

Docker & Kubernetes

Teil 3

Teil 3 – Installation und Nutzung

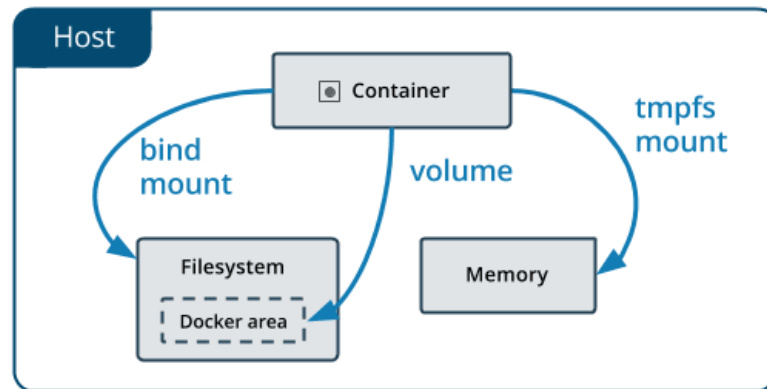
- Docker
 - Ressourcen (Netzwerk, Volumes)
 - Logging
 - Registry
 - Beispiele
- Portainer
- Docker Compose

Docker CLI – *volume* Kommando

- Volumes (Wiederholung :-)
 - Im Host-Filesystem
 - Zugriff nur aus Containern
 - ***Verwaltung durch Docker***



CLI-Kommandos



Docker CLI – *volume* Kommando

- Volumes auflisten

```
$ docker volume list
```



```
Terminal - mb-GLS02VMZ - ~/Desktop/Schulung-Docker
```

Datei	Bearbeiten	Ansicht	Terminal	Reiter	Hilfe
-------	------------	---------	----------	--------	-------

```
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $ docker volume list  
DRIVER          VOLUME NAME  
local           9eb65f59f8c72f02561fa10caa05355ecb386e6c13fcc0ef6b162a443ae5c73  
local           488dc9ad18ea0cbca0c5c34d474ade85e1077b2b587f83531405532dfcc2ab5  
local           621c940eebc2be98105c6abdfa33755353e7b1b5d06000c7154b17872db84bf0  
local           8164df8ec93bf622066d4469ef80eca7a294d68b66d915c010d34c46c24e13a71  
local           e13cd4755b841e8ca2c26b7bf8ed5f5ba1ab7729f80176d76a18b2a7c9076f557  
local           portainer_portainer-docker-extension-desktop-extension_portainer_data  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GLS02VMZ ~/Desktop/Schulung-Docker $
```

Docker CLI – *volume* Kommando

- Ein Volume erzeugen

```
$ docker volume create [OPTIONS] [<volume name>]
```

```
$ docker volume create testvolume
```



Mit der Option „--driver“ kann ein Treiber angegeben werden. Die in Docker installierten Volume-Treiber können mit „docker info“ ausgelesen werden.

```
Terminal - mb@mb-GL502VMZ ~/Desktop/Schulung-Docker
```

Datei	Bearbeiten	Ansicht	Terminal	Reiter	Hilfe
-------	------------	---------	----------	--------	-------

```
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
testvolume  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker volume list  


| DRIVER | VOLUME NAME                                                           |
|--------|-----------------------------------------------------------------------|
| local  | portainer_portainer-docker-extension-desktop-extension_portainer_data |
| local  | testvolume                                                            |

  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
```

Docker CLI – *volume* Kommando

- Ein Volume untersuchen

```
$ docker volume inspect <volume name>
```

```
$ docker volume inspect testvolume
```



```
Terminal - mb@mb-GL502VMZ ~/Desktop/Schulung-Docker
Datei Bearbeiten Ansicht Terminal Reiter Hilfe
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker volume create testvolume
testvolume
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker volume list
DRIVER      VOLUME NAME
local       portainer portainer-docker-extension-desktop-extension_portainer_data
local       testvolume
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker volume inspect testvolume
[
  {
    "CreatedAt": "2022-08-31T08:52:41Z",
    "Driver": "local",
    "Labels": {},
    "Mountpoint": "/var/lib/docker/volumes/testvolume/_data",
    "Name": "testvolume",
    "Options": {},
    "Scope": "local"
  }
]
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
```


