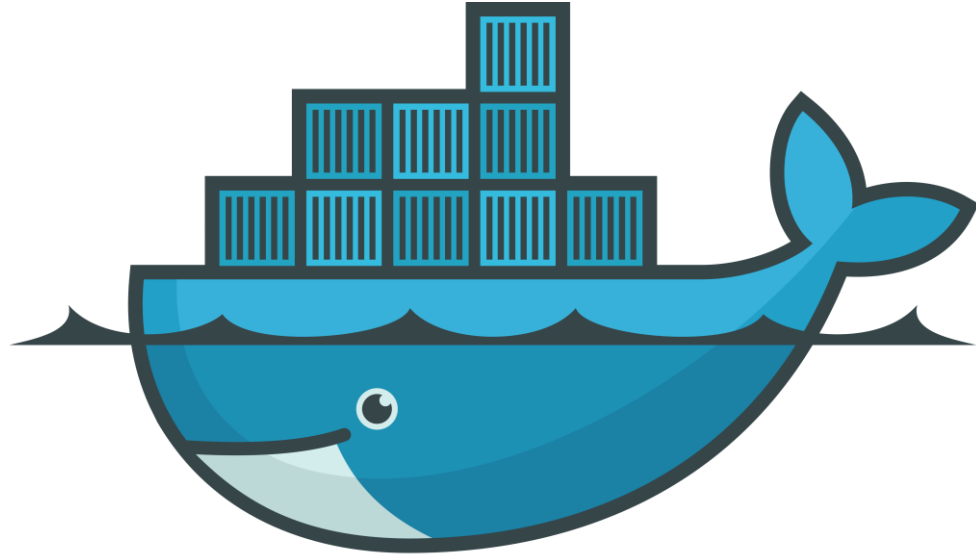


Container / Docker

- <https://github.com/LNA-DEV/Workshop-Docker>

- Wenig Overhead
- Sehr gute Portabilität
- Höhere Effizienz
- Bessere Developer-Experience
- Schnelle Startgeschwindigkeit
- Einfaches Dependencymanagement

