

Manubrain Plattform

Inhaltsverzeichnis

// Anlegen der Ordnerstruktur

// Vergabe der Ports

// Starten der Plattform

//

//

//

//






1. Anlegen der Ordnerstruktur

- Zum Aufsetzen der Plattform muss im Vorfeld eine Ordnerstruktur erstellt werden, welche im Repository vorkonfiguriert ist.
- Die Ordner können allerdings auch individuell verteilt werden





Docker-Ordner

- Im Ordner „docker“ befindet sich die docker compose yaml Datei, welche unter Docker Compose ausgeführt wird
- Des Weiteren befindet sich ein Ordner Node_Red mit dem Dockerfile für Node Red im Ordner, sowie eine Ordner-Vorlage zur Installation

Standardisierte Ordnerstruktur

Name	Änderungsdatum	Typ
 docker	04.11.2022 11:19	Dateiordner
 grafana	04.11.2022 11:19	Dateiordner
 influxdb	04.11.2022 11:19	Dateiordner
 jupyter	04.11.2022 11:19	Dateiordner
 nodered	04.11.2022 11:19	Dateiordner

docker

Name	Änderungsdatum	Typ
 Node_Red	04.11.2022 11:19	Dateiordner
 Ordner	04.11.2022 11:19	Dateiordner
 Python	04.11.2022 11:19	Dateiordner
 compose	03.11.2022 17:48	YML-Datei

1. Anlegen der Ordnerstruktur

- Sollte ein individueller Pfad gewünscht werden kann dies in der Docker Compose Datei eingestellt werden
- Der relative oder absolute Pfad des Ordners muss hierfür jeweils unter dem Begriff source mit Anführungszeichen eingefügt werden
 - Beispiel: 'C:\Users\Admin\Desktop\nodered'
- Wichtig: Der referenzierte Ordner muss bereits auf dem System existieren und wird nicht automatisch angelegt.

```
compose.yml
version: '3'
services:
  nodered:
    build: 'Node_Red\'
    image: custom-node-red
    container_name: nodered
    volumes:
      - type: bind
        source: '..\nodered'
        target: /data
    restart: on-failure:10
    environment:
      - TZ=Europe/Amsterdam
    ports:
      - "1880:1880"
  timeseriesdb:
    image: influxdb:1.8
    environment:
      - DOCKER_INFLUXDB_INIT_USERNAME = root
      - DOCKER_INFLUXDB_INIT_PASSWORD = geheim
    restart: on-failure:10
    volumes:
      - type: bind
        source: '..\influxdb'
        target: /var/lib/influxdb
    ports:
      - "8000:8086"
```

