|  |  |
| --- | --- |
| üK Modul 223 |  |
| Lernjournal |  |
| Phearum Svay |  |

# Lernjournal

Heute (Mittwoch, 17. März 2021) haben wir gelernt, wie wir richtig mit Docker arbeiten. Vorher kannte ich nur die Basics von Docker. Also wie ich ein Container erstellen kann, damit eine Datenbank darauf laufen kann. Wir haben zuerst die Commands zum Erstellen eines Containers mit einem Image angeschaut.

## Container erstellen

Mit dem folgenden Befehl wird ein Container erstellt:

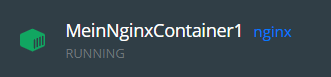
*docker run [OPTIONS] IMAGE [COMMAND] [ARG…]*

Die Brackets sind optionale Angaben. In diesem Befehl braucht es mindestens ein Image. Als Aufgabe mussten wir einen Container mit einem Namen der eigenen Wahl erstellen. Dieser Container musste mit dem Image *nginx* erstellt werden.

Der Befehl für diese Aufgabe sollte folgendermassen aussehen:

*docker run --name MeinNginxContainer1 nginx*

Damit wird jetzt ein Container erstellt mit dem Namen «MeinNginxContainer1» und dem Image «nginx».



### Port & Volume zuweisen

Die nächste Aufgabe bzw. eine Erweiterung der letzten Aufgabe ist es ein Container zu erstellen, der einen Port und Volume zugewiesen hat. Dieser Port wird 8080 (80:80) sein damit dann per localhost eine index.html aufgerufen werden kann. Die index.html Seite wird sich im Volume befinden, der auch dem Container zugewiesen wird.

Beim Befehl wird zusätzlich -v [VOLUME\_PATH] und -p [PORT] dazu kommen:

*docker run --name MeinNginxContainer1 -v C:\Users\svayp\Desktop\MeinNginxContainerOrdner1:/usr/share/nginx/html -p 80:80 nginx*

Jetzt existiert ein Container mit einem gewählten Namen, zugewiesenen Volume, Port und läuft auf nginx.



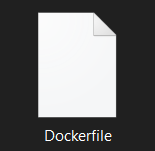
*Reflexion:* Nach dieser Aufgabe wurde mir klarer, wie genau der Befehl zum Erstellen eines Containers aufgebaut ist. Das Verständnis für die einzelnen «Teile» im Befehl (-v Volume, -p Port, --name NameDesContainers) wurde auch klarer für mich. Beim Erstellen eines Containers ist es noch möglich -d zu verwenden, was «detached» heisst. Jedoch mach dies kein Unterschied, weil CTRL + C und -d die gleiche Aufgabe erfüllen.

## Image erstellen

Nachdem der Container mit einer HTML-Seite funktionierte, ging es zur nächsten Aufgabe. Ein Image ist ein Speicherabbild oder ein Datenträgerabbild und eine Kopie eines Datenträgers oder Datenspeichers. Diese Images werden vom Container gebraucht, um überhaupt ein «Inhalt» zu haben. In der Informatik sind Images meistens Erstellungsoftware. Erstellungssoftware ist, wie es der Name eigentlich schon erklärt, Software, die dir hilft Sachen zu erstellen. Besser gesagt sind es meistens Gerüste, die dir helfen eine bestimmte Aufgabe zu erfüllen. Images kommen mit Funktionen, die alle eine bestimmte Aufgabe erledigen. Es gibt z.B. images von Windows, macOs oder Unixoid/Unix/Linux die einfachen Betriebssysteme sind. Von MariaDB oder CentOS gibt es auch Images. MariaDB ist ein DB-Managementsystem und CentOS ein UNIX System. So viel zu Images.

Die nächste Aufgabe ist es ein eigenes Image anhand des Containers zu erstellen. Das eigene Image beinhaltet dann das nginx Image und die HTML-Seite. Es ist eigentlich dann eine Erweiterung des nginx Image.

Um ein Image zu erstellen, wird der *docker build* Befehl verwendet. Jedoch gibt es dazu noch eine Voraussetzung, wenn ein eigenes Image erstellt wird. Es muss ein Dockerfile vorhanden sein. Das Dockerfile ist ein Text Dokument, welches alle Commands enthält, die helfen ein Image zu montieren. Dieses File ist aber kein .txt-File, sondern ein einfaches File ohne Dateiendung. Also sieht das File folgendermassen aus:





### Dockerfile

Ein Image besteht aus read-only Layers. Jedes dieser Layer repräsentiert eine Dockerfile Instruktion. Der Inhalt einess Dockerfiles:



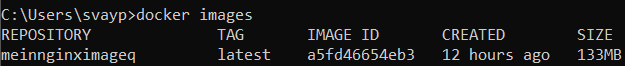
Zurück zum Erstellen eines eigenen Image. Sobald dieser Dockerfile steht ist es möglich sein eigenes Image zu erstellen. Der Befehl ist so aufgebaut:

*docker build [OPTIONS] PATH | URL | -*

Wie immer sind die Brackets nicht notwendig. Der PATH ist der Pfad in dem das Dockerfile liegt. Für die Aufgabe sieht der konkrete Befehl so aus:

*docker build -t meinnginximage1 C:\Users\svayp\Desktop\MeinNginxContainerOrdner1*

Wird der Befehl ausgeführt, erstellt es ein Image namens «meinnginximageq». Der Name eines Image darf nur im lowercase sein und keine anderen Zeichen als Buchstaben beinhalten.



Damit wurde ein eigenes Image erstellt.

## Images auf Repository pushen

Es gibt eine Website namens «dockerhub», die per hub.docker.com erreicht werden kann. Wenn du ein Account auf dieser Seite besitzt, kannst du Images auf dein Repository auf dieser Seite pushen.

## Netzwerk erstellen

### Container auf Netzwerk zuweisen