Docker CLI – *volume* Kommando

- Volumes aufräumen

```
$ docker volume prune
```

Prune entfernt alle ungenutzten, lokalen Volumes.



```
Terminal - mb@mb-GL502VMZ ~/Desktop/Schulung-Docker
Datei Bearbeiten Ansicht Terminal Reiter Hilfe
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker volume list
DRIVER      VOLUME NAME
local       9eb65f59f8c72f02561fa10caa05355ecb386e6c13fccf0ef6b162a443ae5c73
local       408dc9ad18ea0cbea0c5c34d474ade85e1077b2b587f83531405532dfcfc2ab5
local       621c940eebc2be98105c6abdfa33755353e7b1b5d06000c7154b17872db84bf0
local       8164df8ec93bf622066d4469ef80ec7a294d68b66d915c010d34c46c24e13a71
local       e13c4d755b841e8ca2c26b7bf8ed5f5a1ab7729f80176d76a18b2a7c9076f557
local       portainer_portainer-docker-extension-desktop-extension_portainer_data
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker volume prune
WARNING! This will remove all local volumes not used by at least one container.
Are you sure you want to continue? [y/N] y
Deleted Volumes:
9eb65f59f8c72f02561fa10caa05355ecb386e6c13fccf0ef6b162a443ae5c73
e13c4d755b841e8ca2c26b7bf8ed5f5a1ab7729f80176d76a18b2a7c9076f557
408dc9ad18ea0cbea0c5c34d474ade85e1077b2b587f83531405532dfcfc2ab5
621c940eebc2be98105c6abdfa33755353e7b1b5d06000c7154b17872db84bf0
8164df8ec93bf622066d4469ef80ec7a294d68b66d915c010d34c46c24e13a71
Total reclaimed space: 0B
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker volume list
DRIVER      VOLUME NAME
local       portainer_portainer-docker-extension-desktop-extension_portainer_data
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
```


Docker CLI – *network* Kommando

- Netzwerke (Wiederholung :-)
 - Treiber
 - Bridge (default)
 - Mehrere Container sollen auf einem Docker-Host kommunizieren
 - Host – keine Isolierung zwischen Host und Container
 - Overlay – für Kommunikation in Clustern (multi-host)
 - Container auf mehreren Docker-Hosts sollen miteinander kommunizieren
 - None
 - Macvlan – Container bekommen eigene MAC-Adressen
 - Installierte Treiber mit „docker info“ auslesbar

Docker CLI – *network* Kommando

- Netzwerke auflisten

```
$ docker network list
```

[illegible]

Docker CLI – *network* Kommando

- Ein Netzwerk erzeugen

```
$ docker network create [OPTIONS] [<network name>]
```

```
$ docker network create --driver bridge testnet
```

Eigene bridge-Netzwerke besitzen DNS auf Basis der Container-Namen.

Wichtige Optionen:

```
--driver=<name>
```

- Den Treiber festlegen

```
--gateway=<ip-number>
```

- Das Gateway definieren

```
--subnet=<subnet>
```

- Subnet im CIDR-Format

```
--ip-range=<ip-range>
```



```
Terminal - mb@mb-GL502VMZ ~/Desktop/Schulung-Docker
```

Datei	Bearbeiten	Ansicht	Terminal	Reiter	Hilfe
-------	------------	---------	----------	--------	-------

```
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker network create --driver bridge testnet  
1da68a1427a5b6d54b48206843f3caff127b359a406e021e82027b3dda71d282  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $  
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker network list
```

NETWORK ID	NAME	DRIVER	SCOPE
0cd7db3256d3	bridge	bridge	local
ca963f396951	host	host	local
fdf6391eda4e	none	null	local
5a6e66d0800d2	portainer_portainer-docker-extension-desktop-extension_default	bridge	local
1da68a1427a5	testnet	bridge	local

```
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
```

Docker CLI – *network* Kommando

- Informationen über ein Netzwerk

```
$ docker network inspect <network name>
```

```
$ docker network inspect testnet
```

