**Docker Installation**

Download from docker website and run the setup

sudo apt-get update

sudo apt-get install -y docker.io

**Checking Docker Service is running**

sudo systemctl status docker

sudo systemctl start docker && sudo systemctl enable docker

**Creating Docker Image, Run Container and Connecting to Container**

We need four command for that

create Create container from image

ps check the run containers

start start container

attach attach the I/O

rm remove the container

**CREATION**

**docker create -it ubunt:16.04 bash**

above command will download the ubuntu image to your system

**docker run –name webserver -v $(pwd):/usr/share/nginx/html -d -p 8080:80 nginx**

**docker run -it ubuntu bash**

**docker run hello-world testing image**

docker run basic command

--name give any name to image

-v mapping local directory to container directory

-d run container in detached mode

-p mapping port of local machine to container

here 8080 is representing local and 80 is for container

local:container

Nginx image name from docker hub

**docker pull ubuntu:latest**

above command will pull latest ubuntu images to local system

CHECKING

docker ps -a

above command will list all downloaded images in system

STARTING CONTAINER

docker start image\_id

above command is used to start particular image with ID

LOGIN TO DOCKER / CONNECTING TO DOCKER

docker attach image\_id

above command will start image and connect to container

LOGOUT FROM CONTAINER

exit

above command used to come out from container

DELETING IMAGES

docker kill contanair\_id

above command will stop the container

docker rm container\_id

above command is used to delete the container

docker rmi image\_id

above command is used to delete the image

MAPING LOCAL DIRECTORY TO DOCKER IMAGE AT CREATION TIME

docker create -it -v $(pwd): /var/www/ ubuntu:16.04 bash

above command will map current working directory to container /var/www directory

STARTING AND CREATING IMAGE AT SAME TIME

docker run -it -d ubuntu:16.04 bash

above command combine the start and create command

STOPING THE CONTAINER

docker stop image\_id

above command will stop the container

CREATING A DOCKERFILE

create a file with name Dockerfile without any extension and write following code

# Dockerfile

FROM nginx:alpine # this specify the image name

VOLUME /usr/share/nginx/html #map container directory

EXPOSE 80 #map the port 80 of container

Now runt the following command from Dockerfile directory

docker build . -t webserver:v1

the . is specify the Dockerfile location

-t is used to tagged the image with name webserver:v1

With this we are not pulling image from docker hub we create our own images

To see all images, run below command

docker images

now run below command to build the image and run the container

docker run -v $(pwd):/usr/share/nginx/html -d -p 8080:80 webserver:v1

to automate the above command, then we need docker compose

install docker-compose from docker website

check docker-compose version

docker-compose –verison

now create another file with name docker-compose.yml and enter following code

# docker-compose.yml

version: ‘2’

services:

webserver:

build: .

ports:

* “8080:80”

volumes:

* .:/usr/share/nginx/html

Save file and run below command

Docker-compose up (-d)

-d means in detached mode

docker-comose ps

docker-compose stop

docker commit container\_id

above command will save the container as a new image

docker tag container\_id any\_name

above command will give name to container

docker logs container\_id

above command will show what’s going wrong with container

**ctrl + p then ctrl + q** command is used to exit from docker attach command keep the container still running state

if the container is running in one windows, and we want another session to it then run

command

docker exec -ti container\_id bash

above command will connect to same container, if one session end, another also end

**DOCKER FLOW**

Docker Images 🡪 Container 🡪 Stop Container 🡪 Commit Container

**NETWORKING BETWEEN CONTAINERS**

**VOLUMES / SHARING DATA BETWEEN CONTAINER AND HOST & CONTAINERS**

Volumes are shared folder. Used to share data between container and host and

containers.

**Two types of Volumes are there**

1. Permanent if container delete, volumes on host still exist
2. Ephemeral if container delete, volumes also delete

**Note:**

Volumes are not part of images

**Sharing Database between Host & Container**

Create a folder using mkdir command

Then spin a container using below command to share the volume

**docker run -ti -v /path\_of\_directory\_on\_host:/path\_on\_container ubuntu bash**

**Sharing Database between Containers**

**docker run -ti –name server1 -v /share\_data ubuntu bash**

above command create volume with name share\_data but not sharing with host

**docker run -ti --name server2 --volumes-from server1 ubuntu bash**

above command will share the volumes of server1 with server2 container

**Note:**

Volumes shared between container is example of ephemeral

DOCKER REGISTERIES

docker repository are location from where we can download or pull docker

images.

Searching Images in docker repository

docker search image\_name

Login to docker from command line

docker login

docker pull ubuntu

docker push image\_id

above command will push the image to dockerhub.io

DOCKER FILES

Dockerfile is a small program to create image

**docker build -t name-of-image .**

. represent the location of Dockerfile

FROM image\_name

MAINTNER Rizwan khan <rizwan@gmail.com>

RUN echo “any command need to run on container”

CMD echo “hello container”

ADD used to add a local file

ADD run.sh /run.sh

ENV ENV DB\_HOST=db.production.example.com

ENV DB\_PORT=1234

ENTRYPOINT

EXPOSE EXPOSE 8080 use for port mapping

VOLUME VOLUME [‘/host/path/’ “/container/path/”]

VOLUME [“/shared-data”]

WORKDIR /install/

USER Rizwan

<https://www.docs.docker.com/engine/reference/builder/>

for more commands

Save above file with name Dockerfile and run below command

**docker build -t hello .**