**Docker**

Docker is a containerization tool.

Virtualization -- Fixed hardware allocation.

Containerization - No Fixed Hardware

Process isolation ( Dependency in os is removed )

+++++++++++++++++++++++++

In comparison to the traditional virtualization functionalities of hypervisors,

Docker containers eliminate the need for a separate guest operating system for every new virtual machine.

Docker implements a high-level API to provide lightweight containers that run processes in isolation.

A Docker container enables rapid deployment with minimum run-time requirements. It also ensures better management and simplified portability.

This helps developers and operations team in rapid deployment of an application.

+++++++++++++++++++++++++

Dowload DockerDesktop from google

# docker --version

We should be comformatable with four terms

1) Docker Images

Combinations of binaries / libraries which are necessary for one software application.

2) Docker Containers

When image is executed comes into running condition, it is called container.

3) Docker Host

Machine on which docker is installed, is called as Docker host.

4) Docker Client

Terminal used to run docker run commands ( Git bash/power shell /putty/cmd prompt )

+++++++++++

Docker Commands

--------------------

Working on Images

-------------------------

1 To download a docker image

docker pull image\_name (hub.docker.com)

2 To see the list of docker images

docker image ls

(or)

docker images

3 To delete a docker image from docker host

docker rmi image\_name/image\_id

4) To upload a docker image into docker hub

docker push image\_name

5) To tag an image

docker tag image\_name ipaddress\_of\_local\_registry:5000/image\_name

or

docker tag image\_name dockerhub id /image\_name

6) To build an image from a customised container

docker commit container\_name/container\_id new\_image\_name

7) To create an image from docker file

docker build -t new\_image\_name

8) To search for a docker image

docker search image\_name

9) ) To delete image

Docker rmi image\_name /image\_id

10) To delete all images that are not attached to containers

docker system prune -a

++++++++++++++++++++++++++++++++++++++++++++++

Working on containers

-----------------------------

1) To see the list of all running continers

docker container ls

2) To see the list of running and stopped containers

docker ps -a

3) To start a container

docker start container\_name/container\_id

4) To stop a running container

docker stop container\_name/container\_id

5) To restart a running container

docker restart container\_name/container\_id

To restart after 10 seconds

docker restart -t 10 container\_name/container\_id

6) To delete a stopped container

docker rm container\_name/container\_id

7) To delete a running container

docker rm -f container\_name/container id

8) To stop all running containers

docker stop $(docker ps -aq)

9) To restart all containers

docker restart $(docker ps -aq)

20) To remove all stopped containers

docker rm $(docker ps -aq)

21) To remove all contianers(running and stopped)

docker rm -f $(docker ps -aq)

22) To see the logs generated by a container

docker logs container\_name/container\_id

23) To see the ports used by a container

docker port container\_name/container\_id

++++++++++++++++++++++++++++++++++++++++++++++++

Run command options

-it for opening an interactive terminal in a container

--name Used for giving a name to a container

-d Used for running the container in detached mode as a background process

-p Used for port mapping between port of container with the dockerhost port.

-P Used for automatic port mapping ie, it will map the internal port of the container

with some port on host machine.

This host port will be some number greater than 3000

-v Used for attaching a volume to the container

--link Used for linking the container for creating a multi container architecture

-e Used for passing environment varaibles to the container

++++++++++++++++++++++++++++++++++++++++++++++++++=

# docker images ( There are no images )

To download tomcat image

# docker pull tomee

# docker images

# docker pull ubuntu

If you do not specify the version, by default, we get latest version

I want to download jenkins

# docker pull jenkins

**To create a container from an image**

# docker run --name c1 -p 7070:8080 tomee

**To check the tomcat is running or not**

localhost:7070

( 7070 is port number mapped in docker host)

Lets remove the container ( Open another gitbash/powershell terminal)

# docker stop c1

# docker rm -f c1

# docker run --name mytomcat -p 7070:8080 **-d** tomee

( The above command runs tomcat in detached mode , so we get out # prompt back )

