Docker commands

Assuming docker machine name is default

* Docker-machine start default => default is machine name =>start docker machine
* Docker-machine stop default
* Docker-machine restart default

After starting docker machine you may need to run:

* Docker-machine env
* Followed by another command eval which will be displayed there to configure the shell

Other commands:

* Docker info
* Docker --version
* Docker-compose --version
* Docker-machine --version
* Docker ps => containers which are running
* Docker container ls
* Docker images
* Docker image ls
* Docker run hello-world => if image hello-world doesn’t exist locally, docker pulls from the hub, creates a container from this image
* Docker run –d –p 80:80 --name webserver nginx => docker start the webserver nginx on ip address port 80
* Docker container start servername
* Docker container stop servername
* Docker container rm –f servername
* Docker image rm imageName
* -d => runs in daemon mode => runs in background
* Docker logs containerID =>gives log details of container id provided
* Docker logs --tail 5 containerId => gives 5 last recent log details of container id provided
* Docker ps –a => view containers you ran
* Docker ps –q => gives container id of container running in background
* Docker stop containerName or docker stop containerId => stops gracefully the container and waits 10 seconds and if not stopped then it sends the kill signal
* Docker search searchKeyWord => searches the dockerhub for repositories
* Docker pull nameToPull:version
* Docker ps –l => last container you ran
* To remove a container..list out the ids, if for example the id of a container is *idf34h5gjd7gh* you need not enter the whole long id but instead you can enter the first minimum 3 characters and it will be deleted as each id generated is unique
* Docker rm idf
* To remove an image..list out the ids, if for example the id of an image is *ghn4kjkj6las* you need not enter the whole long id but instead you can enter the first minimum 3 characters and it will be deleted as each id generated is unique
  + Docker rmi –f ghn
* Docker history containerName => view history of container and commands used

Errors:

* If on docker run something error server misbehaving arises, check internet connection DNS, use google’s public dns settings 8.8.8.8 and 8.8.4.4
* Files/images/containers created should not be placed in another directory but rather should be in same directory of Docker and they can be in sub-directories

Images conceptually similar to classes

Layers conceptually similar to inheritance

Containers conceptually similar to isntances

Running a simple php script on Docker

* Create a directory in the host docker directory
* Create a php script in this directory and add some html to it with one line of php echo code
* Run this command :

docker run --rm -p 8000:80 -v $(pwd):/var/www/html php:apache

* Docker should download the latest version of the image and run at **localhost:8000**

Breaking down the docker command we just ran:

* **--rm** this tells docker to remove the container after the command is completed. When you will exit by pressing **Ctrl + c** the container will stop and remove itself from your system
* **-p 8000:80** this tells docker to map port 80 within the container to port 8000 on the host
* **-v $(pwd):/var/www/html** this tells docker mount a volume. You typically pass in a path to a folder on your root directory, a colon and then a path to the folder in the container. But for this we are just mounting the current directory from our terminal into the directory Apache serves by default
* **Php:apache** we specify the image to use for php

For running a project on Docker we can either deploy from dockerub or github directly on our local server or we can specify the images needed using a docker-compose.yml file followed by docker-compose commands.

If we take a laravel project for example,

* we clone the project from github into our host’s docker directory,
* navigate to the project directory using the terminal,
* create a docker-compose.yml file into the root of the project including the required images and ports and connectivity configurations to database
* Create a vhost.conf file