Docker-Compose

Big Data und Cloud Computing

22. Mai 2024

Fabian Schmidt | OTH Amberg-Weiden

Gliederung

Ostbayerische Technische Hochschule Amberg-Weiden

1. Recap: Docker
2. Docker-Compose
   - Warum Docker-Compose?
   - Umgebungsvariablen
   - Arbeiten mit mehreren Compose Dateien
   - GPU Unterstützung
   - Networking
3. Praktisches Beispiel

Recap: Docker
Warum Docker?

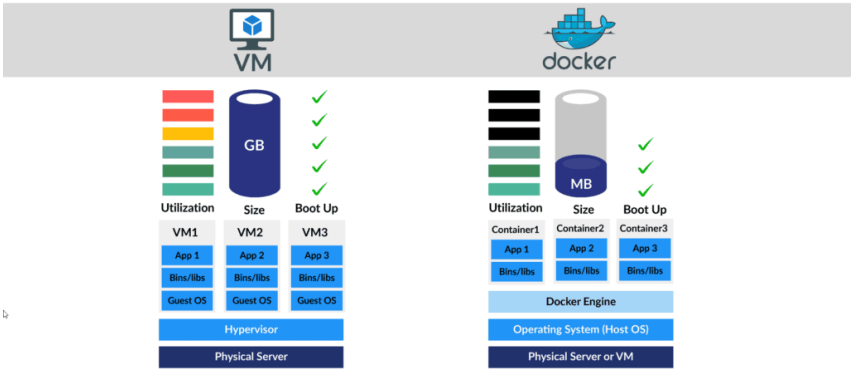
[1] https://shorturl.at/QbN0E

Recap: Docker

Was kann Docker?

- Selbstständig
- Isoliert
- Unabhängig
- Portabel

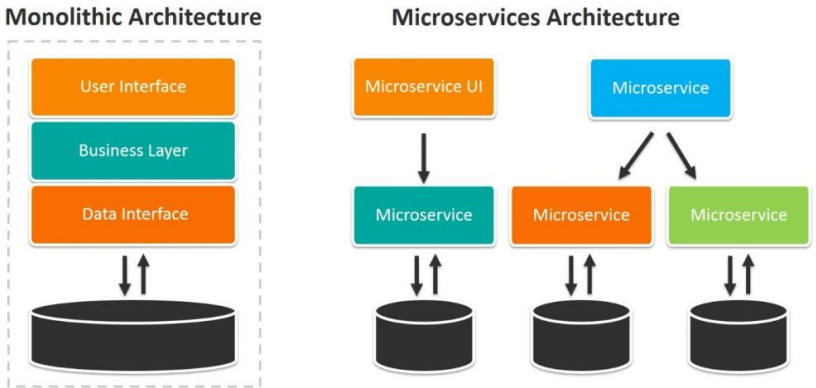
Recap: Docker
Was ist Docker?



[2] https://k21academy.com/docker-kubernetes/docker-vs-virtual-machine/

Recap: Docker

Monolithic vs Microservice Architektur



[3] https://s7280.pcdn.co/wp-content/uploads/2018/10/key1-1024x496.jpg.optimal.jpg

Docker-Compose

Warum Docker-Compose?

- Orchestrierung von Containern
- Standartisierte Entwicklungsumbegung
- Effiziente Zusammenarbeit
- Container re-use
- Portabel zwischen Umgebungen

Docker-Compose
Umgebungsvariablen

Osthayerische Technische Hochschule Amberg-Weiden

- Konfigurationsinformationen an den Container zur Laufzeit geben
- Datenbank Konfiguration, Service URLs, API Keys, Anwendungsspezifische Konstanten
- Setzen von Umgebungsvariablen über Compose-Datei oder Docker-CLI

Docker-Compose

Umgebungsvariablen - Compose-Datei (1/4)

Nutzen von Umgebungsvariablen über:

- .env-Datei

```
$ cat .env
TAG=v1.5

$ cat compose.yml
services:
  web:
    image: "webapp:${TAG}"
```

```
$ docker compose config

services:
  web:
    image: 'webapp:v1.5'
```

Docker-Compose

Umgebungsvariablen - Compose-Datei (2/4)

Nutzen von Umgebungsvariablen über:

- .env-Datei
- compose environment Attribut

```
web:
  environment:
    - DEBUG=1
```

Docker-Compose

Umgebungsvariablen - Compose-Datei (3/4)

Nutzen von Umgebungsvariablen über:

- .env-Datei
- compose environment Attribut
- compose env_file Attribut

```
web:
  env_file:
    - web-variables.env
```

Docker-Compose

Umgebungsvariablen - Compose-Datei (4/4)

Nutzen von Umgebungsvariablen über:

- .env-Datei
- compose environment Attribut
- compose env_file Attribut
- Umgebungsvariablen von Host-Maschine

```
db:
  image: "postgres:${POSTGRES_VERSION}"
```

Docker-Compose

Umgebungsvariablen - Docker-CLI (1/2)

- .env-Datei mit CLI übergeben

```
$ cat .env
TAG=v1.5

$ cat ./config/.env.dev
TAG=v1.6


$ cat compose.yml
services:
  web:
    image: "webapp:${TAG}"
```

```
$ docker compose config
services:
  web:
    image: 'webapp:v1.5'
```

```
$ docker compose --env-file ./config/.env.dev config
services:
  web:
    image: 'webapp:v1.6'
```

Docker-Compose

Umgebungsvariablen - Docker-CLI (1/2)

