Kaushik Dey

Kaushikdey1984@yahoo.com

docker volumes,images, build with docker hub

Docker documentation

To check the volume

[root@ip-172-31-21-247 ~] # docker volume list

DRIVER VOLUME NAME

To create the volume

[root@ip-172-31-21-247 ~] # docker volume create kaushikvol1

kaushikvol1

[root@ip-172-31-21-247 ~] # docker volume list

DRIVER VOLUME NAME

local kaushikvol1

To inspect the volume

[root@ip-172-31-21-247 ~] # docker volume inspect kaushikvol1

[

{

"CreatedAt": "2023-05-03T01:21:29Z",

"Driver": "local",

"Labels": {},

"Mountpoint": "/var/lib/docker/volumes/kaushikvol1/\_data",

"Name": "kaushikvol1",

"Options": {},

"Scope": "local"

}

]

[root@ip-172-31-21-247 ~] #

docker run -it --mount source=kaushikvol1,destination=/kaushikvol1 centos bash

[root@ip-172-31-45-37 ec2-user]# docker volume list

DRIVER VOLUME NAME

local kaushikvol1

[root@ip-172-31-45-37 ec2-user]# docker inspect kaushikvol1

[

{

"CreatedAt": "2023-05-08T02:23:43Z",

"Driver": "local",

"Labels": {},

"Mountpoint": "/var/lib/docker/volumes/kaushikvol1/\_data",

"Name": "kaushikvol1",

"Options": {},

"Scope": "local"

}

]

[root@ip-172-31-45-37 ec2-user]# cd /var/lib/docker/volumes

[root@ip-172-31-45-37 volumes]# ls

backingFsBlockDev kaushikvol1 metadata.db

[root@ip-172-31-45-37 volumes]# history

* clear
* docker --version
* yum install docker -y
* clear
* docker --version
* docker version
* systemctl status docker
* systemctl start docker
* systemctl status docker
* systemctl enable docker
* clear
* docker version
* clear
* docker images
* docker ps
* docker ps -a
* docker run ubuntu
* docker images
* docker ps -a
* docker ps
* docker pull nginx
* clear
* docker images
* docker run -it centos bash
* clear
* docker images
* docker volume list
* docker volume create kaushikvol1
* docker volume list
* docker volume inspect kaushikvol1
* docker volume list
* docker inspect kaushikvol1
* cd /var/lib/docker/volumes
* ls
* history

[root@ip-172-31-45-37 volumes]# cd..

bash: cd..: command not found

[root@ip-172-31-45-37 volumes]# cd ..

[root@ip-172-31-45-37 docker]# cd ..

[root@ip-172-31-45-37 lib]# cd

[root@ip-172-31-45-37 ~]# pwd

/root

[root@ip-172-31-45-37 ~]# cd /var/lib/docker/volumes

[root@ip-172-31-45-37 volumes]# pwd

/var/lib/docker/volumes

[root@ip-172-31-45-37 volumes]# ls

backingFsBlockDev kaushikvol1 metadata.db

[root@ip-172-31-45-37 volumes]# cd kaushikvol1

[root@ip-172-31-45-37 kaushikvol1]# ls

\_data

[root@ip-172-31-45-37 kaushikvol1]# pwd

/var/lib/docker/volumes/kaushikvol1

[root@ip-172-31-45-37 kaushikvol1]# cd \_data

[root@ip-172-31-45-37 \_data]# pwd

/var/lib/docker/volumes/kaushikvol1/\_data

[root@ip-172-31-45-37 \_data]# ls

fileformcontainer.txt

[root@ip-172-31-45-37 \_data]# echo "newvol1" >> propertyfile.txt

[root@ip-172-31-45-37 \_data]# ls

fileformcontainer.txt propertyfile.txt

[root@ip-172-31-45-37 \_data]# ll

total 8

-rw-r--r-- 1 root root 25 May 8 02:23 fileformcontainer.txt

-rw-r--r-- 1 root root 8 May 8 03:24 propertyfile.txt

[root@ip-172-31-45-37 \_data]# docker run -it --mount source=kaushikvol1,destination=/kaushikvol1 centos bash

