

# 1 Docker

---



## 1 Wat is Docker nu eigenlijk?

Docker is een open source platform dat het makkelijk maakt om je applicaties te ontwikkelen in containers. Deze containers zorgen ervoor dat je projecten die erin zitten in een geïsoleerde omgeving werken. Het maakt niet uit welke besturingssysteem je gebruikt, want dat wordt allemaal meegegeven in de container


Docker wordt steeds populairder. Sinds 2015 is het gebruik van Docker met 25% gestegen onder bedrijven. momenteel gebruiken





---

## 1.2 De drie belangrijkste componenten van Docker

Binnen Docker zijn er drie heel belangrijke essentiële onderdelen:

- **Dockerfile** 

Een Dockerfile kan je vergelijken met het DNA van een applicatie. Deze bevat de instructies/code voor het maken van een image.
- **Image** 

Een image is een kant-en-klaar softwarepakket met alle afhankelijkheden. Vanuit een image kunnen meerdere containers worden gestart.
- **Container** 

Een container is een draaiende instantie van een image. Containers zijn lichtgewicht en geïsoleerd.

---

## 1.3 Waarom Docker gebruiken?

### ☒ Voordelen

- ✓ **Tijdbesparing** – Developers kunnen snel en eenvoudig een consistente ontwikkelomgeving opzetten.
- ✓ **Samenwerking** – Teams kunnen eenvoudig dezelfde configuratie delen.
- ✓ **Ruimtebesparing** – Je hoeft geen zware afhankelijkheden lokaal te installeren.

### ☒ Nadelen

- ✗ **Steile leercurve** – Docker kan lastig zijn om te leren voor beginners.
  - ✗ **Geen GUI-ondersteuning** – Docker is primair bedoeld voor backend-processen.
  - ✗ **Command-line based** – Werken met Docker vereist kennis van terminal-commando's.
-

## 2 Hoe installeer je Docker?

---



### 2.1 Docker installeren

Er zijn verschillende manieren om Docker te installeren. In deze documentatie leg ik stap voor stap uit hoe je Docker kunt installeren. Heel belangrijk dat ik enkel uitleg hoe je dit kan doen op een windows-systeem. Voor Windows gebruiken we **Docker Desktop**.

Docker Desktop kan op twee manieren worden geïnstalleerd:

1. Via de Installer
2. Via de Console

Beide methoden worden hieronder uitgelegd.

---

#### 2.1.1 Download Docker Desktop

Voordat we Docker kunnen installeren, moeten we Docker Desktop downloaden. Dit bestand is nodig voor zowel de installatie via de Installer als via de Console.

Je kunt Docker Desktop downloaden via de officiële website: [🔗 Docker Desktop download](#)

Klik op de link en vervolgens op **Download Docker Desktop**.

---

### 2.2 Installatie via de Installer

Zorg ervoor dat je de Docker Desktop installer hebt gedownload voordat je begint.

Stappen:

1. Dubbelklik op het gedownloade **Docker Desktop Installer** bestand om de installatie te starten. 
2. Er verschijnt een venster waarin je kunt kiezen of je een desktop shortcut wilt toevoegen. 
3. Klik op **OK** en wacht tot Docker alle bestanden heeft uitgepakt. 
4. Zodra de installatie is voltooid en je het bericht **Installation Succeeded** ziet, klik je op **Close**. 
5. Open Docker Desktop en ziezo Docker is geïnstalleerd! 



### 2.3 Installatie via de Console

Ook voor deze methode moet je eerst de Docker Desktop installer downloaden.

Stappen:

1. Open de **Command Prompt** (cmd) of **PowerShell**.

2. Navigeer naar de map waar de Docker Desktop Installer is opgeslagen met het commando:

```
cd Downloads
```

3. Dit commando moet je uitvoeren om Docker Desktop te installeren: Via de Terminal:

```
"Docker Desktop Installer.exe" install
```

Via PowerShell:

```
Start-Process 'Docker Desktop Installer.exe' -Wait install
```

Via de windows command prompt:

```
start /w "" "Docker Desktop Installer.exe" install
```

4. Na een succesvolle installatie krijg je een bericht in de console dat het proces is voltooid. 🎉



5. Open **Docker Desktop**. 🖥️
6. Kies of je een desktop shortcut wilt en klik op **Verder**. ➡️
7. Klik op **Close** en start Docker opnieuw op. 🔄

---

## 3 De commando's van Docker 🖥️

---

Je kan de Docker commands opverdelen in 5 soorten.

