Docker Cheatsheet

(c) 2021 Lino Figueroa (lino.figueroa@softwareone.com)

Definitions

Container

A standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another.

Image (container image)

Read-only template that contains a set of instructions for creating a container that can run on a container platform.

Image name

Label that is made up of slash-separated name components, optionally prefixed by a registry hostname.

Tag / image tag

Label that convey useful information about a specific image version or variant.

By default the most recent version of an image, if not specified at construction, is "latest".

Run containers

Detached or interactive

`-d` (preferred) parameter indicates that the container will be run dettached from the console (fire and forget).

`-it` indicates that the console will be attached to the container (interactive).

By default contaienrs are launched attached to the console but no interactive.

Run a container

```
docker run --name <container_name> <container_image:tag>
```

If no name specified docker will generate a random one.

Important: names are used for container communication.

Run a container mapping ports

```
docker run --name <container_name> -p host_port:container_port <container_image:tag>
```

`-p` parameter can be specified several times to map several ports.

`-P` parameter will map all the container ports to host ports that will be automatically assigned starting from 30000.

Run a container mapping volumes

```
docker run --name <container_name> -v host_file_or_folder:container_file_or_folder
<container_image:tag>
```

Run a container sending environment variables

```
docker run --name <container_name> -e VARIABLE_NAME=variable_value <container_image:tag>
```

Manage containers

View containers

```
docker ps
```

View a list of running containers

```
docker ps -a
```

View a list of all the container (running, stopped, paused, etc.)

Stop a container

```
docker stop <container_name_or_id>
```

Kill a container

```
docker kill <container_name_or_id>
```

Stops the container without waiting for a gracefully stop. Sends KILL signal.

Start a stopped container

```
docker start <container_name_or_id>
```

Will be started with the launching parameters specified in run (ports, volumes, environment variables, etc.)

Delete a container

```
docker rm <container_name_or_id>
```

If the container is running should be stopped or killed or use the `-f` parameter.

```
docker rm -f <container_name_or_id>
```

Run a shell in a container

```
docker exec -it <container_name> <program_name>
```

`<program_name>` can be `sh`, `bash` or any other shell.

View logs

```
docker logs <container_name>
```

Shows logs from standard out.

`-f` parameter activates `follow` mode and console are updated with new logs.

Inspect container

```
docker inspect <container_name>
```

Returns information about the given container.

Manage images

Build

```
docker build -t <image_name:image_tag> .
```

Builds the image using the Dockerfile present in the current folder

View images

```
docker images
```

Delete image

```
docker rmi <image_name:image_tag>
```

Retag image

```
docker tag <origin_image:origin_tag> <destination_image:destination_tag>
```

(Used before pushing images to a Registry)

Push image to Docker hub

1. Login in docker hub

```
docker login --username=yourusername --password yourpassword
```

2. Tag image

```
docker image tag <image_name:tag> <yourhubusername>/<image_name:tag>
```

Example:

```
docker image tag basicweb:latest impalah/basicweb:latest
```

3. Push

```
docker push <yourhubusername>/<image_name:tag>
```

Docker volumes

A volume is a mapping between a host folder (or a volume object) with a folder in the container.

Files in the mapped unit in the container will be stored in a folder in the host computer.

Create a volume

```
docker volume create <volume_name>
```

List volumes

```
docker volume ls
```

Inspect volume

Get information about a volume.

```
docker inspect <volume>
```

Generally, the folder of volumes will be under `/var/lib/docker` (see below for Windows).

Windows: WSL mapping

Docker on windows uses WSL (Linux subsystem) and maps linux directories to $wsl drive.

More generally `/var/lib/docker/` maps to `\\wsl$\docker-desktop-data\version-pack-data\community\docker\`

Delete volume

```
docker volume rm <volume>
```

Attach volume to container

If volume does not exists, it will be created.

```
docker run -d --name devtest -v myvol2:/app nginx:latest
```

Bind mounts

This operation will bind a folder in the host computer with a folder in the container.

The difference is that no volume folder will be created.

Host folder path should be absolute. If only a name is specified docker will create a volume.

```
docker run -d -it --name devtest -v C:/Users/Impalah/projects:/app nginx:latest
```

(test, entering nginx): `docker exec -it devtest sh`