[root@94baf3926469 /]# ls

bin etc kaushikvol1 lib64 media opt root sbin sys usr

dev home lib lost+found mnt proc run srv tmp var

[root@94baf3926469 /]# cd kaushikvol1

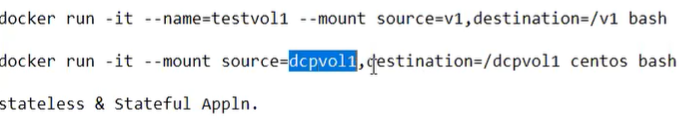
[root@94baf3926469 kaushikvol1]# ls

fileformcontainer.txt propertyfile.txt

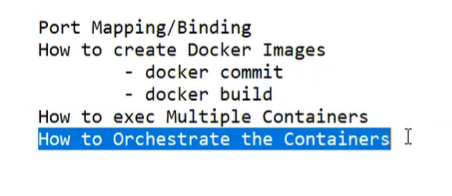
[root@94baf3926469 kaushikvol1]#

Please note that :

Container also running in state. In such scenario we required external volume.



With the help of orchestration tool we can add volumes, external volumes and clear the volume



PORT MAPPING & BINDING

Run tomcat with port mapping

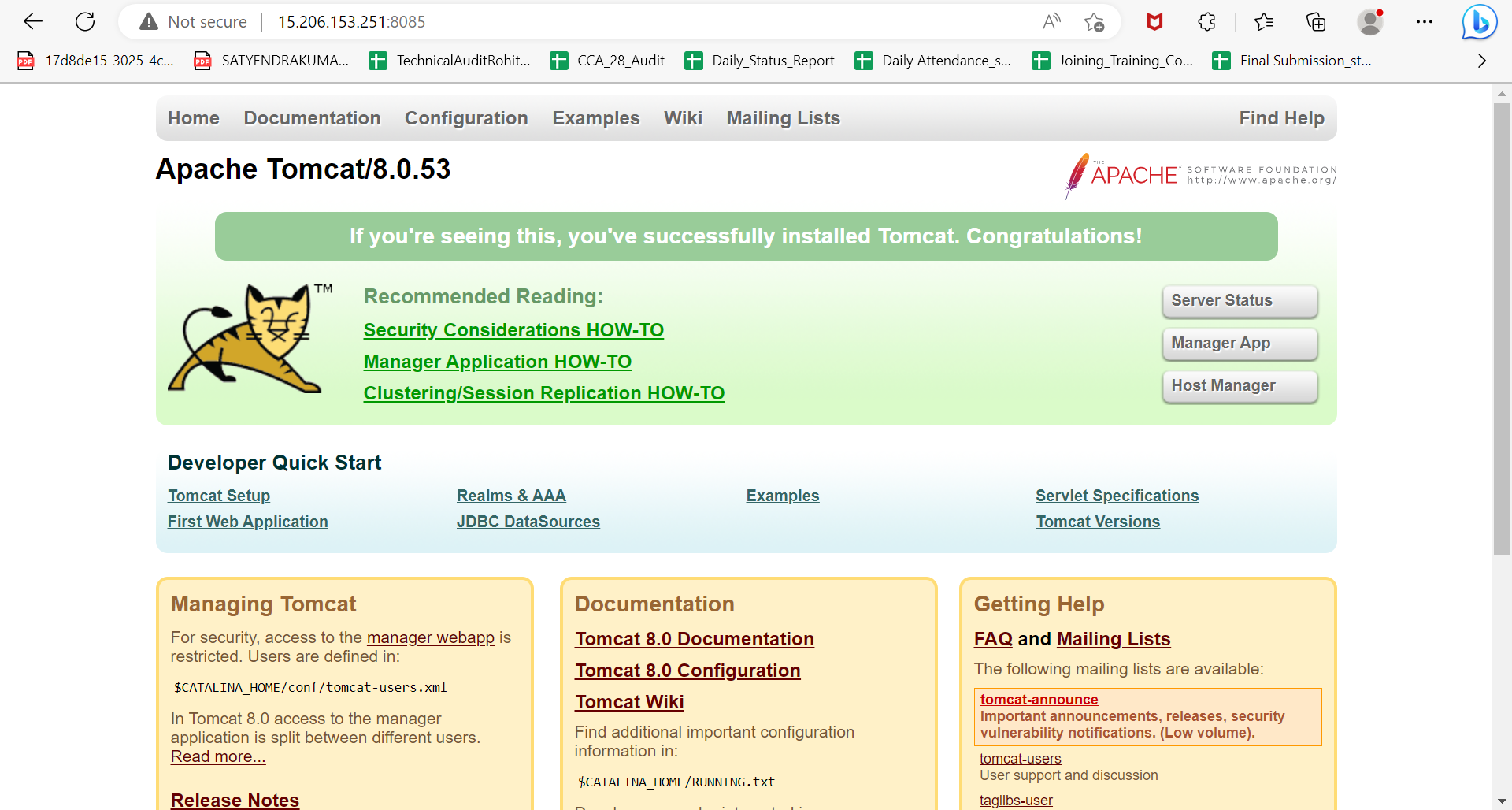
Host\_port:container\_port

A picture containing shape

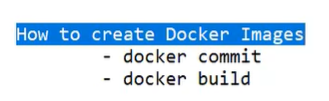
Description automatically generated

docker run -it -p 8085:8080 tomcat:8.0

in this demo we never install tomcat in our VM and it is running inside the container so, that’s why port mapping and binding is required.



DOCKER IMAGES



[root@ip-172-31-45-37 ec2-user]# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

debian latest 34b4fa67dc04 5 days ago 124MB

nginx latest 6efc10a0510f 3 weeks ago 142MB

ubuntu latest 08d22c0ceb15 2 months ago 77.8MB

centos latest 5d0da3dc9764 19 months ago 231MB

tomcat 8.0 ef6a7c98d192 4 years ago 356MB

[root@ip-172-31-45-37 ec2-user]# docker run -it debian

root@83f56f24030b:/# ls

bin dev home lib64 mnt proc run srv tmp var

boot etc lib media opt root sbin sys usr

root@83f56f24030b:/# git --version

bash: git: command not found

root@83f56f24030b:/# apt-get update && apt-get install -y git

Install inside that debian container

root@83f56f24030b:/# git –version

git version 2.30.2

in this debian container we added git install, so we can create our custom images from this debian container, so the command is

[root@ip-172-31-45-37 ec2-user] # docker commit 83f56f24030b iimdatascience1/dcpdebian-git:v1.0

[root@ip-172-31-45-37 ec2-user] # docker images

[root@ip-172-31-45-37 ec2-user] # docker run -it iimdatascience1/dcpdebian-git:v1.0

DOCKER Files

[root@ip-172-31-45-37 ec2-user] # mkdir dcp-docker

[root@ip-172-31-45-37 ec2-user] # cd dcp-docker