1. De installatie
2. De images
3. Docker hub
4. Algemene COmmands
5. Containers

Ik ga voor jullie over deze 4 commands gaan. Ik zeg 4, omdat we de installatie hierboven al hebben gezien. Als u niet meer goed weet hoe dat je dat moet doen, ga gerust nog eens kijken.

### 3.1 De Images 📷

---

1. Dit eerste commando is om een Docker image te maken vanaf een DockerFile.

```
docker build -t my-image .
```

- **-t** is een tag die je aan de image geeft. Dit kan een naam zijn, een naam met een versie, of een naam met een versie en een tag.
  - **.** is de directory waar de DockerFile zich bevindt.
- 

2. Dit is om een Docker image te maken vanuit een DockerFile zonder de cache.

```
docker build -t <image_name> . --no-cache
```

- **<image\_name>** Dit is de naam van de image die je wilt maken.
  - **.** is de directory waar de DockerFile zich bevindt.
  - **--no-cache** is een optie die de cache uitschakelt. Dit kan handig zijn als je de image wilt maken van een DockerFile dat je hebt aangepast.
- 

3. Dit commando wordt gebruikt om een lijst van alle Docker images te laten zien

```
docker images
```

- Dit commando laat alle Docker images zien die op je computer zijn. Je kunt ook een speciaal commando gebruiken om een specifieke image te zoeken. Bijvoorbeeld: **docker images -a** om alle images te laten zien, inclusief de niet-gebruikte images.
- 

4. Dit commando is om een Docker image te verwijderen.

```
docker rmi <image_name>
```

- **rmi** Dit betekent verwijderen.
  - **<image\_name>** Dit is de naam van de image die je wilt verwijderen.
- 

5. Dit is om alle images die je niet gebruikt te verwijderen.

```
docker image prune
```

- **prune** Dit betekent afknippen of verwijderen van iets dat niet is gebruikt.
-

## 3.2 De Docker Hub

Docker hub is een platform waar je je eigen Docker images kunt opslaan en delen met anderen. Je kunt ook images van anderen downloaden en gebruiken. Om Docker Hub te gebruiken moet je eerst een account aanmaken. Daarna kun je je eigen images uploaden en downloaden van images van anderen.

1. Dit commando wordt gebruikt voor het inloggen in Docker

```
docker login -u <username>
```

- **login** Dit betekent inloggen.
- **-u** Dit is de optie voor het inloggen met een gebruikersnaam.
- Dit is de naam van je gebruikersnaam op Docker Hub.

- 
2. Dit commando wordt gebruikt om een Docker image te uploaden naar Docker Hub.

```
docker push <username>/<image_name>
```

- **push** Dit betekent dat je de image upload.
- Dit is de naam van je gebruikersnaam op Docker Hub.
- **<image\_name>** Dit is de naam van de image die je wilt uploaden.

- 
3. Dit commando wordt gebruikt om een image te zoeken in de Docker Hub.

```
docker search <image_name>
```

- **search** Dit betekent zoeken.
- **<image\_name>** Dit is de naam van de image die je wilt zoeken.

- 
4. Dit commando wordt gebruikt om een image te downloaden van Docker Hub.

```
docker pull <image_name>
```

- **pull** Dit betekent dat je de image download.
- **<image\_name>** Dit is de naam van de image die je wilt downloaden.

