docker_cheatsheet

Docker builds images and runs them as isolated containers; Compose runs several containers together from one file. These are the commands you use every day, grouped by task. Podman accepts the same commands: switch to it and every command updates. Click a command to copy it.

:images — 12 cmds

Build, pull, tag and push images.

List images

```
$ docker images
```

Pull an image

```
$ docker pull nginx:1.27
```

Build an image from the current folder

```
$ docker build -t app:1.0 .
```

Build from another Dockerfile

```
$ docker build -f Dockerfile.prod -t app:prod .
```

Build without the cache

```
$ docker build --no-cache -t app:1.0 .
```

Build for several platforms

```
$ docker buildx build --platform linux/amd64,linux/arm64 -t app:1.0 .
```

Tag an image for a registry

```
$ docker tag app:1.0 registry.example.com/app:1.0
```

Push an image

```
$ docker push registry.example.com/app:1.0
```

Show the layers of an image

```
$ docker history app:1.0
```

Inspect an image

```
$ docker image inspect app:1.0
```

Remove an image

```
$ docker rmi app:1.0
```

Save an image to a file and load it back

```
$ docker save -o app.tar app:1.0   $ docker load -i app.tar
```

:run — 10 cmds

Start containers with the options you need most.

Run a throwaway interactive shell

```
$ docker run -it --rm alpine sh
```

Run in the background with a name and a port

```
$ docker run -d --name web -p 8080:80 nginx
```

Pass environment variables

```
$ docker run -e NODE_ENV=production --env-file .env app
```

Mount the current folder

```
$ docker run --rm -v "$PWD":/app -w /app node:22 npm test
```

Use a named volume

```
$ docker run -d -v pgdata:/var/lib/postgresql/data postgres:17
```

Restart unless stopped by hand

```
$ docker run -d --restart unless-stopped app
```

Limit memory and CPU

```
$ docker run --memory 512m --cpus 1 app
```

Join a network

```
$ docker run -d --network mynet --name api app
```

Override the entrypoint

```
$ docker run -it --entrypoint sh app
```

Run as another user

```
$ docker run --user 1000:1000 app
```

:containers

11 cmds

List, stop, remove and inspect containers.

List running containers

```
$ docker ps
```

List all containers, stopped too

```
$ docker ps -a
```

Show chosen columns

```
$ docker ps --format 'table {{.Names}}\t{{.Status}}\t{{.Ports}}'
```

Stop, start or restart a container

```
$ docker stop web   $ docker start web
$ docker restart web
```

Remove a container

```
$ docker rm web
```

Stop and remove a running container

```
$ docker rm -f web
```

Rename a container

```
$ docker rename web web-old
```

Live CPU and memory of every container

```
$ docker stats
```

Processes inside a container

```
$ docker top web
```

Inspect a container

```
$ docker inspect web
```

Get one field, like the IP address

```
$ docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' web
```

:debug

10 cmds

Look inside running containers.

Print container logs

```
$ docker logs web
```

Follow the last 100 lines

```
$ docker logs -f --tail 100 web
```

Logs from the last 10 minutes

```
$ docker logs --since 10m web
```

Open a shell in a container

```
$ docker exec -it web sh
```

Run a command as root

```
$ docker exec -u root web id
```

Copy a file out of a container

```
$ docker cp web:/etc/nginx/nginx.conf .
```

Copy a folder into a container

```
$ docker cp ./site web:/usr/share/nginx/html
```

Show files changed in a container

```
$ docker diff web
```

Show published ports

```
$ docker port web
```

Stream events from the engine

```
$ docker events
```

:compose 13 cmds

Run multi-container apps from compose.yaml.

Start everything in the background

```
$ docker compose up -d
```

Rebuild images, then start

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

Stop and remove containers and networks

```
$ docker compose down
```

Also remove the volumes

```
$ docker compose down -v
```

List the services

```
$ docker compose ps
```

Follow the logs of a service

```
$ docker compose logs -f web
```

Open a shell in a service

```
$ docker compose exec web sh
```

Run a one-off command

```
$ docker compose run --rm web npm test
```

Restart a service

```
$ docker compose restart web
```

Pull newer images

```
$ docker compose pull
```

Combine files for an environment

```
$ docker compose -f compose.yaml -f compose.prod.yaml up -d
```

Show the final merged config

```
$ docker compose config
```

Rebuild and sync on file changes

```
$ docker compose watch
```

:networks 6 cmds

Let containers find each other by name.

List networks

```
$ docker network ls
```

Create a network

```
$ docker network create mynet
```

Connect a running container

```
$ docker network connect mynet web
```

Disconnect a container

```
$ docker network disconnect mynet web
```

Inspect a network

```
$ docker network inspect mynet
```

Remove a network

```
$ docker network rm mynet
```

:volumes 5 cmds

Data that outlives the container.

List volumes

```
$ docker volume ls
```

Create a volume

```
$ docker volume create pgdata
```

Inspect a volume

```
$ docker volume inspect pgdata
```

Remove a volume

```
$ docker volume rm pgdata
```

Back up a volume to a tar file

```
$ docker run --rm -v pgdata:/data -v "$PWD":/backup alpine tar czf /backup/pgdata.tgz -C /data .
```

:cleanup 7 cmds

Get disk space back.

Show disk usage

```
$ docker system df
```

Remove stopped containers, unused networks and dangling images

```
$ docker system prune
```

Remove everything unused, volumes included

```
$ docker system prune -a --volumes
```

Remove stopped containers

```
$ docker container prune
```

Remove unused images

```
$ docker image prune -a
```

Remove unused volumes

```
$ docker volume prune
```

Clear the build cache

```
$ docker builder prune
```

:system 6 cmds

Log in, switch engines and check versions.

Log in to a registry

```
$ docker login registry.example.com
```

Log out

```
$ docker logout registry.example.com
```

Search Docker Hub

```
$ docker search postgres
```

List and switch contexts

```
$ docker context ls   $ docker context use remote
```

Show client and engine versions

```
$ docker version
```

Show engine-wide information

```
$ docker info
```

Updated Nov 8, 2026 · checked against Docker Engine 27+ and Compose v2 · sources: Docker CLI reference, Docker Compose CLI reference

Made by Lucas Bittencourt. Free, no ads, no tracking. Spotted a mistake? Tell me.