docker image **pull *nome*:*tag***	Scarica l'immagine specificata
docker image **rm *nome*:*tag***	Rimuove l'immagine specificata
docker image **ls**	Stampa a video la lista delle immagini scaricate
docker **search *ricerca***	Cerca le immagini su hub.docker.com
docker container **run *immagine***	Avvia un container con l'immagine specificata, accetta altri attributi
docker container run **-it *immagine* /bin/bash**	Avvia un container con una shell di tipo bash
docker container run **-d** -it -***immagine*** /bin/bash	Avvia in background un container
docker run **-name=*nome*** alpine **sleep 10**	Avvia un container con il nome specificato e il comando sleep
CTRL+P CTRL+Q	Combinazione per uscire dalla shell mantenendo il container attivo
docker ps	Stampa a video la lista dei container attivi
docker stop ***container***	Blocca l'esecuzione di un container
docker container ls **-a**	Visualizza lo stato dei container spenti
ps -elf	Dentro ad un container, permette di stampare i processi attivi
docker attach ***container***	Torna nella bash di un container
docker container start ***container***	Avvia un container stoppato
docker container prune	Elimina tutti i container
docker run **--restart always|unless-stopped|on-failed**	Specifica la politica di riavvio (SEMPRE, MAI, IN CASO DI ECCEZ)
docker **volume create *nome***	Genera un volume
docker volume **ls**	Mostra i volumi
docker volume **inspect *nome***	Ispeziona il volume
docker volume **rm *nome***	Elimina un volume
docker volume **inspect *nome***	Visualizza il volume
docker run **-v *nome*:/*test*** …	Avvia un container e collega il volume specificato alla cartella test
docker run **-v *path-utente*:*cartella-conainer*** …	Mapping cartelle
docker run **-p *portavm*:*portacontainer*** …	Port Forwarding

DOCKERFILE

Implementazione:

docker **image build --tag local:*dockerfile* .** — Crea un'immagine dal dockerfile specificato (il p.to infondo specifica di considerare la cartella corrente come path del dockerfile) (--tag da un nome all'immagine)

Esecuzione:

docker **run *nomeImmagine***

Struttura dockerfile:

FROM *immagine*
COMANDO attributo1, attributo2, …
 # Questo è un commento

Comandi:

RUN *comando*	Esegue il comando LINUX specificato (eseguito prima di preparare l'immagine)
CMD ["*comando*", "*attributi*"]	Esegue il comando specificato
COPY "*sorgente*", "*destinazione*"	Copia, solitamente un file script.sh
ADD *sorgente destinazione*	Permette di copiare da altre cartelle ma anche scaricare dal web e decomprimere
LABEL *attributo*="*valore*"	Indica informazioni interne al dockerfile (creatore, data creazione, …)
WORKDIR *cartella*	Imposta la cartella di lavoro (alternativa a cd)
ENV *nome*="*valore*"	Genera una variabile d'ambiente
EXPOSE *porta*	Specifica quale porta esporre all'esterno del container (usare comunque -p …)
USER *utente*	Imposta l'utente che eseguirà i comandi
VOLUME ["cartella host"]	Crea una cartella nel container e la collega ad una cartella dell'host
ARG *nome=valore default*	Variabile di "passaggio", per passare informazioni alla generazione dell'immagine
ENTRYPOINT ["*comando*"]	Come CMD ma senza pass parametri, combinabile con CMD per param opzionali

Passaggio parametri:

Il comando CMD "supporta" il passaggio parametri

dockerfile:

FROM alpine
CMD ["echo", "ciao mondo"]

terminale:

docker build --tag testparametri .
docker run testparametri echo ciao mamma

Comandi utili:

echo → stampa a video **sh** → esegue lo script.sh **touch** → genera file **cat** → stampa contenuto file
useradd → crea un utente