Docker cheat sheet

Field-tested Docker and Compose commands for running servers – searchable and copyable in one click. With warnings on commands that delete data.

daniele.michelin // cheat sheet

Images

List local images
```
docker images
```

Pull an image with a pinned tag
```
docker pull nginx:1.27
```

Build an image from the Dockerfile in this folder
```
docker build -t myapp:1.0 .
```

Retag an image for a registry
```
docker tag myapp:1.0 registry.example.com/myapp:1.0
```

! Delete an image
```
docker rmi myapp:1.0
```

Running & managing containers

List running containers
```
docker ps
```

List all containers including stopped ones
```
docker ps -a
```

Start a detached container with a port mapping
```
docker run -d --name web -p 8080:80 nginx
```

Throwaway container with an interactive shell
```
docker run --rm -it alpine sh
```

Restart the container after reboot or crash
```
docker run -d --restart unless-stopped nginx
```

Stop a container gracefully (SIGTERM, then SIGKILL)
```
docker stop web
```

! Force-stop and delete a container
```
docker rm -f web
```

Logs, exec & diagnostics

Last 100 log lines, then follow live
```
docker logs -f --tail 100 web
```

Open a shell in a running container
```
docker exec -it web sh
```

Run a single command in the container
```
docker exec web nginx -t
```

CPU and RAM usage of all containers (snapshot)
```
docker stats --no-stream
```

Volumes & networks

Mount a named volume into a container
```
docker run -v data:/var/lib/mysql mariadb
```

Back up a volume to a tar archive
```
docker run --rm -v data:/d -v $(pwd):/b alpine tar czf
/b/data.tgz -C /d .
```

! Delete a volume – the data is gone
```
docker volume rm data
```

List networks
```
docker network ls
```

Create a user-defined bridge (DNS by container name)
```
docker network create appnet
```

Docker Compose

Start the stack in the background
```
docker compose up -d
```

Rebuild images and start the stack
```
docker compose up -d --build
```

Follow the logs of one service live
```
docker compose logs -f web
```

Pull new images, recreate changed services
```
docker compose pull && docker compose up -d
```

Shell inside a service container
```
docker compose exec web sh
```

Stop the stack and remove containers and network
```
docker compose down
```

! Like down, but also deletes the volumes
```
docker compose down -v
```

Cleaning up with prune

Disk usage by images, containers, volumes
```
docker system df
```

! Delete untagged (dangling) images
```
docker image prune
```

! Delete all unused volumes – data loss possible
```
docker volume prune
```

! Delete stopped containers, unused networks and dangling images
```
docker system prune
```

! Radical: everything unused incl. images and volumes
```
docker system prune -a --volumes
```

Resource limits

Limit memory to 512 MB
```
docker run -d --memory 512m nginx
```

Limit to 1.5 CPU cores
```
docker run -d --cpus 1.5 nginx
```

Change the limits of a running container
```
docker update --memory 1g --memory-swap 1g web
```

Set log rotation for a container
```
docker run -d --log-opt max-size=10m --log-opt max-
file=3 nginx
```

Find OOM kills from the last hour
```
docker events --since 1h --filter event=oom
```