Informationen über das Netzwerk, inkl.
aller verbundenen Container.



```
Terminal - mb@mb-GL502VMZ ~/Desktop/Schulung-Docker
Datei Bearbeiten Ansicht Terminal Reiter Hilfe
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker network inspect testnet
[
  {
    "Name": "testnet",
    "Id": "1da68a1427a5b6d54b48206843f3caff127b359a406e021e82027b3dda71d282",
    "Created": "2022-08-31T09:42:32.64337747Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.19.0.0/16",
          "Gateway": "172.19.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
  }
]
```

Docker CLI – *network* Kommando

- Ein Netzwerk entfernen

```
$ docker network rm <network name>
```

```
$ docker network rm testnet
```

Es können nur Netzwerke entfernt werden, mit denen keine Container mehr verbunden sind.



```
Terminal - mb@mb-GL502VMZ ~/Desktop/Schulung-Docker
Datei Bearbeiten Ansicht Terminal Reiter Hilfe
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker network list
NETWORK ID          NAME                DRIVER            SCOPE
0cd7db3256d3        bridge             bridge            local
ca963f396951        host               host              local
fdf6391eda4e        none              null              local
5a6e66d080d2        portainer_portainer-docker-extension-desktop-extension_default bridge            local
1da68a1427a5        testnet            bridge            local
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker network rm testnet
testnet
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker network list
NETWORK ID          NAME                DRIVER            SCOPE
0cd7db3256d3        bridge             bridge            local
ca963f396951        host               host              local
fdf6391eda4e        none              null              local
5a6e66d080d2        portainer_portainer-docker-extension-desktop-extension_default bridge            local
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
```

Docker CLI – *network* Kommando

- Netzwerk aufräumen

```
$ docker network prune
```

Es können nur Netzwerke entfernt werden, mit denen keine Container mehr verbunden sind.



```
Terminal - mb@mb-GL502VMZ ~/Desktop/Schulung-Docker
Datei Bearbeiten Ansicht Terminal Reiter Hilfe
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker network list
NETWORK ID          NAME                DRIVER              SCOPE
0cd7db3256d3        bridge             bridge              local
b2b923c16331        docker_gwbridge    bridge              local
ca963f396951        host               host                local
f32zr2zby77b        ingress            overlay             swarm
fdf6391eda4e        none               null                local
nvncx59ugnkv        ovnet              overlay             swarm
5a6e66d080d2        portainer_portainer-docker-extension-desktop-extension_default bridge          local
0edc21d46290        testnet            bridge              local
f0ed0acc7af4        testnet1           bridge              local
b0aff7000e74        testnet0815        bridge              local

mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker network prune
WARNING! This will remove all custom networks not used by at least one container.
Are you sure you want to continue? [y/N] y
Deleted Networks:
ovnet
testnet
testnet0815
testnet1

mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker network list
NETWORK ID          NAME                DRIVER              SCOPE
0cd7db3256d3        bridge             bridge              local
b2b923c16331        docker_gwbridge    bridge              local
ca963f396951        host               host                local
f32zr2zby77b        ingress            overlay             swarm
fdf6391eda4e        none               null                local
5a6e66d080d2        portainer_portainer-docker-extension-desktop-extension_default bridge          local
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
```

Docker CLI – *network* Kommando

- Container mit Netzwerk verbinden/lösen

```
$ docker network connect <network-name> <container-name>
$ docker network disconnect
```

```
$ docker network create testnet
$ docker container run --name alpine --rm -i -t -d alpine sh -c "while true; do ls; sleep 3; done"
$ docker network connect testnet alpine
$ docker network disconnect testnet alpine
```

Netzwerke können nachträglich mit laufenden Containern verbunden werden und/oder von diesen gelöst werden.

Container beim Start verbinden:

```
$ docker container run --network=testnet alpine
```

A screenshot of a terminal window titled "Terminal - mb@mb-GL502VMZ ~/Desktop/Schulung-Docker". The terminal shows a series of commands and their outputs. The first part shows the user running a loop command in an alpine container. Then, they run "docker network connect testnet alpine" and "docker network disconnect testnet alpine". The terminal output shows the container's file system being listed repeatedly, and the network connection status being updated. The terminal has a menu bar with "Datei", "Bearbeiten", "Ansicht", "Terminal", "Reiter", and "Hilfe". There are also some icons on the right side of the terminal window.

