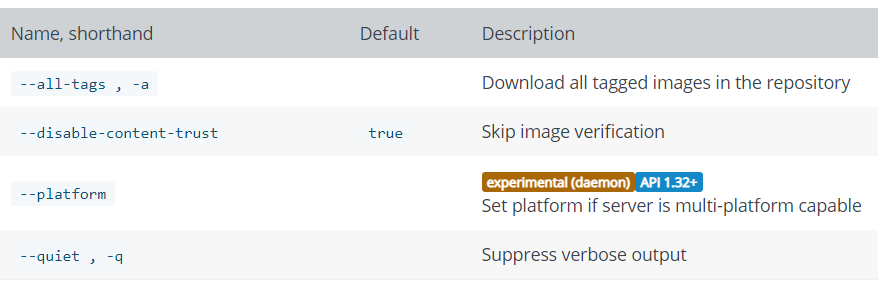
**Pull:**

**If we just want image, not container. Then we use pull**

* **Docker pull <image name>**

**To pull all the tags of image.**

* **Docker pull -a**



If we don’t specify the tag. It will pull LATEST by default. And also, by default it pulls from docker hub public repository. If we want to use any other repository, use the below command.

* **Docker pull <registry>/<image>:<tag>**

**images:**

**to check the images, present locally**

* **Docker images 🡪 to check the images**

**To check all the images including intermediate images**

* **Docker images -a**
* **Docker images –all**

**To see only image IDs**

* **Docker images -q**

**To remove an image.**

* **Docker rmi <image ID or name>**
* **Docker rmi <image id or name> -f 🡪 force remove**

**To remove multiple images.**

* **Docker rmi <image1> <image2>**
* **sudo docker rmi $(sudo docker images -aq) 🡪 to remove all ids at once**

**Containers:**

**To check the containers running currently**

* **Docker ps**

**To check the old stopped containers.**

* **Docker ps -a**

**To remove the container.**

* **Docker rm <container ID or name>**

**To remove multiple containers.**

* **Docker rm <container1> <container2>**
* **Docker ps -aq or -a -q 🡪 to see all the ids**
* **Docker rm $(docker ps -a -q) 🡪to remove all the ids at once**

**Run:**

* We can run docker container as in attached, detached or interactive mode
* By default, it runs in attached mode. If we close the terminal after running the container, the application gets terminated
* If we add **-d** in the command, the terminal runs in background. It is called detached mode
* The docker way where we cannot enter something but only watching what happening is called attached mode
* Whenever we want interactive, we use **-it**
* **Docker run -it <id or name> /bin/bash**
* **Docker run -p 80:80 nginx:1.12**

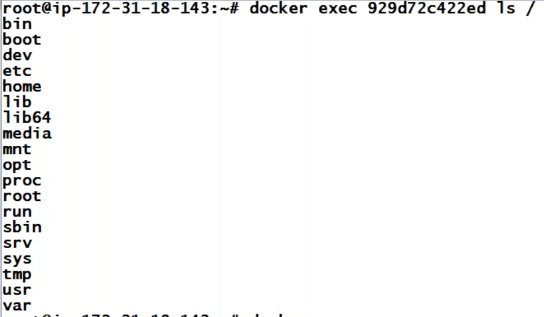
tag is nothing but version of image. If we don’t mention version, it will take it by default as latest version

we can also use below options while running the image to overwrite the image configuration.

|  |  |  |
| --- | --- | --- |
| --entrypoint |  | Overwrite the default ENTRYPOINT of the image |
| --env , -e |  | Set environment variables |
| --env-file |  | Read in a file of environment variables |
| --expose |  | Expose a port or a range of ports |
| --hostname , -h |  | Container host name |
| --ip |  | IPv4 address (e.g., 172.30.100.104) |
| --label , -l |  | Set meta data on a container |
| --label-file |  | Read in a line delimited file of labels |
| --link |  | Add link to another container |
| --memory , -m |  | Memory limit |
| --name |  | Assign a name to the container |
| --mount |  | Attach a filesystem mount to the container |
| --restart | no | Restart policy to apply when a container exits |
| --rm |  | Automatically remove the container when it exits |
| --ulimit |  | Ulimit options |
| --user , -u |  | Username or UID (format: <name|uid>[:<group|gid>]) |
| --volume , -v |  | Bind mount a volume |
| --publish , -p |  | Publish a container’s port(s) to the host |