Osthayerische Technische Hochschule Amberg-Weiden

- .env-Datei mit CLI übergeben
- Variablen direkt in CLI definieren

```
$ docker compose run -e DEBUG=1 web python console.py
```

Docker-Compose

Arbeit mit mehreren Compose-Dateien

- Merge: Zusammenführen und überschreiben von Compose-Dateien
  - Für single-value Optionen ersetzt der neue Wert den alten

```yaml
services:
  myservice:
    # ...
    command: python app.py
```

```yaml
services:
  myservice:
    # ...
    command: python otherapp.py
```

```yaml
services:
  myservice:
    # ...
    command: python otherapp.py
```

Docker-Compose

Arbeit mit mehreren Compose-Dateien

- Merge: Zusammenführen und überschreiben von Compose-Dateien
  - Für single-value Optionen ersetzt der neue Wert den alten
  - Multi-value Optionen werden Concateniert

```yaml
services:
  myservice:
    # ...
    expose:
      - "3000"
```

```yaml
services:
  myservice:
    # ...
    expose:
      - "4000"
      - "5000"
```

```yaml
services:
  myservice:
    # ...
    expose:
      - "3000"
      - "4000"
      - "5000"
```

Docker-Compose

Arbeit mit mehreren Compose-Dateien

- Merge: Zusammenführen und überschreiben von Compose-Dateien
  - Für single-value Optionen ersetzt der neue Wert den alten
  - Multi-value Optionen werden Concateniert
  - environment, labels, volumns, devices werden gemergt; Lokale Version hat Vorrang

```yaml
services:
  myservice:
    # ...
    environment:
      - FOO=original
      - BAR=original
```

```yaml
services:
  myservice:
    # ...
    environment:
      - BAR=local
      - BAZ=local
```

```yaml
services:
  myservice:
    # ...
    environment:
      - FOO=original
      - BAR=local
      - BAZ=local
```

Docker-Compose

Arbeit mit mehreren Compose-Dateien

- Merge: Zusammenführen und überschreiben von Compose-Dateien
- Extend: Teilen gemeinsamer Konfigurationen
  - Service von anderer Datei erweitern

```yaml
services:
  webapp:
    build: .
    ports:
      - "8000:8000"
    volumes:
      - "/data"
```

```yaml
services:
  web:
    extends:
      file: common-services.yml
      service: webapp
```

Docker-Compose

Arbeit mit mehreren Compose-Dateien

- Merge: Zusammenführen und überschreiben von Compose-Dateien
- Extend: Teilen gemeinsamer Konfigurationen
  - Service von anderer Datei erweitern
  - Service in gleicher Datei erweitern

```yaml
services:
  web:
    build: alpine
    extends: webapp
  webapp:
    environment:
      - DEBUG=1
```

Docker-Compose

Arbeit mit mehreren Compose-Dateien

- Merge: Zusammenführen und überschreiben von Compose-Dateien
- Extend: Teilen gemeinsamer Konfigurationen
  - Service von anderer Datei erweitern
  - Service in gleicher Datei erweitern
  - Kombination beider Möglichkeiten

```yaml
services:
  web:
    extends:
      file: common-services.yml
      service: webapp
    environment:
      - DEBUG=1
    cpu_shares: 5

  important_web:
    extends: web
    cpu_shares: 10
```

Docker-Compose

Arbeit mit mehreren Compose-Dateien

- Merge: Zusammenführen und überschreiben von Compose-Dateien
- Extend: Teilen gemeinsamer Konfigurationen
- Include: Einbinden serperater compose Dateien

```yaml
include:
  - my-compose-include.yaml  #with serviceB declared
services:
  serviceA:
    build: .
    depends_on:
      - serviceB #use serviceB directly as if it was declared in this Compose file
```

Docker-Compose
GPU Unterstützung

Ostbayerische Technische Hochschule Amberg-Weiden

```yaml
services:
  test:
    image: nvidia/cuda:12.3.1-base-ubuntu20.04
    command: nvidia-smi
    deploy:
      resources:
        reservations:
          devices:
            - driver: nvidia
              count: 1
              capabilities: [gpu]
```

Docker-Compose
Networking

Angenommen die Anwendung liegt im Order ***myapp*** mit dieser Compose-Datei

```yaml
services:
  web:
    build: .
    ports:
      - "8000:8000"
  db:
    image: postgres
    ports:
      - "8001:5432"
```

Docker-Compose

Networking

- Standardmäßig erzeugt compose ein Netzwerk. Jeder Container ist über dieses Netzwerk erreichbar
- beim ausführen von ***docker compose up*** passiert folgendes:
  1. Ein Netzwerk Namens ***myapp_default*** wird erzeugt
  2. Container mit Konfiguration von ***web*** wird erzeugt und dem Netzwerk hinzugefügt
  3. Container mit Konfiguration von ***db*** wird erzeugt und dem Netzwerk hinzugefügt

Osthayerische Technische Hochschule Amberg-Weiden

- ***HOST_PORT:CONTAINER_PORT***
- für ***db-service***:
  - HOST_PORT: 8001
  - CONTAINER_PORT: 5432
- db-service Zugriff innerhalb des web-Containers: ***postgres://db:5432***
- db-service Zugriff von Host-Maschine: ***postgres://localhost:8001***

Praktisches Beispiel

Wordpress-Webseite mit Docker-Compose

siehe `docker-compose.yml`

Quellen

Ostbayerische Technische Hochschule Amberg-Weiden

- Docker Docs