

Container

Proseminar Softwareentwicklung in der Wissenschaft

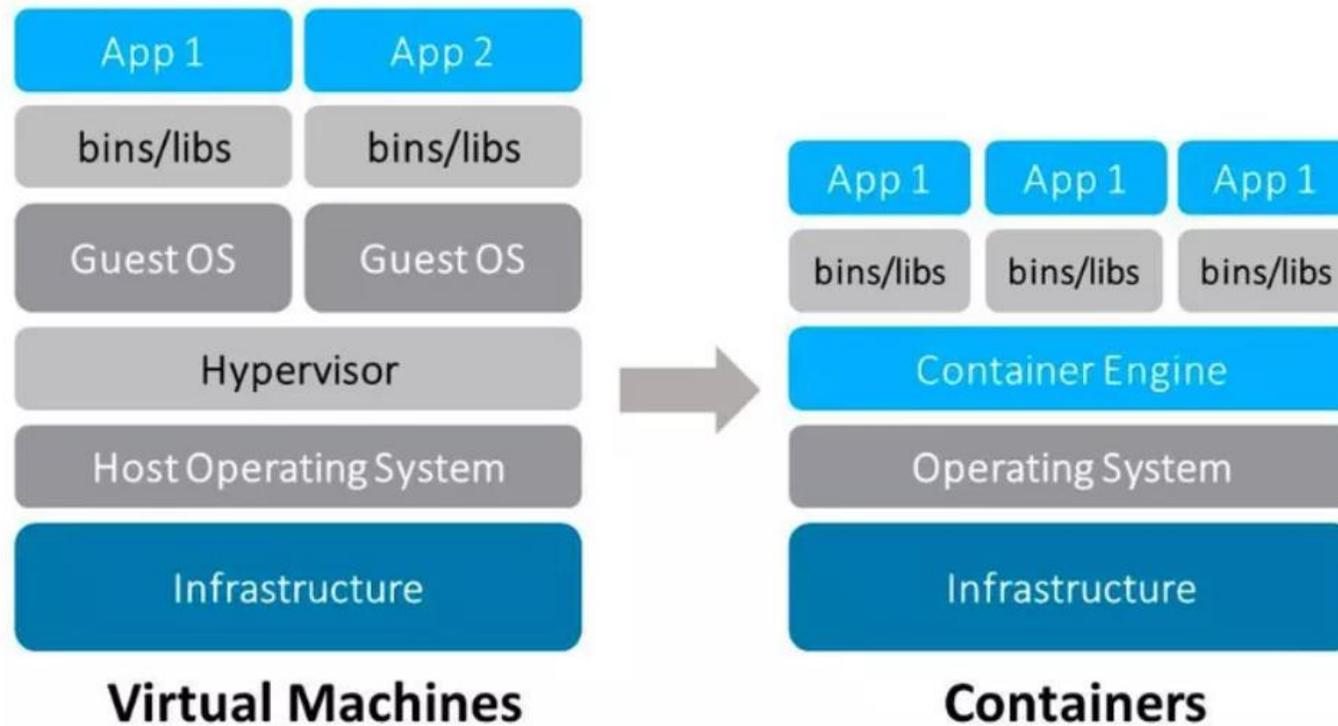
Pascal Kröger

Juni 2020

Inhalt

1. Was sind Container?
2. Warum?
3. Einführung in Docker
4. Podman
5. Weitere Container Engines
6. Zusammenfassung
7. Ausblick

Was sind Container?

