docker image prune

Estimated reading time: 7 minutes

Description

Remove unused images

API 1.25+ (https://docs.docker.com/engine/api/v1.25/) The client and daemon API must both be at least 1.25 (https://docs.docker.com/engine/api/v1.25/) to use this command. Use the `docker version` command on the client to check your client and daemon API versions.

Usage

```
docker image prune [OPTIONS]
```

Options

Name, shorthand	Default	Description
`--all , -a`		Remove all unused images, not just dangling ones
`--filter`		Provide filter values (e.g. 'until=')
`--force , -f`		Do not prompt for confirmation

Parent command

Command	Description
docker image (https://docs.docker.com/engine/reference/commandline/image)	Manage images

Related commands

Command	Description
docker image build (https://docs.docker.com/engine/reference/commandline/image_build/)	Build an image from a Dockerfile

Command	Description
docker image history (https://docs.docker.com/engine/reference/commandline/image_history/)	Show the history of an image
docker image import (https://docs.docker.com/engine/reference/commandline/image_import/)	Import the contents from a tarball to create a filesystem image
docker image inspect (https://docs.docker.com/engine/reference/commandline/image_inspect/)	Display detailed information on one or more images
docker image load (https://docs.docker.com/engine/reference/commandline/image_load/)	Load an image from a tar archive or STDIN
docker image ls (https://docs.docker.com/engine/reference/commandline/image_ls/)	List images
docker image prune (https://docs.docker.com/engine/reference/commandline/image_prune/)	Remove unused images
docker image pull (https://docs.docker.com/engine/reference/commandline/image_pull/)	Pull an image or a repository from a registry
docker image push (https://docs.docker.com/engine/reference/commandline/image_push/)	Push an image or a repository to a registry
docker image rm (https://docs.docker.com/engine/reference/commandline/image_rm/)	Remove one or more images
docker image save (https://docs.docker.com/engine/reference/commandline/image_save/)	Save one or more images to a tar archive (streamed to STDOUT by default)
docker image tag (https://docs.docker.com/engine/reference/commandline/image_tag/)	Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE

Extended description

Remove all dangling images. If `-a` is specified, will also remove all images not referenced by any container.

Examples

Example output:

```
$ 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
Deleted Images:
untagged: alpine:latest
untagged: alpine@sha256:3dcdb92d7432d56604d4545cbd324b14e647b313626d99b889d0626
de158f73a
deleted: sha256:4e38e38c8ce0b8d9041a9c4fefe786631d1416225e13b0bfe8cfa2321aec4bb
a
deleted: sha256:4fe15f8d0ae69e169824f25f1d4da3015a48feeeeebb265cd2e328e15c6a869
f
untagged: alpine:3.3
untagged: alpine@sha256:4fa633f4feff6a8f02acfc7424efd5cb3e76686ed3218abf4ca0fa4
a2a358423
untagged: my-jq:latest
deleted: sha256:ae67841be6d008a374eff7c2a974cde3934ffe9536a7dc7ce589585eddd83af
f
deleted: sha256:34f6f1261650bc341eb122313372adc4512b4fceddc2a7ecbb84f0958ce5ad6
5
deleted: sha256:cf4194e8d8db1cb2d117df33f2c75c0369c3a26d96725efb978cc69e046b87e
7
untagged: my-curl:latest
deleted: sha256:b2789dd875bf427de7f9f6ae001940073b3201409b14aba7e5db71f408b8569
e
deleted: sha256:96daac0cb203226438989926fc34dd024f365a9a8616b93e168d303cfe4cb5e
9
deleted: sha256:5cbd97a14241c9cd83250d6b6fc0649833c4a3e84099b968dd4ba403e609945
e
deleted: sha256:a0971c4015c1e898c60bf95781c6730a05b5d8a2ae6827f53837e6c9d38efde
c
deleted: sha256:d8359ca3b681cc5396a4e790088441673ed3ce90ebc04de388bfcd31a0716b0
6
deleted: sha256:83fc9ba8fb70e1da31dfcc3c88d093831dbd4be38b34af998df37e8ac538260
c
deleted: sha256:ae7041a4cc625a9c8e6955452f7afe602b401f662671cea3613f08f3d9343b3
5
deleted: sha256:35e0f43a37755b832f0bbea91a2360b025ee351d7309dae0d9737bc96b6d080
9
deleted: sha256:0af941dd29f00e4510195dd00b19671bc591e29d1495630e7e0f7c44c1e6a8c
0
deleted: sha256:9fc896fc2013da84f84e45b3096053eb084417b42e6b35ea0cce5a3529705ea
c
deleted: sha256:47cf20d8c26c46fff71be614d9f54997edacfe8d46d51769706e5aba94b16f2
b
deleted: sha256:2c675ee9ed53425e31a13e3390bf3f539bf8637000e4bcfbb85ee03ef4d910a
1

Total reclaimed space: 16.43 MB
```

Filtering

The filtering flag (`--filter`) format is of “key=value”. If there is more than one filter, then pass multiple flags (e.g., `--filter "foo=bar" --filter "bif=baz"`)

The currently supported filters are:

- until (`<timestamp>`) - only remove images created before given timestamp
- label (`label=<key>` , `label=<key>=<value>` , `label!=<key>` , or `label!=<key>=<value>`) - only remove images with (or without, in case `label!=...` is used) the specified labels.

