|  |  |  |
| --- | --- | --- |
| **Images** | | |
| docker pull <image|repository>[:<tag>] | Fertiges Docker Image herunterladen |  |
| docker push <image|repository>[:<tag>] | Schiebt ein Image auf ein Repository |  |
| docker build <directory> -t <image>[:<tag>] | Erzeugt Image aus Definition in Dockerfile im gegeb. dir | -t: tag value |
| docker images -a -q | Alle Images anzeigen | -a: all / -q: quiet |
| docker rmi -f <image> [<image>...] | Löscht Images | -f: force |
| docker run --name <name> -d -p source:target -t -w <workdir> <image> | Startet das gebaute/heruntergeladene Image (erzeugt neue Instanz) [workdir in Windows mit extra / prefixen] | -t: tty / -d: detach / -p: publish ports /  --name: container name / -w: workdir |
| **Container** | | |
| docker ps -a -q | Alle Container anzeigen | -a: all / -q: quiet |
| docker rm -f <container> [<container>...] | Löscht Container | -f: force |
| docker stop <container> [<container>...] | Fährt Container herunter |  |
| docker start <container> [<container>...] | Startet Container |  |
| docker restart <container> [<container>...] | Stop & Start |  |
| docker logs -f <container> | Logs eines Containers anzeigen | -f: follow |
| docker exec -it -u <user> <container> bash | In Container einloggen (geht nur mit cmd.exe) | -i: interactive / -t: tty / -u: run as user |
| docker cp <src> <dest> | Dateien vom/zum Container kopieren |  |
| **Allgemein** | | |
| docker info | U.a. Uptime anzeigen |  |
| docker stats | Docker status anzeigen (live stream) |  |
| docker <command> --help | Hilfe zu einem Befehl |  |
| **Compose** | | |
| docker-compose | Startet einen Docker-Verbund |  |
| docker-compose pull <name> | Holt das Image <name> wie in docker-compose.yml def. |  |
| **Sonstiges** | | |
| $(docker ps -a -q) | Parameter, wann immer Listen angegeben werden können (rm, rmi, stop, start, restart) |  |

## Get all keys available for docker ps --format

docker ps --format='{{json .}}' creates a stream:

{ ... }

{ ... }

{ ... }

# read everything into memory `slurp`, then just process the first item and finally get its keys

docker ps --format='{{json .}}' | jq --slurp '.[0] | keys'

# get the first element from the stream, then get its keys

docker ps --format='{{json .}}' | jq --null-input 'range(1) as $i | input | keys'