Docker Notes:

How Docker works:

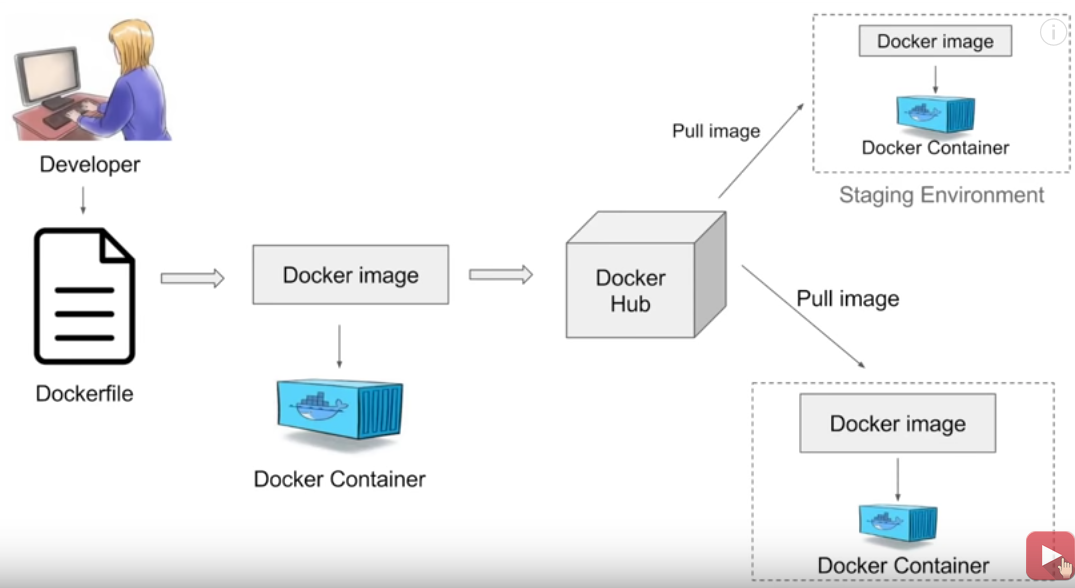
Developer Designs:

Docker File – describes steps to create a Docker image.

Docker Image -all application and its dependencies

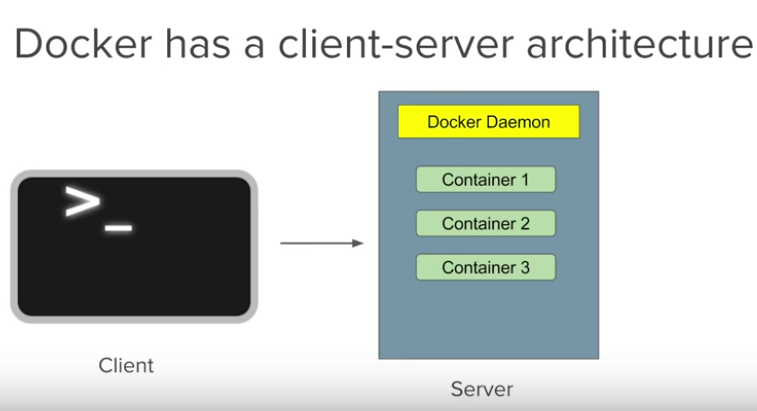
Docker Container- is what you get when you run the Docker Image

Docker Hub -Has a lot of publicly available images.



Virtualization vs Containerization

Docker has a client-server architecture



The daemon(sever) receives the commands from the Docker client through CLI or REST API’s

Docker client and daemon can be present on the same host(machine) or different hosts.

To install docker:

<https://docs.docker.com/v17.12/toolbox/toolbox_install_windows/>

Basic :

docker version

docker -v || docker -version : docker version number

docker info : will give information about the number of container/running/paused/stopped etc.

docker --help: eg. docker images[command] --help

docker login: use it to login into Docker Hub(need to create an account, similar to GIT)

———————————— Images

docker images

docker pull : you can use this to pull an image from Docker Hub

docker rmi: remove one or more images [use docker images -q to get the image id you want to remove and then use docker rmi ‘imageid’]

————————————

Containers :

docker ps : show information about the container

docker run : run a container [eg docker run imagename, if you don’t have the the image then it will down load from docker hub ]

docker run -it unbuntu [this will take us inside ubuntu]

docker start : start container

docker stop: stop the container

————————————

System :

docker stats : memory usage, containers running etc.

docker system df :

docker system prune: remove unused data(anything that is stopped) You will be able to reclaim space.

