



Development Fundamental

Dokumentation

Marlen Müller

22. Oktober 2025 ZLI, Zürich

Inhaltsverzeichnis

Einrichtung der Entwicklungsumgebung.....	3
Erstellung der README.md	4
Was ist README.md?.....	4
Wie benutzt man README.md?	5
Cheat Sheet	6
Verwendung von Git (Commit, Push).....	7

Klonen des Repositories

Um einen Repository zu klonen muss man das Terminal auf Git Bash öffnen. Dort muss man den **Befehl: git**

```
mmarl@LaptopMarlen MINGW64 ~  
$ cd ~/projects  
  
mmarl@LaptopMarlen MINGW64 ~/projects  
$ git clone https://github.com/ICT-BLJ/docker-nodejs-sample  
fatal: destination path 'docker-nodejs-sample' already exists and is not an empty directory.  
  
mmarl@LaptopMarlen MINGW64 ~/projects  
$
```

clone *LinkVonDatei* eingeben und auf Enter drücken. Davor muss man noch den Ordner anwählen, welchen einen Ordner anwählt, indem man die Datei rein klonen muss. Dies kann man mit dem Befehl: **cd ~/Speicher Ort**. Auf dem Bild kann man sehen, dass es nicht funktioniert hat. Der Grund dafür ist, dass ich den Vorgang bereits gemacht habe und die Datei bereits auf meinem Computer existiert.

Einrichtung der Entwicklungsumgebung

Verwenden Sie Docker Init Erstellen von Assets manuell

Innerhalb der `docker-nodejs-sample` Verzeichnis, ausführen die `docker init` Befehl in einem Terminal. `docker init` bietet einige Standard Konfiguration, aber Sie müssen ein paar Fragen zu Ihrer Anwendung beantworten. Das folgende Beispiel finden Sie unter Beantwortung der Aufforderungen von `docker init` und Nutzung Die gleichen Antworten für Ihre Aufforderungen.

```
$ docker init  
Welcome to the Docker Init CLI!  
  
This utility will walk you through creating the following files with sensible default  
- .dockerignore  
- Dockerfile  
- compose.yaml  
- README.Docker.md  
  
Let's get started!  
  
? What application platform does your project use? Node  
? What version of Node do you want to use? 18.0.0  
? Which package manager do you want to use? npm  
? What command do you want to use to start the app? node src/index.js  
? What port does your server listen on? 3000
```

Bildquelle:

<https://docs.docker.com/guides/nodejs/containerize/>

Auf der Seite docs.docker.com hat man eine Anleitung, wie man die **Docker-Assets initialisiert** und wie man **deren Anwendung ebenfalls ausführen** kann. Auf dem ersten Bild kann man mehr oder weniger erkennen, wie die Anleitung für die Terminal Version aussieht.

Für mich persönlich hat es sehr geholfen und es war klar und unkompliziert erklärt. Ausserdem hat es im Terminal auch etwa so ausgesehen. Als

erstes musste man wie beschrieben **docker init** schreiben, damit die Fragen kommen. Anhand der Pfeile und dem Tabulator, kann man die verschiedenen Einstellungen ändern. Später kann man dies auch im Code machen. Nun müssen wir es nur die Ausführung der Anwendung möglich machen. Wenn man ein wenig weiter unten scrollt, kommt man auch zu dem Abschnitt.

