**Docker Cleanup Process**

**Prune unused Docker objects**

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**

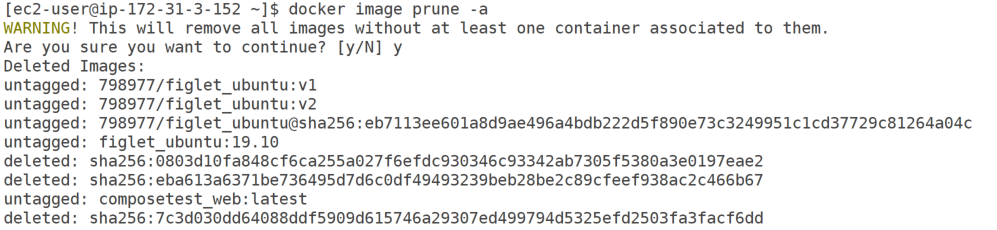
The **docker image prune** command allows you to clean up unused images. By default, **docker image prune** only cleans up **dangling images**. A **dangling image** is one that is not tagged and is not referenced by any container. To remove dangling images:

$ docker image prune



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

$ docker image prune -a





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"

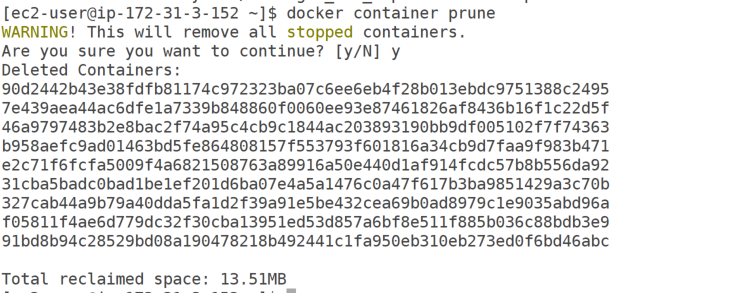
for more options:

<https://docs.docker.com/engine/reference/commandline/image_prune/>

**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



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"

for more options <https://docs.docker.com/engine/reference/commandline/container_prune/>

**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.

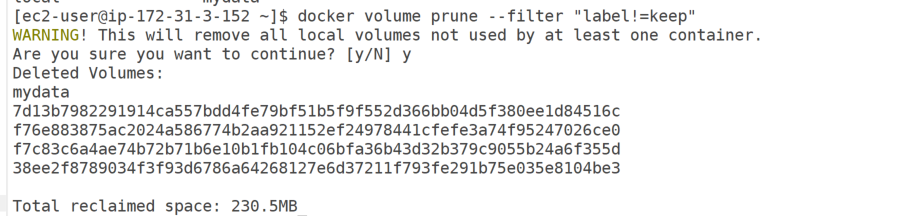
$ docker volume prune

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:

ref: <https://docs.docker.com/engine/reference/commandline/volume_ls/>

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

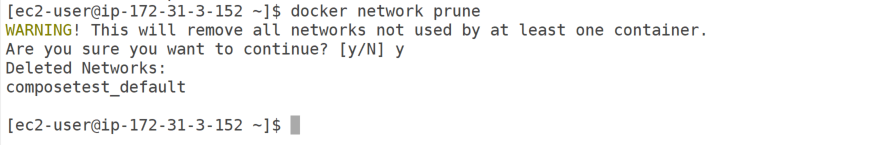


for more options <https://docs.docker.com/engine/reference/commandline/volume_prune/>

**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



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:

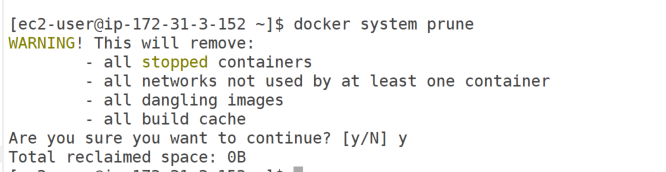
$ docker network prune --filter "until=24h"

for more options <https://docs.docker.com/engine/reference/commandline/network_prune/>

**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



$ docker system prune --volumes

