**Docker Commands Cheat Sheet**

1. [docker ps](https://docs.docker.com/engine/reference/commandline/ps/) — Lists running containers. Some useful flags include: **-a / -all** for all containers (default shows just running) and —**-quiet** /-**q** to list just their ids (useful for when you want to get all the containers).
2. [docker pull](https://docs.docker.com/engine/reference/commandline/pull/) — Most of your images will be created on top of a base image from the [Docker Hub](https://hub.docker.com/) registry. [Docker Hub](https://hub.docker.com/) contains many pre-built images that you can pull and try without needing to define and configure your own. To download a particular image, or set of images (i.e., a repository), use **docker pull**.
3. [docker build](https://docs.docker.com/engine/reference/commandline/build/) — The **docker build** command builds Docker images from a Dockerfile and a “context”. A build’s context is the set of files located in the specified PATH or URL. Use the -t flag to label the image, for example **docker build -t my\_container** . with the . at the end signalling to build using the currently directory.
4. [docker run](https://docs.docker.com/engine/reference/run/) — Run a docker container based on an image, you can follow this on with other commands, such as **-it bash** to then run bash from within the container. Also see [Top 10 options for docker run — a quick reference guide for the CLI command](https://medium.com/the-code-review/top-10-docker-run-command-options-you-cant-live-without-a-reference-d256834e86c1). **docker run my\_image -it bash**
5. [docker logs](https://docs.docker.com/engine/reference/commandline/logs/)— Use this command to display the logs of a container, you must specify a container and can use flags, such as **--follow** to follow the output in the logs of using the program. **docker logs --follow my\_container**
6. [docker volume ls](https://docs.docker.com/engine/reference/commandline/volume_ls/) — This lists the [volumes](https://docs.docker.com/storage/volumes/), which are the preferred mechanism for persisting data generated by and used by Docker containers.
7. [docker rm](https://docs.docker.com/engine/reference/commandline/rm/) — Removes one or more containers. **docker rm my\_container**
8. [docker rmi](https://docs.docker.com/engine/reference/commandline/rmi/)— Removes one or more images. **docker rmi my\_image**
9. [docker stop](https://docs.docker.com/engine/reference/commandline/stop/) — Stops one or more containers. **docker stop my\_container** stops one container, while **docker stop $(docker ps -a -q)** stops all running containers. A more direct way is to use docker kill my\_container, which does not attempt to shut down the process gracefully first.
10. Use them together, for example to clean up all your docker images and containers:

* kill all running containers with **docker kill $(docker ps -q)**
* delete all stopped containers with **docker rm $(docker ps -a -q)**
* delete all images with **docker rmi $(docker images -q)**