# docker container ls

To start jenkins

# docker run --name myjenkins -p 9090:8080 -d jenkins/jenkins

To check for jenkins (Open browser )

http://localhost:9090

To create ubuntu container

# docker run --name myubuntu -it ubuntu

Observation: You have automatically entered into ubuntu

# ls ( To see the list of files in ubuntu )

# exit ( To comeout of container back to host )

+++++++++++++

Case 1:

Start tomcat as a container and name it as "webserver". Perform port mapping and run this container in detached mode

# docker run --name webserver -p 7070:8080 -d tomee

To access homepage of the tomcat container

Launch any browser

http://localhost:7070

++++++++++++++++++++++++++++++++

Case 2:

Start jenkins as a container in detached mode , name is as "devserver", perform port mapping

# docker run -d --name devserver -p 9090:8080 jenkins

To access home page of jenkins ( In browser)

http://localhost:9090

++++++++++++++++++++++++++++++++++++++++

Case 3: Start nginx as a container and name as "appserver", run this in detached mode , perform automatic port mapping

Generally we pull the image and run the image

Instead of pulling, i directly

# docker run --name appserver -P -d nginx

( if image is not available, it perform pull operation automatically )

( Capital P , will perform automatic port mapping )

How to check nginx is running or not? ( we do not know the port number)

To know the port that is reserved for nginx )

# docker port appserver

80/tcp -> 0.0.0.0:49153(for example)

80 is nginx port

49153 is dockerhost port

or

# docker container ls ( to see the port of nginx and docker host )

To check nginx on browser

http://localhost:49153

++++++++++++++++++++++++++

To start centos as container

# docker run --name mycentos -it centos

# exit ( To come back to dockerhost )

++++++++++++++++++++++++++++++++++++++++++++

Creating Customized docker images

When ever docker container is deleted all the software that have installed within the container will also deleted

If we can save the container as an images ,then we can preserve the software

The creation of customized docker images can be done in two ways

1.Using docker commit command

2.Using dockerfile (more powerfull)

Using docker commit

1. > docker run –name c11 -it ubuntu

Update apt repository container not docker fost

1. > docker commit c11 myubuntu (user defined name)
2. Check docker images ls

Myubuntu

1. Run the customizes images

* docker run –name c22 -it myubuntu
* #git –version
* #exit

1. Pushing image into Dockerhub

* docker login –usernsme
* password
* docker tag <image name> myubuntu archanareddy22/myubuntu:latest
* docker push archanareddy22/myubuntu:latest

Docker file

Using docker file

--------------------

This is a simple text file, which uses predefinied keywords for creating customized docker images.

Key words used in docker file ( case sensitive )

1) FROM -- used to specify the base image from which the docker file has to be created.

2) MAINTAINER -- This represents name of the organization or the

author who created this docker file.

3) CMD -- This is used to specify the initial command that should be executed when the container starts.

4) ENTRYPOINT - used to specify the default process that should be executed when container starts.

It can also be used for accepting arguments from the CMD instruction.

5) RUN -- Used for running linux commands within the container. It is generally helpful for installing the software in the container.

6) USER -- used to specify the default user who should login into the container.

7) WORKDIR --

Used to specify default working directory in the container

8) COPY -- Copying the files from the host machine to the container.

9) ADD -- Used for copying files from host to container, it can also be used for downloading files from remote servers.

10) ENV -- used for specifying the environment variables that should be passed to the container.

EXPOSE -- Used to specify the internal port of the container

VOLUME -- used to specify the default volume that should be attached to the container.

LABEL -- used for giving label to the container

STOPSIGNAL -- Used to specify the key sequences that have to be passed in order to stop the container.

+++++++++++++++++++++

1. Create Dockerfile

# step specify the base image

FROM ubuntu

MAINTAINER archanareddy

RUN apt-get update

#setup startup command

CMD [ "echo","Hello" ]

Write the code in notepad and save as dockerfie

1. Now in powershell get into the path pf dockerfile

* docker build -t <image name> .
* docker run <image name>
* docker tag <imagename> archanareddy22/<hub image name>
* docker push