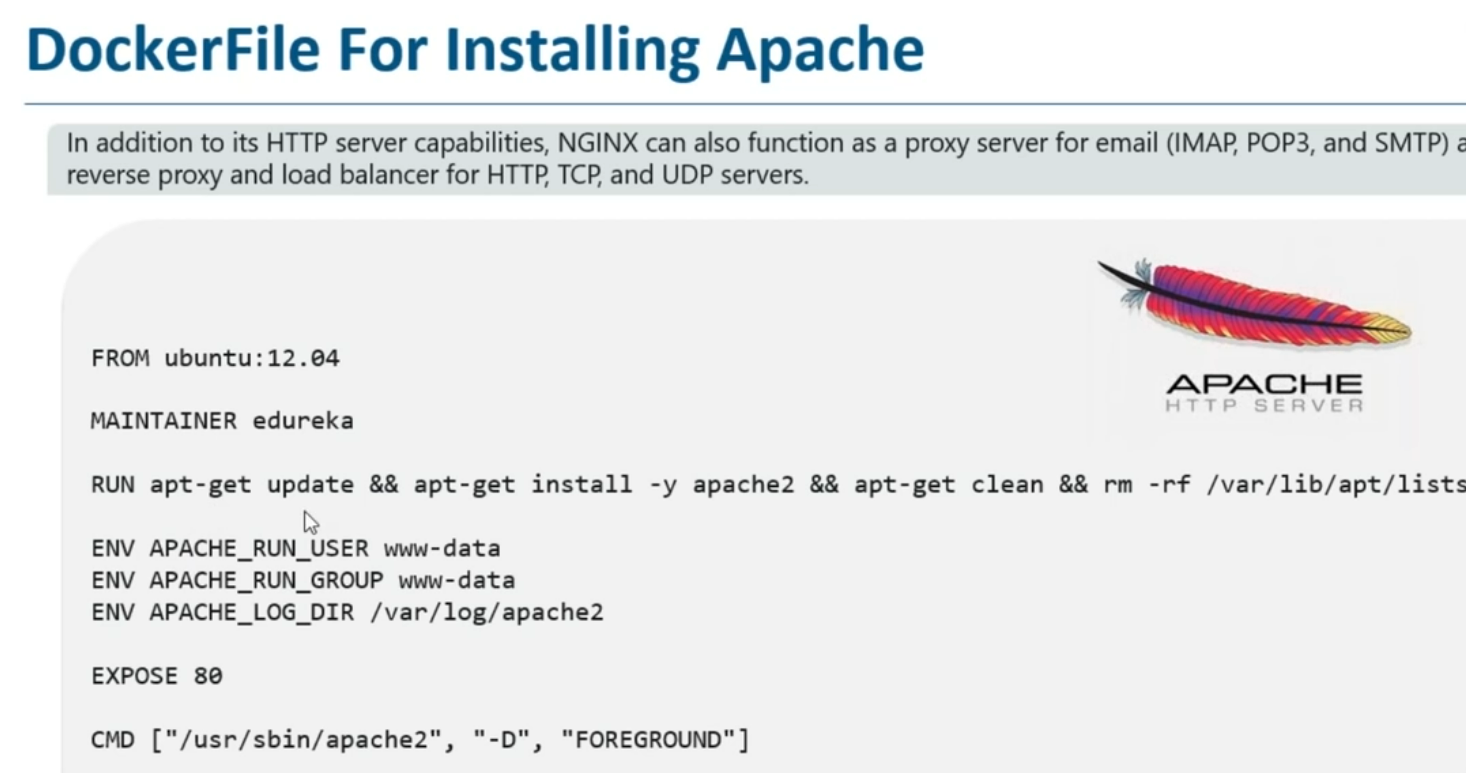
Docker file is json language





CMD:we can change the command while creating container, it will append the command

In file [“echo hi”] is there but while creating conatainer docker run -d venky echo helloboss mention it will append hi and create with helloboss

Cmd = [“echo”, ” hi”]

Entrypoint = [“echo ”]

Case2:🡪 if we not mention parameter then only it will go cmd other wise even in this scenario also if mention something like ( docker run -d venky hello hoe r u?) it will create by hello how r u?

