Docker CheatSheet

Tobias Köhler, Niklas Nikisch

Hochschule Mannheim, Fakultät für Informatik

Build

Baut ein Image aus einem Dockerfile im Momentanen Verzeichnis und *tagged* es.
docker build -t appname .

Listet alle Docker Images auf.
docker images

Entfernt das angegeben Image.
docker image rm ImageId

Ship

Einloggen um auf die Registry der Dockerseite zuzugreifen.
docker login

Lädt das getaggte Image in die Registry hoch.
docker push username/repository:**tag**

Tagged das Image für späteres hochladen.
docker tag <**image**> **username**/repository:**tag**

Zieht ein Image oder ein Repository aus der Registry.
docker pull appname

Suche in der Registry nach einem Image.
docker search name

Others

Listet alle laufenden Container auf.
docker container ls

Stoppt den angegebenen Container.
docker container stop ContainerID

Entfernt den angegebenen Container.
docker container rm ContainerID

Listet alle laufenden Services auf.
docker service ls ContainerID†

Run

Mit **docker run** können Container anhand von Images erstellt und gestartet werden. Das angegebene Baseimage wird zuerst lokal gesucht. Findet es keins, sucht es auf dem Docker Hub.

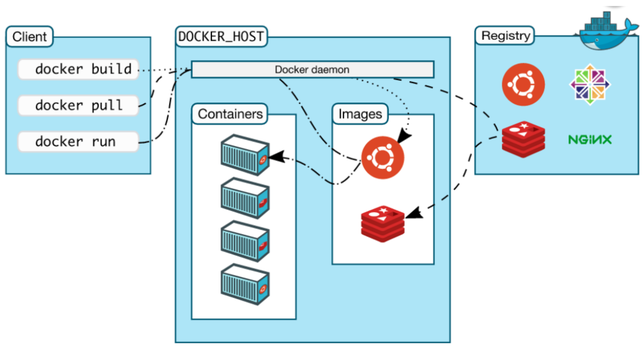
docker run appname

Startet die App aus dem Repository:
docker run username/repository:tag

Options:

- **-d** – Startet den Container Im Hintergrund
- **-it** – Startet den Container im interactive-Modus, sodass man auf dem Container arbeiten kann.
- **-p** <**host_port**>:<**container_port**> – Mappt einen Port vom Container auf das Hostsystem
- **–expose** – Öffnet einen oder mehrere Ports
- **-P** – Mappt alle offenen Ports vom Container auf zufällige Hostsystem-Ports.
- **–name** – Gibt dem Container einen Namen, der anstelle der ID für Befehle verwendet werden kann
- **-v** <**Datei_oder_Ordner_zum_mounten**> [>**Ziel_auf_dem_Container**>] – Mountet den angegebenen Ordner vom Hostsystem auf den Container, sodass dieser auf Dateien zugreifen kann.
- **-e** <**VARIABLEN_NAME**> = <**VARIABLEN_WERT**> – Definiert eine Umgebungsvariable für den Container.
- **–link** <**container_name**> [:<**name_in_new_container**>] – Linkt einen bestehenden Container in den neuen, sodass sich der Container mit dem anderen Verbinden kann.

Docker Ecosystem



Dockerfile

```
# Nutzen eines Vaterimages
FROM (Vaterimage)

# Setzt den Ordner in dem alle Befehle ausgeführt werden
WORKDIR (Ordner)

# Kopiert Daten von (src) nach (des)
ADD (url) (des)
COPY (src) (des)

# Gibt einen Port frei
EXPOSE (Nummer)

# Definiert ein Argument. Kann mit $<ARG_NAME> darauf zugreifen.
ARG (Argument)

# Definiert eine Umgebungsvariable
ENV (Umgebungsvariable)

# Startet ein Programm mit übergebenen Parametern
CMD ["Programm", "Parameter"]

# Führt einen beliebigen Befehl auf dem Container aus
RUN (Befehl)
```

TIPP: Da bei jedem RUN ein neuer Container gecacht wird, kann man Befehle mit && aneinander hängen um die Anzahl der Images/Layer zu reduzieren.

Bsp.: RUN apt-get update && apt-get upgrade

Tobias Köhler

Niklas Nikisch