**exec:**

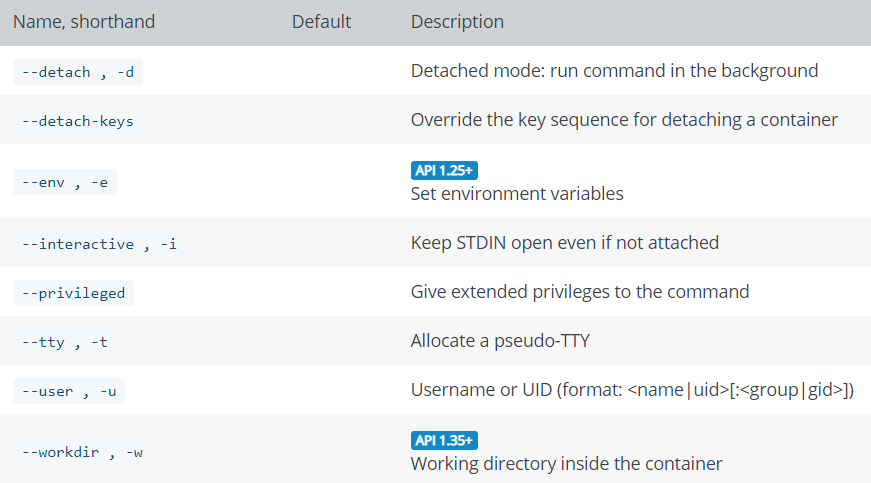
* The docker exec command runs a new command in a running container.



* As above, we can execute any command without entering to the container



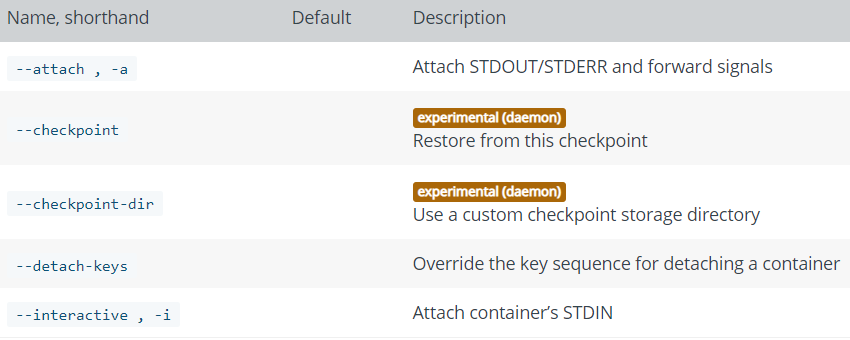
* With the above command. We can ssh into container and run commands
* /bin/bash is used there to get the bash shell in container.



**Start:**

Docker start command to used to start the stopped container

* **Docker start <container name or ID>**
* **Docker start -i <name or ID 🡪 this is used to ssh into container after starting it**



**Stop:**

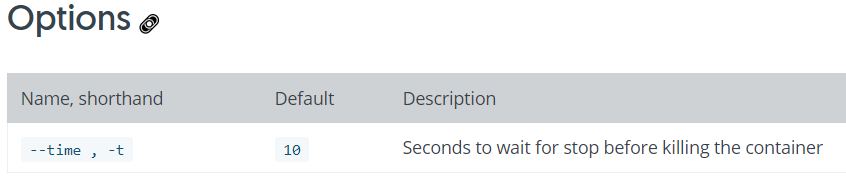
It is to stop the running container

* **Docker stop <name or ID>**
* **Docker stop -t 10 <name or ID> 🡪 to stop the container after waiting for 10 sec. we can also use –time.**

**Restart:**

To restart one or more containers.

* **Docker restart <name or ID>**



Docker has whatever required for the application. If we want, we can install anything

* **Docker logs <id or name> 🡪 to see the logs generated by app**
* **Docker ps <id or name> 🡪 for the history of container**
* **Docker top <container id> 🡪 to check the processes in container**