

Understanding the Dockerfile Commands

#beginners#docker#serverless#tutorial

"Blueprint for creating a docker image"

Docker builds images automatically by reading the instructions from a Dockerfile.

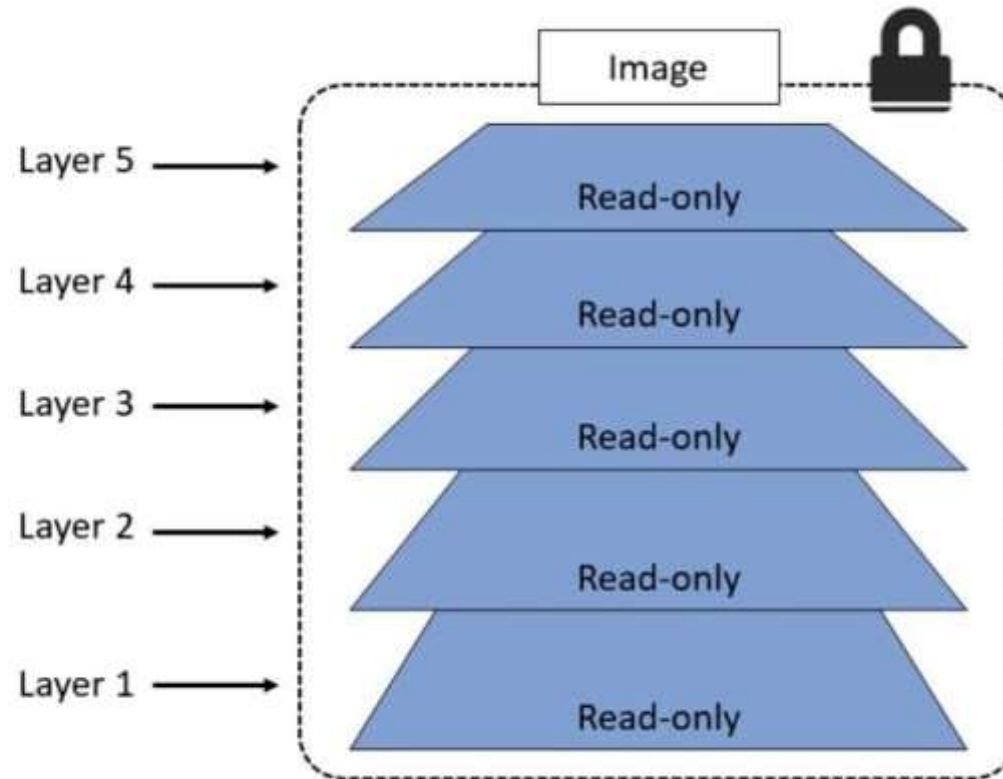
It is a text file without any .txt extensions that contains all commands in order, needed to build a given image.

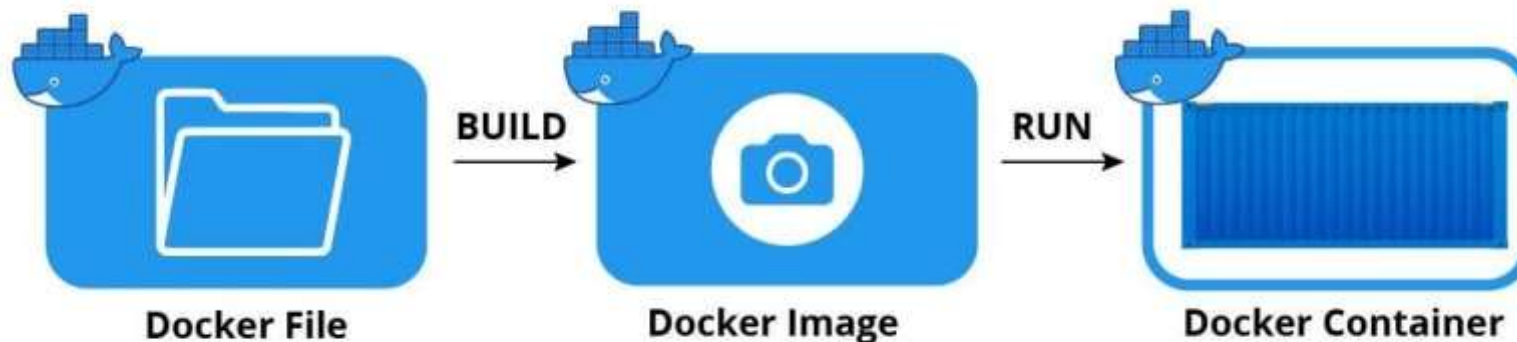
It is always named **Dockerfile**.