Entrypoint = [“echo ”]

Cmd = [” hi”] 🡪 here we mention parameter that in cmd so will check parameter cmd and it will take

ENTRY POINT: we cant change command while creating container, it will execute as it is

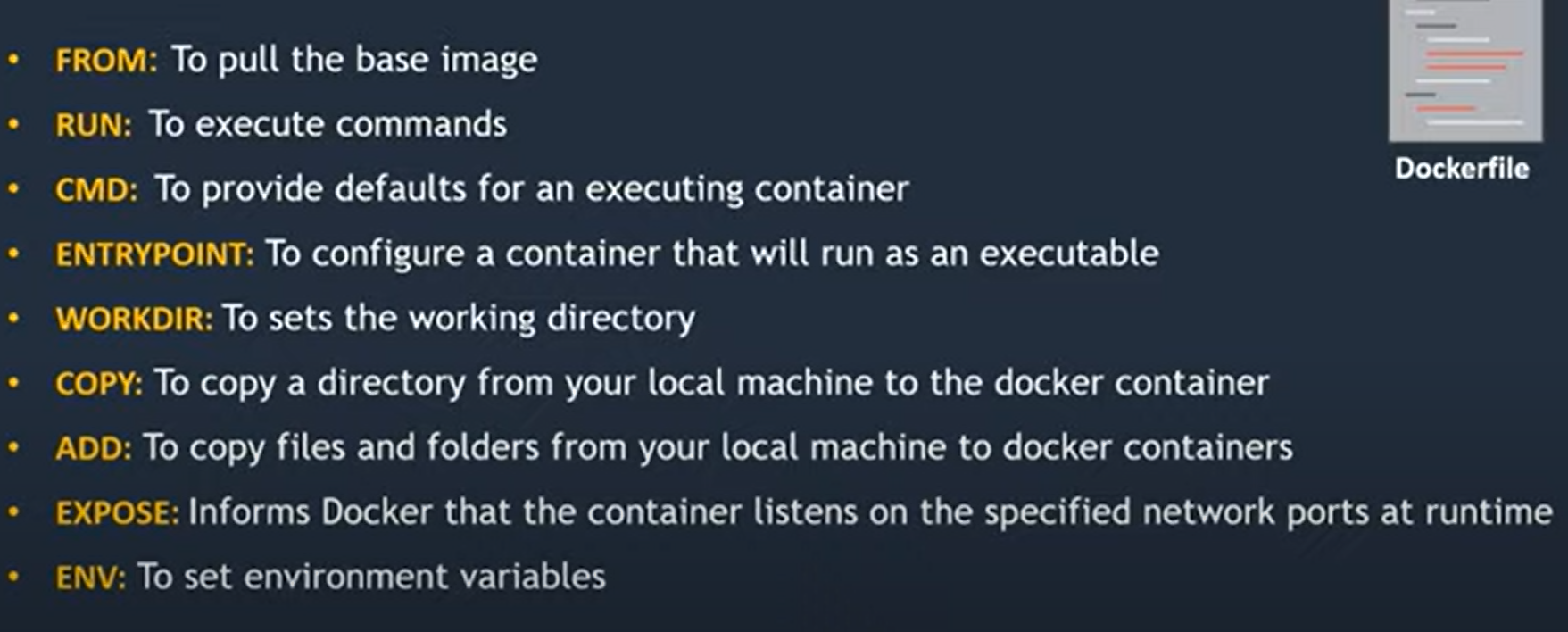
If we r using entry point ,if we not mention any parameter if cmd not there it wont ececute nything simply create container

Build: docker build -t venky:1.0 . 🡪 if file is in same path

Docker build -t venky -f /opt/dockerfile 🡪 if docker file is in different path we have to specify the path

Run: docker run -d –name demo.container -p 8082:8080 venky:1.0

8080 exposed in docker file so in docker hub it accessed 8082 here 8080 send request to 8082



TOMCAT FILE

Tomcat server deployment

FROM ubuntu

MAINTAINER venkyy82@gmail.com

RUN mkdir /opt/tomcat/

RUN chmod 755 /opt/tomcat

WORKDIR /opt/tomcat

ADD apache-tomcat-10.0.27.tar.gz /opt/tomcat

RUN apt update

RUN apt install default-jdk -y

RUN java -version

WORKDIR /opt/tomcat/apache-tomcat-10.0.27/webapps

ADD SampleWebApp.war /opt/tomcat/apache-tomcat-10.0.27/webapps

RUN /opt/tomcat/apache-tomcat-10.0.27/bin/startup.sh start

EXPOSE 8080

#CMD ["/opt/tomcat/apache-tomcat-10.0.27/bin/startup.sh", "start"]

nginx file

#

# NOTE: THIS DOCKERFILE IS GENERATED VIA "update.sh"

#

# PLEASE DO NOT EDIT IT DIRECTLY.

#

FROM debian:bullseye-slim

LABEL maintainer="NGINX Docker Maintainers <docker-maint@nginx.com>"

