



Docker Einführung – Cheatsheet



Docker auf Windows installieren

1. Voraussetzungen:

- Windows 10 oder 11 (64-bit)
- WSL2 (Windows Subsystem for Linux 2) aktiviert (wird bei der Installation unterstützt)

2. Installation:

- Lade Docker Desktop von <https://www.docker.com/products/docker-desktop> herunter.
- Installiere es und aktiviere dabei die WSL2-Integration.
- Nach der Installation: Docker Desktop starten.

3. Testen, ob alles funktioniert:

```
docker --version
```



Images – Vorlagen für Container

Ein **Image** ist eine schreibgeschützte Vorlage, die alles enthält, was eine Anwendung braucht: Quellcode, Laufzeit, Bibliotheken und Abhängigkeiten. Container werden aus Images erstellt.

Befehl	Beschreibung
<code>docker pull <image></code>	Image aus Docker Hub laden
<code>docker images</code>	Lokale Images auflisten
<code>docker rmi <image></code>	Image löschen
<code>docker build -t name:tag .</code>	Image aus Dockerfile bauen



Container – Laufende Instanzen von Images

Ein **Container** ist eine laufende Instanz eines Images. Er ist leichtgewichtig, portabel und isoliert. Du kannst ihn dir wie eine kleine virtuelle Maschine vorstellen – nur schneller und effizienter.

Befehl	Beschreibung
<code>docker run <image></code>	Container starten
<code>docker run -it <image> /bin/bash</code>	Interaktiver Start mit Shell
<code>docker run -d <image></code>	Im Hintergrund starten
<code>docker run -p 8080:80 <image></code>	Port-Weiterleitung (host:container)
<code>docker ps</code>	Laufende Container anzeigen
<code>docker ps -a</code>	Alle Container (auch gestoppte) anzeigen
<code>docker stop <container></code>	Container stoppen
<code>docker start <container></code>	Gestoppten Container starten
<code>docker rm <container></code>	Container löschen
<code>docker logs <container></code>	Logs anzeigen
<code>docker exec -it <container> <cmd></code>	Befehl im laufenden Container ausführen

Volumes – Daten persistent speichern

Volumes ermöglichen es, Daten außerhalb eines Containers zu speichern, sodass diese beim Löschen des Containers erhalten bleiben. Ideal für Datenbanken oder Logs.

Befehl	Beschreibung
<code>docker volume create <name></code>	Volume erstellen
<code>docker volume ls</code>	Volumes anzeigen
<code>docker run -v <host_path>:<container_path></code>	Volume mounten
<code>docker volume inspect <name></code>	Details anzeigen

Dockerfile – Eigene Images erstellen

Ein **Dockerfile** ist eine einfache Textdatei, die Schritt für Schritt beschreibt, wie ein Image gebaut wird. Du definierst dort z. B. Basis-Image, kopierte Dateien und Startbefehle.

```
# Basis-Image
FROM openjdk:17-jdk-alpine

# Arbeitsverzeichnis setzen
WORKDIR /app

# JAR-Datei kopieren
COPY target/app.jar app.jar

# Befehl beim Start
ENTRYPOINT ["java", "-jar", "app.jar"]
```

Image bauen:

```
docker build -t mein-image .
```

🔗 Docker Compose – Mehrere Container koordinieren

Docker Compose erlaubt es, mehrere Container (z. B. App + Datenbank + Webserver) mit einer einzigen Datei (`docker-compose.yml`) zu konfigurieren und zu starten. Ideal für Entwicklungsumgebungen.

```
version: '3.8'
services:
  web:
    image: nginx
    ports:
      - "8080:80"
  app:
    build: .
    ports:
      - "8081:8080"
```

Befehl	Beschreibung
<code>docker-compose up</code>	Container starten (laut YML-Datei)
<code>docker-compose down</code>	Container stoppen und aufräumen



Aufräumen – Platz schaffen

Docker speichert standardmäßig vieles im Cache. Die folgenden Befehle helfen, ungenutzte Daten zu entfernen:

```
docker system prune      # Alles Ungenutzte löschen
docker container prune   # Ungenutzte Container löschen
docker image prune       # Ungenutzte Images löschen
```