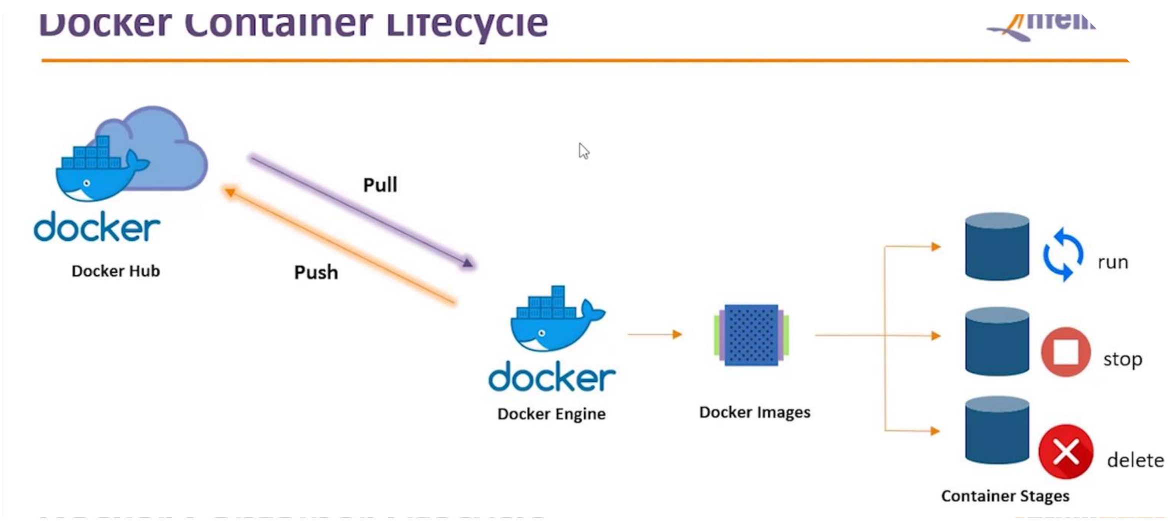


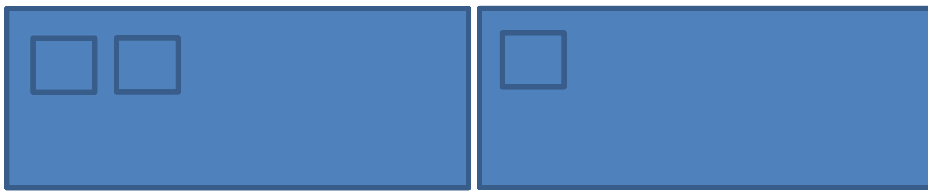
Life cycle of Container

You pull an image from docker hub. When you pull an image, it becomes a container

Container is running

It is stopped

It is removed



COMMON DOCKER COMMANDS

1: This command helps you know the installed version of the docker software on your system

```
$ docker –version
```

2: This command helps you pull images from the central docker repository.

```
$ docker pull <image-name>
```

3. This command helps you in listing all the docker images, downloaded on your system.

```
$ docker images
```

4: This command helps in running containers, from their image name.

```
$ docker run <image-name>
```

5: This command helps in listing all the containers which are running in the system

```
$ docker ps
```

6: If there are any stopped containers, they can be seen by adding the "-a" flag in this command

```
$ docker ps –a
```

7: For logging into/accessing the container, one can use the exec command

```
$ docker exec <container-id>
```

8. For stopping a running container, we use the "stop" command

```
$ docker stop <container-id>
```

9: This command kills the container by stopping its execution immediately. The difference between 'docker kill' and 'docker stop'. 'docker stop' gives the container time to shutdown gracefully, in situations when it is taking too much time for getting the container to stop, one can opt to kill it

```
$ docker kill <container-id>
```

10: To remove a stopped container from the system, we use the "rm" command

```
$ docker rm <container-id>
```

11: To remove an image from the system we use the command "rmi"

```
$ docker rmi <image-id>
```

PUSHING THE CONTAINER ON DOCKERHUB

Step 1: The first step is to login, it can be done using the following command:

```
$ docker login
```

Step 2: Finally, for pushing your container on DockerHub, use the following command

```
$ docker push <username>/<container-id>
```

Step 3: You can verify the push on DockerHub

```
$ docker exec –it<container-id> bash
```

Now anyone, who wants to download this container. Can simply pass the following command:

```
docker pull containername/apache
```

SAVING CHANGES TO A DOCKER CONTAINER

Step 1: Pull the docker container using the command:

```
$ docker pull Ubuntu
```

Step 2: Run the container using the commands

```
$ docker run –it –d Ubuntu
```

Step 3: Access the container using the command:

```
$ docker exec –it<container-id> bash
```

Step 4: Install apache2 on this container, using the following commands

```
$ apt-get update
$ apt-get install apache2
```

Step 5: Exit the container, and save the container using this command. The saved container will be converted into an image with the name specified is

```
$ docker commit <container-id><username>/<container-name>
```

The username must match with the username which you created on dockerhub. The container-name name can be anything.

CREATING A DOCKERFILE

Step 1: First, create a folder docker, in the home directory

```
$ mkdir docker
```

Step 2: Enter into this directory, and create a file called 'Dockerfile', with the same contents as the Sample Dockerfile. Add the following content in the Dockerfile.

```
 $ cd docker
$ sudo nano Dockerfile
```

```
FROM ubuntu
RUN apt-get update
RUN apt-get -y install apache2
ADD . /var/www/html
ENTRYPOINT apachectl -D FOREGROUND
ENV name Devops Learning
```

Step 3: Create one more file called, index.html with the following contents can verify the push on DockerHub.

$ sudo nano index.html

```
<html>
<body>
Hello World
</body>
</html>
```

Step 4: Now pass the following command

$ docker build <directory-of-dockerfile> -t <name of container>
Sudo docker build . –t newimage

Step 5: Finally, run this built image, using the following command:

$ docker run –it –p 81:80 –d <name of container>

sudo docker run –it –p 81:80 –d newimage

Step 6: Now navigate to the server IP address on port 81

Step 7: Finally, login into the container, and check the variable $name, it will have the same value, as given in the Dockerfile.

Sudo docker run
sudo docker images

```
sudo docker pull nginx
```

```
sudo docker run --name docker-nginx -p 80:80 nginx
```

```
Check in Browser : 80
```

```
sudo docker rm docker-nginx
sudo docker run --name docker-nginx -p 80:80 -d nginx
```

```
sudo docker stop docker-nginx
```