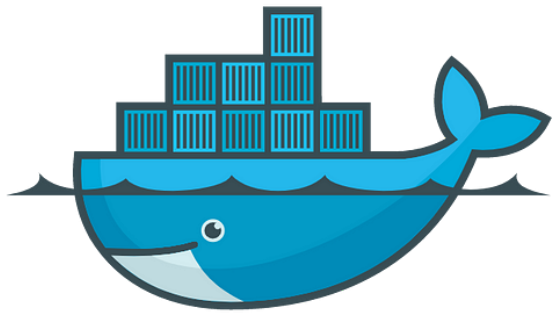


How to docker?

Docker something

Julia Winkler

19.06.2024

Gliederung

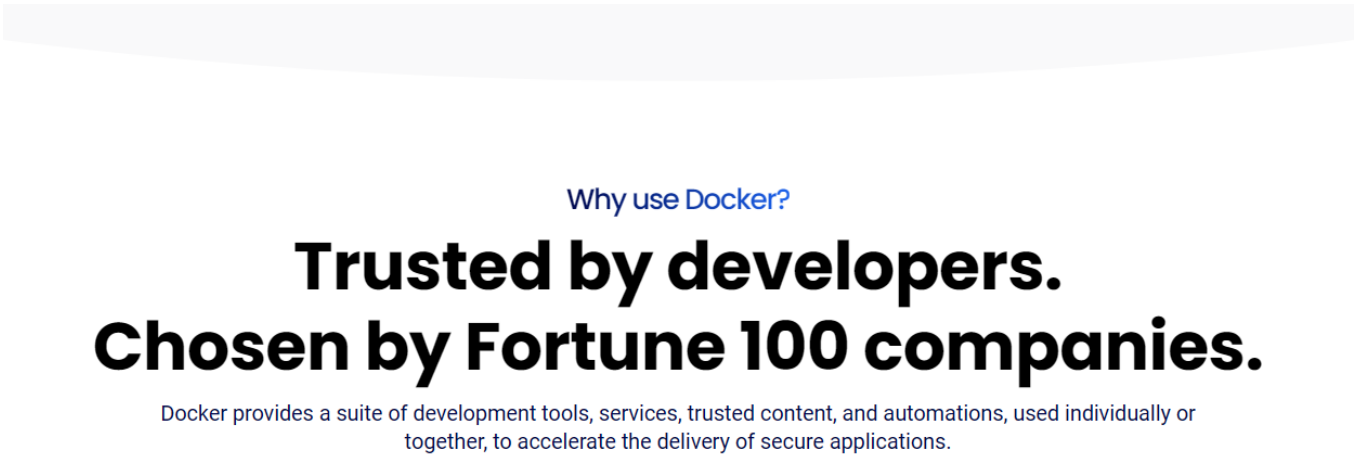
Wiso, Weshalb, Warum?

Why use Docker?

Trusted by developers.
Chosen by Fortune 100 companies.

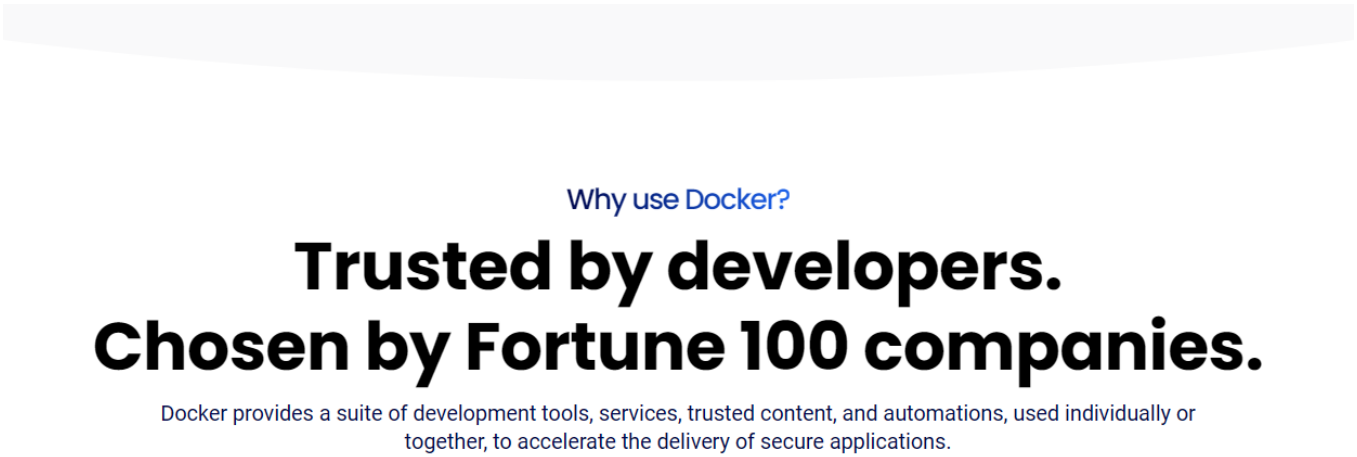
Docker provides a suite of development tools, services, trusted content, and automations, used individually or together, to accelerate the delivery of secure applications.

