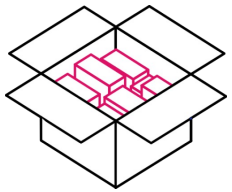
Einführung in Docker

Hoai Viet Nguyen – TH Köln

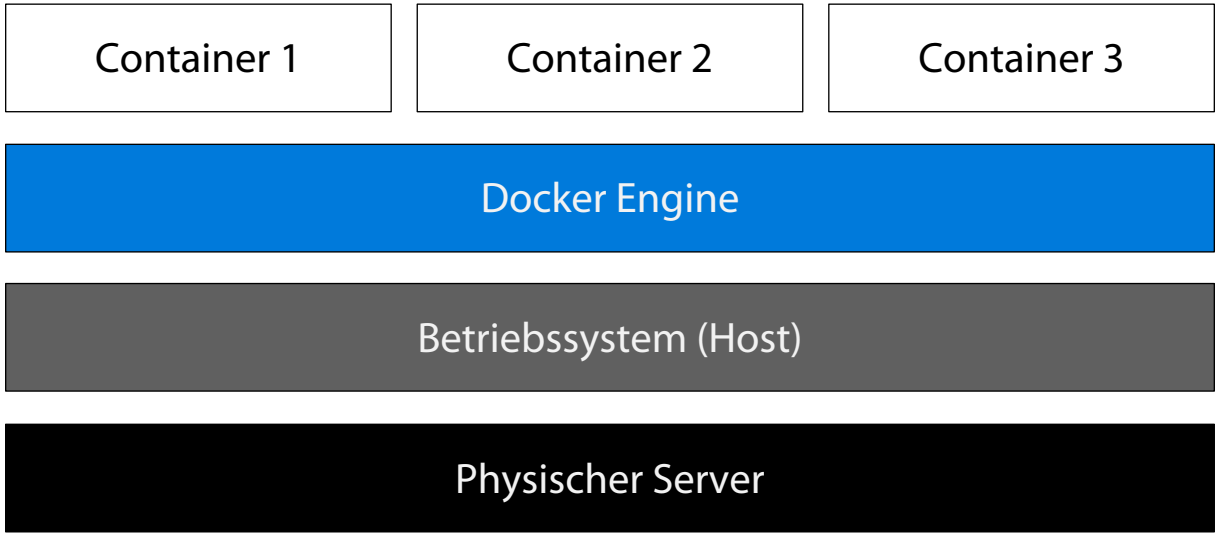
Technology
Arts Sciences
TH Köln



Docker

- Frei verfügbare Open Source Containerisierungstechnologie
- Für jedes bekannte Betriebssystem verfügbar
  - Windows
  - MacOS
  - Linux
- Viele Basis-Container-Images vorhanden
- Sehr weit verbreitet