Docker – Overlay-Netzwerk

- Nutzung mit Docker-Swarm
 - `docker swarm init` auf Master
 - `docker swarm join` auf anderen Docker Servern
 - `docker network create --attachable --driver overlay ...` auf Master
- Nutzung der Overlay-Netzwerke mit den Containern/Services der Docker-Server im Swarm
- <https://docs.docker.com/network/network-tutorial-overlay/>

Docker – Logging

- „12 Factors“ => XI. Logs
 - Anwendung/Service kümmert sich nicht darum
 - Strom der Ereignisse wird auf *stdout* und *stderr* geschrieben
 - Laufzeitumgebung kümmert sich um das Sammeln
- `docker logs <container>` liest von *stdout* und *stderr*
- Logging-Driver können das Verhalten ändern
- Sammlung durch verschiedene Log-/Event-Collectors möglich (Fluentd, Splunk, ...)
- <https://docs.docker.com/config/containers/logging/local/>

Docker – Container Registry

- K kümmert sich um die Verwaltung von Images
- Private & public Repositories sind möglich
- Versionierung / Speicherung mit Tags

- Anmeldung mit

```
docker login -p <passwd> -u <user> <server>
```

- Abmeldung mit

```
docker logout <server>
```

Docker – Registry Möglichkeiten

- Docker Hub – Angebot von Docker Inc.
- Docker Registry ... self hosted ... open source
<https://docs.docker.com/registry/>
- Kommerzielle Angebote:
 - GitLab, DigitalOcean, AWS, GCP, Azure, ...
 - Canister ... <https://canister.io/>
- Harbor ... <https://goharbor.io/>
 - Open source, mit UI
 - Für self hosting
 - Aus dem CNCF-Stack

Docker – Registry Möglichkeiten

- Docker Hub – Angebot von Docker Inc.
- Docker Registry ... self hosted ... open source
<https://docs.docker.com/registry/>
- Kommerzielle Angebote:
 - GitLab, DigitalOcean, AWS, GCP, Azure, ...
 - Canister ... <https://canister.io/>
- Harbor ... <https://goharbor.io/>
 - Open source, mit UI
 - Für self hosting
 - Aus dem CNCF-Stack

Docker – Docker Registry

- Installation als Docker-Container

```
$ docker network create registry
$ docker container run -d -p 5000:5000 --network registry --restart=always --name registry registry:2
```



- Ein Image bauen und in die Registry senden

```
$ docker image build --file Dockerfile-1.test --tag localhost:5000/testimage:v1 .
$ docker image push localhost:5000/testimage:v1
```



Aufbau einer eigenen, einfachen Registry.

Eigenes Netzwerk für Nutzung von DNS.