Wiso, Weshalb, Warum?



"a sandboxed process on your machine that is isolated from all other processes on the host machine"

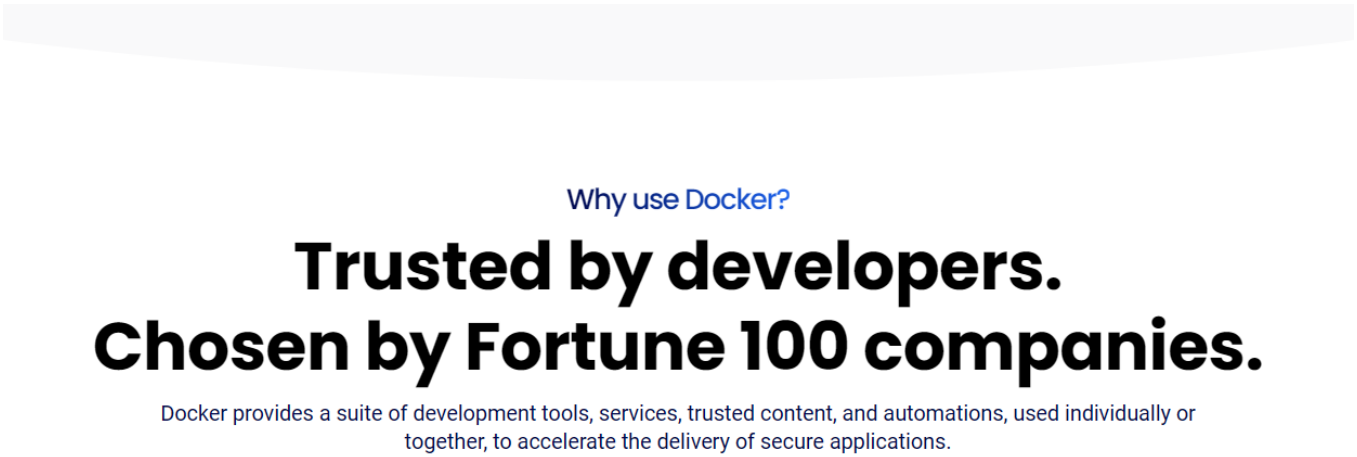
Wiso, Weshalb, Warum?



"a sandboxed process on your machine that is isolated from all other processes on the host machine"

"It works on my computer"

Wiso, Weshalb, Warum?



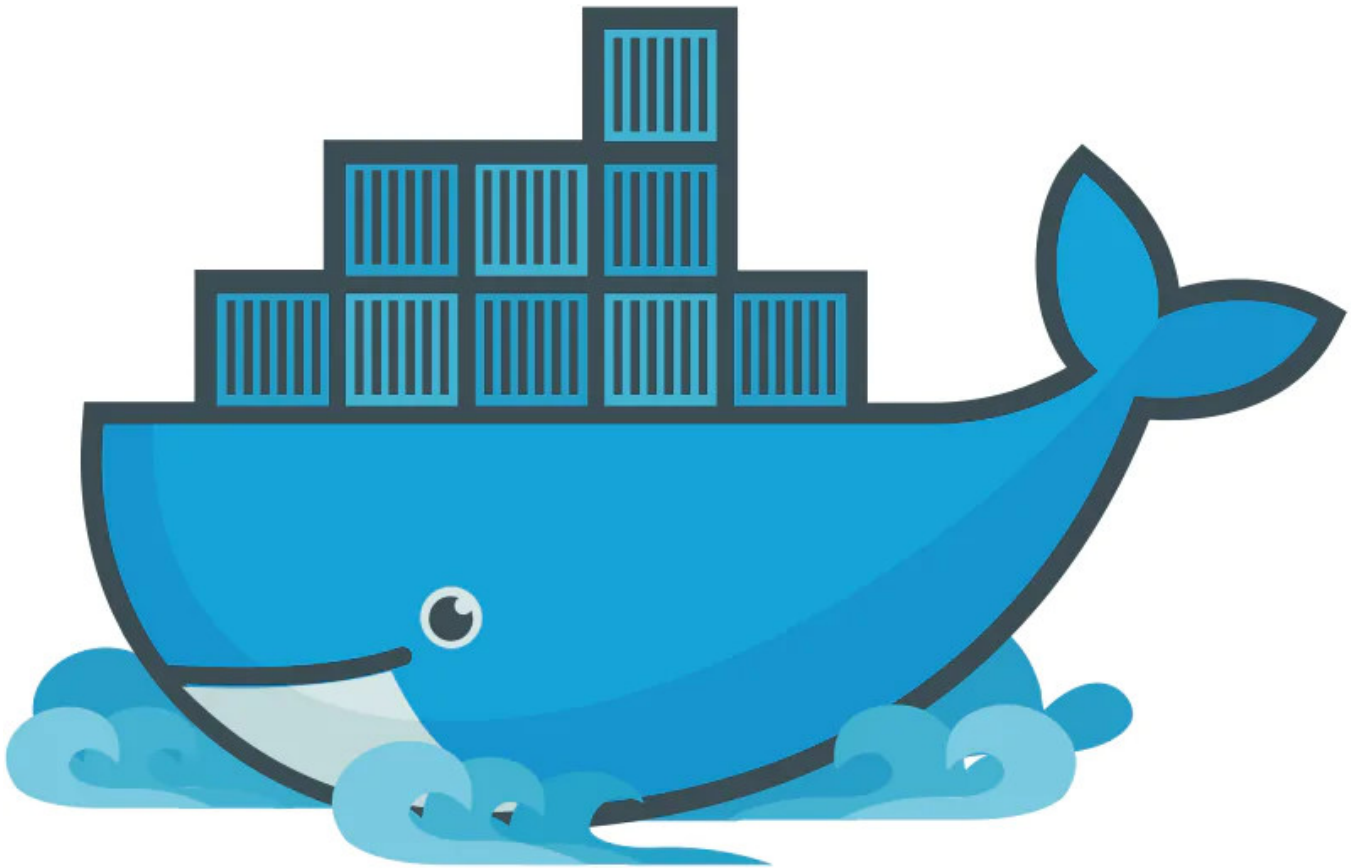
"a sandboxed process on your machine that is isolated from all other processes on the host machine"

