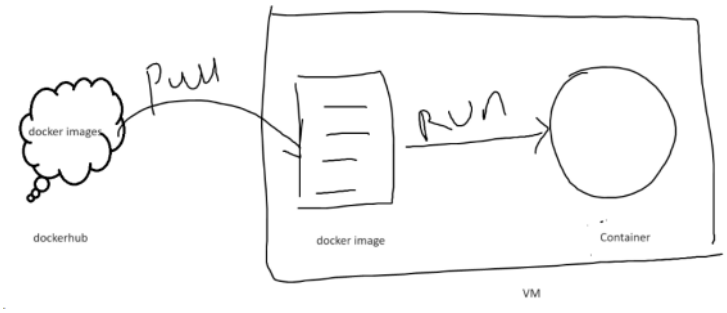
Docker-Day-3

22 December 2023 19:51

==========

dockerfile

==========

- It is a normal script file contains the set of instructions which are useful to build a Docker image.
    - dockerfile ==> docker image ==> container
    - 

► Now let's instructions that we use inside the dockerfile

=====

FROM

=====

- In every dockerfile the first line is **FROM** instruction.
- The **FROM** used to pull the base image & that we are looking to customize it.
- Search of dockerfile in internet we can use every file started with from instruction.

  Ex:-
  FROM ubuntu:latest

======

LABEL

======

- LABEL instruction used to stores meta data information about docker Image, like who is the author of the Docker Image.
- As best practice purpose only we use LABEL instruction.

  Ex:-
  LABEL owner chaitanya
  LABEL team XYZ

=====

RUN

=====

- RUN instruction used execute shell commands, I mean any Linux commands

  Ex:-
  RUN apt-get install wget -y
  RUN apt-get install tree -y
  RUN apt-get install vim -y

usecase-1:

- Create a custom docker image with git presence
  FROM ubuntu:latest
  LABEL owner chaitanya
  RUN apt-get install git -y

- To Create docker image based out of the docker file we use command

  docker build -t custom_git:latest -f dockerfile

  docker images

  docker run -it custom_git:latest /bin/bash

  git --version

usecase-2:

- Use single RUN instruction for instead of multiple run instructions

  Way-1
  FROM ubuntu:latest
  LABEL owner chaitanya
  RUN apt-get install git -y
  RUN apt-get install wget -y
  RUN apt-get install tree -y
  RUN apt-get install vim -y

  Way-2
  FROM ubuntu:latest
  LABEL owner chaitanya
  RUN apt-get install git -y && \
  apt-get install wget -y && \
  apt-get install tree -y && \
  apt-get install vim -y

=====
CMD
=====

- CMD instruction is used to provide default command & parameters to docker container.
- We can easily override the default commands that are mentioned in CMD while spinning up container.
- If we have multiple CMD instructions in dockerfile the last CMD instruction only will be applied.

Usecase3: Create docker image with default echo command for docker container using CMD

- **Create docker file**
  FROM ubuntu
  LABEL Chaitanya
  RUN apt-get update
  CMD ["echo", "Hello World"]
- The each string in shell command will be double quoted, then only CMD will consider it as command.
- **Create docker image**
  docker build -t basic_cmd:1.0 -f dockerfile
  docker images
  docker run -d basic_cmd:1.0
- The container will gets spin-up & execute the default echo command inside container & container will get stopped automatically since there is no long running command inside it.

Usecase4: Create docker image such way that it should be up & running as long you stop it explicitly using CMD

- **Create docker file**
  FROM ubuntu
  LABEL owner chaitanya
  RUN apt-get update -y
  RUN apt-get install iputils-ping -y
  CMD ["ping", "google.com"]
- **Create docker image**
  docker build -t basic_cmd:2.0 -f dockerfile
  docker images
  docker run basic_cmd:2.0

Usecase5:- Override the shell commands in CMD instruction

- docker run basic_cmd:2.0 hostname
- The hostname command overrides the default ping command & prints the hostname of the container.

============
ENTRYPOINT
============

- Similar to CMD instruction, ENTRYPOINT instruction also is used to provide default command & parameters to docker container.

- **What is the difference between CMD and ENTRYPOINT?**
  - You cannot override the ENTRYPOINT instruction by adding command-line parameters to the docker run command.

  - By opting for this instruction, you imply that the container is specifically built for such use.

Usecase6:- Create docker image with default echo command for docker container using ENTRYPOINT

- **Create docker file**

  FROM ubuntu
  MAINTAINER sofija
  RUN apt-get update
  ENTRYPOINT ["echo", "Hello World"]

- **Create docker image**
  docker build -t basic_entrypoint:1.0 -f dockerfile /root
  docker images
  docker run basic_entrypoint:1.0

- Prints the Hello World & container will be stopped automatically

Usecase7:- Create docker image such way that it should be up & running as long you stop it explicitly using CMD

- **Create docker file**
  FROM ubuntu
  LABEL owner chaitanya
  RUN apt-get update -y
  RUN apt-get install iputils-ping -y
  ENTRYPOINT ["ping", "google.com"]