ENV NGINX\_VERSION 1.23.2

ENV NJS\_VERSION 0.7.7

ENV PKG\_RELEASE 1~bullseye

RUN set -x \

# create nginx user/group first, to be consistent throughout docker variants

&& addgroup --system --gid 101 nginx \

&& adduser --system --disabled-login --ingroup nginx --no-create-home --home /nonexistent --gecos "nginx user" --shell /bin/false --uid 101 nginx \

&& apt-get update \

&& apt-get install --no-install-recommends --no-install-suggests -y gnupg1 ca-certificates \

&& \

NGINX\_GPGKEY=573BFD6B3D8FBC641079A6ABABF5BD827BD9BF62; \

found=''; \

for server in \

hkp://keyserver.ubuntu.com:80 \

pgp.mit.edu \

; do \

echo "Fetching GPG key $NGINX\_GPGKEY from $server"; \

apt-key adv --keyserver "$server" --keyserver-options timeout=10 --recv-keys "$NGINX\_GPGKEY" && found=yes && break; \

done; \

test -z "$found" && echo >&2 "error: failed to fetch GPG key $NGINX\_GPGKEY" && exit 1; \

apt-get remove --purge --auto-remove -y gnupg1 && rm -rf /var/lib/apt/lists/\* \

&& dpkgArch="$(dpkg --print-architecture)" \

&& nginxPackages=" \

nginx=${NGINX\_VERSION}-${PKG\_RELEASE} \

nginx-module-xslt=${NGINX\_VERSION}-${PKG\_RELEASE} \

nginx-module-geoip=${NGINX\_VERSION}-${PKG\_RELEASE} \

nginx-module-image-filter=${NGINX\_VERSION}-${PKG\_RELEASE} \

nginx-module-njs=${NGINX\_VERSION}+${NJS\_VERSION}-${PKG\_RELEASE} \

" \

&& case "$dpkgArch" in \

amd64|arm64) \

# arches officialy built by upstream

echo "deb https://nginx.org/packages/mainline/debian/ bullseye nginx" >> /etc/apt/sources.list.d/nginx.list \

&& apt-get update \

;; \

\*) \

# we're on an architecture upstream doesn't officially build for

# let's build binaries from the published source packages

echo "deb-src https://nginx.org/packages/mainline/debian/ bullseye nginx" >> /etc/apt/sources.list.d/nginx.list \

\

# new directory for storing sources and .deb files

&& tempDir="$(mktemp -d)" \

&& chmod 777 "$tempDir" \

# (777 to ensure APT's "\_apt" user can access it too)

\

# save list of currently-installed packages so build dependencies can be cleanly removed later

&& savedAptMark="$(apt-mark showmanual)" \

\

# build .deb files from upstream's source packages (which are verified by apt-get)

&& apt-get update \

&& apt-get build-dep -y $nginxPackages \

&& ( \

cd "$tempDir" \

&& DEB\_BUILD\_OPTIONS="nocheck parallel=$(nproc)" \

apt-get source --compile $nginxPackages \

) \

# we don't remove APT lists here because they get re-downloaded and removed later

\

# reset apt-mark's "manual" list so that "purge --auto-remove" will remove all build dependencies

# (which is done after we install the built packages so we don't have to redownload any overlapping dependencies)

&& apt-mark showmanual | xargs apt-mark auto > /dev/null \

&& { [ -z "$savedAptMark" ] || apt-mark manual $savedAptMark; } \

\

# create a temporary local APT repo to install from (so that dependency resolution can be handled by APT, as it should be)

&& ls -lAFh "$tempDir" \

&& ( cd "$tempDir" && dpkg-scanpackages . > Packages ) \

&& grep '^Package: ' "$tempDir/Packages" \

&& echo "deb [ trusted=yes ] file://$tempDir ./" > /etc/apt/sources.list.d/temp.list \

# work around the following APT issue by using "Acquire::GzipIndexes=false" (overriding "/etc/apt/apt.conf.d/docker-gzip-indexes")

# Could not open file /var/lib/apt/lists/partial/\_tmp\_tmp.ODWljpQfkE\_.\_Packages - open (13: Permission denied)

# ...

# E: Failed to fetch store:/var/lib/apt/lists/partial/\_tmp\_tmp.ODWljpQfkE\_.\_Packages Could not open file /var/lib/apt/lists/partial/\_tmp\_tmp.ODWljpQfkE\_.\_Packages - open (13: Permission denied)

&& apt-get -o Acquire::GzipIndexes=false update \

;; \

esac \

\

&& apt-get install --no-install-recommends --no-install-suggests -y \

$nginxPackages \

gettext-base \

curl \

&& apt-get remove --purge --auto-remove -y && rm -rf /var/lib/apt/lists/\* /etc/apt/sources.list.d/nginx.list \