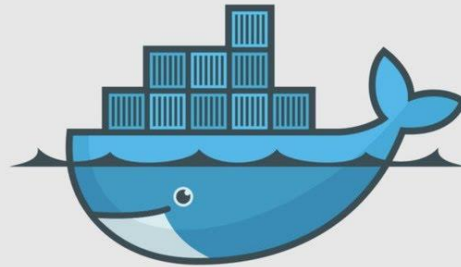


docker

-



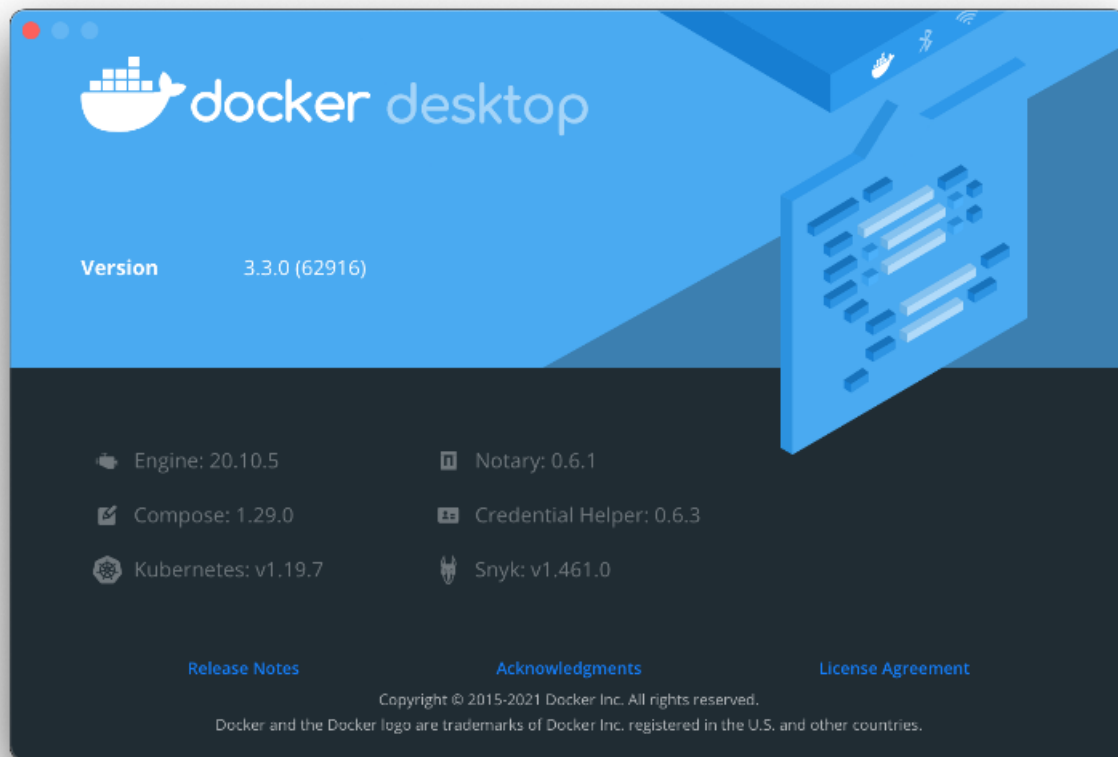
Docker Hub



docker

- `docker ps [OPTIONS]`
- `docker pull [OPTIONS] NAME[:TAG|@DIGEST]`
- `docker run [OPTIONS] IMAGE [COMMAND] [ARG...]`
- `docker images [OPTIONS] [REPOSITORY[:TAG]]`

- Übung 01 - Webserver
-



```
# syntax=docker/dockerfile:1
FROM ubuntu:18.04
COPY . /app
RUN make /app
CMD python /app/app.py
```

- Übung 02 und 03 - Dockerfile
-

- Volumes
- Bind Mounts
- (Tmpfs Mounts)

- Docker managed
- Mehr Isolation vom Host
- Für Docker gemacht =>
 - Flexibler
 - Können zwischen Containern geteilt werden
- Erstellen: `docker volume create my_volume`
- Benutzen: `docker run -d --name my_container -v my_volume:/data my_image`

- Übung 04 - Volumes

- Dateisystemmount
- Kontrolle über den Pfad der Daten
- Weniger Sicherheitsüberprüfungen wie Volumes

- Benutzen: `docker run -d --name my_container -v /host/data:/container/data my_image`

- Übung 05 - Mounts
-

- Komplexe Apps
- Mehrere Container
- Einfache (wiederverwendbare) Konfiguration => yaml
- Development oder Singlehost Production Deployments

```
1  version: '3.8'
2
3  services:
4    web:
5      image: nginx:latest
6      ports:
7        - "8080:80"
8      volumes:
9        - ./data:/usr/share/nginx/html
10     networks:
11       - frontend
12
13     db:
14       image: mysql:5.7
15       environment:
16         MYSQL_ROOT_PASSWORD: example
17       volumes:
18         - db_data:/var/lib/mysql
19       networks:
20         - backend
21
22     networks:
23       frontend:
24       backend:
25
26     volumes:
27       db_data:
```

```
1  version: '3.8'
2
3  services:
4    web:
5      build:
6        context: ./app
7      ports:
8        - "5000:5000"
9      environment:
10        - REDIS_HOST=redis
11      networks:
12        - my_network
13
14     redis:
15       image: redis:alpine
16       ports:
17         - "6379:6379"
18       networks:
19         - my_network
20
21     networks:
22       my_network:
```

- Übung 06 - Compose
-

- Erlaubt Kommunikation
- Verschiedene Driver
 - Bridge (default)
 - Host
 - None
- Commands
 - `docker network create --driver <driver_name> <network_name>`
 - `docker network ls`
 - `docker network connect <network_name> <container_name>`
- Auch in Compose nutzbar

- Übung 07 - Networks

- Von Docker ausgehend 2015 ins Leben gerufen
- Ermöglicht Kompatibilität mit anderen Container Runtimes
- Von der Linux Foundation verwaltet
- Gesponsert von: AWS, Docker, Google, RedHat, Microsoft, Oracle...
- Ermöglicht das Ausführen eines Containers mit anderen Container Runtimes
- Nur eine Registry => Verschiedene Container in einer Registry



- Übung 08 – Freie Übung
-

Erfahren Sie mehr unter www.itelio.com

Copyright itelio GmbH, Änderungen vorbehalten.

Die Garantien für itelio Produkte und Services werden ausschließlich in der entsprechenden, zum Produkt oder Service gehörigen Garantieerklärung beschrieben. Aus dem vorliegenden Dokument sind keine weiterreichenden Garantieansprüche abzuleiten. itelio übernimmt keine Verantwortung für die Richtigkeit und Vollständigkeit der Angaben in diesem Dokument.

itelio

itelio GmbH

Franz-Larcher-Str. 4

D-83088 Kiefersfelden

Fon +49-8033-6978- 0

Fax +49-8033-6978-91

info@itelio.com | www.itelio.com