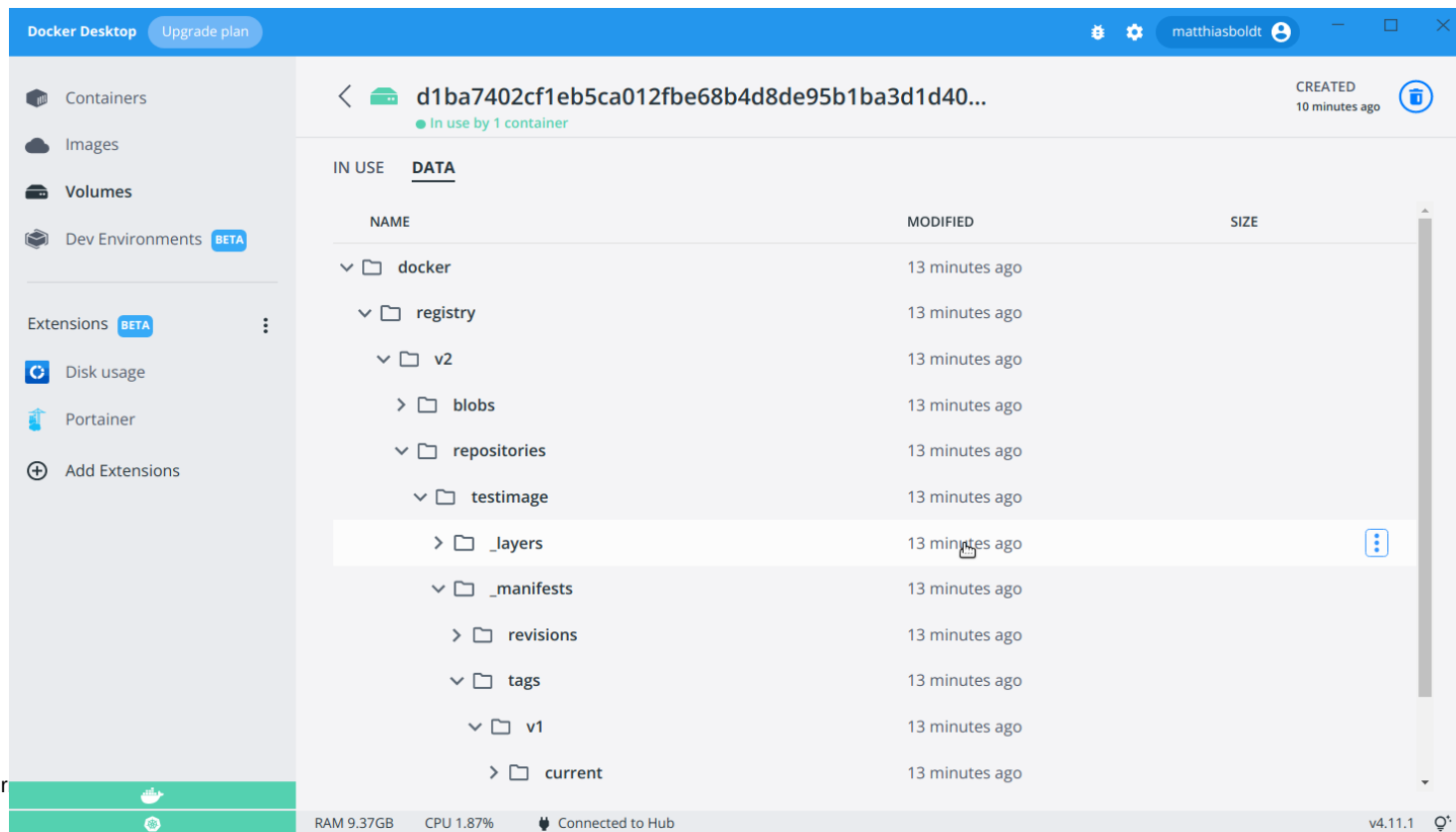
Kommandos in Datei „Registry.sh“

```
Terminal - mb@mb-GL502VMZ ~/Desktop/Schulung-Docker
Datei Bearbeiten Ansicht Terminal Reiter Hilfe

mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker image build --file Dockerfile-1.test --tag localhost:5000/testimage:v1 .
[+] Building 0.1s (5/5) FINISHED
=> [internal] load build definition from Dockerfile-1.test 0.0s
=> => transferring dockerfile: 216B 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/alpine:latest 0.0s
=> CACHED [1/1] FROM docker.io/library/alpine:latest 0.0s
=> exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:ef07e415a6f477fe79e0fclade18ac0441b174d7042c13199a5f9bfd13ce02f6 0.0s
=> => naming to localhost:5000/testimage:v1 0.0s
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker image push localhost:5000/testimage:v1
The push refers to repository [localhost:5000/testimage]
994393dc58e7: Pushed
v1: digest: sha256:a5be6edee921579cfeba07196e3a890539489f0ba3fef57bed45387fff34e75c size: 527
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
```


