Learning Consolidation

Containerize RESTful Services and Database by Using Docker

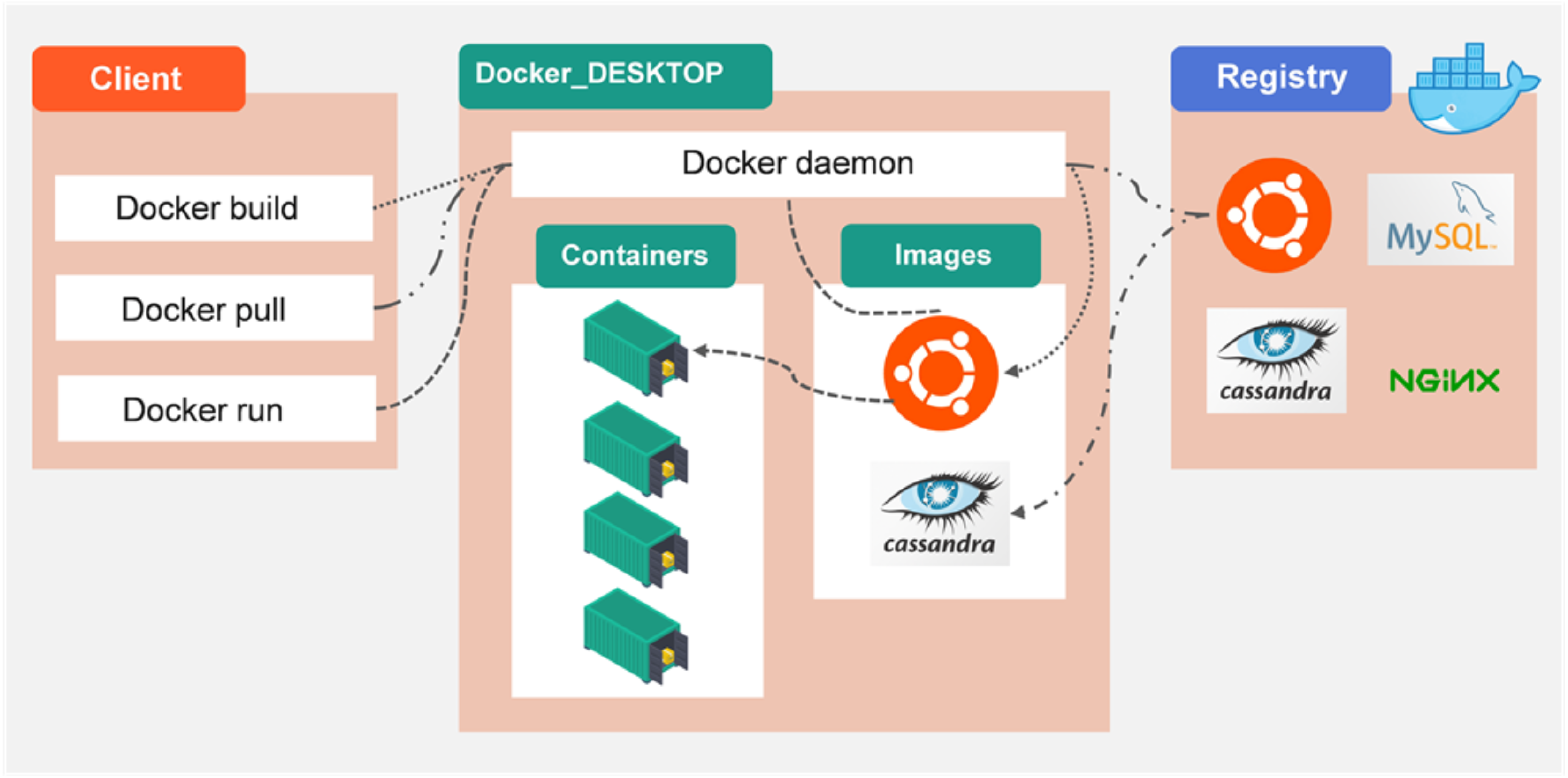
Menu

00:01:00:00:10 01/ 10

Learning Objectives

- Networking in Docker
- Docker Cheat Sheet

Docker Architecture



Docker Architecture (contd.)

- Docker uses a client-server architecture.
- The Docker client talks to the Docker daemon, which does the heavy lifting of building, running, and distributing the Docker containers.
- The Docker client and daemon can run on the same system or can connect to a Docker client on a remote Docker daemon.
- The Docker client and daemon communicate using a REST API over UNIX sockets.
- The Docker client is the PowerShell or command prompt where the Docker commands are run.
- The Docker Desktop is the host for running the Docker containers and building the images and contains the Docker daemon.
- The registry is the Docker hub where the predefined Docker images are present.

