DOCKER CLI CHEAT-SHEET

INSTALLATION

Docker Desktop is available for Mac, Linux and Windows

https://docs.docker.com/desktop

View example projects that use Docker

https://github.com/docker/awesome-compose

Check out our docs for information on using Docker

https://docs.docker.com

NETWORKING

List Networks:
docker network ls

Create Network:
docker network create network_name

Run Container with Network:
docker run --network network_name image_name

VOLUME

List Volumes:
docker volume ls

Create Volume:
docker volume create volume_name

Run Container with Volume
docker run -v volume_name:/path image_name

DOCKER-COMPOSE

Compose File:
docker-compose.yml

Start Services:
docker-compose up

Start Services in Background:
docker-compose up -d

Stop Services:
docker-compose down

Scale Services:
docker-compose up --scale service_name=num_instances

DOCKER REGISTRY

Push Image:
docker push image_name

Pull Image:
docker pull image_name

DOCKER IMAGE MANAGEMENT

WORKING WITH IMAGES

List Images:
docker images

Show Image Details:
docker inspect <image_name>

Build Image from Dockerfile:
docker build -t <image_name>:<tag> <path_to_dockerfile>

Pull Image from Registry:
docker pull <image_name>:<tag>

Tag an Image:
docker tag <source_image>:<source_tag> <target_image>:<target_tag>

Remove Image:
docker rmi <image_name>:<tag>

Remove Dangling Images:
docker image prune

MANAGING IMAGE LAYERS

List Image Layers:
docker history <image_name>:<tag>

View Disk Usage by Image:
docker system df

Remove Unused Data (Images, Containers, Volumes, Networks):
docker system prune

ADVANCED IMAGE OPERATIONS

Build Image with Build Context:
docker build -t <image_name>:<tag> -f <dockerfile_path> <build_context_path>

List Tags for an Image:
docker images <image_name>

Prune All Unused Images:
docker image prune -a

Delete All Images (Use with Caution):
docker rmi $(docker images -q)

Build Image with Build Arguments:
docker build --build-arg <arg_name>=<arg_value> -t <image_name>:<tag><path_to_dockerfile

CONTAINERIZING WITH IMAGES

Create Container from Image:
docker create --name <container_name> <image_name>:<tag>

Start Container from Image:
docker run -d <image_name>:<tag>

Start Container from specific Image:
docker run <image_name>@<digest>

Commit Changes to New Image:
docker commit <container_id> <new_image_name>:<tag>

Export Image to Tarball:
docker save -o <output_file>.tar <image_name>:<tag>

Load Image from Tarball:
docker load -i <input_file>.tar

Push Image to Registry:
docker push <image_name>:<tag>

DOCKER CONTAINER MANAGEMENT

RUNNING CONTAINERS

Run a container:
docker run <image_name>

Run in Detached Mode:
docker run -d <image_name>

Container with Name:
docker run --name <container_name> <image_name>

Port Mapping:
docker run -p <host_port>:<container_port> <image_name>

Environment Variables:
docker run -e <variable_name>=<value> <image_name>

Remove After exit / Auto Remove:
docker run --rm <image_name>

Mount Volume:
docker run -v <volume_name>:/path/in/container <image_name>

Host Directory Mount:
docker run -v /host/path:/container/path <image_name>

Read-Only Volume:
docker run -v <volume_name>:/path/in/container:ro <image_name>

Interactive Mode:
docker run -it <image_name> /bin/bash

Name & Network:
docker run --name <name> --network <network> <image>

Background, Port, Environment:
docker run -d -p <host>:<container> -e <var>=<val> <image>

MORE CONTAINER ACTIONS

Pause All Containers:
docker pause $(docker ps -q)

Unpause All Containers:
docker unpause $(docker ps -q)

Stop All Containers:
docker stop $(docker ps -aq)

Restart All Containers:
docker restart $(docker ps -aq)

DELETING CONTAINERS

Stop and Delete a Running Container:
docker stop <container_id>
docker kill <container_id>
docker rm <container_id>

Force Remove a Running Container:
docker rm -f <container_id>

Delete All Stopped Containers:
docker container prune

Delete Containers by Name:
docker rm <container_name>

Delete All Containers (Stopped and Running):
docker rm -f $(docker ps -aq)

Delete All Exited Containers:
docker rm $(docker ps -aq -f status=exited)

Delete Containers Matching a Pattern:
docker ps -a | grep "<pattern>" | awk '{print $1}' | xargs -I {} docker rm {}

DELETING CONTAINERS WITH VOLUMES

Delete Container & Volume:
docker rm -v <container_id>

Delete Unused Volumes:
docker volume prune

Delete Specific Volume:
docker volume rm <volume_name>

INFO & STATS

Show the logs of a container:
docker logs CONTAINER

Show stats of running containers:
docker stats

Show processes of container:
docker top CONTAINER

Show installed docker version:
docker version

Get detailed info about an object:
docker inspect NAME

Show all modified files in a container:
docker diff CONTAINER

Show mapped ports of a container:
docker port CONTAINER
docker port web

CREATING VOLUMES AND MOUNTING

Create a Volume:
docker volume create <volume_name>

Container with Mounted Volume:
docker run -v <volume_name>:/path/in/container <image_name>

Host Directory Mount:
docker run -v /host/path:/container/path <image_name>

Read-Only Volume:
docker run -v <volume_name>:/path/in/container:ro <image_name>

MANAGING CONTAINERS

List Running Containers:
docker ps

List All Containers:
docker ps -a

Inspect Container:
docker inspect <container_id>

Access Container Shell:
docker exec -it <container_id> /bin/bash

Stop Container:
docker stop <container_id>

Start Container:
docker start <container_id>

Restart Container:
docker restart <container_id>

Pause Container:
docker pause <container_id>

Unpause Container:
docker unpause <container_id>

Rename Container:
docker rename <old_name> <new_name

Copy Files to/from Container:
docker cp <src_path> <container_id>:<dest_path>
docker cp <container_id>:<src_path> <dest_path>

Get Container Logs:
docker logs <container_id>

Attach to Container's Terminal:
docker attach <container_id>