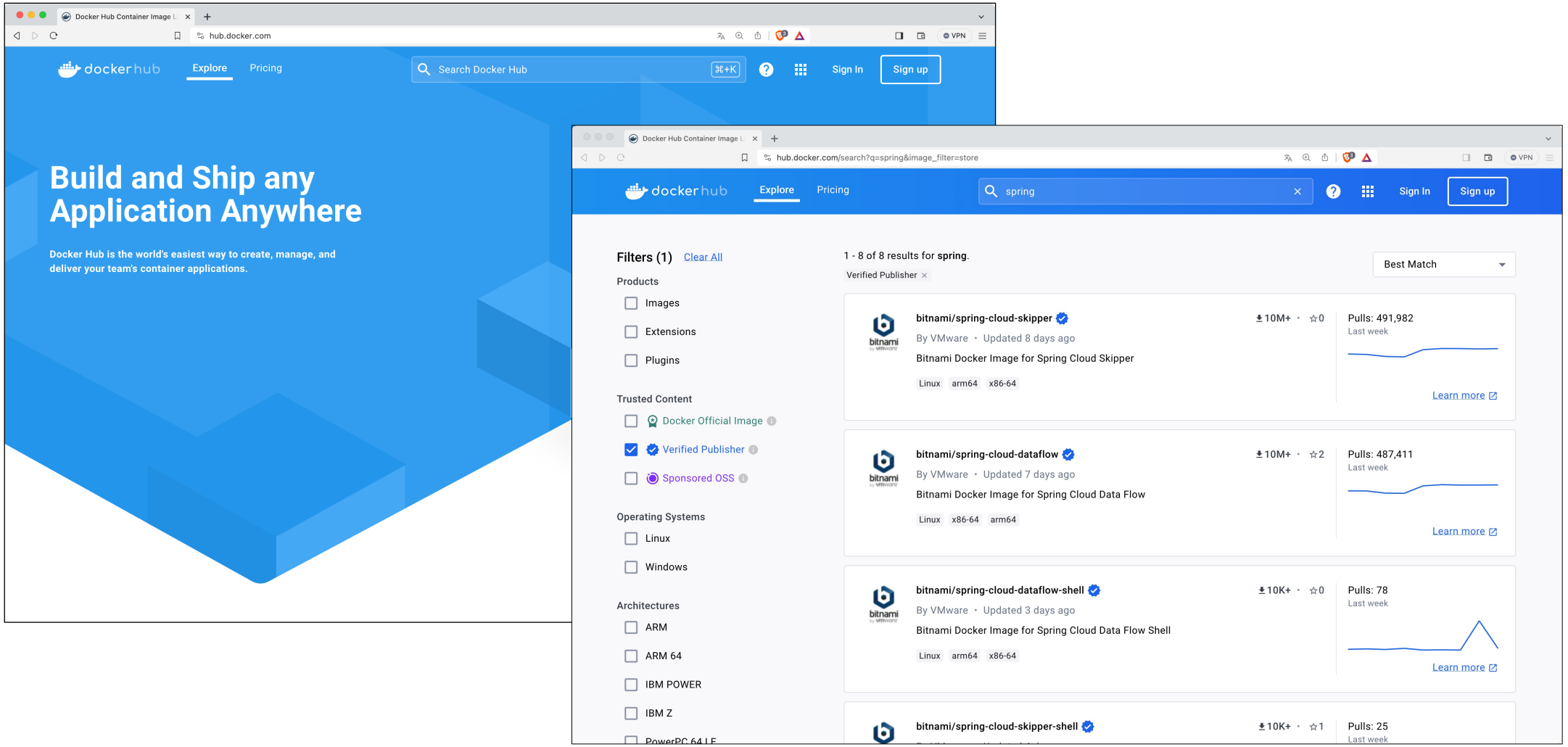
Docker Aufbau



Docker Image

- Ein read-only Template, aus dem Container erzeugt werden
- Beschreibt die Installation und Konfiguration von
  - Betriebssystem
  - Laufzeitumgebungen
  - Bibliotheken
  - Anwendungen
  - Ausführung der Quelltexte
- Docker Images können von jeder Person definiert, erweitert und veröffentlicht werden

dockerhub


Docker Hub Container Image
hub.docker.com
dockerhub
Explore
Pricing
Search Docker Hub
⌘+K
Sign In
Sign up
Build and Ship any Application Anywhere
Docker Hub is the world's easiest way to create, manage, and deliver your team's container applications.
hub.docker.com/search?q=spring&image_filter=store
spring
Filters (1)
Clear All
Products
Images
Extensions
Plugins
Trusted Content
Docker Official Image
Verified Publisher
Sponsored OSS
Operating Systems
Linux
Windows
Architectures
ARM
ARM 64
IBM POWER
IBM Z
PowerPC 64 LE
1 - 8 of 8 results for spring.
Verified Publisher
Best Match
bitnami/spring-cloud-skipper
By VMware • Updated 8 days ago
Bitnami Docker Image for Spring Cloud Skipper
Linux arm64 x86-64
10M+ ☆0
Pulls: 491,982
Last week
Learn more
bitnami/spring-cloud-dataflow
By VMware • Updated 7 days ago
Bitnami Docker Image for Spring Cloud Data Flow
Linux x86-64 arm64
10M+ ☆2
Pulls: 487,411
Last week
Learn more
bitnami/spring-cloud-dataflow-shell
By VMware • Updated 3 days ago
Bitnami Docker Image for Spring Cloud Data Flow Shell
Linux arm64 x86-64
10K+ ☆0
Pulls: 78
Last week
Learn more
bitnami/spring-cloud-skipper-shell
10K+ ☆1
Pulls: 25
Last week

Dockerfile

- Text-basierte Datei
- Jedes Docker Image basiert auf einem Dockerfile
- Enthält Anweisungen für das Deployment eines Containers
- Anweisungen werden zeilenweise sequenziell ausgeführt
- Anweisungen können einen Layer anlegen
- Reihenfolge der Anweisungen relevant, da Layers gecached werden

