Exercise Worksheet

www.vomtom.at

From the Course:
Understanding Docker Run, Dockerfile, Docker-Compose for Beginners

Build your own images with custom configuration using docker compose and the docker-compose.yml file

Now you know two things:

1. You can use a Dockerfile to generate an image
2. You can mount folders inside your directory automatically

We're one step close to building our dream dev-environment.

What if we desperately need a different configuration than provided in the php:7.2-apache image? We change it and rebuild it using our own Dockerfile to our own image.

Use this docker-compose.yml file:

```
version: '3'

services:
  phpapp:
    build:
      context: ./
      dockerfile: Dockerfile
    image: phpapp:123
    ports:
      - "8080:80"
    volumes:
      - "./:/var/www/html"
    container_name: my-php-app
```

And this Dockerfile for starter:

```
FROM php:7.2-apache
```

And this index.php:

```
<?php

phpinfo();
```

Then head to the terminal and type in:

```
docker-compose up
```

- Observe that the image gets built
- The image name is "phpapp:123" and not the directory_name… anymore.
- It mounts again the volume

- It forwards the port 80
- And the container name is “my-php-app”

Open the http://localhost:8080 . It should show you the php information.

This isn’t impressive *yet*. But what if you desperately need mysqli and the php-intl extension installed inside your docker container. Surely you can enter the container by “docker exec -it my-php-app /bin/bash” and then apt-get install etc… but there is a better way. Why not directly embed it into the container?

```
Ctrl-c
```

- Stop the container

```
docker-compose rm
```

- Remove the container

Extend the Dockerfile so that it looks like this:

```
FROM php:7.2-apache

RUN apt-get -y update \
&& apt-get install -y libicu-dev \
&& docker-php-ext-configure intl \
&& docker-php-ext-install intl

RUN docker-php-ext-install mysqli && docker-php-ext-enable mysqli
```

Then run

```
docker-compose up --build
```

- This should rebuild your containers
- You should see a lot of compiler-output
- Once done it should open apache

http://localhost:8080 and text-search for “mysqli” and “intl”. There should be these packages available now.

```
Ctrl-c
```

- Stop the running container

```
docker-compose rm
```

- remove the container