Docker image consists of read-only layers each of which represents a Dockerfile instruction. The layers are stacked and each one is created by the change from the previous layer. For example, if I create a base layer of ubuntu and then in second instruction I install Python it will create a second layer. Likewise, if I do any changes by the instructions(RUN , COPY , ADD) it will create a new layer in that image.

Containers are read-write layers that are created by docker images.

In simple words, a Dockerfile is a set of instructions that creates a stacked-layer for each instruction that collectively makes an image(which is a prototype or template for containers)





Frequently used Dockerfile commands -

FROM - Defines a base image, it can be pulled from docker hub

(for example- if we want to create a javascript application with node as backend then we need to have node as a base image, so it can run node application.)

RUN - Executes command in a new image layer(we can have multiple run commands)

CMD - Command to be executed when running a container(It is asked to have one CMD command, If a Dockerfile has multiple CMDs, it only applies the instructions from the last one.

EXPOSE - Documents which ports are exposed (It is only used for documentation)

ENV - Sets environment variables inside the image

COPY - It is used to copy your local files/directories to Docker Container.

ADD - It is more feature-rich version of the COPY instruction. COPY is preferred over ADD. Major difference b/w ADD and COPY is that ADD allows you to copy from URL that is the source can be URL but in COPY it can only have local ones.

ENTRYPOINT - Define a container's executable (You cannot override and ENTRYPOINT when starting a container unless you add the --entrypoint flag.)

VOLUME - It defines which directory in an image should be treated as a volume. The volume will be given a random name which can be found using docker **inspect** command.

WORKDIR - Defines the working directory for subsequent instructions in the Dockerfile(Important point to remember that it doesn't create a new intermediate layer in Image)

#Basic Dockerfile

```
FROM ubuntu:18.04
```

```
COPY . /app
```

RUN make /app

CMD python /app/app.py

Each instruction creates one layer:

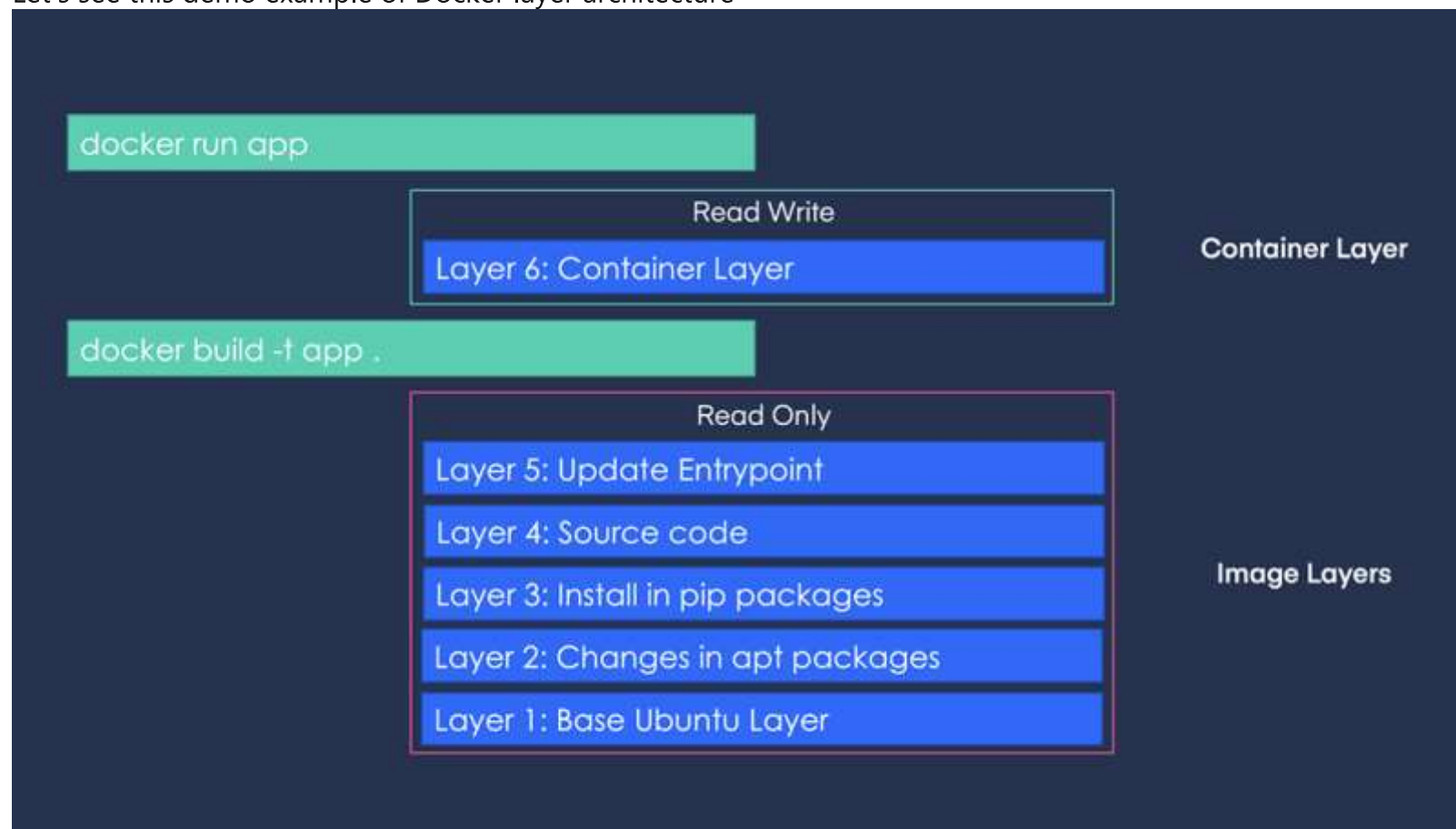
FROM creates a layer from the ubuntu:18.04 Docker image.

COPY adds files from your Docker client's current directory.

RUN builds your application with make.

CMD specifies what command to run within the container.

Let's see this demo example of Docker layer architecture-



If some files should be prevented from being copied into the Docker image(it can be sensitive informations like .env files which contains API keys or any other files that are not much important), a .dockerignore file can be added at the same level as the Dockerfile where files that should not be copied over into the Docker image can be specified. By this if we are using a COPY or ADD instruction in a Dockerfile to specify the files to be added into a Docker image, any file specified in the .dockerignore file will be ignored and not added into the Docker image.

Shell and Exec forms

All three instructions (RUN, CMD and ENTRYPOINT) can be specified in shell form or exec form.

Shell form

RUN apt-get install python3

CMD echo "Hello world"

ENTRYPOINT echo "Hello world"

Exec form

This is the preferred form for CMD and ENTRYPOINT instructions. ["executable", "param1", "param2", ...]

Difference between RUN, CMD and ENTRYPOINT?

RUN - RUN instruction allows you to install your application and packages required for it. It executes any commands on top of the current image and creates a new layer by committing the results. Often you will find multiple RUN instructions in a Dockerfile.

RUN apt-get install python

CMD - CMD instruction allows you to set a default command, which will be executed only when you run container without specifying a command. If Docker container runs with a command, the default command will be ignored. If Dockerfile has more than one CMD instruction, all but last CMD instructions are ignored.

CMD "echo" "Hello World!"

ENTRYPOINT - ENTRYPOINT instruction allows you to configure a container that will run as an executable. It looks similar to CMD, because it also allows you to specify a command with parameters. The difference is ENTRYPOINT command and parameters are not ignored when Docker container runs with command line parameters.

Prefer ENTRYPOINT to CMD when building executable Docker image and you need a command always to be executed. Additionally use CMD if you need to provide extra default arguments that could be overwritten from command line when docker container runs.

Choose CMD if you need to provide a default command and/or arguments that can be overwritten from command line when docker container runs.

To know more about it -