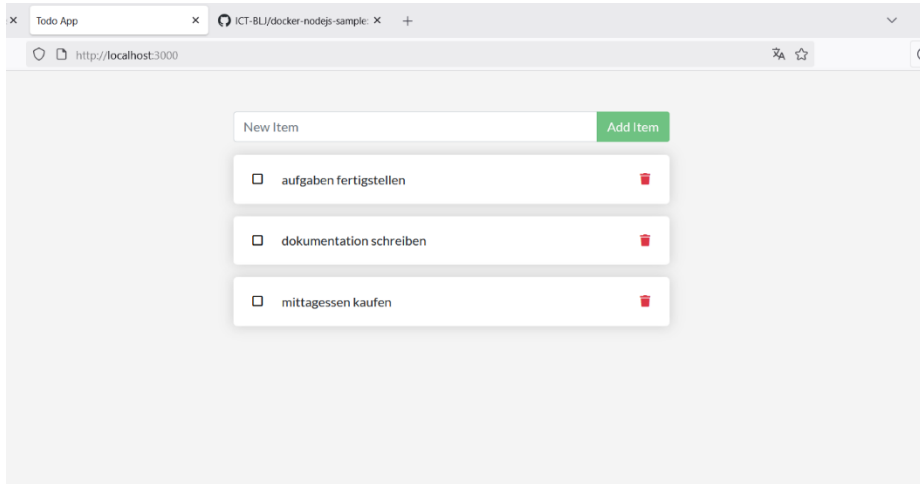
Führen Sie die Anwendung aus

Innerhalb der `docker-nodejs-sample` Verzeichnis, führen Sie den folgenden Befehl in einem Terminal.

```
$ docker compose up --build
```

Dies muss man ebenfalls ins Terminal eingeben um anschliessend den Container zu erstellen. Sobald der Container erstellt wurde, kann man den angegebenen folgenden

Link: <http://localhost:3000/> anklicken. Daraufhin wird ein neuer Tab geöffnet. Hat man alles richtig gemacht und der Container läuft, erscheint ein Eingabefeld für eine **ToDo-Liste**. Hat man aber einen Fehler kann die Seite nicht geladen werden. Im Falle eines Fehlers, kann man den, wie bereits auch schon erwähnt, beheben.



Auch wenn man die Seite neu lädt, bleiben die folgenden Punkte gespeichert. Jedoch werden sie nicht gespeichert, wenn man den Container deaktiviert. Beim Einschalten des Containers wird alles wieder zurückgesetzt. Ausserdem ist die Seite funktionsunfähig, wenn der Container nicht aktiviert ist.

Erstellung der README.md

Was ist README.md?

README.md ist eine **Textdatei**, die ein Projekt beschreiben. Meistens betrifft dies GitHub. Darauf kann man herausfinden, was das Projekt macht und wie man es benutzt, wie auch wer dieses Projekt erstellt hat. Kurzgesagt ist es eine Art von Dokumentation auf GitHub. Das .md am Schluss steht für Markdown und formatiert den Text. Dazu habe ich noch ein Cheat Sheet, welches auch in dieser Dokumentation erhalten ist.

```
### **Aufgabenstellung:**  
  
1. **Repository clonen:**  
- Zuerst muss man den Speicherort wählen. In meinem Fall wähle es der Ordner **Projects**. Im Terminal schreibt man dann **cd ~/projects** um den Speicherort anzuwählen.  
- Danach muss man das Repository mit **git clone** *URL von Repository* auf den Computer clonen.  
  
2. **Installation der notwendigen Pakete:**  
- Pakete werden automatisch in Docfiles installiert.  
- man braucht folgendes:  
  - Docker Engine  
  - Docker CLI  
  - Docker Compose  
  - WSL2 Integration  
- sichergehen ob alles stimmt:  
  - docker --version  
  - docker compose version  
  - java -version  
  - mvn -v  
  - git --version  
  
3. **Docker-Konfiguration und -Installation:**  
- Docker herunterladen  
- PC neustarten  
- Docker testen mit Terminal [docker run hello-world]  
  
4. **Starten der Applikation in einem Docker-Container:**  
- Docker auf Desktop öffnen.  
- Warten bis wann es grün (aktiviert) ist.  
- Prüfen ob Dockerfile vorhanden ist.  
- Docker image in Terminal öffnen  
- Container starten  
- Container stoppen und evtl. löschen
```

Wie benutzt man README.md?

README.md funktioniert ganz simpel. Man schreibt eine Dokumentation, in meinem Fall jetzt auf **Visual Studio Code**. Dabei benutzt man wie bereits erwähnt benutzt man Markdown, um den Text zu formatieren.

Wie man es hier im Bild sehen kann, ist die Darstellung sehr einfach. Mit dem Markdown habe

ich hauptsächlich Wörter fett ****XX**** oder kursiv **XX** formatiert. Dazu wird im Cheat Sheet mehr zusammengefasst.