[root@ip-172-31-45-37 dcp-docker] # vi Dockerfile

[root@ip-172-31-45-37 dcp-docker] # cat Dockerfile

[root@ip-172-31-45-37 dcp-docker] # docker build -t iimdatascience1/dcp-deb-git-vim .

Successfully built 2c2aeb328b36

Successfully tagged iimdatascience1/dcp-deb-git-vim:latest

[root@ip-172-31-45-37 dcp-docker] # docker images

Text

Description automatically generated



PUBLISH DOCKER IMAGES TO DOCKER HUB

Before login inside the docker hub from docker daemon first we need to create access token and with the help of access token we should connect with docker hub.

Step 1 : login in dockerhub

<https://hub.docker.com/>

Graphical user interface, text, application, email

Description automatically generated

Step 2 : Now we should go with account settings, the screenshot is given below.

Graphical user interface, application

Description automatically generated

Step 3 : Now we should create access token , the screenshot is given below.

Graphical user interface, application

Description automatically generated

Step 4 : Now click on New Access Token, the screenshot is given below.

Graphical user interface, text, application, email

Description automatically generated

Step 5 : now we should run with following commands to see the effects.

[root@ip-172-31-45-37 dcp-docker]# docker login -u iimdatascience1

Password: dckr\_pat\_XR7\_NEn8wxykHbQMEe4LWYOph\_Y

WARNING! Your password will be stored unencrypted in /root/.docker/config.json.

Configure a credential helper to remove this warning. See

https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded

[root@ip-172-31-45-37 dcp-docker] #

[root@ip-172-31-45-37 dcp-docker] # docker push iimdatascience1/dcpdebian-git:v1.0

The push refers to repository [docker.io/iimdatascience1/dcpdebian-git]

4d72adc2489a: Pushed

d925e0fae4e6: Mounted from library/debian

v1.0: digest: sha256:4bbcb5e5e26be0987f3dcbce08bb2845d7223effcc009c605fb8961af2ae3c95 size: 741

[root@ip-172-31-45-37 dcp-docker] #

Now, in our docker hub repo our images is pushed , the screenshot is given below.

Graphical user interface, text, application, email

Description automatically generated

