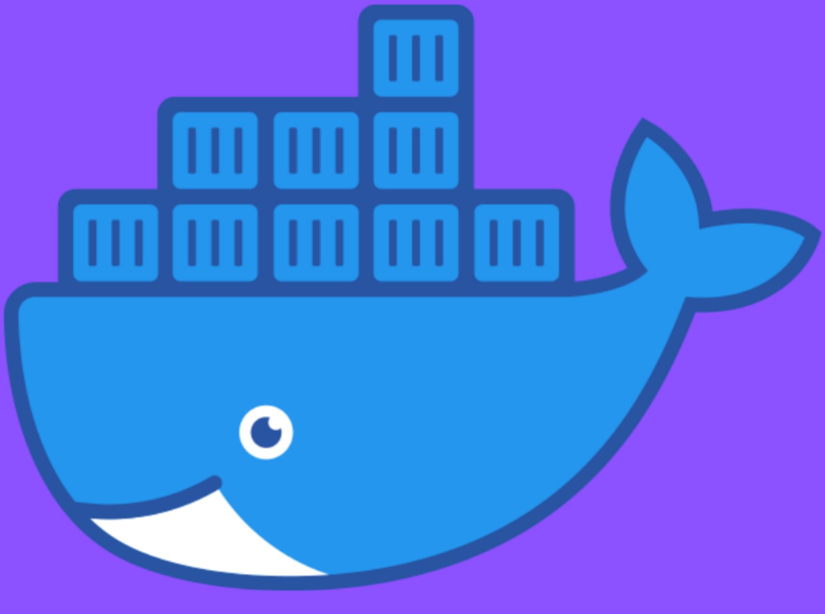
DOCKER

THE ULTIMATE CHEAT SHEET

"The easiest way to understand Docker commands"



WWW.THECHIEF.IO

The Ultimate Docker Cheat Sheet

Installation

Linux

For more information, see here

```
curl -sSL https://get.docker.com/ | sh
```

Mac

For more information, see here

Use this link to download the dmg.

```
https://download.docker.com/mac/stable/Docker.dmg
```

Open the downloaded file and follow the installation instructions.

Windows

For more information, see here

Use the msi installer:

```
https://download.docker.com/win/stable/InstallDocker.msi
```

Open the downloaded file and follow the installation instructions.

Docker Registries & Repositories

Login to a Registry

```
docker login
```

```
docker login localhost:8080
```

Logout from a Registry.

```
docker logout
```

```
docker logout localhost:8080
```

Searching an Image

```
docker search nginx
```

```
docker search --filter stars=3 --no-trunc nginx
```

Pulling an Image

```
docker image pull nginx
```

```
docker image pull eon01/nginx localhost:5000/myadmin/nginx
```

Pushing an Image

```
docker image push eon01/nginx
```

```
docker image push eon01/nginx localhost:5000/myadmin/nginx
```

Running Containers

Create and Run a Simple Container

> -Start an ubuntu:latest image
>
> - Bind the port `80` from the **CONTAINER** to port `3000` on the **HOST**
> - Mount the current directory to `/data` on the CONTAINER
> - Note: on **windows** you have to change `-v ${PWD}:/data` to `-v "C:\Data":/data`

```
docker container run --name infinite -it -p 3000:80 -v ${PWD}:/data
ubuntu:latest
```

Creating a Container

```
docker container create -t -i eon01/infinite --name infinite
```

Running a Container

```
docker container run -it --name infinite -d eon01/infinite
```

Renaming a Container

```
docker container rename infinite infinity
```

Removing a Container

```
docker container rm infinite
```

A container can be removed only after stopping it using `docker stop` command. To avoid this, add the `--rm` flag while running the container.

Updating a Container

```
docker container update --cpu-shares 512 -m 300M infinite
```

Running a command within a running container

```
docker exec -it infinite sh
```

In the example above, `bash` can replace `sh` as an alternative (if the above is giving an error).

Starting & Stopping Containers

Starting

```
docker container start nginx
```

Stopping

```
docker container stop nginx
```

Restarting

```
docker container restart nginx
```

Pausing

```
docker container pause nginx

```

Unpausing

```
docker container unpause nginx
```

Blocking a Container

```
docker container wait nginx
```

Sending a SIGKILL

```
docker container kill nginx
```

Sending another signal

```
docker container kill -s HUP nginx
```

Connecting to an Existing Container

```
docker container attach nginx
```

Getting Information about Containers

From Running Containers

Shortest way:

```
docker ps
```

Alternative:

```
docker container ls
```

From All containers.

```
docker ps -a
```

```
docker container ls -a
```

Container Logs

```
docker logs infinite
```

'tail -f' Containers' Logs

```
docker container logs infinite -f
```

Inspecting Containers

```
docker container inspect infinite
```

```
docker container inspect --format '{{ .NetworkSettings.IPAddress }}' $(docker ps -q)
```

Containers Events

```
docker system events infinite
```

Public Ports

```
docker container port infinite
```

Running Processes

```
docker container top infinite
```

Container Resource Usage

```
docker container stats infinite
```

Inspecting changes to files or directories on a container’s filesystem

```
docker container diff infinite
```

Managing Images

Listing Images

```
docker image ls
```

Building Images

From a Dockerfile in the Current Directory

```
docker build .
```

From a Remote GIT Repository

```
docker build github.com/creack/docker-firefox
```