"It works on my computer"

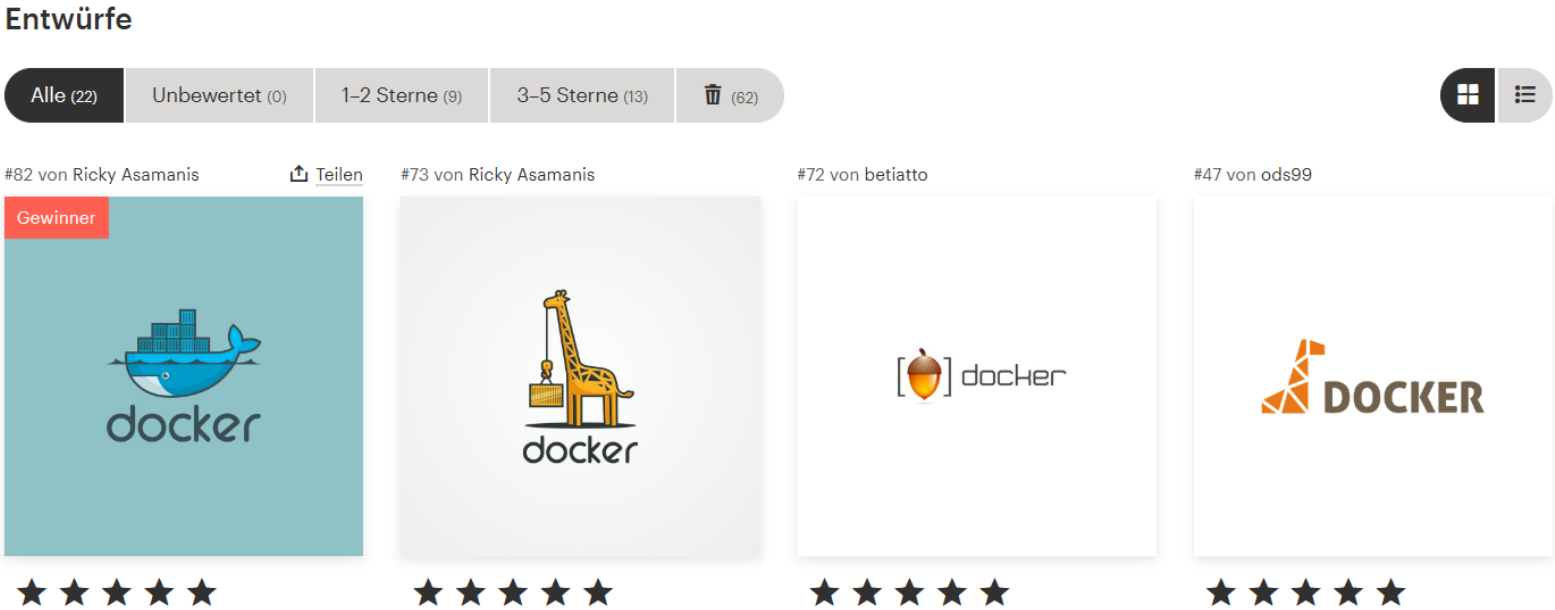
"faster onboarding and testing while also simplifying the deployment of services"

Wer ist Moby Dock?

