

# Prune unused objects

## Prune images and containers

<https://docs.docker.com/config/pruning/>

## Prune images

The docker image prune command allows you to clean up unused images. By default, docker image prune only cleans up dangling images. A dangling image is one that is not tagged and is not referenced by any container. To remove dangling images:

```
$ docker image prune
```

```
WARNING! This will remove all dangling images.  
Are you sure you want to continue? [y/N] y
```

To remove all images which are not used by existing containers, use the -a flag:

```
$ docker image prune -a
```

```
WARNING! This will remove all images without at least one container associated to them.  
Are you sure you want to continue? [y/N] y
```

By default, you are prompted to continue. To bypass the prompt, use the -f or --force flag.

You can limit which images are pruned using filtering expressions with the --filter flag. For example, to only consider images created more than 24 hours ago:

```
$ docker image prune -a --filter "until=24h"
```

Other filtering expressions are available. See the docker image prune reference for more examples.

## Prune containers

When you stop a container, it is not automatically removed unless you started it with the `--rm` flag. To see all containers on the Docker host, including stopped containers, use `docker ps -a`. You may be surprised how many containers exist, especially on a development system! A stopped container's writable layers still take up disk space. To clean this up, you can use the `docker container prune` command.

```
$ docker container prune
```

```
WARNING! This will remove all stopped containers.
```

```
Are you sure you want to continue? [y/N] y
```

By default, you are prompted to continue. To bypass the prompt, use the `-f` or `--force` flag.

By default, all stopped containers are removed. You can limit the scope using the `--filter` flag. For instance, the following command only removes stopped containers older than 24 hours:

```
$ docker container prune --filter "until=24h"
```

Other filtering expressions are available. See the [docker container prune reference](#) for more examples.

## Prune volumes

Volumes can be used by one or more containers, and take up space on the Docker host. Volumes are never removed automatically, because to do so could destroy data.

```
$ docker volume prune
```

```
WARNING! This will remove all volumes not used by at least one container.
```

```
Are you sure you want to continue? [y/N] y
```

By default, you are prompted to continue. To bypass the prompt, use the `-f` or `--force` flag.

By default, all unused volumes are removed. You can limit the scope using the `--filter` flag. For instance, the following command only removes volumes which are not labelled with the `keep` label:

```
$ docker volume prune --filter "label!=keep"
```

Other filtering expressions are available. See the [docker volume prune reference](#) for more examples.

# Prune networks

Docker networks don't take up much disk space, but they do create iptables rules, bridge network devices, and routing table entries. To clean these things up, you can use `docker network prune` to clean up networks which aren't used by any containers.

```
$ docker network prune
```

```
WARNING! This will remove all networks not used by at least one container.
```

```
Are you sure you want to continue? [y/N] y
```

By default, you are prompted to continue. To bypass the prompt, use the `-f` or `--force` flag.

By default, all unused networks are removed. You can limit the scope using the `--filter` flag. For instance, the following command only removes networks older than 24 hours:

```
$ docker network prune --filter "until=24h"
```

Other filtering expressions are available. See the [docker network prune reference](#) for more examples.

---

Revision #1

Created 13 October 2020 09:55:36 by Clinton

Updated 13 October 2020 09:59:59 by Clinton