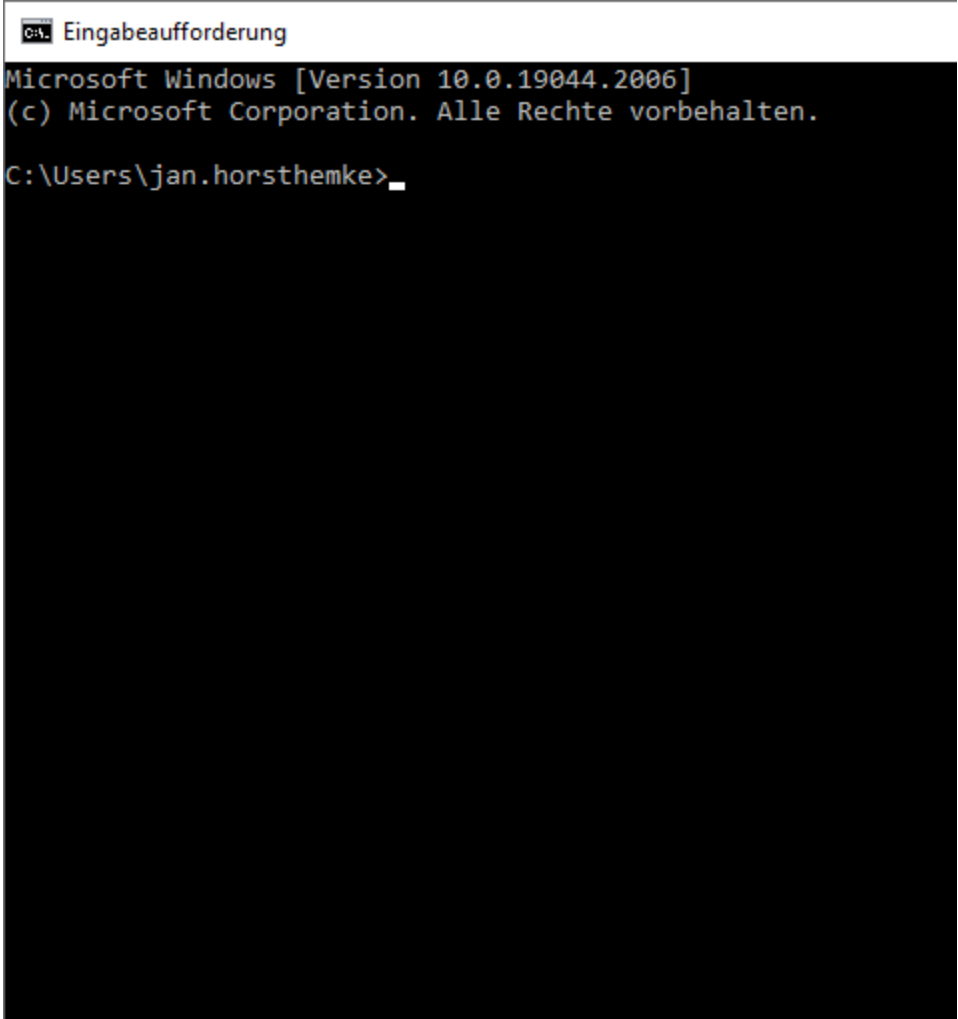
1.2 Vergabe der Ports

- Im zweiten Schritt sollte die Portbelegung überprüft werden
- Unter Windows kann dies mit **cmd** durch folgenden Befehl für die Standardports erfolgen.
 - `netstat -ano | find "1880,8000,3000,2010,8200"`
- Die Ports können in der Docker compose Datei angepasst werden. Hierbei gilt der erste Port ist der Port im Hostsystem, der zweite Port der im Docker Netzwerk
 - Beispiel: "Host:Docker" bzw. "2000:1880"

```
compose.yml
version: '3'
services:
  nodered:
    build: 'Node_Red\'
    image: custom-node-red
    container_name: nodered
    volumes:
      - type: bind
        source: '..\nodered'
        target: /data
    restart: on-failure:10
    environment:
      - TZ=Europe/Amsterdam
    ports:
      - "1880:1880"
  timeseriesdb:
    image: influxdb:1.8
    environment:
      - DOCKER_INFLUXDB_INIT_USERNAME :
      - DOCKER_INFLUXDB_INIT_PASSWORD :
    restart: on-failure:10
    volumes:
      - type: bind
        source: '..\influxdb'
        target: /var/lib/influxdb
    ports:
      - "8000:8086"
```

1.3 Starten der Plattform

- Zum Implementieren der Plattform muss Docker Desktop unter Windows aktiv sein
- Anschließend kann die Plattform mittels cmd implementiert werden
- Hierfür muss als Erstes in den Plattform Ordner gewechselt werden und dort die Docker Compose Datei ausgeführt werden
 - Mit dem Befehl **cd** kann der Ordner gewechselt werden
 - Mit dem Befehl **dir** der Ordnerinhalt angezeigt werden



```
C:\Users\jan.horsthemke>
```

























1.3 Starten der Plattform

- Der Start der Plattform wird durch den Befehl `docker compose up` eingeleitet
- Hierdurch werden die Images heruntergeladen, falls diese noch nicht vorhanden sind
- Des Weiteren werden die Container erstellt und gestartet

Containers [Give Feedback](#)

A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. [Learn more](#)