Instead of Specifying a Context, You Can Pass a Single Dockerfile in the URL or Pipe the File in via STDIN

```
docker build - < Dockerfile
```

```
docker build - < context.tar.gz
```

Building and Tagging

```
docker build -t eon/infinite .
```

Building a Dockerfile while Specifying the Build Context

```
docker build -f myOtherDockerfile .
```

Building from a Remote Dockerfile URI

```
curl example.com/remote/Dockerfile | docker build -f - .
```

Removing an Image

```
docker image rm nginx
```

Loading a Tarred Repository from a File or the Standard Input Stream

```
docker image load < ubuntu.tar.gz
```

```
docker image load --input ubuntu.tar
```

Saving an Image to a Tar Archive

```
docker image save busybox > ubuntu.tar
```

Showing the History of an Image

```
docker image history
```

Creating an Image From a Container

```
docker container commit nginx
```

Tagging an Image

```
docker image tag nginx eon01/nginx
```

Pushing an Image

```
docker image push eon01/nginx
```

Networking

Creating Networks

Creating an Overlay Network

```
docker network create -d overlay MyOverlayNetwork
```

Creating a Bridge Network

```
docker network create -d bridge MyBridgeNetwork
```

Creating a Customized Overlay Network

```
docker network create -d overlay \
  --subnet=192.168.0.0/16 \
  --subnet=192.170.0.0/16 \
  --gateway=192.168.0.100 \
  --gateway=192.170.0.100 \
  --ip-range=192.168.1.0/24 \
  --aux-address="my-router=192.168.1.5" --aux-address="my-switch=192.168.1.6" \
  --aux-address="my-printer=192.170.1.5" --aux-address="my-nas=192.170.1.6" \
  MyOverlayNetwork
```

Removing a Network

```
docker network rm MyOverlayNetwork
```

Listing Networks

```
docker network ls
```

Getting Information About a Network

```
docker network inspect MyOverlayNetwork
```

Connecting a Running Container to a Network

```
docker network connect MyOverlayNetwork nginx
```

Connecting a Container to a Network When it Starts

```
docker container run -it -d --network=MyOverlayNetwork nginx
```

Disconnecting a Container from a Network

```
docker network disconnect MyOverlayNetwork nginx
```

Exposing Ports

Using Dockerfile, you can expose a port on the container using:

```
EXPOSE <port_number>
```

You can also map the container port to a host port using:

```
docker run -p $HOST_PORT:$CONTAINER_PORT --name <container_name> -t <image>
```

e.g.

```
docker run -p $HOST_PORT:$CONTAINER_PORT --name infinite -t infinite
```

Security

Guidelines for Building Secure Docker Images

1. Prefer minimal base images
2. Dedicated user on the image as the least privileged user
3. Sign and verify images to mitigate MITM attacks
4. Find, fix and monitor for open source vulnerabilities
5. Don’t leak sensitive information to docker images
6. Use fixed tags for immutability
7. Use COPY instead of ADD
8. Use labels for metadata
9. Use multi-stage builds for small secure images
10. Use a linter

You can find more information on Snyk's 10 Docker Image Security Best Practices blog post.

Cleaning Docker

Removing a Running Container

```
docker container rm nginx
```

Removing a Container and its Volume

```
docker container rm -v nginx
```

Removing all Exited Containers

```
docker container rm $(docker container ls -a -f status=exited -q)
```

Removing All Stopped Containers

```
docker container rm `docker container ls -a -q`
```

Removing a Docker Image

```
docker image rm nginx
```

Removing Dangling Images

```
docker image rm $(docker image ls -f dangling=true -q)
```

Removing all Images

```
docker image rm $(docker image ls -a -q)
```

Removing all Untagged Images

```
docker image rm -f $(docker image ls | grep "^<none>" | awk "{print $3}")
```

Stopping & Removing all Containers

```
docker container stop $(docker container ls -a -q) && docker container rm
$(docker container ls -a -q)
```

Removing Dangling Volumes

```
docker volume rm $(docker volume ls -f dangling=true -q)
```

Removing all unused (containers, images, networks and volumes)

```
docker system prune -f
```

Clean all

```
docker system prune -a
```

Docker Swarm

Installing Docker Swarm

```
curl -ssl https://get.docker.com | bash
```

Initializing the Swarm

```
docker swarm init --advertise-addr 192.168.10.1
```

Getting a Worker to Join the Swarm

```
docker swarm join-token worker
```

Getting a Manager to Join the Swarm

```
docker swarm join-token manager
```

Listing Services

```
docker service ls
```

Listing nodes

```
docker node ls
```

Creating a Service

```
docker service create --name vote -p 8080:80 instavote/vote
```

Listing Swarm Tasks

```
docker service ps
```

Scaling a Service

```
docker service scale vote=3
```

Updating a Service

```
docker service update --image instavote/vote:movies vote
```

```
docker service update --force --update-parallelism 1 --update-delay 30s nginx
```

```
docker service update --update-parallelism 5--update-delay 2s --image instavote/vote:indent vote
```

```
docker service update --limit-cpu 2 nginx
```

```
docker service update --replicas=5 nginx
```

Connect Deeper

This work was first published in Painless Docker Course.

Join our online community FAUN and subscribe to our podcast The DevOps Fauncast.

Visit our publication The Chief I/O, subscribe to the newsletter and get cloud native insights from experts.