Wer ist Moby Dock?



Wer ist Moby Dock?

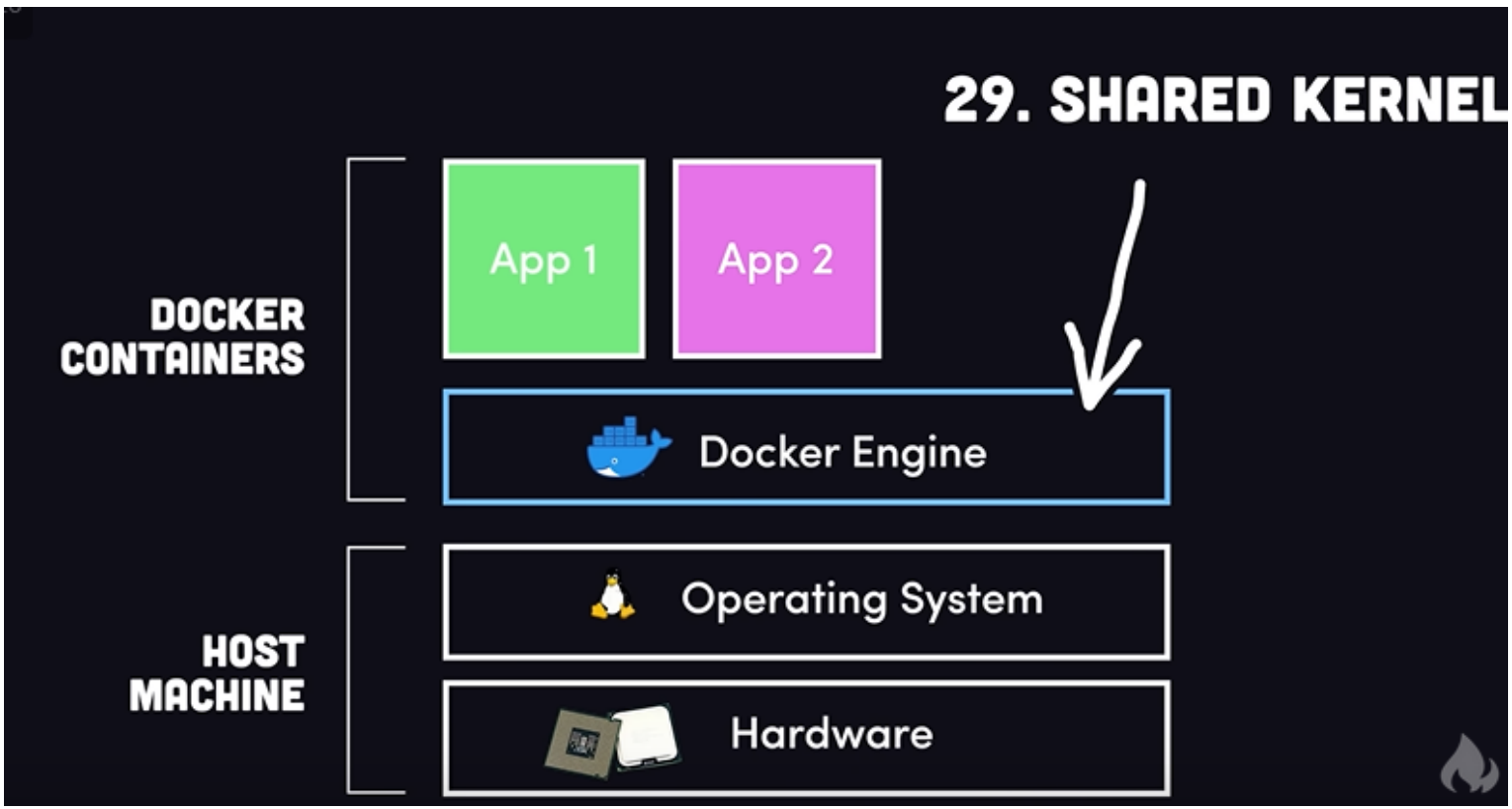


Wettbewerb zum Icon für Docker

Was ist Docker?