Showing 6 items

	NAME	IMAGE	STATUS	PORT(S)	STARTED	ACTIONS
<input type="checkbox"/>	 docker 5 containers	-	Running (5/5)	-		  
<input type="checkbox"/>	 jupyter-1 6d0323f352d4	jupyter/datascience-notebook	Running	8200	4 seconds ago	  
<input type="checkbox"/>	 nodered af214488419f	custom-node-red:latest	Running	1880	4 seconds ago	  
<input type="checkbox"/>	 timeseriesdb-1 25febacd6a4e	influxdb:1.8	Running	8000	5 seconds ago	  
<input type="checkbox"/>	 grafana-1 6d4652703738	grafana/grafana:latest	Running	3000	4 seconds ago	  
<input type="checkbox"/>	 chronograf-1 f5a3ac6f11cd	chronograf:latest	Running	2010	5 seconds ago	  

```
Eingabeaufforderung - docker compose up

Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\Workstation>cd Desktop/Manubrain/docker

C:\Users\Workstation\Desktop\Manubrain\docker>docker compose up
[+] Running 5/0
 - Container nodered                Running
 - Container docker-grafana-1       Created
 - Container docker-jupyter-1       Created
 - Container docker-timeseriesdb-1 Created
 - Container docker-chronograf-1    Created
Attaching to docker-chronograf-1, docker-grafana-1, docker-jupyter-1,
docker-timeseriesdb-1 | ts=2022-11-04T14:56:01.010554Z lvl=info msg=
docker-timeseriesdb-1 | ts=2022-11-04T14:56:01.010604Z lvl=info msg=
docker-timeseriesdb-1 | ts=2022-11-04T14:56:01.127115Z lvl=info msg=
docker-timeseriesdb-1 | ts=2022-11-04T14:56:01.128452Z lvl=info msg=
roughput_bytes_per_second=50331648 throughput_bytes_per_second_burst=
docker-timeseriesdb-1 | ts=2022-11-04T14:56:01.128494Z lvl=info msg=
_open op_event=start
docker-grafana-1      | logger=settings t=2022-11-04T14:56:01.313056
ed=2022-10-18T08:51:02Z
```

1.4 Einrichten der Zeitreihendatenbank

- Die Einrichtung der Zeitreihendatenbank erfolgt über das Terminal
 - Hierfür wird der Container im terminal geöffnet

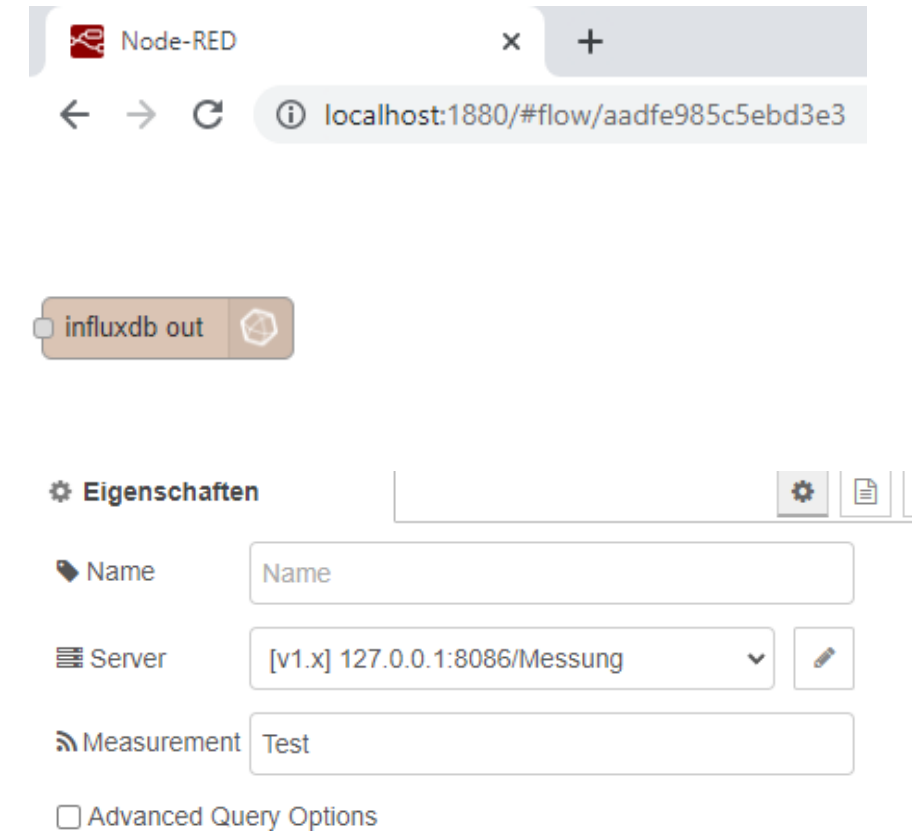
	Container Name	Image	Status	Ports	Created	Actions
<input type="checkbox"/>	timeseriesdb-1 25febacd6a4e	influxdb:1.8	Running	8000	3 minutes ago	View details Open in terminal Pause Restart Open with browser
<input type="checkbox"/>	grafana-1 6d4652703738	grafana/grafana:latest	Running	3000	3 minutes ago	
<input type="checkbox"/>	chronograf-1 f5a3ac6f11cd	chronograf:latest	Running	2010	3 minutes ago	