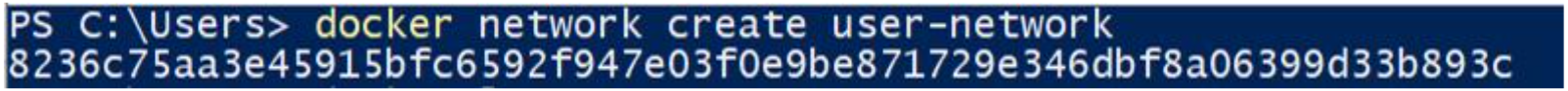
Communication Between Containers

- A Spring Boot application uses MySQL or MongoDB to store data.
- The Spring Boot application and the database can be Dockerized.
- Both containers must communicate with each other in the Docker environment.
- Since Docker communication happens through UNIX sockets, the Docker Desktop application is used.
- Docker Desktop is an easy-to-install application for Mac or Windows environments that helps to build and share containerized applications.
- Docker Desktop includes the Docker daemon `Dockerd`, the Docker client `Docker`, `Docker compose`, Docker Content Trust, Kubernetes, and Credential Helper.
- Docker expects the containers that need to communicate with each other to run on the same network.

Create a Docker Network

- The command below is used to create a Docker network:

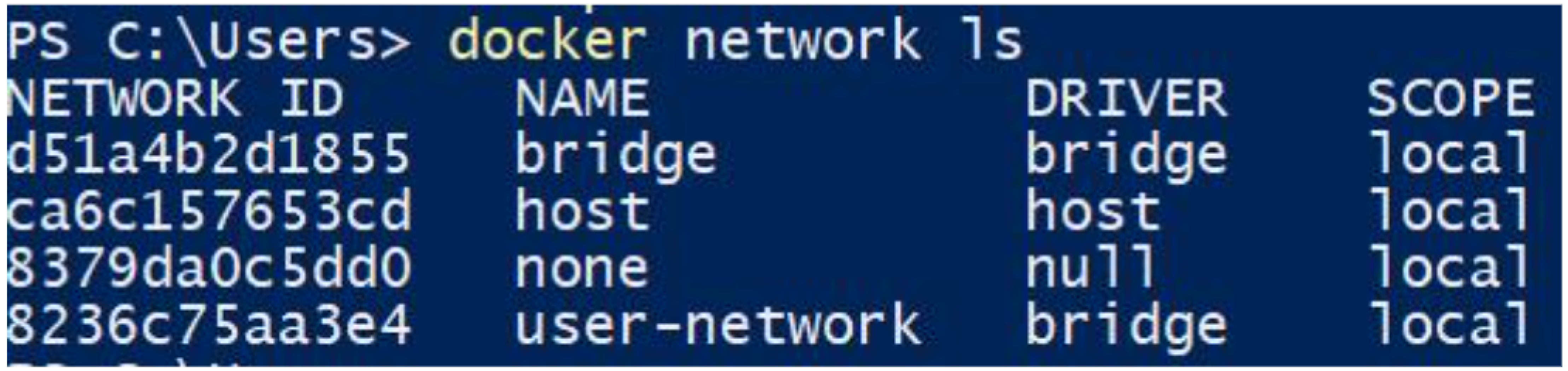
```
Docker network create <name of the network>
```



```
PS C:\Users> docker network create user-network
8236c75aa3e45915bfc6592f947e03f0e9be871729e346dbf8a06399d33b893c
```

- To view all the networks in Docker, use the command below:

```
Docker network ls
```



```
PS C:\Users> docker network ls
NETWORK ID     NAME           DRIVER    SCOPE
d51a4b2d1855   bridge         bridge    local
ca6c157653cd   host           host      local
8379da0c5dd0   none           null      local
8236c75aa3e4   user-network   bridge    local
```

Dockerize MySQL

1. Pull the MySQL image using the following command:

```
Docker pull mysql
```

2. Run the image to create the container on the network created earlier.
   - Note that in MySQL a password is required to connect to the MySQL shell.

```
Docker run -it --network user-network --name mysqlservice -e
MYSQL_ROOT_PASSWORD=root -d mysql
```

3. Execute the MySQL shell from the Docker container.

```
Docker exec -it mysqlservice bash
```

4. Enter the bash and give `mysql -u root -p` and enter the password ‘root’ as specified in step 2.

Dockerize the Spring Boot Application

- To Dockerize the Spring Boot application built earlier:
  - Create a Docker image of the Spring Boot application
  - Run the Docker image and create a container
- The Docker image of the application must be built from scratch.
- The image will not be available in Docker hub like Mongo or MySQL.

Docker Housekeeping

- Kill all running containers
  - `docker kill $(docker ps -q)`
- Delete all stopped containers
  - `docker rm $(docker ps -a -q)`
- Delete all exited containers
  - `docker rm $(docker ps -q -f status=exited)`
- Delete all images
  - `docker rmi $(docker images -q)`