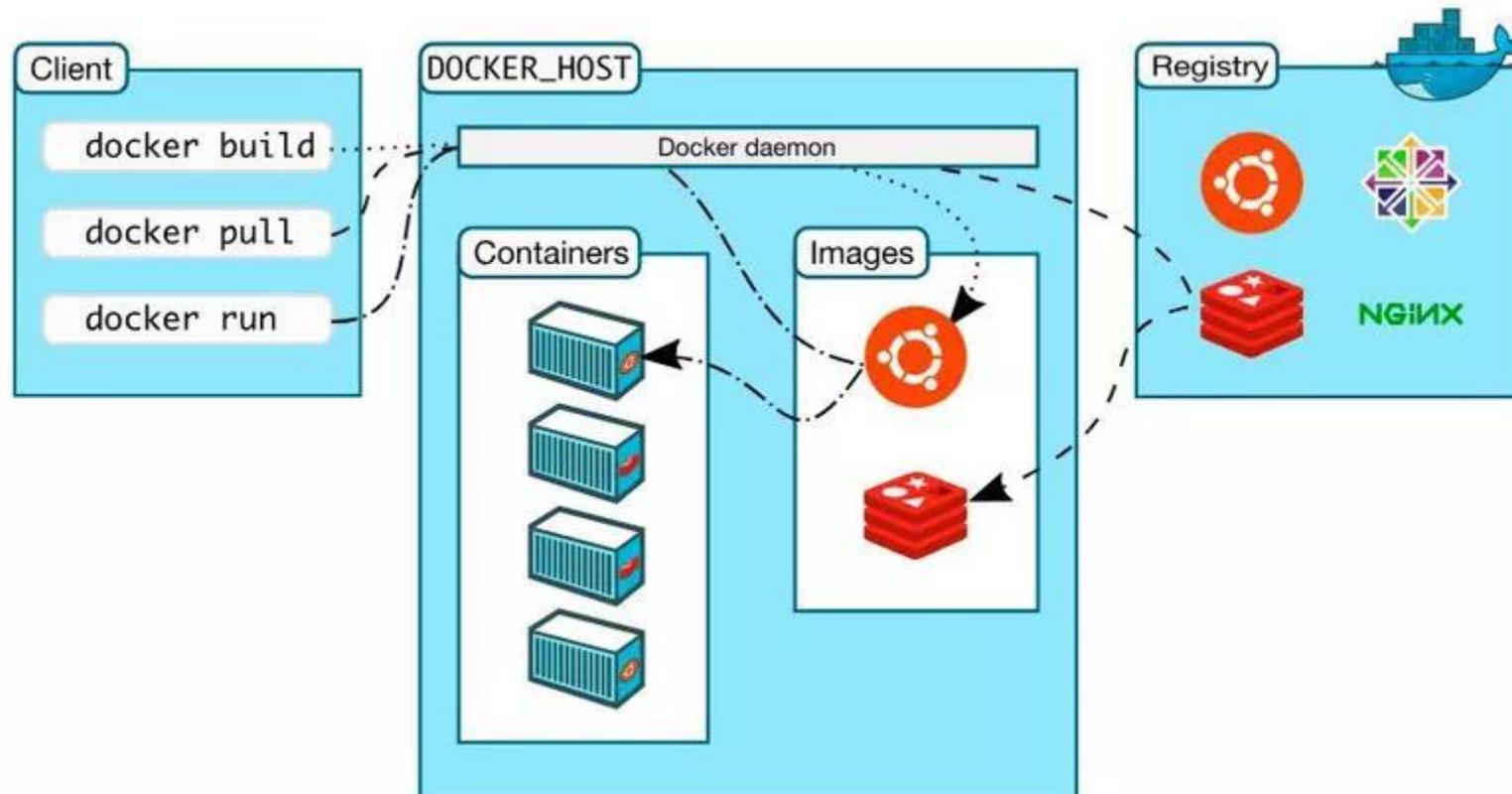


(Bild: Dev-Insider.de)

Warum?

- Container enthält alles, was die Applikation benötigt
- Abstrahierung des Betriebssystems
- Trennung von Abhängigkeiten
- “Leichtgewichtig”

Docker



(Bild: Dev-Insider.de)

Dockerfile

- Wird benötigt um Image zu bauen
- Beschreibt Inhalt des Images/Containers

```
1 FROM mcr.microsoft.com/dotnet/core/sdk:3.1-buster
2 WORKDIR /src
3 EXPOSE 80
4 COPY . .
5 RUN dotnet restore "./SIW20.csproj"
6 RUN dotnet build "./SIW20.csproj" -c Release -o /app/build
7
8 RUN dotnet publish "./SIW20.csproj" -c Release -o /app/publish
9 ENTRYPOINT ["dotnet", "SIW20.dll"]
10
```

Dockerfile

- Image ohne SDK 200 MB
- Image mit SDK 700 MB

```
1 FROM mcr.microsoft.com/dotnet/core/aspnet:3.1-buster-slim AS base
2 WORKDIR /app
3 EXPOSE 80
4
5 FROM mcr.microsoft.com/dotnet/core/sdk:3.1-buster AS build
6 WORKDIR /src
7 COPY . .
8 RUN dotnet restore "./SIW20.csproj"
9 RUN dotnet build "./SIW20.csproj" -c Release -o /app/build
10
11 FROM build AS publish
12 RUN dotnet publish "./SIW20.csproj" -c Release -o /app/publish
13
14 FROM base AS final
15 WORKDIR /app
16 COPY --from=publish /app/publish .
17 ENTRYPOINT ["dotnet", "SIW20.dll"]
```

