Docker

Ben Gavan

June 26, 2020

1 Orientation and Setup

1.1 Test Docker Version

```
docker --version
```

$$hygv \tag{1}$$

2 The Dockerfile

2.1 Where is the code - What directory the code is in

2.2 What port

3 Building an image

To build an image without binding to a port

```
docker build -t <image name> .
```

where '.' represents the current directory.

To build an image binding to a port

```
 docker run -d -p 80:80 my-app
```

3.1 Building an image from a Dockerfile embedded in the source

Run this in the directory you are building the dockerfile from

```
docker build -f path/to/Dockerfile .
```

4 View all images

To view all images built on the current machine/user

```
docker images
```

5 Running an image

Running an image without binding to a port

```
content...
```

Running an image binding to port 8081 externally and then routing to port 8080 internally inside of the container

```
$ docker run -d -p 8081:8080 go-docker
```

-d means run this detached, as a daemon, eg, not dependent on the terminal session
-p means map ports; mapping <host machine port>:<to docker container port>

6 Information about an image

6.1 History of an image

```
docker image history <image-name>
```

7 List of current running containers

To view the list of current containers

```
  docker container ls
```

8 Stopping a container

To stop a container, using the container ID (found in Section 7)

```
$ docker container stop <container ID>
```

9 Pushing a container to docker hub

```
docker push <username>/<app name>
```

for example, for the tutorial when you sign up to Docker,

```
docker push bengavan/cheers2019
```

10 Deploying a Docker container on AWS

10.1 Deploying a Docker container to AWS from Docker hub

10.2 Deploying a Docker container to AWS from Docker hub

11 Dockerfile templates

11.1 Simple Go image

```
# To be built from the /Services directory using:
# docker build -f path/to/Dockerfile .

FROM golang:alpine as builder
RUN mkdir /build
ADD . /build/
WORKDIR /build

RUN apk add --no-cache git \
  && go get github.com/johnnadratowski/golang-neo4j-bolt-driver \
  && go get github.com/lib/pq \
  && go get golang.org/x/crypto/bcrypt \
  && apk del git

#RUN go get github.com/johnnadratowski/golang-neo4j-bolt-driver
RUN CGO_ENABLED=0 GOOS=linux go build -a -installsuffix cgo -ldflags '-extldflags "-static"' -o main ./Authentication

FROM scratch
COPY --from=builder /build/main .
COPY Shared/utils Shared/utils
WORKDIR .
CMD ["./main"]
```