Container Basics

**Container:** It is a standard unit of software that can run a particular application/service and its associated processes. It is light weighted environment and use underlying operating system kernel.

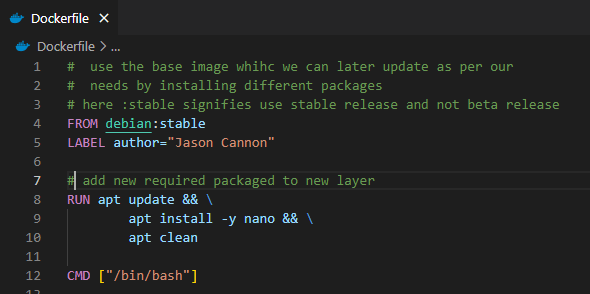
Containers have very small footprint as compared to VM, so they are very quick to start and stop. Virtual machines typically are large have UI and rum multiple apps together which is opposite of container.

Container image is required to start the container. These images are usually stored in docker repository. DockerFile is used to build the container image from base image.

**Base Image:** It is tiny and plain but functional operating system image.

Docker use layered file system and we can add more layers to base image to prepare what we need. Newly added layers do not update base image and keep it intact by adding new changes to new layer. Docker make this possible using layered file system.

Sample DockerFile



# Basic Docker Commands

## Run docker

Command: docker run -dit Debian

Command checks image named Debian is present locally first before starting the container, if it is not present then it fetched image from repo. If we run container without dit option then it will start and immediately stop in background and won’t continue running

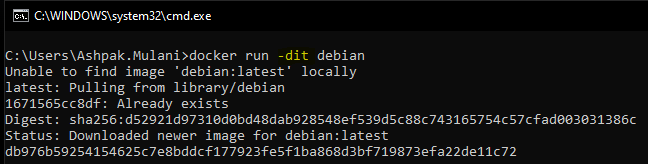
Options:

-dit :

**D** is for detach which allows container to run in background.

**I** is for interactive which allows us to type commands in container.

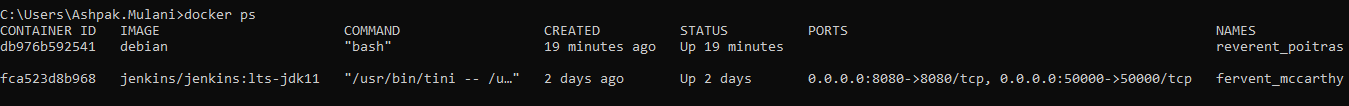
**T** is forterminal which we use to interact with docker

On successful execution command returns docker ID (long string) 

## List docker

Command: docker ps

List all running containers on system. This doesn’t list images. Containers and images are different thing. Container is running image instance.



## Stop docker

Command: docker stop <<docker name or ID>>

Successful execution returns docker ID as show below.



## Inspect docker

To get detailed information about docker running container we use this command. It gives all the details around running container like environment variables, network settings, disk details etc..

Command: docker inspect <<docker ID>>

Successful execution returns lots of details around running docker container.

Sample output   
  
 