Docker

Docker is a linux based tool which runs the application in a container (linux VM) with all the required environment to run the application.

To run docker we need a VM setup application which is, Docker Toolbox is used for Windows 10 home and lower versions, for above versions Docker Windows CE is used. CE is custom edition with stable version, Enterprise edition is unstable with latest releases and features like beta version. After successfully installing the docker from docs.docker/toolbox we can run commands

docker version - gives the current version of both docker client and the server, better maintain latest. It also determines that docker engine can communicate with the server

docker info - gives info about everything like how many containers etc. are running, IP address, root dir, operating system which is linux and versions, about docker environment and all config values of the engine.

Docker has so many commands which are increasing day by day, so to avoid the complexity docker introduced the management commands which have the all the commands related to it as child.

For example, “docker run” will run the container, but “docker container run” will make command more understandable. Here container is management command. In CLI enter just “docker” command to get the list of management and normal common commands list and their each info.

Syntax: docker <command> (options) -old way but still works in new versions of docker

docker <command> <sub-command> (options) -new way works in new versions

There two basic things in docker one is the client and other is the server. Docker server runs in VM not in the windows. If we open oracle VM(downloaded during docker installation) we can see the default server is running but in linux VM. So we run docker version cmd in windows cmd it will only give the client info which is docker in windows but error of not finding server. We need to tell it to communicate with the server inside the local VM. To do that we use commands related to “docker-machine”.

Run “docker-machine ls” command it will give all the servers running (currently shows default which is also seen running in oracle VM gui, we can shutdown server, change the allocated hardware settings from oracle VM gui, apart from it we don’t use it. We will do these steps by docker-machine start etc in cli directly). Run command docker it will give the help in choosing commands related to docker server machine, there we will find env command which will setup the server environment. Run docker env to get the details of the server running in the linux VM and run the command mentioned in it to let client running in windows communicate with the docker server running in the linux VM. Now running docker version will give the details of both the client and server in the cli.

Commands explained with example redis images

Docker search redis - searches for the already existing redis images in docker hub

Docker run -d redis - runs the redis found image as a container inside VM. By default docker will run command in foreground, to make it run in background use –d (detached), by default takes latest version or else we can mention docker run –d run:latest or redis:3.2 for 3.2 version

Docker ps – ps means processes , it will give all the docker containers and inside images running list, it also gives friendly names to call easily under container id

Docker inspect <friendly name | container –id> for all details about running container

Docker logs <friendly name | container –id> display messages written by containers in error logs or output

To access this redis server which is running inside the VM by client we need the port to communicate, we can simplify during running the container like

Docker run –d –name myRedisPort -p 6379:6379 redis:latest Here <host-port>:<container-port> we are stating which port it should run in host for that port of container. Redis runs by default 6379 inside container so we are stating also run this in 6379 of host. By default it is 0.0.0.0:6379 if we mention like above, we can mention full IP like …. –p 127.0.01:6379:6379, if we mention only one port … -p 6379 this is way of creating a dynamic port. It takes 6379 of container port and runs in random host port, by this we can run multiple ports.But to know which port is assigned we can run “docker port redisGiven Name 6379” this will give the dynamic assigned port. (we can verify all by docker ps command)

Everytime we close the container then deata inside will be lost, to store it in host we use binding directories which are called volumes(-v) . Any data which needs to be saved on docker host but not inside containers should be stored in **/opt/docker/data/**redis:/data (bold part is common and then our image comes and then data). Command looks like “docker run –d --name redisMapped –v /opt/docker/data/redis:/data redis” this will create a container of redis image with redisMapped name and binding volume in mentioned path. ($PWD gives the current directory). Since redis is a sever application it runs and we can see inside docker ps list, but if we run OS images like docker run Ubuntu then it will not present in docker ps list,cause it itself is single entity so to access its running process we can give docker run Ubuntu ps, to access the bash of ubunut give docker run –it Ubuntu bash (this will run bash inside Ubuntu OS, -it is used to include interaction commands we give at end).

# Deploy static file as container

To deploy docker images we need **dockerfile** which has all the info of dependencies required by our application. This is a base image file which acts as instruction file. Docker images are built based on contents of docker file….continue further from katacoda.com/courses/docker