Exercise Worksheet

www.vomtom.at

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

Running multiple docker containers from the command line step by step

This shows the difference between containers and images. We will create two *containers* (linux1, linux2) based on the same *image* (ubuntu)

```
docker run -it -d --rm --name linux1 ubuntu /bin/bash
```

- creates container “linux1”
- additional flags:
  - -d starts the container as “detached”. Use “docker attach” to attach to it later on.
  - --rm cleans up the container after stopping. The container will be removed, basically the same as “docker rm container_identifier” after stopping the container. So everything is kept tidy.
  - --name will give the container a dedicated name, which makes it easier to address the container later on.

```
docker run -it -d --rm --name linux2 ubuntu /bin/bash
```

- Creates container “linux2”

```
Cli1: docker attach linux1
```

- Attaches to container linux1

```
Cli1: ls
```

- Lists the file system on linux1

```
Cli1: mkdir mylinux1
```

- Creates a new directory on container linux1

```
Cli1: ls
```

- Shows that “mylinux1” was created

```
Cli2: docker attach linux2
```

- Attaches to container linux2

```
Cli2: ls
```

- Shows that the directory of linux2 is different than linux1, although they are both from the same image “ubuntu”
- They are separated, they don’t share their file-system
- The bash process is isolated in the container

```
Cli2: exit
```

- This will exit the bash and also remove the container because of the --rm command

```
Cli2: docker ps -a
```

- Shows only one container which is running, the other one got removed

```
Cli1: exit
```

- Exits the only running container linux1 and deletes the container

```
docker ps -a
```

- Shows nothing anymore