Docker – Docker Registry

- Daten werden in einem Volume gehalten



Docker – Docker Registry

- WEB-Viewer für die eigene Docker Registry

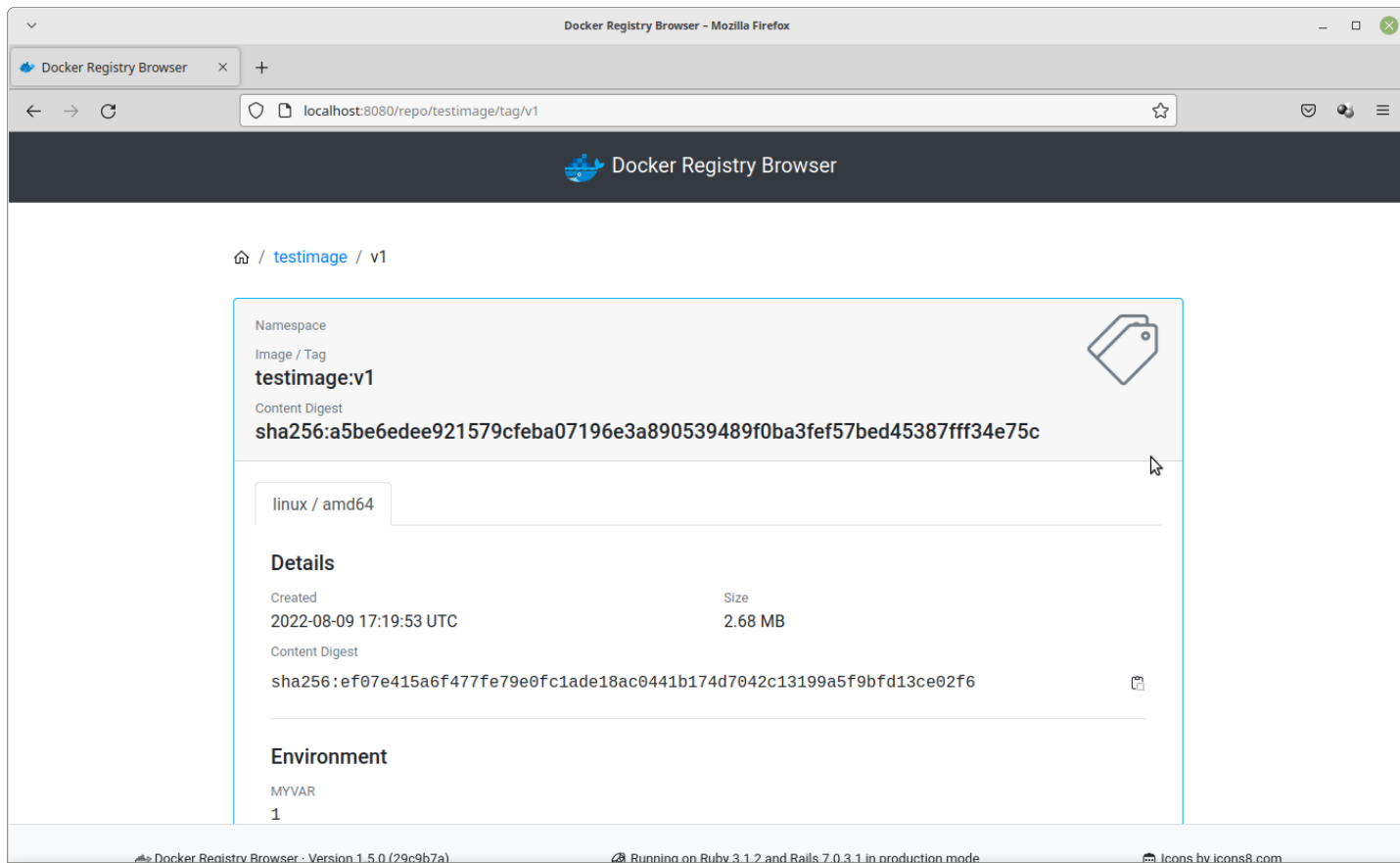
```
$ docker container run -d --name registry-browser --network registry --env  
DOCKER_REGISTRY_URL=http://registry:5000 -p 8080:8080 klausmeyer/docker-registry-browser
```



- Open-Source-Projekt von Klaus Meyer
- <https://github.com/klausmeyer/docker-registry-browser>
- <https://github.com/klausmeyer/docker-registry-browser/blob/master/docs/README.md>
- Der Viewer ist über <http://localhost:8080> zu erreichen

Docker – Docker Registry

Ansicht der Information
zum erzeugten Image
im Browser über
<http://localhost:8080>



Docker – Shell-Login

- Shell-login in einen laufenden Linux-Container
- Für Debugging, Tests und Inspektion
- Nutzung von „docker container exec ...“

```
$ docker container exec -it <container name> <shell>
```

```
$ docker container exec -it alpine /bin/sh
```



- Welche Shell gestartet wird, hängt vom Basis-System ab

Docker – Portainer

- Nahezu alle Docker-CLI Kommandos aus UI erreichbar
- Komplette Übersicht im Gegensatz zum UI von Docker Desktop
- Als Extension in Docker Desktop installierbar
- Auf Linux Server unter Docker Engine separat installierbar
- In Datei „CMD-Docker-auf-Linux-Server.txt“ ist die Installation beschrieben
- ... sehen wir uns Portainer einmal an ...
 - Docker Objekte verwalten
 - Image aus TAR-Archiv importieren