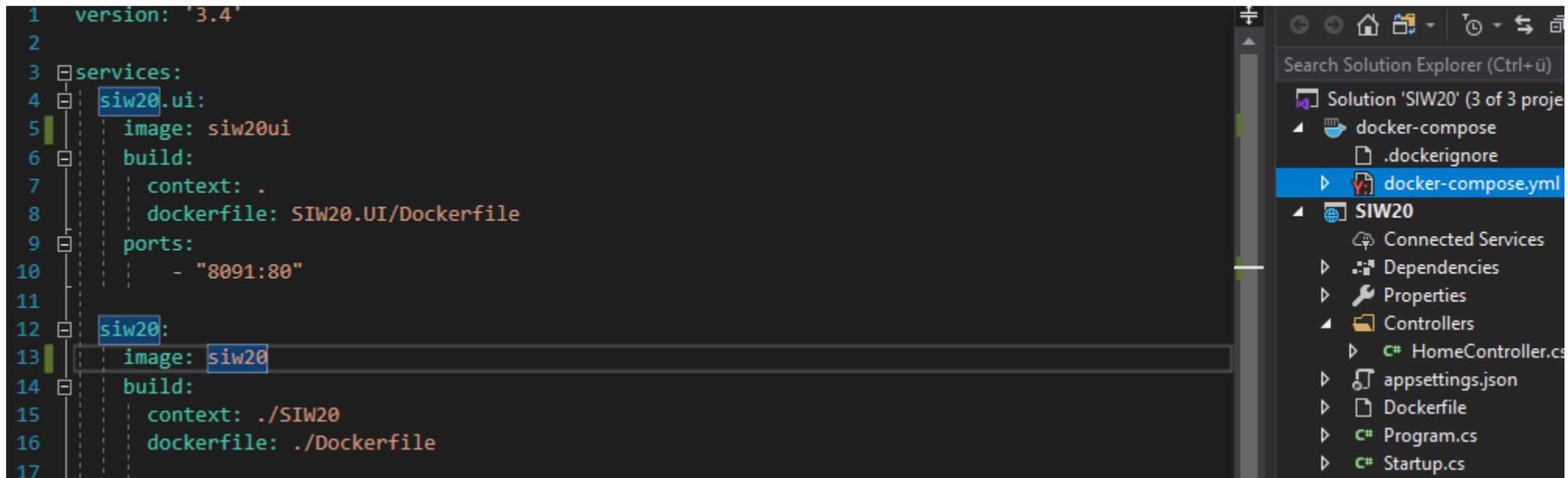
Demo einer API

Docker Compose

- Multi Container
- Netzwerk zwischen Containern
- docker-compose.yml

Docker-compose.yml

```
1 version: '3.4'
2
3 services:
4   siw20.ui:
5     image: siw20ui
6     build:
7       context: .
8       dockerfile: SIW20.UI/Dockerfile
9     ports:
10      - "8091:80"
11
12   siw20:
13     image: siw20
14     build:
15       context: ./SIW20
16       dockerfile: ./Dockerfile
17
```



The image shows a screenshot of Visual Studio Code. The main editor window displays the content of a Docker-compose.yml file. The file is structured as follows:

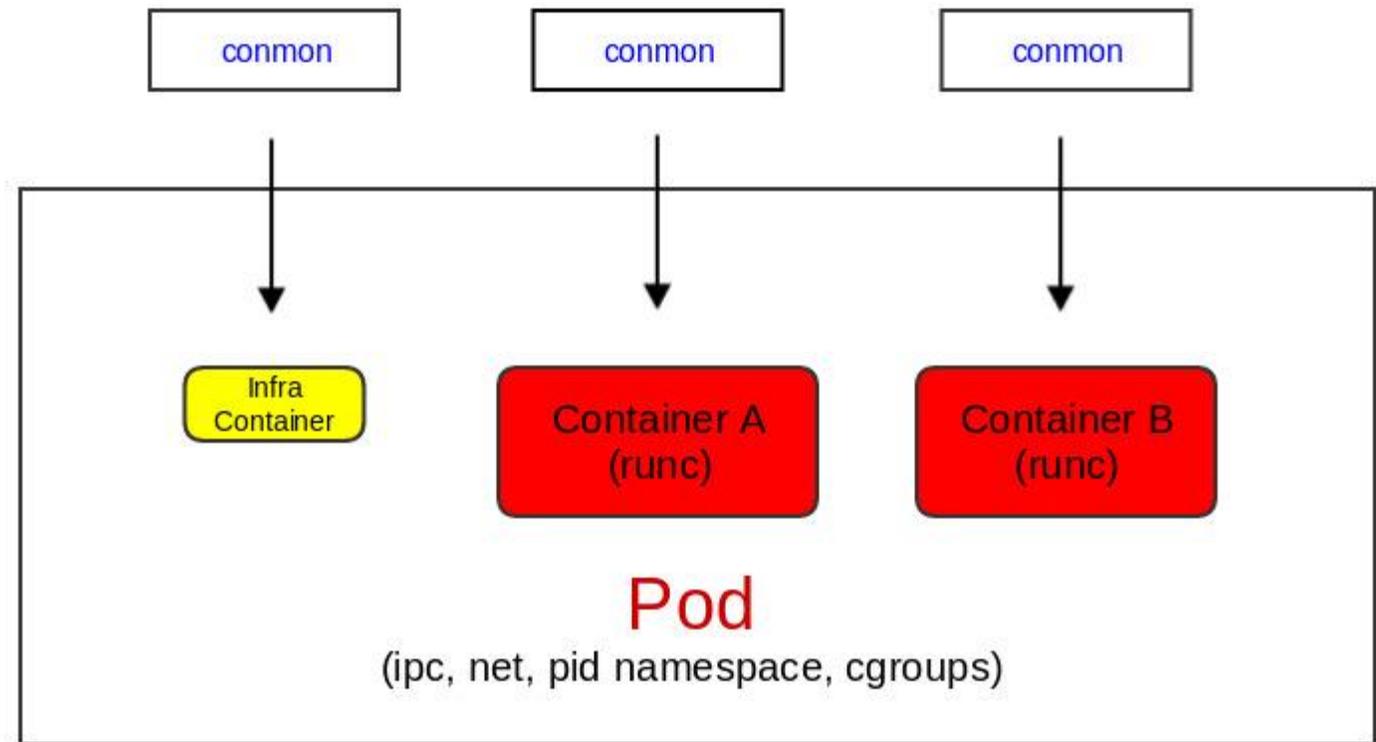
- version: '3.4'
- services:
 - siw20.ui:
 - image: siw20ui
 - build:
 - context: .
 - dockerfile: SIW20.UI/Dockerfile
 - ports:
 - "8091:80"
 - siw20:
 - image: siw20
 - build:
 - context: ./SIW20
 - dockerfile: ./Dockerfile

The Solution Explorer on the right side of the screen shows a project named 'SIW20'. The file 'docker-compose.yml' is highlighted in the file list. Other files visible in the Solution Explorer include 'docker-compose', '.dockerignore', 'Connected Services', 'Dependencies', 'Properties', 'Controllers', 'HomeController.cs', 'appsettings.json', 'Dockerfile', 'Program.cs', and 'Startup.cs'.

Demo einer API mit UI

Podman

- Podman alias Docker
- Daemonless
- Container Engine muss nicht unter Root laufen



(Bild: Netways.de)

Andere Container Engines

- Singularity
- Shifter

Zusammenfassung

- Container sind Abstrahierung VM
- Enthalten alles nötige um app zu betreiben
- “Leichtgewichtig”
- Portabel

Ausblick

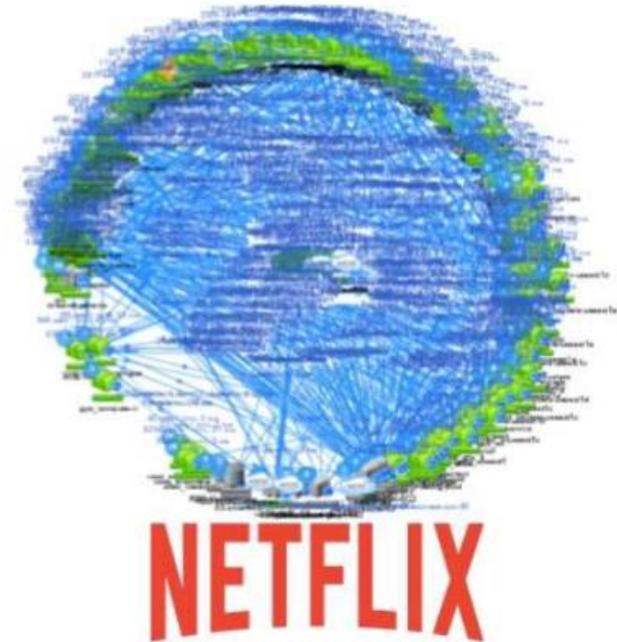
- Persistenter Speicher
- Microservices
- Container Orchestration



kubernetes

(Bild: Teuto.net)

500+ microservices



(Bild: Slideshare.net)

Literatur

- <https://docs.docker.com/>
- <https://podman.io/>
- <https://www.netways.de/blog/2019/05/31/podman-ist-dem-docker-sein-tod/>
- https://www.reddit.com/r/docker/comments/7y2yp2/why_is_singularity_used_as_opposed_to_docker_in/
- https://tin6150.github.io/psg/blogger_container_hpc.html