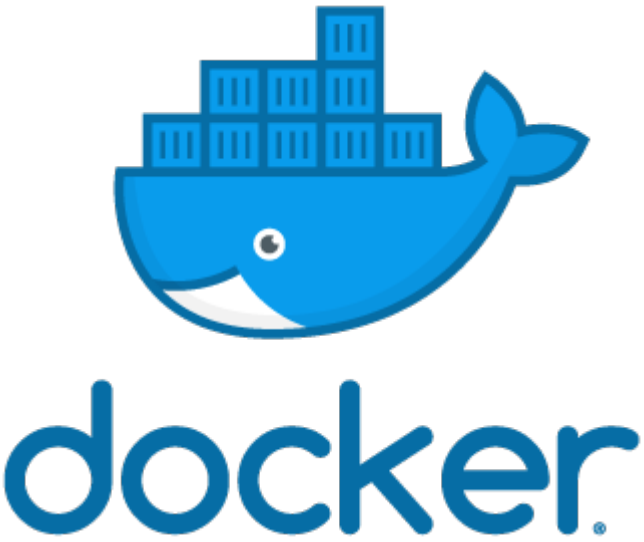
Docker: Stop All Containers



Now and then, especially when working on a development environment, you need to stop multiple Docker containers. Quite often, you need to stop all of the currently running containers. I'm going to show you one of the possible ways.

Docker: Stop a Container

You need to use a container name or container ID with the **docker stop** command.

For example, I have an **nginx** load balancer container:

```
root@s5:~ # docker ps -f name=nginx
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
32cd3e477546 nginx:latest "nginx -g 'daemon of…" 11 months ago
Up About a minute 0.0.0.0:80->80/tcp, 0.0.0.0:443->443/tcp
nginx
```

Based on this output, I can stop my **nginx** container like this:

```
root@s5:~ # docker stop nginx
nginx
```

… or like that:

```
root@s5:~ # docker stop 32cd3e477546
```

32cd3e477546

Docker: Stop Multiple Containers

Since I also have a MariaDB container named db, I might need stop it together with nginx.

Here’s the info on the db container:

```
root@s5:~ # docker ps -f name=db
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
c745794419a9 mariadb:latest "docker-entrypoint.s…" 9 months ago Up 4 seconds 3306/tcp db
```

If I ever decide to stop both nginx and db together, I can do it like this:

```
root@s5:~ # docker stop nginx db
nginx
db
```

Docker: Stop All Containers

As you can see from previous examples, **docker stop** simply takes a list of containers to stop. If there’s more than one container, just use space as a delimiter between container names or IDs.

This also allows us to use a clever shell expansion trick: you can some other command, and pass its output to the docker stop container.

For instance, this shows us the list of all the IDs for currently running **Docker** containers:

```
root@s5:~ # docker ps -q
510972d55d8c
1b8b1657736e
c745794419a9
32cd3e477546
```

What we can do now is pass the result of this command as the parameter for the docker stop command:

```
root@s5:~ # docker stop $(docker ps -q)
510972d55d8c
1b8b1657736e
c745794419a9
32cd3e477546
```

And just to check, running docker ps now won't show any running containers:

```
root@s5:~ # docker ps -q
```

IMPORTANT: make sure you double-check what you're doing! Specifically, run docker ps -q, compare it to docker ps, this kind of thing. Because once containers stopped you may not have an easy way to generate the list of same containers to restart.

In my case, I'm just specifying them manually as the parameters for docker start:

```
root@s5:~ # docker start 510972d55d8c 1b8b1657736e c745794419a9 32cd3e477546
510972d55d8c
1b8b1657736e
c745794419a9
32cd3e477546
```

That's it for today! Hope you enjoyed this quick how-to, let me know if you have any questions, **Docker** and whatnot!

See Also

- Docker Software
- List containers in Docker
- Restart Stopped Docker containers
- Docker Inspect
- Remove Unused Docker volumes