# Docker Images / Volumes

## Modul 169

# Inhalt

- **Repetition**
- **Dockerfile**
  *Befehle / Caching / Multistage Build*
- **Übungen**
  *zu Docker Image und Dockerfile*
- **Docker Hub**
  - eigene **Apps containerisieren**
- Docker Volumen / Mouts
- **Übungen**
  *zu Docker Volumen / Mounts*

# Regeln 

## *INP24C* **spezial**

# §1 Fokus und Geräte

Die **digitalen Geräte**: 📱, 💻, etc.

- immer nur auf **Aufforderung der Lehrkraft**
- immer nur zur **Bearbeitung der gestellten Aufgaben**

**Private Aktivitäten sind untersagt**: *unter anderem Social Media, Spiele, Videos, private E-Mails/Chats, Surfen, Shoppen, etc.*

**1. Verwarnung**

- **Mündliche** Ermahnung durch Lehrperson

**2. Verwarnung**

- 👨‍🏫 Das Gerät ist für den **Rest der Lektion bei der Lehrperson** zu hinterlegen.
- 🚨 **Absenz**, wenn dadurch nicht gearbeitet werden kann!
- 🗣️ **Meldung an den Berufsbildner**.

# §2 Ruhe und Umgangsformen

Die Konzentration der Mitschüler muss gewährleistet sein.

- **Lärm ist zu vermeiden**
  z.B. laute Gespräche, Geräusche, Rufen.
- **Freundlicher, höflicher und respektvoller** Umgangston

### 1. Verwarnung

- **Mündliche** Ermahnung durch Lehrperson.
- Evtl. auf separaten Arbeitsplatz versetzen.

### 2. Verwarnung

- 🚪 Für den Rest der Lektion **aus dem Unterricht gewiesen**.
- 🚨 Die gesamte Lektion gilt als **Absenz**.
- 🗣️ **Meldung an den Berufsbildner**.

# Was versteht Ihr unter Git?

- Was ist der Unterschied von Git und GitHub?

# Wofür dient ein Dockerfile?

- Wieso sollte man es mit Git versionieren?

# 📖 Auftrag

Lesen Sie auf der Modulwebseite Woche 2

- Image Builden
- Caching beim Erstellen von Docker-Images
- Multistage Builds

- 🧬 Einzelarbeit
- 🕐 15 Min

# 📝 Auftrag

Machen Sie auf der Modulwebseite Woche 2

- Einfaches Dockerfile

- 🧬 Einzelarbeit
- 🕐 45 Min

# 📝 Auftrag

Machen Sie auf der Modulwebseite Woche 2

- Dockerhub Tutorial

- 🧬 Einzelarbeit
- 🕐 45 Min

# Docker Volumes & Bind Mounts

# Demo: Daten im Container

```
docker run -it --name volume-test -w /app ubuntu
echo "Hallo Welt" > hallo.txt
exit
docker container rm volume-test
docker run -it --name volume-test -w /app ubuntu
ls -la
```

→ Was wird das Resultat sein?

# Wie können Daten eines Containers erhalten bleiben?

💡 **Wenn ein Container gelöscht wird, sind die Daten im Container weg.**

# 📖 Auftrag

Lesen Sie auf der Modulwebseite Woche 3

- Docker Volume und Mounts

- 🧬 Einzelarbeit
- 🕐 15 Min

# Docker Volumes vs Bind Mounds

| Merkmal | Docker Volumes | Bind Mounts |
|---|---|---|
| Verwaltung | Von Docker verwaltet | Vom Host verwaltet |
| Speicherort | Im Docker-Standardverzeichnis | Pfad im Host-Dateisystem |
| Portabilität | Hoch (kann einfach zwischen Hosts verschoben werden) | Gering *(abhängig vom Host)* |
| Sicherheit | Besser isoliert | Weniger isoliert, da direkt auf den Host zugegriffen wird |
| Performance | Gut, Optimiert für Docker | Langsam bei vielen Files, Sync mit Host-System |

# *Docker Volumes*: Vor- und Nachteile

## Vorteile

- Einfache Verwaltung
- Portabilität
- bessere Performance

## Nachteile

- weniger Kontrolle über den Speicherort
- zusätzliche Komplexität

# *Bind Mounts*: Vor- und Nachteile

## Vorteile

- Direkter Zugriff auf Host-Dateisystem
- Flexibilität
- Einfachheit bei Entwicklung

## Nachteile

- weniger portabel
- Sicherheitsrisiken
- Abhängigkeit von Host-Umgebung

# Wann welche Variante?

## Volumes

- Docker Volumes **in Produktion** und wann immer möglich
- Docker Volumes immer für Daten die nicht in Git versionisiert sind
    - **Datenbanken** und co.

## Bind Mounts

- Bind Mounts bei der Entwicklung **für Quellcode** und **Konfigurationsdateien** die selbst in Git versionisiert sind.
- **Nie** in Produktion!

# Demo: Docker Volume und Bind Mounts

```
docker volume ls
docker volume create sqlite-volume
docker volume inspect sqlite-volume

docker run -it --rm -w /apps alpine/sqlite:3.51.2 test.db
docker run -it --rm -w /apps -v sqlite-volume:/apps alpine/sqlite:3.51.2 test.db
docker run -it --rm -w /apps -v .:/apps alpine/sqlite:3.51.2 test.db
```

```
CREATE TABLE contacts (
    contact_id INTEGER PRIMARY KEY,
    name TEXT NOT NULL
);
INSERT INTO contacts (contact_id, name) VALUES (2, 'New Name');
INSERT INTO contacts (contact_id, name) VALUES (1, 'Name');
```

# 📝 Auftrag

Machen Sie die Übungen unter Woche 3

- Übungen Mounts
- Übungen Volumes

- 🧬 Einzelarbeit
- 🕐 bis zum Ende