Cheat Sheet

~ Markdown

Überschriften:

#	Überschrift 1
##	Überschrift 2
###	Überschrift 3
####	Überschrift 4
#####	Überschrift 5
#####	Überschrift 6

Textformatierung:

Kursiv	<i>kursiv</i>
Fett	Fett
Fett und Kursiv	<i>Fett und Kursiv</i>
~~durchgestrichen~~	durchgestrichen

Liste

Unsortiert:

- Punkt 1
- Punkt 2
 - Unterpunkt 1
 - Unterpunkt 2

Sortiert:

1. Punkt
2. Punkt
 1. Unterpunkt
 2. Unterpunkt

Link:

[Text](https://XXX.ch)

Bild:

! [Alternativtext](https://XXX.ch/bild.jpg)

Zitat:

> Zitat

>>verschachteltes Zitat

Aufgabenliste:

- [x] erledigt
- [] Noch nicht erledigt

Tabelle:

Trennlinie = ---

Spalte 1	Spalte 2	Spalte 3
-----	-----	-----
Zeile 1	Daten	Daten
Zeile 2	Daten	Daten

HTML in Markdown:

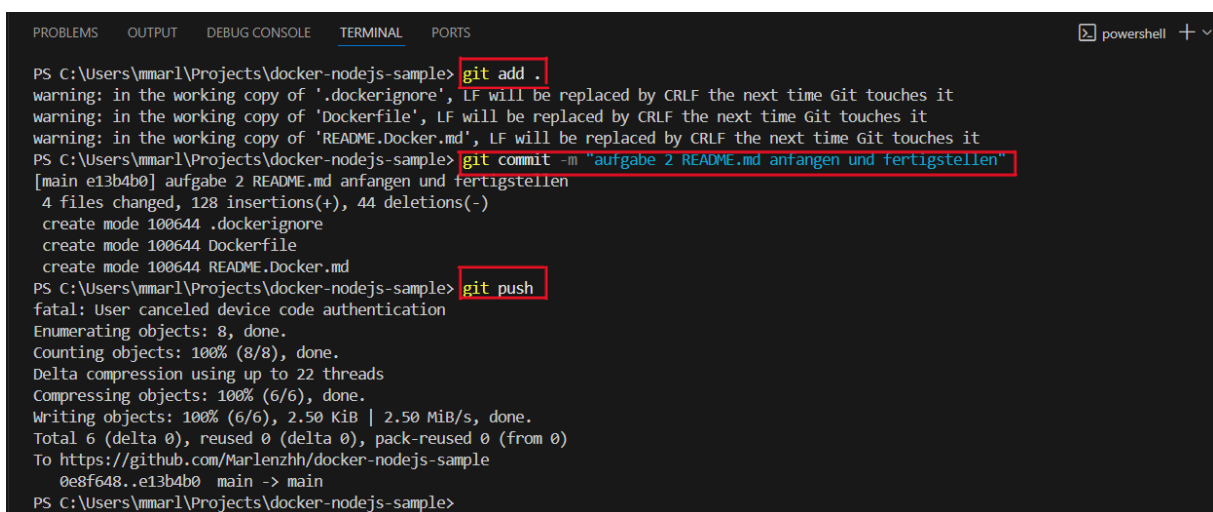
Fetter Text und
<i>Kursiver Text</i>

Verwendung von Git (Commit, Push)

Im Wochenbericht haben wir gelernt, wie man Dateien auf GitHub pusht.

1. **git status:** => Status überprüfen
2. **git add . :** => geänderte Dateien zur Commit Liste hinzufügen
3. **git commit -m «Kommentar»:** => Änderung commiten mit einer passenden Nachricht
4. **git push:** => Datei auf GitHub pushen

Wenn man die Dateien einzeln commiten will, benutzt man `git add datei1.java datei2.py`.



```
PS C:\Users\mmarl\Projects\docker-nodejs-sample> git add .
warning: in the working copy of '.dockerignore', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Dockerfile', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'README.Docker.md', LF will be replaced by CRLF the next time Git touches it
PS C:\Users\mmarl\Projects\docker-nodejs-sample> git commit -m "aufgabe 2 README.md anfangen und fertigstellen"
[main e13b4b0] aufgabe 2 README.md anfangen und fertigstellen
4 files changed, 128 insertions(+), 44 deletions(-)
create mode 100644 .dockerignore
create mode 100644 Dockerfile
create mode 100644 README.Docker.md
PS C:\Users\mmarl\Projects\docker-nodejs-sample> git push
fatal: User canceled device code authentication
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 22 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 2.50 KiB | 2.50 MiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Marlenzhhd/docker-nodejs-sample
0e8f648..e13b4b0 main -> main
PS C:\Users\mmarl\Projects\docker-nodejs-sample>
```