**Commands**

# Check the version of Docker installed

docker --version

# Run a Docker container with a Python hello world application

docker run -p 5000:5000 in28min/hello-world-python:0.0.1.RELEASE

# Run a Docker container with a Java hello world application

docker run -p 5000:5000 in28min/hello-world-java:0.0.1.RELEASE

# Run a Docker container with a Node.js hello world application

docker run -p 5000:5000 in28min/hello-world-nodejs:0.0.1.RELEASE

# Run a Docker container with a Node.js hello world application in detached mode

docker run -d -p 5000:5000 in28min/hello-world-nodejs:0.0.1.RELEASE

# Run a Docker container with a Python hello world application in detached mode

docker run -d -p 5001:5000 in28min/hello-world-python:0.0.1.RELEASE

# View the logs of a Docker container

docker logs 04e52ff9270f5810eefe1f77222852dc1461c22440d4ecd6228b5c38f09d838e

# View the logs of a Docker container with a given container ID or name

docker logs c2ba

# List Docker images

docker images

# List running containers

docker container ls

# List all containers, including stopped ones

docker container ls -a

# Stop a running container with a given container ID

docker container stop f708b7ee1a8b

# Run a Docker container with a REST API in detached mode

docker run -d -p 5001:8080 in28min/hello-world-rest-api:0.0.1.RELEASE

# Pull a Docker image from a registry

docker pull mysql

# Search for Docker images in a registry

docker search mysql

# Show the history of a Docker image

docker image history in28min/hello-world-java:0.0.1.RELEASE

# Show the history of a Docker image with a given image ID

docker image history 100229ba687e

# Inspect a Docker image

docker image inspect 100229ba687e

# Remove a Docker image

docker image remove mysql

# Remove a Docker image with a given image name and tag

docker image remove in28min/hello-world-java:0.0.1.RELEASE

# Remove a Docker container with a given container ID or name

docker container rm 3e657ae9bd16

# List all containers, including stopped ones

docker container ls -a

# Pause a running container with a given container ID or name

docker container pause 832

# Unpause a paused container with a given container ID or name

docker container unpause 832

# Stop a running container with a given container ID or name

docker container stop 832

# Inspect a Docker container with a given container ID or name

docker container inspect ff521fa58db3

# Remove all stopped containers

docker container prune

# Show Docker system information

docker system

# Show Docker disk usage

docker system df

# Show detailed Docker system information

docker system info

# Remove all unused Docker resources

docker system prune -a

# Display the top resource-consuming processes of a Docker container with a given container ID or name

docker top 9009722eac4d

# Show live resource usage statistics of a Docker container with a given container ID or name

docker stats 9009722eac4d

# Run a Docker container with a Java hello world application, limited to 512MB memory

docker container run -p 5000:5000 -d -m 512m in28min/hello-world-java:0.0.1.RELEASE

# Run a Docker container with a Java hello world application, limited to 512MB memory and CPU quota of 50%

docker container run -p 5000:5000 -d -m 512m --cpu-quota=50000 in28min/hello-world-java:0.0.1.RELEASE

# Show Docker system events

docker system events

# Show live resource usage statistics of a Docker container with a given container ID or name

docker container stats 4faca1ea914e3e4587d1d790948ec6cb8fa34f26e900c12632fd64d4722fd59a

# Show live resource usage statistics of a Docker container with a given container ID or name

docker stats 42f170966ce613d2a16d7404495af7b3295e01aeb9142e1fa1762bbdc581f502

# Change directory to the Python hello world project

cd /in28Minutes/git/devops-master-class/projects/hello-world/hello-world-python

# Build a Docker image with a given tag

docker build -t in28min/hello-world-python:0.0.2.RELEASE .

# Run a Docker container with a Python hello world application, mapped to port 5000

docker run -p 5000:5000 -d in28min/hello-world-python:0.0.2.RELEASE

# Show the history of a Docker image with a given image ID

docker history e66dc383f7a0

# Push a Docker image to a registry

docker push in28min/hello-world-python:0.0.2.RELEASE

# Change directory to the Node.js hello world project

cd ../hello-world-nodejs/

# Build a Docker image with a given tag

docker build -t in28min/hello-world-nodejs:0.0.2.RELEASE .

