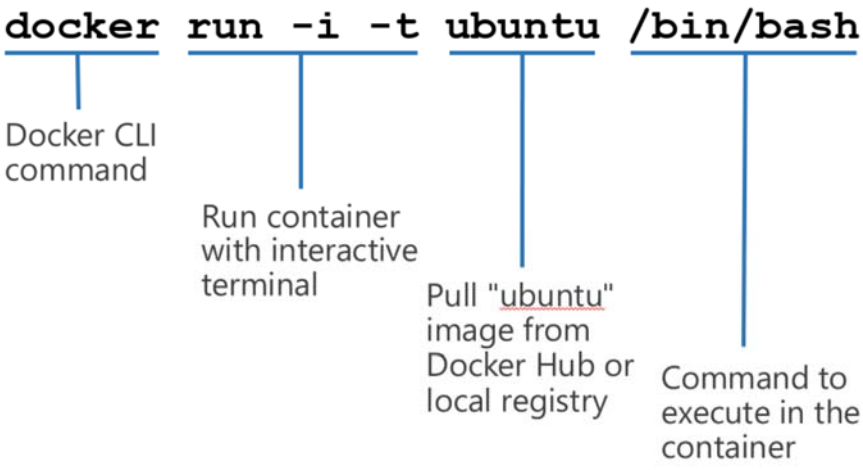
Docker CMD

Beispiel:

Mount c:\folder1 von Server1 auf C:\ContainerFolder in Containter1

```
Docker run –it –v c:\folder1: C:\ContainerFolder Container1
```



Übersicht der wichtigsten Befehle

`docker build -t "imagename"`	buildet ein Docker Image aus einer Datei namens „`Dockerfile`“ (ohne Dateiendung), das Image bekommt den Namen der mit `-t` als Parameter übergeben wurde.
`docker run -d -p 8080:80 "imagename"`	erstellt und startet einen Container basierend auf den Images, `-p` mappt den port `8080` (des Hostsystems) im Container auf den port `80`. `-d` startet den Container im detached mode, der Container wird im Hintergrund gestartet und die console wird nicht von der Ausgabe des Docker Containers blockiert.
`docker container ls -a`	zeigt alle laufenden Container an, der Parameter `-a` zeigt auch alle derzeit nicht laufenden an. Dieser Command ersetzt `docker ps` welcher aber auch weiterhin noch funktioniert.
`docker images`	zeigt alle gebauten oder heruntergeladenen Images an
`docker search microsoft`	zeigt alle Images von Microsoft
`docker image rm IMAGE`	entfernt das angegebene Image
`docker start CONTAINER`	startet den angegeben Container
`docker logs -f (CONTAINER | SERVICENAME)`	zeigt den log output des Containers `-f :` mit dem Parameter `-f` wird der folgende log output live ausgegeben.
`docker attach (CONTAINER | SERVICENAME)`	hängt den Standard Input und Output auf die aktuelle Konsole. Man kann wie mit ssh eine Verbindung in den Container aufbauen. Achtung: dies ist keine richtige ssh Verbindung. Das merkt man insbesondere, wenn man die Verbindung wieder auflösen will. `CTRL -c` oder `exit` stoppt auch den Container. Die Verbindung kann mit `CTRL-p CTRL-q` gelöst werden.
`docker exec -it CONTAINER /bin/sh`	mit `exec` kann ein Befehl innerhalb des Containers ausgeführt werden. Wird der Befehl `/bin/sh` (wenn eine bash vorhanden ist kann auch `/bin/bash` verwendet werden) angegeben, ist dies eine alternative Variante um auf die shell im Container zuzugreifen. Wichtig ist, dass die Option `-it` angegeben wird. Der Vorteil dieser Variante gegenüber `docker attach` ist, dass die shell wie gewohnt mit `exit` verlassen werden kann und der Container dabei nicht gestoppt wird.
`docker stop CONTAINER`	stoppt den angegeben Container. Der Container wird heruntergefahren.
`docker kill CONTAINER`	der Container wird mit einem Kill signal gestoppt.
`docker rm CONTAINER`	entfernt den Docker Container (nur den Container nicht das Image)
`docker rm $(docker ps -aq)`	löscht alle Container (`docker ps -aq` der parameter `-q` beschränkt die Ausgabe auf die Ids. Der Befehl gibt also die Ids aller Container zurück und mit `$` wird über all diese Ids iteriert und `docker rm` ausgeführt)
`docker image rm $(docker images -q)`	löscht alle Images aus dem lokalen Repository (wie oben beschrieben nur mit `docker images -q`)