Docker

freie Software zur Isolierung von Anwendungen
Containervirtualisierung
"light weight" Virtual Maschine



Wichtige Begriffe

Container

Umgebung in der die tatsächliche Anwendung läuft

Image

Blaupausen, um einen Container zu erstellen

Dockerfile

Anleitung, um ein Image zu erstellen

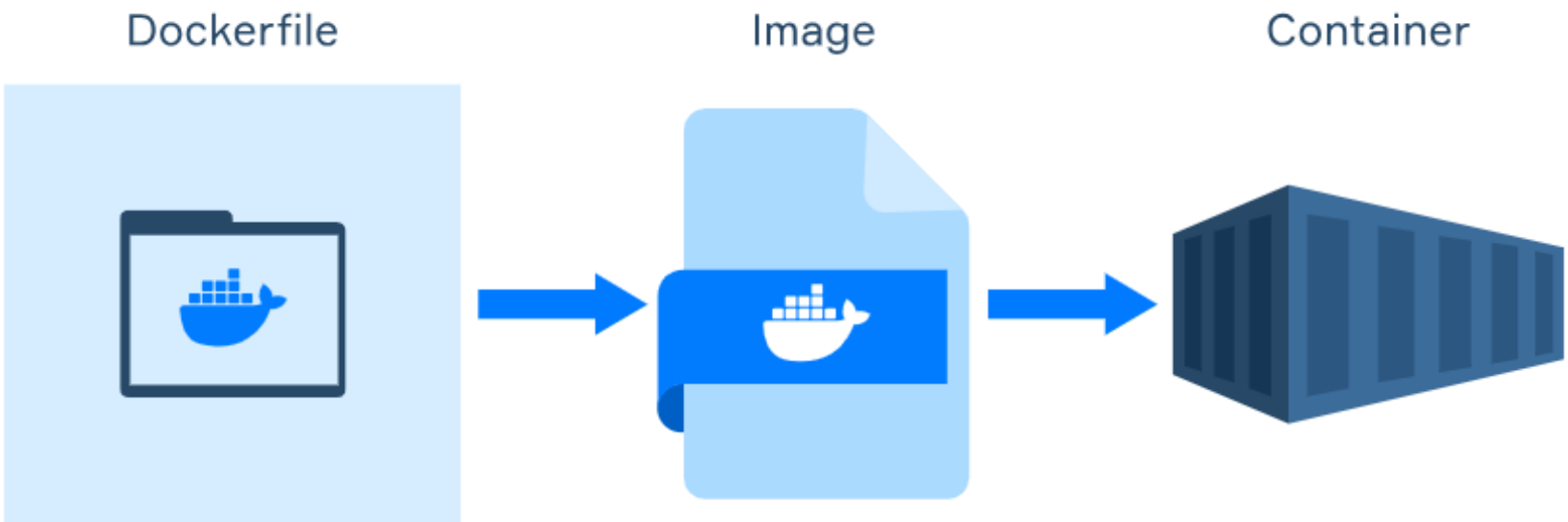
Registry

z.B. Docker Hub, EAC.... Ort an dem viele verschindene Images gespeichert und geteilt werden können