- **Create docker image**
  docker build -t basic_entrypoint:2.0 -f dockerfile /root
  docker images
  docker run basic_entrypoint:2.0

Usecase8:- Try to override the shell commands in ENTRYPOINT instruction

- docker run basic_entrypoint:2.0 hostname

- The hostname command just append as string to the default ping command & prints the hostname

Usecase9:- Using CMD + ENTRYPOINT

- **Create docker file**
  FROM ubuntu
  LABEL Chaitanya
  RUN apt-get update
  ENTRYPOINT ["echo", "Hello"]
  CMD ["World"]

==========
WORKDIR
==========

- WORKDIR instruction used to define the working directory of a Docker container at any given time.
  Ex:- Create dockerfile & image based based out of it without workdir instruction
  FROM ubuntu:latest
  RUN apt-get update -y
  RUN apt-get install git -y
  RUN touch 1.out 2.out 3.out

- Create docker image
  docker build -t custom_img:latest .
  docker run -it custom_img:latest /bin/bash

- Here the we can observe that the default working directory "/" & files are created under "/"

Usecase10:- Create a custom WORKDIR

- **Create docker file**
  FROM ubuntu:latest
  WORKDIR /project
  RUN apt-get update -y
  RUN apt-get install git -y
  RUN touch 1.out 2.out 3.out

- Here the we can observe that the default working directory "/project" & files are created under "/project"

=====
ADD
=====

- The ADD command is used to copy files/directories into a Docker image. It can copy data in three ways:
  - Copy files from the local storage to a destination in the Docker image.
  - Copy a **tar ball from the local storage and extract** it automatically inside a destination in the Docker image.
  - Copy files from a URL to a destination inside the Docker image.

- Ex:- **Create a docker image to copy files from VM to docker container**

  FROM ubuntu:latest
  WORKDIR /project
  RUN mkdir test
  ADD codes /project/test
  ADD abc.out /project/test
  ADD samp.tar.gz /project/test

=====
COPY
=====

- The COPY command is used to copy files/directories into a Docker image. It can copy data in three ways:

- Ex:-

  FROM ubuntu:latest
  WORKDIR /project
  RUN mkdir test
  COPY codes /project/test
  COPY abc.out /project/test

==========
EXPOSE
==========

- EXPOSE instructions informs the docker that the container listens on specific port at runtime.

- Jenkins container will listens on port 8080

- Ex:-
  EXPOSE 8080

==========
Usercase10: Build tomcat docker image with deployable package
==========

- Take Ubuntu as base image
- Install tomcat inside docker Image
- Copy the sample war Artifact
- Expose port 8080, since tomcat listens on port 8080
- Run tomcat as service

Way-1

FROM Ubuntu
LABEL Chaitanya
RUN mkdir /opt/tomcat/
WORKDIR /opt/tomcat
RUN curl -O https://www-eu.apache.org/dist/tomcat/tomcat-8/v8.5.40/bin/apache-tomcat-8.5.40.tar.gz
RUN tar xvfz apache*.tar.gz
RUN mv apache-tomcat-8.5.40/* /opt/tomcat/.

RUN yum -y install java
RUN java -version
WORKDIR /opt/tomcat/webapps
RUN curl -O -L https://github.com/AKSarav/SampleWebApp/raw/master/dist/SampleWebApp.war
EXPOSE 8080
CMD ["/opt/tomcat/bin/catalina.sh", "run"]

Way-2: Actually you don't need to install tomcat on Ubuntu we have docker image for tomcat itself, for your understanding on instructions I craeted dockerfile based on Ubuntu image

FROM tomcat:8.0-alpine
LABEL maintainer="abc@gmail.com"
ADD sample.war /usr/local/tomcat/webapps/
EXPOSE 8080
CMD ["catalina.sh", "run"]

======
ARGS
======

- ARG instruction defines the variables during docker image creation.
- **Usecase: Define variable & call it inside the dockfile using ARG instruction**
    - **Create docker file**
      FROM ubuntu:latset
      LABEL chaitanya
      WORKDIR /
      ARG user1 chai
      RUN echo $user1 > user_name.out
    - **Create docker image**
      docker build -t myarg:1.0 .
      docker run -it myarg:1.0 /bin/bash

      cat /user_name.out
    - Override the user1 value
      docker build -t myarg:2.0 --build-arg user1=tom .

======
ENV
======

- ENV instruction is used to set the Environment variables(system-level/user-level in windows) for the future containers which are created from custom docker image.

  FROM ubuntu:latest
  WORKDIR /project
  ENV http_proxy http://10.0.0.0:8000
  RUN echo $http_proxy

======
VOL(use case step required)
======

- Create a volume inside a container & persist data even container killed/terminated

  FROM ubuntu:latest
  RUN mkdir /data
  WORKDIR /data
  RUN echo "Hello from Volume" > test
  VOLUME /data

Home work:

1. Dockerize jar & spin up container & access application
2. Automate the ci/cd process
    a. Maven build
    b. Sonar-scanning
    c. Dockerize war file
    d. Push docker image to docker registry
    e. Deploy image &Spin as container
    f. Access the application