---

## 3.3 Algemene Commands

1. Dit commando wordt gebruikt om de Docker Daemon ofwel de Docker Engine te starten.

```
docker -d
```

- **-d** Dit betekent dat de Docker Daemon in de achtergrond wordt gestart

---

2. Dit commando wordt gebruikt om help te vragen met DOcker. Je kan **-help** op elk subcommando gebruiken.

```
docker --help
```

- **--help** Dit betekent dat je help wilt. Je kan dit op elk subcommand

---

3. Dit commando wordt gebruikt om de info te laten zien.

```
docker info
```

- **info** Dit betekent dat je de info wilt zien. Dit geeft je informatie over

---

## 3.4 Conatiner Commands

1. Dit commando wordt gebruikt om een nieuwe container te maken vanuit een image. Je kan **-it** gebruiken om een interactieve shell te maken.

```
docker run --name <container_name> <image_name>
```

- **run** Dit betekent dat je een nieuwe container wilt maken.
- **--name** Dit betekent dat je de container een naam wilt geven.
- **<container\_name>** Dit is de naam van de container.
- **<image\_name>** Dit is de naam van de image die je wilt gebruiken.

---

2. Dit commando wordt gebruikt om een bestaande container te starten. Dit commando publiceert ook de container zijn ports naar de host.

```
docker run -p <host_port>:<container_port> <image_name>
```

- **run** Dit betekent dat je een bestaande container wilt starten.
- **-p** Dit betekent dat je de ports wilt publiceren.
- **<host\_port>** Dit is de poort op de host die je wilt gebruiken
- **<container\_port>** Dit is de poort in de container die je wilt gebruiken

- **<image\_name>** Dit is de naam van de image die je wilt gebruiken.
- 

3. Dit commando wordt gebruikt om een container te runnen in de background.

```
docker run -d <image_name>
```

- **run** Dit betekent dat je een container wilt runnen.
  - **-d** Dit betekent dat je de container in de background wilt runnen.
  - **<image\_name>** Dit is de naam van de image die je wilt gebruiken.
- 

4. Dit commando wordt gebruikt om een container te starten of te stoppen.

```
docker start|stop <container_name>
```

- **start** Dit betekent dat je de container wilt starten.
  - **stop** Dit betekent dat je de container wilt stoppen.
  - **<container\_name>** Dit is de naam van de container die je wilt starten of
- 

5. Dit commando wordt gebruikt om een gestopte container te verwijderen.

```
docker rm <container_name>
```

- **rm** Dit betekent dat je de container wilt verwijderen.
  - **<container\_name>** Dit is de naam van de container die je wilt verwijderen
- 

6. Dit commando wordt gebruikt om een shell te openen in een container die aan het runnen is.

```
docker exec -it <container_name> sh
```

- **exec** Dit betekent dat je een commando wilt uitvoeren in een best
  - **-it** Dit betekent dat je een interactieve shell wilt openen.
  - **<container\_name>** Dit is de naam van de container die je wilt openen.
  - **sh** Dit is de shell die je wilt openen.
- 

7. Dit commando wordt gebruikt om de logbestanden van een specifieke container te bekijken.

```
docker logs -f <container_name>
```

- **logs** Dit betekent dat je de logbestanden wilt bekijken.
  - **-f** Dit betekent dat je de logbestanden in real-time wilt bekijken.
  - **<container\_name>** Dit is de naam van de container die je wilt bekijken.
- 

8. Dit commando geeft gedetailleerd info over een specifieke container.

```
docker inspect <container_name>
```

- **inspect** Dit betekent dat je gedetailleerd info wilt krijgen over de container.
  - **<container\_name>** Dit is de naam van de container die je wilt inspecteren.
- 

9. Dit commando geeft een lijst weer van alle Docker containers die op dit moment aan het runnen zijn.

```
docker ps
```

- **ps** Dit betekent dat je een lijst wilt krijgen van alle containers die aan het runnen zijn.
- 

10. Dit commando geeft een lijst weer van alle Docker containers die ooit zijn aangemaakt

```
docker ps --all
```

- **ps** Dit betekent dat je een lijst wilt krijgen van alle containers die aan het runnen zijn.
  - **--all** Dit betekent dat je alle containers wilt zien
- 

11. Deze command wordt gebruikt om de resource gebruik statistieken van actieve Docker containers te bekijken.

```
docker container stats
```