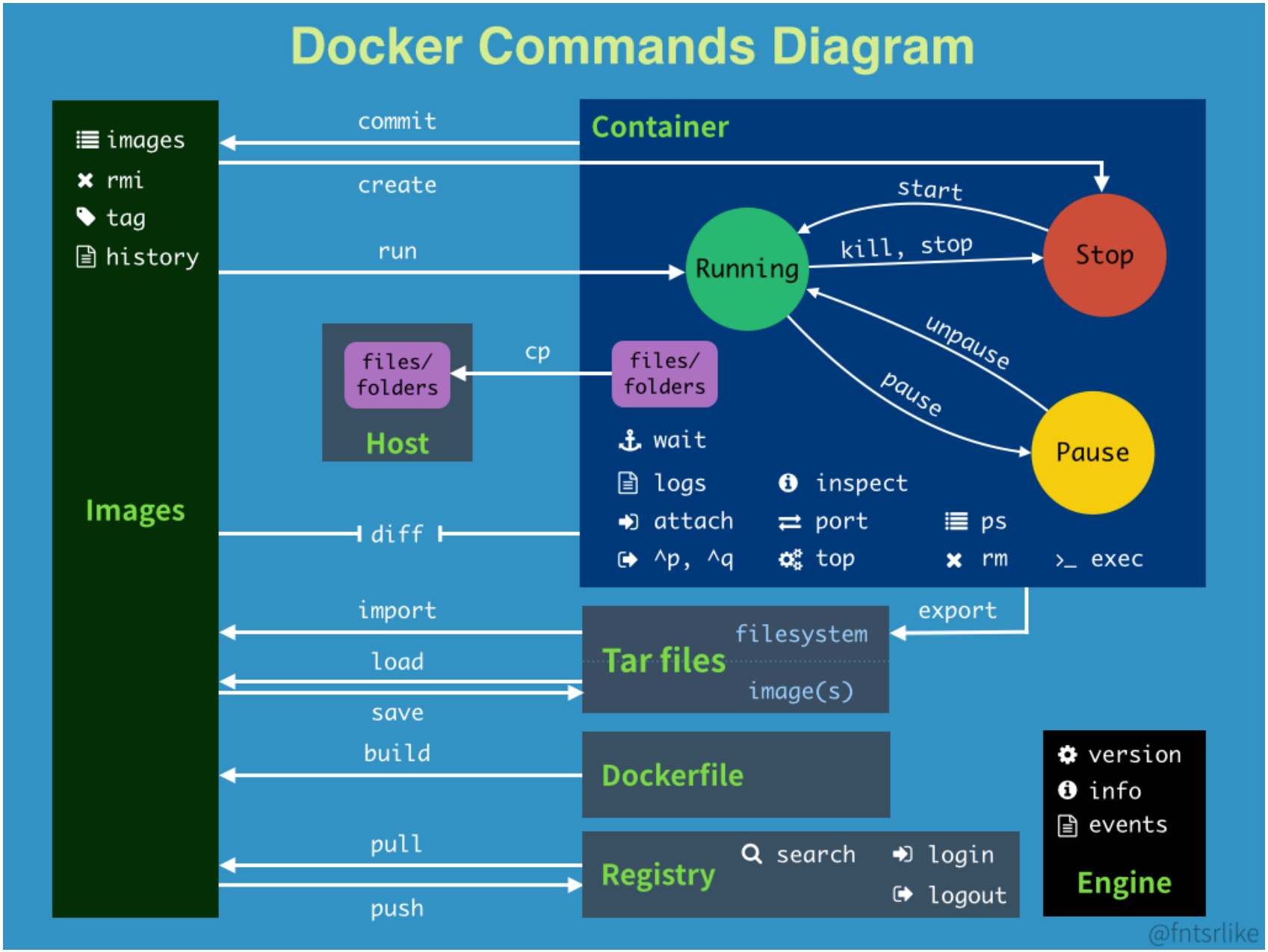
Docker Compose

Orchestrierungstool für Dockerfile
Wrapper für einen oder mehrere Container

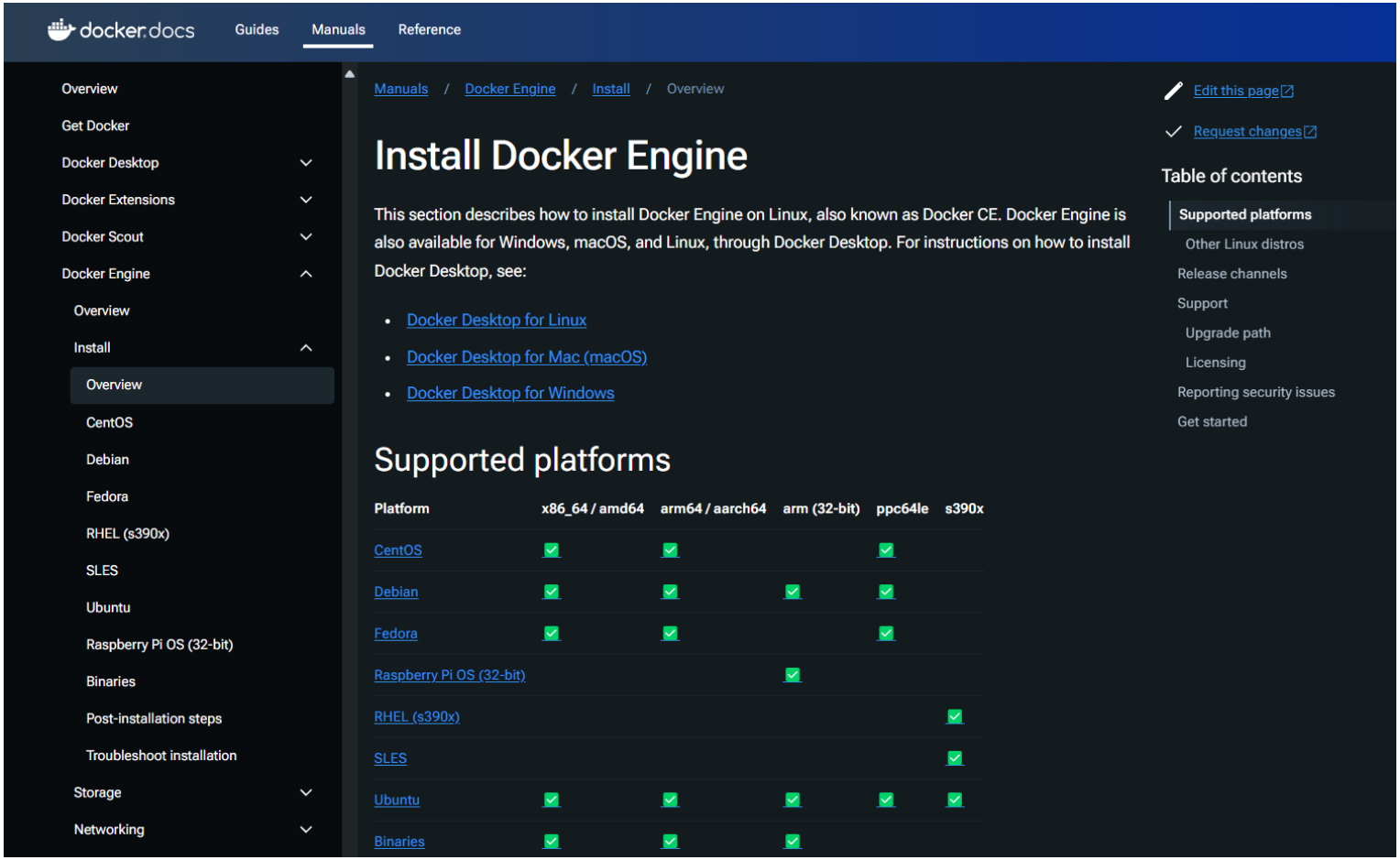
Zusammenhang der Docker Komponenten



Zusammenhang der Docker Komponenten



Wie kreige ich dieses "Docker"?