\

# if we have leftovers from building, let's purge them (including extra, unnecessary build deps)

&& if [ -n "$tempDir" ]; then \

apt-get purge -y --auto-remove \

&& rm -rf "$tempDir" /etc/apt/sources.list.d/temp.list; \

fi \

# forward request and error logs to docker log collector

&& ln -sf /dev/stdout /var/log/nginx/access.log \

&& ln -sf /dev/stderr /var/log/nginx/error.log \

# create a docker-entrypoint.d directory

&& mkdir /docker-entrypoint.d

COPY docker-entrypoint.sh /

COPY 10-listen-on-ipv6-by-default.sh /docker-entrypoint.d

COPY 20-envsubst-on-templates.sh /docker-entrypoint.d

COPY 30-tune-worker-processes.sh /docker-entrypoint.d

ENTRYPOINT ["/docker-entrypoint.sh"]

EXPOSE 80

STOPSIGNAL SIGQUIT

CMD ["nginx", "-g", "daemon off;"]

commands

docker-compose ps -a

1079 mv docker-compose-redis.yml docker-compose.yml

1080 docker-compose ps -a

1081 docker-compose images

1082 docker-compose rmi -f 6057a50d53c1

1083 docker-compose rm opt\_web\_1

1084 docker-compose rmi 6057a50d53c1

1085 docker ps -a

1086 docker rm -f 738c3128a798 b05156bd2ea2

1087 docker ps -a

1088 docker images

1089 docker rmi -f 6057a50d53c1 0dc038a2db82 5d58c024174d 216c552ea5ba e7bb3280b4c7 feb5d9fea6a5

1090 clear

1091 docker images

1092 docker ps -a

1093 docker images

1094 docker-compose ps -a

1095 docker-compose images

1096 clear

1097 ls -la

1098 cat docker-compose.yml

1099 clear

1100 docker-compose up -d

1101 docker-compose images

1102 docker-compose ps -a

1103 clear

1104 ls -la

1105 clear

1106 docker-compose images

1107 vim docker-compose.yml

1108 docker-compose ps -a

1109 docker-compose down

1110 docker-compose ps -a

1111 docker-compose images

1112 clear

1113 docker-compose up -d --scale web=3

1114 vim docker-compose.yml

1115 clear

1116 docker-compose up -d --scale web=3

1117 docker-compose ps -a

1118 docker-compose images

1119 docker-compose stop opt\_web\_2

1120 docker-compose up -d --scale web=2

1121 docker-compose ps -a

1122 clear

1123 docker-compose ps -a

1124 docker-compose stop -t opt\_web\_1

1125 docker-compose stop redis

1126 docker-compose stop opt\_database\_1

1127 docker-compose service

1128 clecar

1129 clear

1130 docker ps -a

1131 docker-compose ps -a

1132 docker stop 827975ea616f

1133 docker-compose ps -a

1134 clear

1139 cp docker-compose.yml docker-compose-redis.yml

1140 vim docker-compose-redis.yml

1141 clear

1142 docker-compose up -f docker-compose-redis.yml -d

1143 docker-compose up --scale=2 -f docker-compose-redis.yml

1146 docker-compose -f docker-compose-redis.yml down

1147 vim docker-compose.yml

1149 docker-compose -f docker-compose-redis.yml down

1150 docker-compose images

1151 docker-compose -f docker-compose-redis.yml up

1152 docker-compose -f docker-compose-redis.yml -f docker-compose.yml down

1154 docker-compose images

1155 docker-compose ps -a

1156 docker-compose -f docker-compose-redis.yml -f docker-compose.yml up

1158 docker-compose -f docker-compose.yml -f docker-compose-redis.yml up

1160 rm -rf docker-compose-redis.yml

1165 docker-compose up --scale=2 -f docker-compose.yml

1168 docker-compose up -d --scale web=2

1169 docker-compose up --scale=2 -f docker-compose.yml web=2

1170 -f docker-compose.yml

1178 docker-compose run

1179 docker-compose run -d

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

version: "3"

services:

web:

image: nginx

ports:

- "8081-8083:80"

database:

image: redis

docker file

#Tomcat server deployment

FROM ubuntu

MAINTAINER venkyy82@gmail.com

RUN apt update -y && apt install nginx -y

EXPOSE 80

CMD ["/usr/sbin/nginx", "-g", "daemon off;"]

Docker compose

version: "3"

services:

web:

image: nginx

ports:

- "8081-8083:80"

database:

image: redis