# Run a Docker container with a Node.js hello world application, mapped to port 5000

docker container run -d -p 5000:5000 in28min/hello-world-nodejs:0.0.2.RELEASE

# Push a Docker image to a registry

docker push in28min/hello-world-nodejs:0.0.2.RELEASE

# Change directory to the Java hello world project

cd ../hello-world-java/

# Build a Docker image with a given tag

docker build -t in28min/hello-world-java:0.0.2.RELEASE .

# Run a Docker container with a Java hello world application, mapped to port 5000

docker run -d -p 5000:5000 in28min/hello-world-java:0.0.2.RELEASE

# Push a Docker image to a registry

docker push in28min/hello-world-java:0.0.2.RELEASE

# Run a Docker container with a Node.js hello world application, mapped to port 5001 and ping google.com

docker run -d -p 5001:5000 in28min/hello-world-nodejs:0.0.3.RELEASE ping google.com

# Run a Docker container with a currency exchange service

docker run -d -p 8000:8000 --name=currency-exchange in28min/currency-exchange:0.0.1-RELEASE

# Run a Docker container with a currency conversion service

docker run -d -p 8100:8100 --name=currency-conversion in28min/currency-conversion:0.0.1-RELEASE

# List Docker networks

docker network ls

# Inspect a Docker network with a given network name

docker network inspect bridge

# Run a Docker container with a currency conversion service, linked to the currency exchange service and environment variable set

docker run -d -p 8100:8100 --env CURRENCY\_EXCHANGE\_SERVICE\_HOST=http://currency-exchange --name=currency-conversion --link currency-exchange in28min/currency-conversion:0.0.1-RELEASE

# Create a Docker network with a given network name

docker network create currency-network

# Stop a running container with a given container ID or name

docker container stop currency-exchange

# Stop a running container with a given container ID or name

docker container stop currency-conversion

# Run a Docker container with a currency exchange service, connected to the currency-network

docker run -d -p 8000:8000 --name=currency-exchange --network=currency-network in28min/currency-exchange:0.0.1-RELEASE

# Run a Docker container with a currency conversion service, connected to the currency-network and environment variable set

docker run -d -p 8100:8100 --env CURRENCY\_EXCHANGE\_SERVICE\_HOST=http://currency-exchange --name=currency-conversion --network=currency-network in28min/currency-conversion:0.0.1-RELEASE

# Check the version of Docker Compose installed

docker-compose --version

# Change directory to the microservices folder

cd ../../microservices/

# Start the services defined in the Docker Compose file

docker-compose up

# Start the services defined in the Docker Compose file in detached mode

docker-compose up -d

# List running containers

docker container ls

# List Docker networks

docker network ls

# Inspect a Docker network with a given network name

docker network inspect microservices\_currency-compose-network

# Stop the services defined in the Docker Compose file

docker-compose down

# List all containers, including stopped ones

docker container ls -a

# Remove all unused Docker resources

docker system prune -a

# Validate the Docker Compose file

docker-compose config

# List Docker images used by the services defined in the Docker Compose file

docker-compose images

# List the running services defined in the Docker Compose file

docker-compose ps

# Display the top resource-consuming processes of the services defined in the Docker Compose file

docker-compose top

# Build a Docker image for a Java hello world application with a given tag

docker build -t in28min/hello-world-java:0.0.1.RELEASE .

# Push the Docker image with the specified tag to a registry

docker push in28min/hello-world-java:0.0.1.RELEASE

# Build a Docker image for a Python hello world application with a given tag

docker build -t in28min/hello-world-python:0.0.1.RELEASE .

# Push the Docker image with the specified tag to a registry

docker push in28min/hello-world-python:0.0.1.RELEASE

# Build a Docker image for a Node.js hello world application with a given tag

docker build -t in28min/hello-world-nodejs:0.0.1.RELEASE .

# Push the Docker image with the specified tag to a registry

docker push in28min/hello-world-nodejs:0.0.1.RELEASE