Docker – einfache Beispiele

- Wordpress

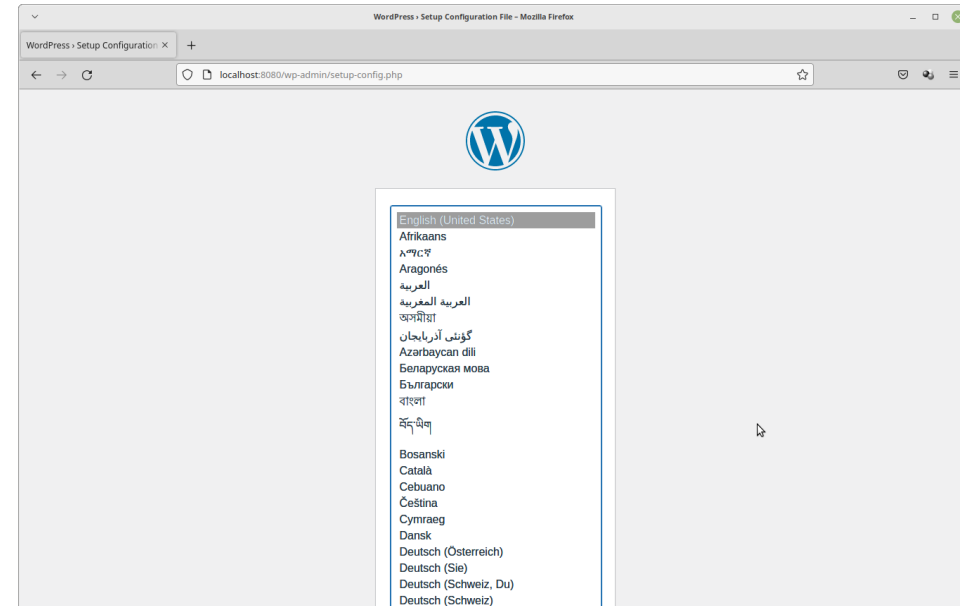
```
$ docker network create wordpress
```

```
$ docker container run --name wordpress --network wordpress -p 8080:80 -d wordpress
```

- Ist über <http://localhost:8080> erreichbar



```
Terminal - mb@mb-GL502VMZ ~/Desktop/Schulung-Docker
Datei Bearbeiten Ansicht Terminal Reiter Hilfe
2ed9d27caef portainer/portainer-docker-extension:2.14.2 "/portainer --admin..." 5 days ago
Up 15 minutes 127.0.0.1:8000->8000/tcp, 127.0.0.1:9000->9000/tcp, 127.0.0.1:9443->9443/tcp
portainer portainer-docker-extension-desktop-extension-service
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker container exec -it wordpress /bin/sh
# ls
index.php wp-admin wp-config-sample.php wp-links-opml.php wp-settings.php
license.txt wp-blog-header.php wp-content wp-load.php wp-signup.php
readme.html wp-comments-post.php wp-cron.php wp-login.php wp-trackback.php
wp-activate.php wp-config-docker.php wp-includes wp-mail.php xmlrpc.php
# ps aux
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
root 1 0.0 0.2 231124 36136 ? Ss 18:40 0:00 apache2 -DFOREGROUND
www-data 23 0.0 0.1 231748 22396 ? S 18:40 0:00 apache2 -DFOREGROUND
www-data 24 0.0 0.0 231624 10672 ? S 18:40 0:00 apache2 -DFOREGROUND
www-data 25 0.0 0.1 231292 18600 ? S 18:40 0:00 apache2 -DFOREGROUND
www-data 26 0.0 0.4 308844 50016 ? S 18:40 0:00 apache2 -DFOREGROUND
www-data 28 0.0 0.0 231624 10708 ? S 18:40 0:00 apache2 -DFOREGROUND
www-data 32 0.0 0.3 311016 48376 ? S 18:42 0:00 apache2 -DFOREGROUND
www-data 33 0.0 0.3 236364 39836 ? S 18:42 0:00 apache2 -DFOREGROUND
www-data 34 0.0 0.3 236092 40200 ? S 18:42 0:00 apache2 -DFOREGROUND
www-data 35 0.0 0.0 231180 8844 ? S 18:42 0:00 apache2 -DFOREGROUND
www-data 36 0.0 0.0 231164 8844 ? S 18:42 0:00 apache2 -DFOREGROUND
root 40 0.6 0.0 2420 584 pts/0 Ss 18:59 0:00 /bin/sh
root 47 0.0 0.0 6700 2860 pts/0 R+ 18:59 0:00 ps aux
# exit
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $
```



