Docker Dockerfile

A Dockerfile is a text document that contains commands that are used to assemble an image. We can use any command that call on the command line. Docker builds images automatically by reading the instructions from the Dockerfile.

The docker build command is used to build an image from the Dockerfile. You can use the -f flag with docker build to point to a Dockerfile anywhere in your file system.

1. $ docker build -f /path/to/a/Dockerfile .

Dockerfile Instructions

The instructions are not case-sensitive but you must follow conventions which recommend to use uppercase.

Docker runs instructions of Dockerfile in top to bottom order. The first instruction must be **FROM** in order to specify the Base Image.

A statement begin with # treated as a comment. You can use RUN, CMD, FROM, EXPOSE, ENV etc instructions in your Dockerfile.

Here, we are listing some commonly used instructions.

FROM

This instruction is used to set the Base Image for the subsequent instructions. A valid Dockerfile must have FROM as its first instruction.

Ex.

1. FROM ubuntu

LABEL

We can add labels to an image to organize images of our project. We need to use LABEL instruction to set label for the image.

Ex.

1. LABEL vendorl = "JavaTpoint"

RUN

This instruction is used to execute any command of the current image.

Ex.

1. RUN /bin/bash -c 'source $HOME/.bashrc; echo $HOME'

CMD

This is used to execute application by the image. We should use CMD always in the following form

1. CMD ["executable", "param1", "param2"?]

This is preferred way to use CMD. There can be only one CMD in a Dockerfile. If we use more than one CMD, only last one will execute.

COPY

This instruction is used to copy new files or directories from source to the filesystem of the container at the destination.

* The source path must be inside the context of the build. We cannot COPY ../something /something because the first step of a docker build is to send the context directory (and subdirectories) to the docker daemon.
* If source is a directory, the entire contents of the directory are copied including filesystem metadata.

WORKDIR

The WORKDIR is used to set the working directory for any RUN, CMD and COPY instruction that follows it in the Dockerfile. If work directory does not exist, it will be created by default.

We can use WORKDIR multiple times in a Dockerfile.

Example:

In the following example I create docker file for lunch chatapp which is written in python.

FROM python:3.8

MAINTAINER priyankamore8997@gmail.com

RUN apt-get upgrade -y

RUN apt-get -y update

RUN apt-get install -y python3-pip

RUN apt-get install -y default-libmysqlclient-dev

RUN pip install --upgrade pip

RUN pip3 install django

RUN pip3 install virtualenv

RUN pip3 install django-environ

RUN pip3 install mysqlclient

COPY /chatapp /chatapp

WORKDIR /chatapp

RUN pip3 install -r requirements.txt

ENV DATABASE\_TYPE=mysql

ENV DATABASE\_NAME=chatapp

ENV USER\_NAME=admin

ENV PASSWORD=admin123

ENV HOST\_NAME=chatapp.ckze0nlll2e9.us-east-2.rds.amazonaws.com

ENV PORT=3306

EXPOSE 8081

WORKDIR /chatapp/fundoo/

CMD python3 manage.py migrate

CMD ["python", "manage.py", "runserver", "0.0.0.0:8081"]