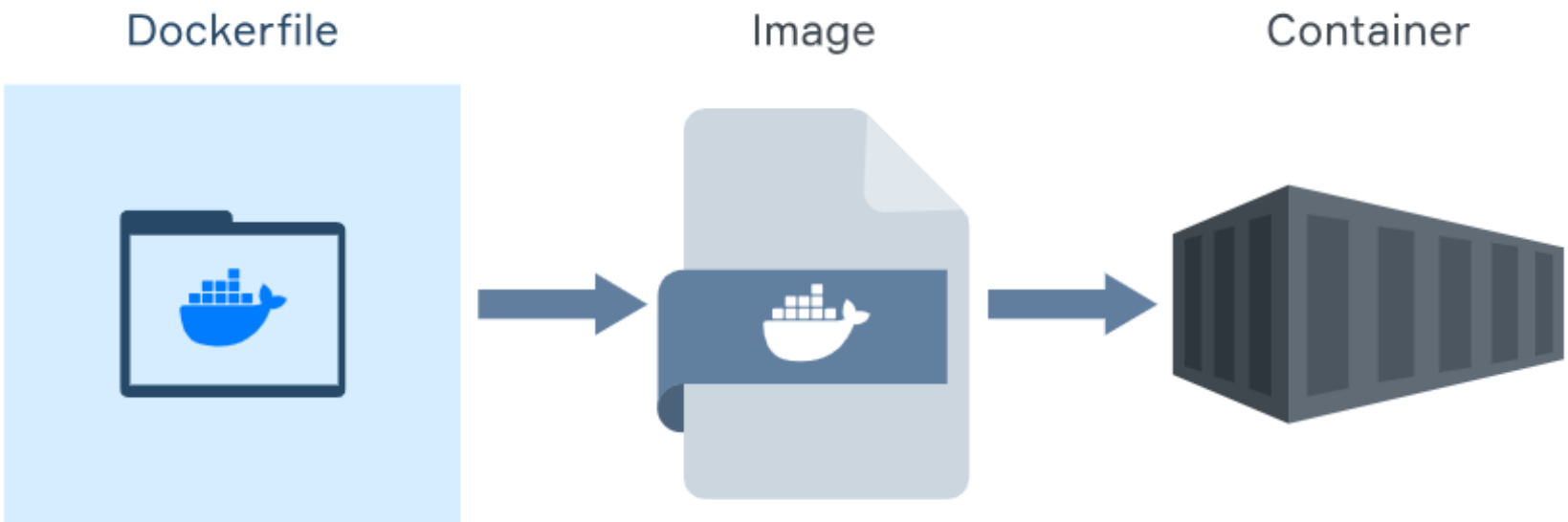


Doku

Hello World

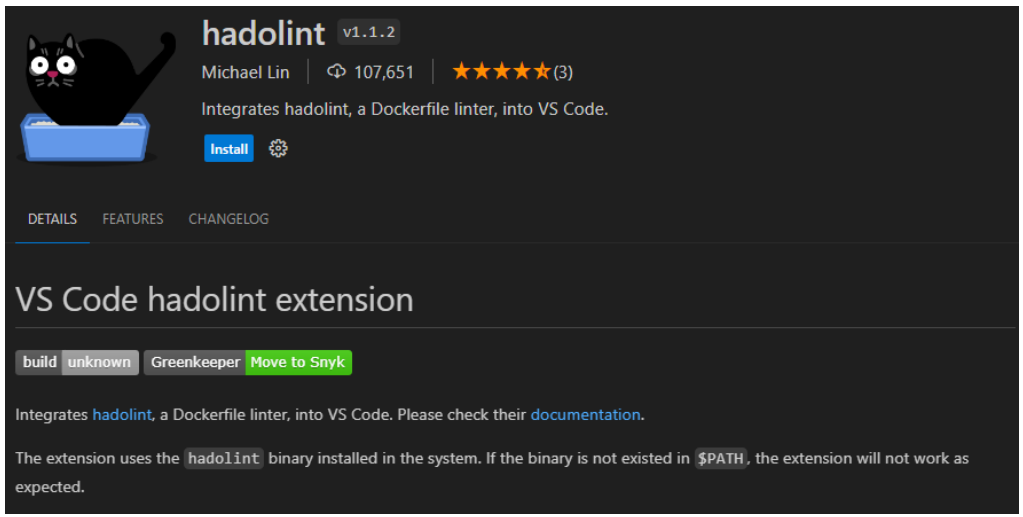
```
> docker -v
> docker --help
> docker run hello-world
```

Zusammenhang der Docker Komponenten



Dockerfile

- Ein Dockerfile ist die Anleitung um ein Image zu erstellen.
- Standardmäßig hießt die Datei 'Dockerfile'
- wieter Optionen mit `docker buildx build`



Plugin für die Arbeit mit Docker

docker build command

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

Build an image from a Dockerfile

`[OPTIONS]`

-f, –file string Name of the Dockerfile (default: "PATH/Dockerfile")

-t, –tag stringArray Name and optionally a tag (format: "name:tag")

PATH in most cases .

Beispiele:

```
docker build . # 'Dockerfile' im aktuellen Ordner
docker build -t myimage:v1 .
docker build -f Docker.cmd .
docker build ./examples/FastAPI/Dockerfile
```

Dockerfile

Beispiel `Dockerfile`:

```
FROM alpine:lastest

CMD [ "echo", "Hello World" ]
```

Weitere Informationen und Instruction
https://docs.docker.com/reference/dockerfile/

CMD vs. ENTRYPOINT

```
> docker build -t example:cmd -f Dockerfile.cmd .
> docker build -t example:entry -f Dockerfile.entry .
```

CMD vs. ENTRYPOINT

```
> docker build -t example:cmd -f Dockerfile.cmd .
> docker build -t example:entry -f Dockerfile.entry .
> docker run example:cmd
> docker run example:cmd hello
> docker run example:entry hello
```

CMD vs. ENTRYPOINT

```
> docker build -t example:cmd -f Dockerfile.cmd .
> docker build -t example:entry -f Dockerfile.entry .
> docker run example:cmd
> docker run example:cmd hello
> docker run example:entry hello
```

- beide definieren den, was nach Container start ausgeführt wird
