Prune unused Docker objects

Estimated reading time: 5 minutes

Docker takes a conservative approach to cleaning up unused objects (often referred to as "garbage collection"), such as images, containers, volumes, and networks: these objects are generally not removed unless you explicitly ask Docker to do so. This can cause Docker to use extra disk space. For each type of object, Docker provides a `prune` command. In addition, you can use `docker system prune` to clean up multiple types of objects at once. This topic shows how to use these `prune` commands.

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
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 (https://docs.docker.com/engine/reference/commandline/image_prune/) 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 (https://docs.docker.com/engine/reference/commandline/container_prune/) 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 (https://docs.docker.com/engine/reference/commandline/volume_prune/) 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 (https://docs.docker.com/engine/reference/commandline/network_prune/) for more examples.

Prune everything

The `docker system prune` command is a shortcut that prunes images, containers, and networks. In Docker 17.06.0 and earlier, volumes are also pruned. In Docker 17.06.1 and higher, you must specify the `--volumes` flag for `docker system prune` to prune volumes.

```
$ docker system prune

WARNING! This will remove:
        - all stopped containers
        - all networks not used by at least one container
        - all dangling images
        - all build cache
Are you sure you want to continue? [y/N] y
```

If you are on Docker 17.06.1 or higher and want to also prune volumes, add the `--volumes` flag:

```
$ docker system prune --volumes

WARNING! This will remove:
        - all stopped containers
        - all networks not used by at least one container
        - all volumes not used by at least one container
        - all dangling images
        - all build cache
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.

pruning (https://docs.docker.com/glossary/?term=pruning), prune (https://docs.docker.com/glossary/?term=prune), images (https://docs.docker.com/glossary/?term=images), volumes (https://docs.docker.com/glossary/?term=volumes), containers (https://docs.docker.com/glossary/?term=containers), networks (https://docs.docker.com/glossary/?term=networks), disk (https://docs.docker.com/glossary/?term=disk), administration