Managing Containers

Docker provides the following commands to manage containers:

- **`docker ps`** : This command is responsible for listing running containers:

```
$ docker ps
CONTAINER ID     IMAGE              COMMAND
CREATED         STATUS           PORTS       NAMES
77d4b7b8ed1f(1)  do180/httpd(2) "httpd -D FOREGROUN
D"(3) 15 hours ago (4) Up 15 hours (5)  80/tcp(6)
 my-httpd-container (7)
```

(1) Each container, when created, gets a **`container ID`**, which is a hexadecimal number and looks like an image ID, but is actually unrelated.

(2) Container image that was used to start the container.

(3) Command that was executed when the container started.

(4) Date and time the container was started.

5 Total container uptime, if still running, or time since terminated.

6 Ports that were exposed by the container or the port forwards, if configured.

7 The container name.

Stopped containers are not discarded immediately. Their local file systems and other states are preserved so they can be inspected for post-mortem analysis. Option **-a** lists all containers, including containers that were not discarded yet:

```
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND
CREATED             STATUS                      POR
TS                NAMES
4829d82fbbff        do180/httpd         "httpd -D FO
REGROUND"   15 hours ago        Exited (0) 3 seconds
ago                         my-httpd-container
```

- **`docker inspect`** : This command is responsible for listing metadata about a running or stopped container. The command produces a **JSON** output:

```
$ docker inspect my-httpd-container
[
{
    "Id": "980e45b5376a4e966775fb49cbef47ee7bbd461be
8bfd1a75c2cc5371676c8be",
...OUTPUT OMITTED...
    "NetworkSettings": {
        "Bridge": "",
        "EndpointID": "483fc91363e5d877ea8f9696854a1
f14710a085c6719afc858792154905d801a",
        "Gateway": "172.17.42.1",
        "GlobalIPv6Address": "",
        "GlobalIPv6PrefixLen": 0,
        "HairpinMode": false,
        "IPAddress": "172.17.0.9",
...OUTPUT OMITTED...
```

This command allows formatting of the output string using the given **go** template with the **-f** option. For example, to retrieve only the IP address, the following command can be executed:

```
$ docker inspect -f '{{ .NetworkSettings.IPAddress
 }}' my-httpd-container
```

- **docker stop** : This command is responsible for stopping a running container gracefully:

```
$ docker stop my-httpd-container
```

Using **docker stop** is easier than finding the container start process on the host OS and killing it.

- **docker kill** : This command is responsible for stopping a running container forcefully:

```
$ docker kill my-httpd-container
```

It is possible to specify the signal with the **`-s`** option:

```
$ docker kill -s SIGKILL my-httpd-container
```

The following signals are available:

SIGNAL	Default action	Description
SIGHUP	Terminate process	Terminate line hangup
SIGINT	Terminate process	Interrupt program
SIGQUIT	Create core image	Quit program
SIGABRT	Create core image	Abort program
SIGKILL	Terminate process	Kill program
SIGTERM	Terminate process	Software termination signal
SIGUSR1	Terminate process	User-defined signal 1
SIGUSR2	Terminate process	User-defined signal 2

- **`docker restart`** : This command is responsible for restarting a stopped container:

```
$ docker restart my-httpd-container
```

The **`docker restart`** command creates a new container with the same container ID, reusing the stopped container state and filesystem.

- **`docker rm`** : This command is responsible for deleting a container, discarding its state and filesystem:

```
$ docker rm my-httpd-container
```

It is possible to delete all containers at the same time. The **`docker ps`** command has the **`-q`** option that returns only the ID of the containers. This list can be passed to the **`docker rm`** command:

```
$ docker rm $(docker ps -aq)
```

Before deleting all containers, all running containers must be stopped. It is possible to stop all containers with:

```
$ docker stop $(docker ps -q)
```

Note

The commands **`docker inspect`**, **`docker stop`**, **`docker kill`**, **`docker restart`**, and **`docker rm`** can use the container ID instead of the container name.