**Cassandra on Docker : Do this by pulling the Docker Images from Docker Hub**

***Step 1. Docker login***

***Step 2. Pull the DataStax Server image***- this has Apache Cassandra with additional capabilities of Search engine, Spark Analytics and Graph components (configurable at docker run)

docker pull datastax/dse-server:latest

***Step 3. Pull DataStax Studio Image***- notebook based development tool for data exploration, data modeling, data visualization, and query profiling. Studio also has the ability to save, import and export notebooks

docker pull datastax/dse-studio:latest

***Step 4. Run container***

docker run command to create new containers from pulled images. Once the container is created you won’t have to perform the run command again (i.e. use docker start/stop container)

***Step 5. Start the DataStax Server Container***

As stated before, the DataStax distribution comes with some additonal integrations for building different models, making it highly sought after for implementing domain driven design patterns.

* The -g flag starts a Node with Graph Model enabled
* The -s flag starts a Node with Search Engine enabled
* The -k flag starts a Node with Spark Analytics enabled

docker run -e DS\_LICENSE=accept --memory 4g --name my-dse -d datastax/dse-server -g -s -k

***Step 6. Start DataStax Studio Container***

The -link parameter provides a way to map a hostname to a container IP address. In this example, we map the database container to Studio container by providing its name, ‘my-dse’. Now Studio can connect to the database using the container name instead of an IP address. (can also do user-defined bridge)

The -p flag is for mapping ports between container and host. The 9091 port is the default address for Studio

docker run -e DS\_LICENSE=accept --link my-dse -p 9091:9091 --memory 1g --name my-studio -d datastax/dse-studio

**Step 5. Connecting Studio**

* Visit the Studio page that is now hosted on your docker container by entering http://localhost:9091 in your browser.
* Select the “Working with CQL” notebook
* When opening the notebook you will see a connection exception. This is because the default connection in studio uses localhost. You will need to change localhost to the DataStax Server Container name ‘*my-dse*’.
* Select the “*Edit Connection*” button to modify the Host/IP connection field from localhost to the DSE Server Container‘s name, *’my-dse’*. Finish by performing a *Test.* If successful, *s*ave the new connection settings.

**Cassandra Docker Cheat Sheet**

**========= Status =========**#Active containers  
$> docker ps

#Container Utilization  
$> docker stats

#Container Details  
$> docker inspect my-dse

#NodeTool Status  
$> docker exec -it my-dse nodetool status

**========== Logs ==========**#Server Logs  
$> docker logs my-dse

#System Out  
$> docker exec -it my-dse cat /var/log/cassandra/system.log

#Studio Logs  
$> docker logs my-studio

**==== Start/Stop/Remove ====**#Start Container  
$> docker start my-dse

#Stop Container  
$> docker stop my-dse

#Remove Container  
$> docker remove my-dse

**======= Additional =======**#Contaier IPAddress  
&> docker inspect my-dse | grep IPAddress

#CQL (Requires IPAddress from above)  
$> docker exec -it my-dse cqlsh [IPAddress]

#Bash  
$> docker exec -it my-dse bash

Json file to Cassandra:

<https://docs.datastax.com/en/datastax_enterprise/5.0/datastax_enterprise/graph/dgl/dglJSON.html>

Very Very Important:

<https://academy.datastax.com/content/docker-tutorial>

Video on Setting an using Cassandra

<https://www.youtube.com/watch?v=-kzbMbFzQZw>

Not very useful

<https://www.youtube.com/watch?v=N9QllqXI1sE>

MongoDB better?

<https://realpython.com/introduction-to-mongodb-and-python/>