- Durch den Befehl „influx“ wird die Datenbank geöffnet
- Anschließend wird eine Datenbank für die Messwerte erstellt
 - CREATE DATABASE Messung
 - SHOW DATABASES

```
Open in external terminal
# influx
Connected to http://localhost:8086 version 1.8.10
InfluxDB shell version: 1.8.10
> SHOW DATABASES
name: databases
name
----
_internal
> CREATE DATABASE Messung
> SHOW DATABASES
name: databases
name
----
_internal
Messung
>
```


1.4 Kommunktion zwischen Node Red und Datenbank

- Zur Einrichtung der Kommunktation zwischen NodeRed und der Datenbank wird als Erstes NodeRed im Explorer geöffnet
 - Hierfür wird in der Adressleiste der Localhost mit dem Port aufgerufen
 - Beispiel: localhost:1880
- Anschließend wird ein „influxdb out“ Element in den Arbeitsbereich gezogen
- Durch einen Doppelklick auf das Element öffnet sich das Konfigurationsfenster
- Als Measurement kann ein beliebiger Name festgelegt werden: Test
- Die Servereinstellungen können durch Klicken auf den Bleistift definiert werden



1.4 Kommunktion zwischen Node Red und Datenbank

- Der Host besitzt die definierte IP-Adresse : 172.16.238.3
- Als Port ist der definierte Port des Docker Netzwerkes zu verwenden: 8086
- Datenbank ist die definierte Datenbank: Messung
- Benutzername kann der Docker Compose Datei entnommen werden: root
- Passwort kann ebenfalls der Docker Compose Datei entnommen werden: geheim

The screenshot shows the 'Eigenschaften' (Properties) dialog box for a database connection. The dialog has a title bar with a gear icon and a document icon. The main content area contains the following fields:

- Name:** A text input field with the placeholder text 'Name'.
- Version:** A dropdown menu with '1.x' selected.
- Host:** A text input field with '172.16.238.3' entered.
- Port:** A text input field with '8086' entered.
- Database:** A text input field with 'Messung' entered.
- Benutzername:** A text input field with 'root' entered.
- Passwort:** A text input field with masked characters (dots) entered.
- Enable secure (SSL/TLS) connection:** A checkbox that is currently unchecked.

1.5 Kommunikation Datenbank mit Python

- Die Kommunikation zwischen Datenbank und Python kann über das influxdb Paket erfolgen.
- Für den externen Zugriff vom Hostsystem aus ist der Port 8000 zu wählen und als host localhost
- Für den Zugriff innerhalb der Dockerumgebung aus Jupyter Notebook ist der Port 8086 zu wählen und als Host 172.16.238.3

Created on Sat Oct 22 17:01:02 2022

@author: Workstation
"""

```
import influxdb
import numpy as np
import matplotlib.pyplot as plt
database = 'Messung'
```

```
client = influxdb.InfluxDBClient(host='localhost',username='root',
                                password='geheim', port=8000) # us
```

```
print(client.get_list_database())
```

```
client.switch_database(database)
```

```
data = client.query('SELECT * FROM Test')
```

```
data = data.raw
```

```
num = data['series'][0]['values']
```

```
humidity = np.array([num[i][1] for i in range(0,len(num))])
```

```
temp = np.array([num[i][2] for i in range(0,len(num))])
```