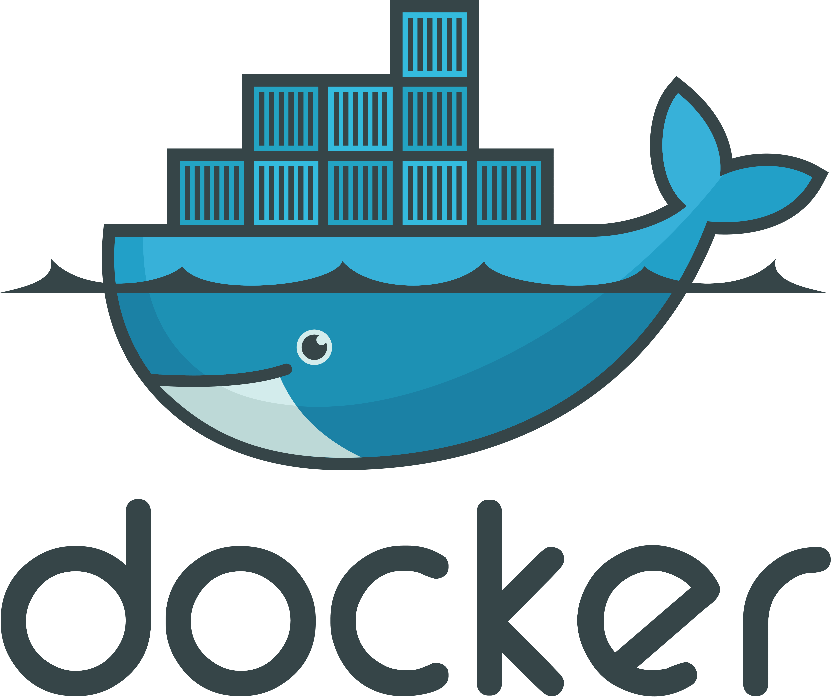
Dokumentation vom Modul 169 Tag 2

Erstellt von: Vladan Vranjes

Klasse Inf2022c

Kontakt: [vvr140992@stud.gibb.ch](mailto:vvr140992@stud.gibb.ch)



Contents

[App Version 1 1](#_Toc1529744775)

[Wichtige Befehle: 2](#_Toc989056613)

[Container erstellen 2](#_Toc601472894)

[Netzwerkverbindungen erhalten 2](#_Toc1959223006)

[Container im Netz starten 2](#_Toc1593800410)

[Extra: 2](#_Toc45001859)

[PrintScreen ihrer Version 1 mit Ihrem Namen als to-do Task 2](#_Toc1468085163)

[PrintScreen der Images bei (Gibb) GitLab 2](#_Toc288220518)

[Aufgabe 1 bei "Ein Image Pushen", Halten Sie alle befehle im Portfolio fest. 2](#_Toc1300364893)

[App Version 2 Frontend 4](#_Toc1980182110)

[PrintScreen ihrer Version 2 mit Ihrem Namen als to-do Task 4](#_Toc849214123)

[PrintScreen der Images bei GitLab 4](#_Toc1456004095)

# App Version 1

## Wichtige Befehle:

### Container erstellen

Im Ordner web-frontend:

docker image build -t todo-app:v1 .

Im Ordner redis-slave:

docker build -t redis-slave:v1 .

Im Ordner redis-master:

docker build -t redis-master:v1 .

### Netzwerkverbindungen erhalten

docker network create todoapp\_network

### Container im Netz starten

docker run --net=todoapp\_network --name=redis-master -d redis-master:v1

docker run --net=todoapp\_network --name=redis-slave -d redis-slave:v1

docker run --net=todoapp\_network --name=frontend -d -p 3000:3000 todo-app:v1

### Extra:

Damit der Container bei einem Fehler nicht gestoppt wird, könnten wir ihn mit dem Parameter --restart unless-stopped ausführen.

# PrintScreen ihrer Version 1 mit Ihrem Namen als to-do Task

Ein Bild, das Text, Screenshot, Zahl, Schrift enthält.

Automatisch generierte Beschreibung

# PrintScreen der Images bei (Gibb) GitLab

Ein Bild, das Text, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

# Aufgabe 1 bei "Ein Image Pushen", Halten Sie alle befehle im Portfolio fest.

Um ein Image zu pushen muss man im Git zuerst ein Access Token generieren.

Drücke dafür auf Edit profile

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Hier kannst du unter Access Tokens einen generieren lassen.

Dieser sieht dann so aus:

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

Gehe jetzt zu deiner VM und nutze den Befehl:

docker login git-registry.gibb.ch

Gib da deinen Nutzernamen ein und als Passwort das eben erstellte Token.

Zukünftig können Sie sich dann mit « docker login -u (nutzer) -p (Token)» einloggen.

Docker login git-registry.gibb.ch -u vvr140992 -p glpat-WrH\_yGy4hsM2NDZfayvh

Tagge als nächstes das Image

docker image tag todo-app:v1 git-registry.gibb.ch/vvr140992/modul\_169\_appv1/todo-app:v1

Pushe dies danach mit:

docker push git-registry.gibb.ch/ vvr140992/modul\_169\_appv1/todo-app:v1

Wiederhole das für die anderen Images auch

# App Version 2 Frontend

Im Ordner redis-slave

docker build -t redis-slave:v1 .

Im Ordner redis-master

docker build -t redis-master:v1 .

Im ordner web-frontendv2

docker image build -t todo-app:v1 .

docker network create todoapp\_network

docker run --net=todoapp\_network --name=redis-master -d redis-master:v1

docker run --net=todoapp\_network --name=redis-slave -d redis-slave:v1

docker run --net=todoapp\_network --name=frontend -d -p 3000:3000 todo-app:v2

# PrintScreen ihrer Version 2 mit Ihrem Namen als to-do Task

Ein Bild, das Text, Screenshot, Schrift, Logo enthält.

Automatisch generierte Beschreibung

# PrintScreen der Images bei GitLab

Tagge als nächstes das Image

docker image tag todo-app:v2 git-registry.gibb.ch/vvr140992/modul\_169\_appv1

/todo-app:v2

Pushe dies danach mit:

docker push git-registry.gibb.ch/vvr140992/modul\_169\_appv1/todo-app:v2

Ein Bild, das Text, Quittung, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung