Solution

ANGULAR + (.dockerignore)

```
FROM node:alpine as node
WORKDIR /app
COPY . .
RUN npm install && npm run build

FROM nginx:alpine
COPY --from=node /app/dist/angular /usr/share/nginx/html
CMD ["nginx", "-g", "daemon off;"]
```

SPRING + (.dockerignore)

```
FROM openjdk:11-jdk-slim-buster
COPY . .
RUN ./gradlew build
CMD ["java", "-jar", "/build/libs/spring.jar"]
```

POSTGRES

```
docker run –p 5432:5432 –e POSGRES_PASSWORD=docker –d postgres
```

Thanks for Lesson #1

Sources:
https://hub.docker.com/
https://docs.docker.com/

Docker networks

Docker volume

Docker-compose

#Practice task

Create docker-compose for our application

Solution

Thanks for Lesson #2

Sources:
https://hub.docker.com/
https://docs.docker.com/