- **container** Dit betekent dat je het over de containers hebt.
  - **stats** Dit betekent dat je de statistieken wilt bekijken.
- 
- 

## 4 Dockerfile

---

### 4.1 Wat is een Dockerfile?



Zoals we allemaal nu al weten, hebben we een Dockerfile nodig om een Docker container te bouwen, of eerlijk een Image. Maar wat is een Dockerfile precies? Dat zal ik voor jullie goed uitleggen. Een Dockerfile is eigenlijk een tekstdocument waarin instructies staan voor het bouwen van Docker images. Het is een soort recept voor het bouwen van een Docker image. Docker zal dit bestand automatisch lezen, en zal aan de hand daarvan de instructies uitvoeren om een Docker image te bouwen. Het is dus een soort script, maar dan voor Docker.

Wij hebben als opdracht gekregen om een eigen Dockerfile te maken. Ik heb een simpele dockerfile gemaakt, met daaraan een python file. Wat dit doet, is als je op de website gaat, dat er "Hello World" komt te staan.



## 4.2 De Dockerfile

Ik ga nu de Dockerfile stap voor stap uitleggen.

---

Als eerste lijn heb ik deze code

```
FROM python:3.9-slim
```

Dit wilt zeggen dat we de officiële Python image gebruiken als de basis van de image.

---

Op mijn 2de lijn code staat:

```
WORKDIR /app
```

Dit is om de directory in te stellen.

---

Daarna heb ik dit:

```
COPY requirements.txt .
```

Dit is om de requirements van de requirements.txt te kopiëren naar de container.

---

Als dat gedaan is, dan hebben we dit:

```
RUN pip install --no-cache-dir -r requirements.txt
```

Dit zorgt er eigenlijk voor dat de vereisten die we daarnet gekopieerd hebben, dat deze worden gedownload.

---

Dan hebben we als voorlaatste regel:

```
COPY . .
```

Dit kopieert de rest van de applicatie naar de container. Dus dit kopieert alle andere files waar we in hebben gecodeerd.

---

Dan hebben we als aller laatste lijn code dit:

```
CMD ["python", "app.py"]
```

Dit stelt het commando in dat moet worden uitgevoerd bij het starten van de container. In dit geval is dat app.py

---

---

## 5 Docker Compose

---

### 5.1 Wat is Docker Compose?

Docker compose is eigenlijk een tool die je helpt om meerdere Docker containers te beheren en te configureren. Docker compose bestaat uit een yaml file. Daarin wordt de configuratie van de containers dan geconfigureert. je kan veel met Docker Compose doen, hier zijn een paar voorbeelden

- Je kan meerdere containers definiëren en configureren
- Je kan containers opstarten en stoppen
- wat ook heel handig is, is dat je containers kan laten communiceren met elkaar

### 5.2 Docker Compose bestand met uitleg

Ik heb een voorbeeld van een Docker Compose bestand gemaakt. Ik zal eerst heel het bestand geven, en zal het daarna uitleggen.

```
version: '3.8'

services:
  web:
    build:
      context: ./web
    ports:
      - "5000:5000"
    depends_on:
      - db
    environment:
      - MONGO_URI=mongodb://db:27017/mydatabase
```

```
db:
  image: mongo
  ports:
    - "27017:27017"
  volumes:
    - mongo_data:/data/db

volumes:
  mongo_data:
```

Als eerste staat er `version: '3.8'`. Dit zegt Docker Compose, dit is welke versie van Docker Compose we gebruiken.

We hebben dan de lijn `services:`, hierin worden de verschillende services gedefinieerd.

De eerste service die we hebben is `web`. Er zijn een paar belangrijke dingen. Als eerste de `ports`, dit is de port dat we gebruiken om onze website op te plaatsen. Dan hebben we de `depends_on`, dit zorgt er dan weer voor dat de database sectie wordt gestart voor dat de website wordt gestart.

De tweede service dat we hebben is de `mongo database`. Deze wordt ook weeral met een port duidelijk gedefinieerd.