Dockerfile Anweisungen

Anweisung	Beschreibung	Kommentar
FROM	Basis-Image festlegen	Muss als erste Anweisung auftreten; nur ein Eintrag pro Build-Stage
ENV	Umgebungsvariablen für Build-Prozess und Container-Laufzeit setzen	—
ARG	Kommandozeilen-Parameter für Build-Prozess deklarieren	Darf vor FROM-Anweisung auftreten
WORKDIR	Aktuelles Verzeichnis wechseln	—
USER	Nutzer und Gruppenzugehörigkeit wechseln	—
COPY	Dateien und Verzeichnisse in das Image kopieren	Legt neuen Layer an
ADD	Dateien und Verzeichnisse in das Image kopieren	Legt neuen Layer an; von Nutzung wird abgeraten
RUN	Befehl im Image während des Build-Prozesses ausführen	Legt neuen Layer an
CMD	Standard-Argumente für Container-Start festlegen	Nur ein Eintrag pro Build-Stage
ENTRYPOINT	Standard-Befehl für Container-Start festlegen	Nur ein Eintrag pro Build-Stage
EXPOSE	Port-Zuweisungen für laufenden Container definieren	Ports müssen beim Starten des Containers aktiv geschaltet werden
VOLUME	Verzeichnis im Image beim Start des Containers im Host-System als Volumen einbinden	—

Quelle: https://www.ionos.de/digitalguide/server/knowhow/dockerfile/

Dockerfile HelloWorld-App mit C

```
# Basis-Image für Container
FROM alpine:latest
RUN apk update && apk add gcc musl-dev --no-cache
# Anlegen des Verzeichnisses /var/c-app
RUN mkdir -p /var/c-app
Wechsel in das Verzeichnis /var/c-app
WORKDIR /var/c-app
# Kopieren von des helloWorld.c-Datei aus lokalem Rechner ins Verzeichnis /var/c-app im Container
COPY helloWorld.c helloWorld.c
# Kompilierung der helloWorld.c-Datei
RUN gcc helloWorld.c -o app
# Ausführungsbefehl wenn docker run ausgeführt wird
CMD ["./app"]
```

```
#include <stdio.h>
int main()
{
     printf("Hello World");
     return 0;
}
```

Dockerfile – Beispiel helloWorld-App mit Kotlin

```
# Definition des Basis-Image
FROM eclipse-temurin:21-jdk-jammy

# Aktualisierung der Paktenliste und Installation von wget und unzip
RUN apt-get update && apt-get -y --no-install-recommends install \
wget unzip

# Setzen der Umgebungsvariable ENV auf den Wert 2.0.20
ENV KOTLIN_VERSION=2.0.20

# Download und entpackung des Kotlin-Compilers mit wget und unzip
RUN wget "https://github.com/JetBrains/kotlin/releases/download/v${KOTLIN_VERSION}/kotlin-compiler-${KOTLIN_VERSION}.zip" &&\
unzip kotlin-compiler-${KOTLIN_VERSION}.zip -d / && mv /kotlinc /usr/lib/

# Hinzufügung des Pfads des Kotlin-Compilers zur Umgebungsvariable PATH
ENV PATH="/usr/lib/kotlinc/bin:${PATH}“

# Kopieren der Kotlin-Datei aus lokalem Rechner in den Container
COPY simpleHelloWorld.kt simpleHelloWorld.kt

# Kompilierung der Kotlin-Datei als Jar-File
RUN kotlinc simpleHelloWorld.kt -d app.jar

# Ausführungsbefehl wenn docker run ausgeführt wird
ENTRYPOINT [ "java", "-jar", "app.jar"]
```

