Docker

# Writing a dockerfile

A Docker image/container is built from a Dockerfile wich contains the code describing the steps that need to be taken in order to build the image.

## Dockerfile structure

Create dockerfile: create a new file and call it “Dockerfile” don’t add any extensions

# use official ubuntu runtime as parent image

FROM ubuntu:18.04

RUN apt-get update \

&& apt-get install -y dos2unix \

&& rm -rf /var/lib/apt/lists/\*

# add the scripts to the container

ADD ./01\_copy\_rawdata.sh /home/Scripts/01\_copy\_rawdata.sh

# fix execute rights if necessairy

RUN chmod 755 /home/Scripts/01\_copy\_rawdata.sh

# convert Windows EOL to UNIX EOL (required when scripts where manipulated in Windows)

RUN dos2unix /home/Scripts/01\_copy\_rawdata.sh

Basic options:

|  |  |
| --- | --- |
| FROM | The base image to build on (look on <https://hub.docker.com/search?q=&type=image>) to find base images that suit your needs |
| RUN | Commands that need to be executed when building the image  Depend on the OS chosen as base image |
| ADD | Usage: ADD “location on host” “location in container”  Add external files or folders to the container. Files and folders specified like this will be **static** meaning that once the image is build, any **changes** made to these files/folders on the host **won’t carry** **over** into the container or the host (and vice versa).  The ADD function can only add files and folders that are available in the folder containing the Dockerfile itself |
| CMD | Usage: CMD [“command1”, “command2”]  Specify commands that will automatically be executed on container start-up |

## docker commands

|  |  |
| --- | --- |
| $docker --help | General info |
| $docker image ls | List all installed images |
| $docker container ls | List all running containers |
| $docker image rm “imagename”:”version” | Remove a image |
| $docker prune | Remove all docker images, containers… (this doesn’t remove docker) |

### docker build

* Build the image from the Dockerfile
* **Usage**: *$docker build --tag=”username”/”imagename”:”version” .*
  + The “username” is only required when you want to push the container to dockerhub
  + The “version” tag isn’t required, when it isn’t specified it will default to “latest”
  + The “.” Indi
* The image name can’t contain upper case letters (or numbers)

### Docker push

* Push a build image to dockerhub so it can be downloaded at any time and place
* **Usage**: *$docker push “username”/”imagename”:”version”*
* In order to push a docker image to dockerhub you need to be logged in: use *$docker login*

### docker run

* Start a container from an image. If the image isn’t present on the host, *“$docker run”* will automatically search dockerhub for the image, download and install it. If the image is already present on the host, it will just launch the container from the image
* **Usage**: *$docker run ‘username’/’imagename’:’version’ ‘command to execute’*
  + The command option isn’t required, when not provided to will default to /bin/bash most of the time
* Docker run has lots of options which can be found here: <https://docs.docker.com/engine/reference/run/>

|  |  |
| --- | --- |
| -it | Keep the output on the attatched foreground in a terminal |
| --rm | Remove all remaining container files on container exit  (containers always preserve the final file structure in memory when exited, this command option stops this preventing cluttering of the system) |
| -v | Mount a volume from the host on the container, these volumes will be **dynamic**  **Usage:** *-v “host-location”:”container-location”* |
| --mount | Same as the -v option, but more readable  Doesn’t seem to work on windows  **Usage:** *--mount src=””,target=””,type=””* |
| --name | Give the container a recognizable name |