Container run

1. **docker create** creates a container but does not start it.
2. **docker rename** allows the container to be renamed.
3. **docker run** creates and starts a container in one operation.

**$ docker run -it ubuntu-ssh-k /bin/bash**

i: interactive, t: with tty

1. **docker rm** deletes a container.
2. **docker update** updates a container's resource limits.

Container info

1. **docker ps** shows running containers.

**$ docker ps**

**CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES**

**e2481c1bad5d ubuntu-ssh-k:latest "/bin/bash" 10 hours ago Up 10 hours hopeful\_carson**

1. **docker logs** gets logs from container. (You can use a custom log driver, but logs is only available for json-file and journald in 1.10)
2. **docker inspect** looks at all the info on a container (including IP address).

To get an IP address of a running container:

**$ docker inspect --format '{{ .NetworkSettings.IPAddress }}' $(docker ps -q)**

**172.17.0.5**

1. **docker events** gets events from container.
2. **docker port** shows public facing port of container.
3. **docker top** shows running processes in container.
4. **docker stats** shows containers' resource usage statistics.
5. **docker diff** shows changed files in the container's FS.

Container start/stop

1. **docker start** starts a container so it is running.
2. **docker stop** stops a running container.
3. **docker restart** stops and starts a container.
4. **docker pause** pauses a running container, "freezing" it in place.
5. **docker unpause** will unpause a running container.
6. **docker wait** blocks until running container stops.
7. **docker kill** sends a SIGKILL to a running container.
8. **docker attach** will connect to a running container.

Image create/remove

1. **docker images** : shows all images.
2. **docker import** : creates an image from a tarball.
3. **docker build** : creates image from Dockerfile.
4. **docker commit** : creates image from a container, pausing it temporarily if it is running.

**$ docker commit b9b2ee7004fe ubuntu-ssh-k**

where the 'b9b2ee7004fe' is the container-id and 'ubuntu-ssh-k' is the new name of the image.

1. **docker rmi** : removes an image
2. **docker load** : loads an image from a tar archive as STDIN, including images and tags

.

1. **docker save** : saves an image to a tar archive stream to STDOUT with all parent layers, tags & versions

.

Container info

Getting Docker Container's IP Address from host machine:

**$ docker inspect --format '{{ .NetworkSettings.IPAddress }}' $(docker ps -q)**

**172.17.0.2**

**172.17.0.3**

Image info

1. **docker history** shows history of image.
2. **docker tag** tags an image to a name (local or registry).

Network create/remove/info/connection

1. **docker network create**
2. **docker network rm**
3. **docker network ls**
4. **docker network inspect**
5. **docker network connect**
6. **docker network disconnect**

Docker Repo

1. **docker login** to login to a registry.
2. **docker logout** to logout from a registry.
3. **docker search** searches registry for image.

**$ docker search mysql**

**NAME DESCRIPTION STARS OFFICIAL AUTOMATED**

**mysql MySQL is a widely used, open-source relati... 2797 [OK]**

**mysql/mysql-server Optimized MySQL Server Docker images. Crea... 182 [OK]**

**...**

1. **docker pull** pulls an image from registry to local machine.

**$ docker pull ubuntu**

**latest: Pulling from ubuntu**

**20ee58809289: Pull complete**

**f905badeb558: Pull complete**

**119df6bf2a3a: Pull complete**

**94d6eea646bc: Pull complete**

**bb4eabee84bf: Pull complete**

**Digest: sha256:85af8b61adffea165e84e47e0034923ec237754a208501fce5dbeecbb197062c**

**Status: Downloaded newer image for ubuntu:latest**

Docker images can consist of multiple layers. In the example above, the image consists of five layers

1. **docker push** pushes an image to the registry from local machine.