

Intern: Docker

Was ist das und wie benutze ich das?



Thomas Witzenrath ■ 15.05.2017

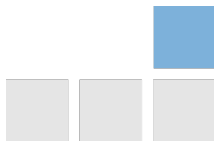


Was ist Docker?

- Docker verpackt Anwendungen in Container
- Benutzt Namespaces und CGroups

Vorteile:

- Anwendung kommt mit allen Dependencies
- saubere Trennung von Anwendungen
- Leichtgewichtig - im Vergleich zu KVM/VMWare/..

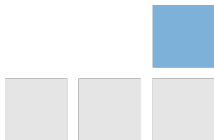




Fertige Docker Images findet man auf <https://hub.docker.com>

```
$ docker run hello-world
```

- holt das hello-world Image
- erzeugt einen neuen Container
- startet den Container und gibt den Output direkt aus





```
$ docker ps [-a]
```

Zeigt laufende (bzw. alle) Container an

```
$ docker stop <container>
```

```
$ docker start <container>
```

Stoppt/Startet einen bereits definierten Container

```
$ docker logs [-f] <container>
```

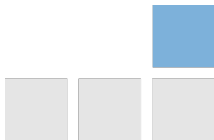
Zeigt Ausgabe (stdout/stderr) des Containers

```
$ docker exec -ti <container> <programm>
```

führt das Programm im Container interaktiv aus

```
$ docker rm <container>
```

löscht den Container





Docker benutzen

`$ docker images`

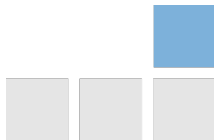
Listet die im System vorhandenen Images

`$ docker pull <name>`

Holt das Image <name> von Docker-Hub

`$ docker rmi <image>`

Löscht ein Image

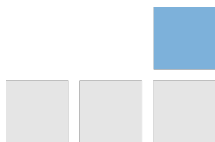


■ Problem

DockerHub Images

- sind nicht unbedingt vertrauenswürdig
- werden potentiell nicht aktualisiert

Daher wollen wir unsere eigenen Images bauen



Ein eigenes Image erzeugen - das Dockerfile

```
FROM alpine:latest
EXPOSE 8080
RUN apk update && apk upgrade
RUN apk add python supervisor
ADD content/ /
RUN chmod +x /initdocker

CMD ['/initdocker']
```

Stellt ein minimales System mit python und supervisor bereit in das unser "Content" kopiert wird. Das Script initdocker startet in diesem Fall supervisord, der einen python-Server startet. Der Port 8080 wird nach aussen exposed.



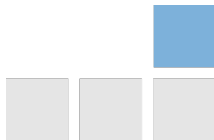
Ein eigenes Image erzeugen - Image bauen

```
$ docker build -t tralios/beispiel .
```

Baut anhand des Dockerfile im aktuellen Pfad ein Image mit dem Namen tralios/beispiel Dieses können wir mit

```
$ docker run tralios/beispiel
```

starten





Container richtig starten

```
$ docker run -d -p 127.0.0.1:8081:8080 --name beispiel tralios/beispiel
```

- erzeugt einen Container aus dem Image tralios/beispiel
- gibt diesem den Namen "beispiel"
- der Container wird im Daemon-Mode (-d) gestartet
- der Port 8080 des Containers wird auf 127.0.0.1:8081 durchgeschleift



Wie gehts weiter?

- docker swarm
- kubernetes
- dokku
- ansible Module `docker_image`, `docker_container`
- ansible-container