- `CMD` kann überschrieben werden
- `ENTRYPOINT` bestimmt den command, neue Parameter werden angehangen

RUN

```
> docker build -t example:single -f Dockerfile.single .
> docker build -t example:multi -f Dockerfile.multi .

# Entstandene Images anschauen
> docker ps -a
> docker images
```

RUN

```
> docker build -t example:single -f Dockerfile.single .
> docker build -t example:multi -f Dockerfile.multi .

# Entstandene Images anschauen
> docker ps -a
> docker images
```

- pro `RUN` baut Docker einen Layer
- Layer werden gecached und nach Möglichkeit wiederverwendet
- versucht `RUN` instructions zu verbinden
- verbessert built-time und Image größe

Python bsp

```
FROM alpine


# Exec form
CMD ["echo", "Hello World."]

#shell form
CMD echo Hello Students
```

Volumes

-

React

```
FROM alpine


# Exec form
CMD ["echo", "Hello World."]

#shell form
CMD echo Hello Students
```

Multistage builds

- 1

React - Multistage

```
FROM alpine


# Exec form
CMD ["echo", "Hello World."]

#shell form
CMD echo Hello Students
```

Dockerfile Best practices

- RUN commands
- Order of COPY
- Volumes
- Multistage

Docker Compose

- Vorteile
- UseCases

Docker Compose zu Python

```
FROM alpine


# Exec form
CMD ["echo", "Hello World."]

#shell form
CMD echo Hello Students
```

Docker Compose Webapp

```
FROM alpine


# Exec form
CMD ["echo", "Hello World."]

#shell form
CMD echo Hello Students
```

title

- OpenDrone Map

es Arbeit

wi arbeit

me work

Cheatsheet

- docker run
- docker build
- docker push, pull
- docker ps -a
- docker rm / rmi
- ...

Coole Quellen und so weiter

- https://www.docker.com/
- https://docs.docker.com/get-started/
-