Docker Commands

11:37 AM

systemctl start docker
systemctl status docker
Sudo -i // will run all command root

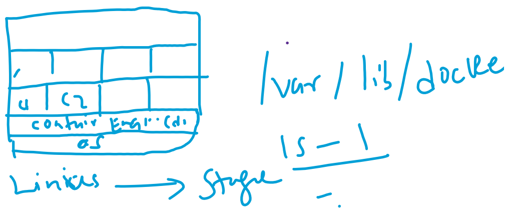
Docker ps -a - To See all running and stop conatiner
docker container ls -l - it will only list running continer
docker container ls -a - stoped and running container

Command	Usage			
docker container prune	Delete all stopped container			
docker container ls -a -s	List all container (stop/running) with size			
docker container rm 23510108d82b	remove cobainer by id or name			
docker images	To list images			
docker run hello-world	To run container from image			
$ docker run [OPTIONS] IMAGE[:TAG|@DIGEST] [COMMAND] [ARG...]				
docker run --name surekha hello-world	To run container from image with custom name to container			
`docker run -it` --name my_ubuntu_co `ubuntu bash`	with extra param			
docker exec -it container_id/container_name bash docker exec -it CONTAINER_ID /bin/bash	If container already running and want to attached/go inside container			
`docker run -d -it` --name my_ubuntu `ubuntu /bin/bash`	It means that the command you initially provided to the container (/bin/bash) will be run in the background and the container will not *stop immediately*			
docker run hello-world	To run container from image			

Docker inspect containerid	Inspect and get ip	
Docker container top cid	Get running process inside container	
Docker container stats	to see all container consuming resource such (ram, cpu etc..)	
Docker container run -d -p 3600:8080 --name testweb nging Netstat -nltp - to see listing port Docker container inspect id		Port forwarding
Docker c rename id newname	Rename container	
Docker conatiner restart id		

/var/lib/docker/ - inside Linux docker get installed here-- it has all conatiner, volumns etc place here..

To access container with initial 3 id		
Kill and stop	abruptly Kill	
Docker container wait cid	It will show exit status	
Docker container pause cid	docker ps -a	it will show paused
Docker container unpause cid	docker ps -a	It will show up running
Docker conatiner port id/nm	To see port mapping	
Docker volumn ls		
Dokcer run -v		
Docker volumn create --name=na		



Containers

Use docker container my_command

create — Create a container from an image.

start — Start an existing container.

run — Create a new container and start it.

ls — List running containers.

inspect — See lots of info about a container.

logs — Print logs. **Docker container logs cid/cname**

stop — Gracefully stop running container.

kill —Stop main process in container abruptly.

rm— Delete a stopped container.

```
docker container run -d --name surekha-self --rm node:latest
```

Images

Use docker image my_command

build — Build an image.

push — Push an image to a remote registry.

ls — List images.

history — See intermediate image info.

inspect — See lots of info about an image, including the layers.

rm — Delete an image

Misc

docker info - no of list con,img etc

docker version — List info about your Docker Client and Server versions.

docker login — Log in to a Docker registry.

— Delete all unused containers, unused networks, and dangling images.

Command combination

```
docker run -d httpd -name test
```

In order to launch this Docker container in the background, I included the *-d* (detach) flag.

```
docker container ls --all
```

remove all containers

```
docker rm -f $(docker ps -a -q)
```

```
docker ps
```

```
docker ps -a
```

```
docker rm test
```

docker container run -i -t -p 1000:8000 --rm my_image

You need to specify both -i and -t to then interact with the container through your terminal shell.

The port is the interface with the outside world. 1000:8000 maps the Docker port 8000 to port 1000 on your machine.

If you had an app that output something to the browser you could then navigate your browser to localhost:1000 and see it.

docker container run -d my_image

```
winpty docker run -it ubuntu
```

Terminology

In the last section, we used a lot of Docker-specific jargon which might be confusing to some. So before we go further, let me clarify some terminology that is used frequently in the Docker ecosystem.

- *Images* - The blueprints of our application which form the basis of containers. In the demo above, we used the `docker pull` command to download the **busybox** image.
- *Containers* - Created from Docker images and run the actual application. We create a container using `docker run` which we did using the busybox image that we downloaded. A list of running containers can be seen using the `docker ps` command.
- *Docker Daemon* - The background service running on the host that manages building, running and distributing Docker containers. The daemon is the process that runs in the operating system which clients talk to.
- *Docker Client* - The command line tool that allows the user to interact with the daemon. More generally, there can be other forms of clients too -
- *Docker Hub* - A registry of Docker images. You can think of the registry as a directory of all available Docker images. If required, one can host their own Docker registries and can use them for pulling images.