Docker – einfache Beispiele

- MySQL-DB-Server

```
$ docker container run --name mysql --network wordpress -e MYSQL_ROOT_PASSWORD=mypwd -d mysql:latest
```



```
Terminal - mb@mb-GL502VMZ ~/Desktop/Schulung-Docker
Datei Bearbeiten Ansicht Terminal Reiter Hilfe
mb@mb-GL502VMZ ~/Desktop/Schulung-Docker $ docker container exec -it mysql /bin/bash
bash-4.4# mysql -h localhost --user=root --password=password
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.0.30 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.00 sec)

mysql>
```

Docker – einfache Beispiele

- Wordpress & MySQL-DB-Server verbinden
 - Beide Container im gleichen bridge Netzwerk
 - In Wordpress-Konfiguration den MySQL-Server angeben



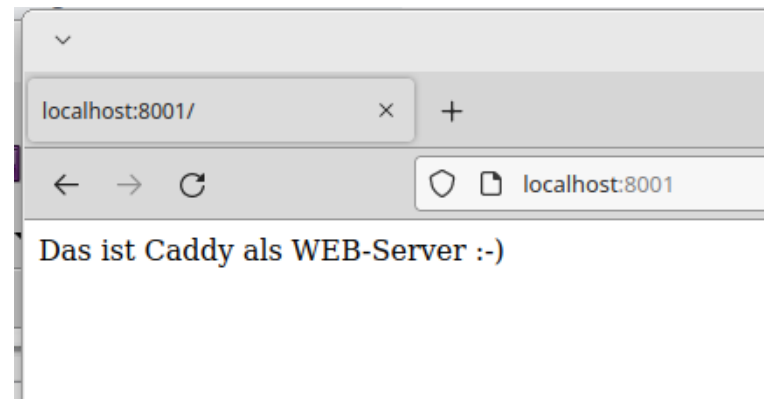
Docker – einfache Beispiele

- WEB-Server mit Caddy

```
$ echo "Das ist Caddy als WEB-Server :-)" > index.html  
$ docker run -d -p 8001:80 -v $PWD/index.html:/usr/share/caddy/index.html -v caddy_data:/data caddy
```



- Erreichbar über <http://localhost:8001>
- Kann sehr einfach als reverse proxy genutzt werden



Docker - Sicherheit

- Mögliche Angriffsvektoren
 - Betriebssystem
 - Netzwerk
 - Docker Daemon
 - Container
 - Prozesse in Containern
 - ...

Docker - Sicherheit

- Container nutzen das Host-OS (Kernel, Betriebssystem, Netzwerk, ...)
 - Kernel-Security-Features
 - Firewall
- Docker Daemon
 - REST-API ... Zugriffsbeschränkung
- Images
 - signed/trusted images
 - docker scan
- Container
 - Zugriffsbeschränkungen für Prozesse
 - Volumes sichern
 - Netzwerke mit minimaler Sichtbarkeit konfigurieren

Docker Compose

- Erweiterung für Docker
- Verwaltung von Docker-Objekten für vollständige Anwendungen (z.B. mit mehreren Containern)
 - Container
 - Netzwerke
 - Volumes
 - ...
- YAML-Files für die Konfiguration
- CLI-Programm „docker-compose“ mit vielen Kommandos *oder*
- Kommando „compose“ im Docker Daemon

Docker Compose

- Nutzung in drei wesentlichen Schritten:
 - Festlegung der Umgebung der Anwendung in Dockerfiles
 - Konfiguration der Services der Anwendung in docker-compose.yml
 - `docker compose up` zum Start der Anwendung (oder `docker-compose up` mit dem extra Binary)

... Bildquellen

- RedHat
- IDG
- Docker
- Earthly
- Aquasec
- dev.to