Kommandozeilenbefehle für Erstellung und Starten von Images

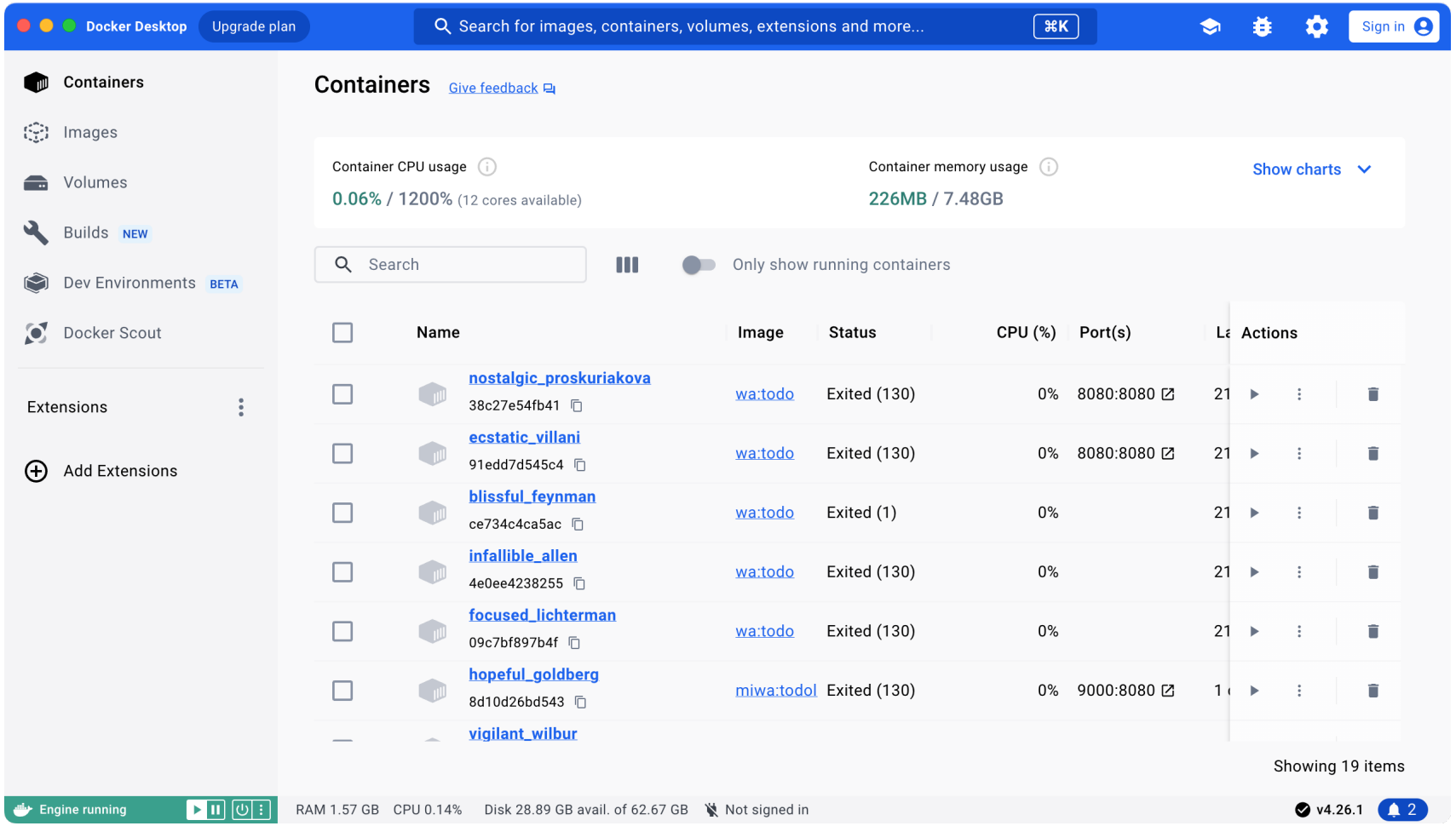
Docker-Befehl	Bedeutung	Henne-Ei-Analogie
docker build	Docker-Image aus Dockerfile erzeugen	Ei mit Erbinformation versehen
docker run <image>	Docker-Container aus Image starten	Küken schlüpft aus Ei

```
# Erzeugen eines Docker-Image mit Tag wa:app basierend auf dem dockerfile im aktuellen Ordner
docker build –t app:latest .

# Docker-Container aus Image mit Tag wa:app erzeugen und starten
docker run --rm –p 8080:8080 app:latest

# Erzeugen und Ausführen
docker build -t app:latest && docker run --rm -p 8080:8080 webapp:latest
```

Docker Dashboard



Weitere Kommandozeilebefehle

Docker Standalone-Befehl	Äquivalenter Docker Management-Befehl	Erklärung
docker ps	docker container ls	Die auf dem Host laufenden Container anzeigen
docker images	docker image ls	Die auf dem Host verfügbaren Images anzeigen
docker inspect <object>	docker <object-type> inspect <object>, z.B. docker image inspect <image>	Informationen zu Docker-Objekten wie Images, Containers, Volumes, etc. anzeigen
docker exec <container_name_or_id> /bin/bash		In den Container einloggen

Demo

Lernzielkontrolle

- Was ist Docker?
- Was ist ein Docker Image?
- Was ist ein Dockerfile?

Zusammenfassung

- Docker ist eine frei verfügbare Containerisierungstechnologie
- Docker Images sind Templates aus denen Container erzeugt werden
- Dockerfile definiert Anweisung für das Deployment eines Containers