**DigitalOcean**

Subscribe

## How To Remove Docker Images, Containers, and Volumes

48

Updated December 21, 2017      👁 1.4m      DOCKER

By: Melissa Anderson

# A Docker Cheat Sheet

## Introduction

Docker makes it easy to wrap your applications and services in containers so you can run them anywhere. As you work with Docker, however, it's also easy to accumulate an excessive number of unused images, containers, and data volumes that clutter the output and consume disk space.

Docker gives you all the tools you need to clean up your system from the command line. This cheat sheet-style guide provides a quick reference to commands that are useful for freeing disk space and keeping your system organized by removing unused Docker images, containers, and volumes.

**How to Use This Guide:**

- This guide is in cheat sheet format with self-contained command-line snippets

- Jump to any section that is relevant to the task you are trying to complete.

The command substitution syntax, `command $(command)`, used in the commands is available in many popular shells such as bash, zsh, and Windows Powershell.

# Purging All Unused or Dangling Images, Containers, Volumes, and Networks

Docker provides a single command that will clean up any resources — images, containers, volumes, and networks — that are dangling (not associated with a container):

```
$ docker system prune
```

To additionally remove any stopped containers and all unused images (not just dangling images), add the `-a` flag to the command:

```
$ docker system prune -a
```

# Removing Docker Images

## Remove one or more specific images

Use the `docker images` command with the `-a` flag to locate the ID of the images you want to remove. This will show you every image, including intermediate image layers. When you've located the images you want to delete, you can pass their ID or tag to `docker rmi`:

**List:**

```
$ docker images -a
```

**Remove:**

```
$ docker rmi Image Image
```

## Remove dangling images

Docker images consist of multiple layers. Dangling images are layers that have no relationship to any tagged images. They no longer serve a purpose and consume disk space. They can be located by adding the filter flag, `-f` with a value of `dangling=true` to the `docker images` command. When you're sure you want to delete them, you can use the `docker images purge` command:

> **Note:** If you build an image without tagging it, the image will appear on the list of dangling images because it has no association with a tagged image. You can avoid this situation by providing a tag when you build, and you can retroactively tag an images with the docker tag command.

**List:**

```
$ docker images -f dangling=true
```

**Remove:**

```
$ docker images purge
```

## Removing images according to a pattern

You can find all the images that match a pattern using a combination of `docker images` and `grep`. Once you're satisfied, you can delete them by using `awk` to pass the IDs to `docker rmi`. Note that these utilities are not supplied by Docker and are not necessarily available on all systems:

**List:**

```
$ docker images -a |  grep "pattern"
```

**Remove:**

```
$ docker images -a | grep "pattern" | awk '{print $3}' | xargs docker rmi
```

## Remove all images

All the Docker images on a system can be listed by adding `-a` to the `docker images` command. Once you're sure you want to delete them all, you can add the `-q` flag to pass the Image ID to `docker rmi`:

**List:**

```
$ docker images -a
```

**Remove:**

```
$ docker rmi $(docker images -a -q)
```

# Removing Containers

## Remove one or more specific containers

Use the `docker ps` command with the `-a` flag to locate the name or ID of the containers you want to remove:

**List:**

```
$ docker ps -a
```

**Remove:**

```
$ docker rm ID_or_Name ID_or_Name
```

## Remove a container upon exit

If you know when you're creating a container that you won't want to keep it around once you're done, you can run `docker run --rm` to automatically delete it when it exits.

**Run and Remove:**

```
$ docker run --rm image_name
```

## Remove all exited containers

You can locate containers using `docker ps -a` and filter them by their status: created, restarting, running, paused, or exited. To review the list of exited containers, use the `-f` flag to filter based on status. When you've verified you want to remove those containers, using `-q` to pass the IDs to the `docker rm` command.

**List:**

```
$ docker ps -a -f status=exited
```

**Remove:**

```
$ docker rm $(docker ps -a -f status=exited -q)
```

## Remove containers using more than one filter

Docker filters can be combined by repeating the filter flag with an additional value. This results in a list of containers that meet either condition. For example, if you want to delete all containers marked as either **Created** (a state which can result when you run a container with an invalid command) or **Exited**, you can use two filters:

**List:**

```
$ docker ps -a -f status=exited -f status=created
```

**Remove:**

```
$ docker rm $(docker ps -a -f status=exited -f status=created -q)
```

## Remove containers according to a pattern

You can find all the containers that match a pattern using a combination of `docker ps` and grep. When you're satisfied that you have the list you want to delete, you can use `awk` and `xargs` to supply the ID to `docker rmi`. Note that these utilities are not supplied by Docker and not necessarily available on all systems:

**List:**

```
$ docker ps -a |  grep "pattern"
```

**Remove:**

```
$ docker ps -a | grep "pattern" | awk '{print $3}' | xargs docker rmi
```

## Stop and remove all containers

You can review the containers on your system with `docker ps`. Adding the `-a` flag will show all containers. When you're sure you want to delete them, you can add the `-q` flag to supply the IDs to the `docker stop` and `docker rm` commands:

**List:**

```
$ docker ps -a
```

**Remove:**

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

# Removing Volumes

## Remove one or more specific volumes - Docker 1.9 and later

Use the `docker volume ls` command to locate the volume name or names you wish to delete.
Then you can remove one or more volumes with the `docker volume rm` command:

**List:**

```
$ docker volume ls
```

**Remove:**

```
$ docker volume rm volume_name volume_name
```

## Remove dangling volumes - Docker 1.9 and later

Since the point of volumes is to exist independent from containers, when a container is removed,
a volume is not automatically removed at the same time. When a volume exists and is no longer
connected to any containers, it's called a dangling volume. To locate them to confirm you want to
remove them, you can use the `docker volume ls` command with a filter to limit the results to
dangling volumes. When you're satisfied with the list, you can remove them all with `docker
volume prune`:

**List:**

```
$ docker volume ls -f dangling=true
```

**Remove:**

```
$ docker volume prune
```

## Remove a container and its volume

If you created an unnamed volume, it can be deleted at the same time as the container with the
`-v` flag. Note that this only works with *unnamed* volumes. When the container is successfully
removed, its ID is displayed. Note that no reference is made to the removal of the volume. If it is
unnamed, it is silently removed from the system. If it is named, it silently stays present.

**Remove:**

```
$ docker rm -v container_name
```

# Conclusion

This guide covers some of the common commands used to remove images, containers, and volumes with Docker. There are many other combinations and flags that can be used with each. For a comprehensive guide to what's available, see the Docker documentation for `docker system prune`, `docker rmi`, `docker rm` and `docker volume rm`. If there are common cleanup tasks you'd like to see in the guide, please ask or make suggestions in the comments.

By: Melissa Anderson                                    ♡ Upvote (48)          ⬚ Subscribe

## Introducing Projects on DigitalOcean

Organize your resources according to
how you work.

READ MORE

## Related Tutorials

How To Install Docker Compose on Ubuntu 18.04

Naming Docker Containers: 3 Tips for Beginners

How to Use Traefik as a Reverse Proxy for Docker Containers on Ubuntu 18.04

How To Provision and Manage Remote Docker Hosts with Docker Machine on Ubuntu 18.04

Webinar Series: Building Blocks for Doing CI/CD with Kubernetes

# 10 Comments

Leave a comment...

Log In to Comment

**kyoungrok0517**  *November 20, 2016*

2  Great article. Thanks!

**piotrekrogowski**  *January 23, 2017*

2  There is -q missing in Remove containers using more than one filter

should be docker rm $(docker ps -a -f status=exited -f status=created **-q**)

**MelissaAnderson**  MOD  *January 23, 2017*

1  The  `-q`  was indeed missing, and we've updated the article. Thank you for taking the time to point it out.

**zakkg3**  *July 25, 2018*

0  @MelissaAnderson note to remove dangling images the command seems to be docker image prune, not docker imageS purge

**lucasbmd**  *February 17, 2017*

3  Super handy article!

I think the "Removing images according to a pattern" and "Remove containers according to a pattern" are mixed up.

For images:

```
docker images | grep "pattern" | awk '{print $3}' | xargs docker rmi
```

and alternatively, I found this one to work better (as multiple tags can link to a same image):

```
docker images | grep "pattern" | awk '{print $1":"$2}' | xargs docker rmi
```

For containers:

```
docker ps -a | grep "pattern" | awk '{print $1}' | xargs docker rm
```

---

**BHa** *July 12, 2017*

1 Thanks for this detailed useful article!

For now as a punch line you can add the new alternate way : `docker system prune`

See https://docs.docker.com/engine/reference/commandline/system_prune/

Can't figure out since this new command released.

---

**cjratliff** *October 6, 2017*

1 This was a life saver. I am a rookie at this and it helped greatly. Thanks!

---

**arangogutierrez** *October 16, 2017*

1 What about the build in CLI tools

```
docker system prune

docker volume prune

docker network prune
```

And you are done!

---

**Tehnicaorg** *January 17, 2018*

2 Thanks for the document.

To remove dangling images the command seems to be `docker image prune`, not `docker images purge`

```
λ docker image prune
WARNING! This will remove all dangling images.
```

```
Are you sure you want to continue? [y/N] y
Deleted Images:
[...]
```

---

Copyright © 2018 DigitalOcean™ Inc.

Community   Tutorials   Questions   Projects   Tags   Newsletter   RSS

Distros & One-Click Apps   Terms, Privacy, & Copyright   Security   Report a Bug   Write for DOnations   Shop