The `until` filter can be Unix timestamps, date formatted timestamps, or Go duration strings (e.g. `10m` , `1h30m`) computed relative to the daemon machine's time. Supported formats for date formatted time stamps include RFC3339Nano, RFC3339, `2006-01-02T15:04:05` , `2006-01-02T15:04:05.999999999` , `2006-01-02Z07:00` , and `2006-01-02` . The local timezone on the daemon will be used if you do not provide either a `Z` or a `+-00:00` timezone offset at the end of the timestamp. When providing Unix timestamps enter seconds[.nanoseconds], where seconds is the number of seconds that have elapsed since January 1, 1970 (midnight UTC/GMT), not counting leap seconds (aka Unix epoch or Unix time), and the optional .nanoseconds field is a fraction of a second no more than nine digits long.

The `label` filter accepts two formats. One is the `label=...` (`label=<key>` or `label=<key>=<value>`), which removes images with the specified labels. The other format is the `label!=...` (`label!=<key>` or `label!=<key>=<value>`), which removes images without the specified labels.

> ✓ **Predicting what will be removed**
>
> If you are using positive filtering (testing for the existence of a label or that a label has a specific value), you can use `docker image ls` with the same filtering syntax to see which images match your filter.
>
> However, if you are using negative filtering (testing for the absence of a label or that a label does *not* have a specific value), this type of filter does not work with `docker image ls` so you cannot easily predict which images will be removed. In addition, the confirmation prompt for `docker image prune` always warns that *all* dangling images will be removed, even if you are using `--filter` .

The following removes images created before `2017-01-04T00:00:00` :

```
$ docker images --format 'table {{.Repository}}\t{{.Tag}}\t{{.ID}}\t{{.CreatedA
t}}\t{{.Size}}'
REPOSITORY          TAG                 IMAGE ID            CREATED AT
             SIZE
foo                 latest              2f287ac753da        2017-01-04 13:42:23
 -0800 PST   3.98 MB
alpine              latest              88e169ea8f46        2016-12-27 10:17:25
 -0800 PST   3.98 MB
busybox             latest              e02e811dd08f        2016-10-07 14:03:58
 -0700 PDT   1.09 MB

$ docker image prune -a --force --filter "until=2017-01-04T00:00:00"

Deleted Images:
untagged: alpine:latest
untagged: alpine@sha256:dfbd4a3a8ebca874ebd2474f044a0b33600d4523d03b0df76e5c598
6cb02d7e8
untagged: busybox:latest
untagged: busybox@sha256:29f5d56d12684887bdfa50dcd29fc31eea4aaf4ad3bec43daf1902
6a7ce69912
deleted: sha256:e02e811dd08fd49e7f6032625495118e63f597eb150403d02e3238af1df240b
a
deleted: sha256:e88b3f82283bc59d5e0df427c824e9f95557e661fcb0ea15fb0fb6f97760f9d
9

Total reclaimed space: 1.093 MB

$ docker images --format 'table {{.Repository}}\t{{.Tag}}\t{{.ID}}\t{{.CreatedA
t}}\t{{.Size}}'

REPOSITORY          TAG                 IMAGE ID            CREATED AT
             SIZE
foo                 latest              2f287ac753da        2017-01-04 13:42:23
 -0800 PST   3.98 MB
```

The following removes images created more than 10 days (`240h`) ago:

```
$ docker images

REPOSITORY          TAG                 IMAGE ID            CREATED
 SIZE
foo                 latest              2f287ac753da        14 seconds ago
 3.98 MB
alpine              latest              88e169ea8f46        8 days ago
 3.98 MB
debian              jessie              7b0a06c805e8        2 months ago
 123 MB
busybox             latest              e02e811dd08f        2 months ago
 1.09 MB
golang              1.7.0               138c2e655421        4 months ago
 670 MB

$ docker image prune -a --force --filter "until=240h"

Deleted Images:
untagged: golang:1.7.0
untagged: golang@sha256:6765038c2b8f407fd6e3ecea043b44580c229ccfa2a13f6d85866cf
2b4a9628e
deleted: sha256:138c2e6554219de65614d88c15521bfb2da674cbb0bf840de161f89ff4264b9
6
deleted: sha256:ec353c2e1a673f456c4b78906d0d77f9d9456cfb5229b78c6a960bfb7496b76
a
deleted: sha256:fe22765feaf3907526b4921c73ea6643ff9e334497c9b7e177972cf22f68ee9
3
deleted: sha256:ff845959c80148421a5c3ae11cc0e6c115f950c89bc949646be55ed18d6a291
2
deleted: sha256:a4320831346648c03db64149eafc83092e2b34ab50ca6e8c13112388f25899a
7
deleted: sha256:4c76020202ee1d9709e703b7c6de367b325139e74eebd6b55b30a63c196abaf
3
deleted: sha256:d7afd92fb07236c8a2045715a86b7d5f0066cef025018cd3ca9a45498c51d1d
6
deleted: sha256:9e63c5bce4585dd7038d830a1f1f4e44cb1a1515b00e620ac718e934b484c93
8
untagged: debian:jessie
untagged: debian@sha256:c1af755d300d0c65bb1194d24bce561d70c98a54fb5ce5b1693beb4
f7988272f
deleted: sha256:7b0a06c805e8f23807fb8856621c60851727e85c7bcb751012c813f122734c8
d
deleted: sha256:f96222d75c5563900bc4dd852179b720a0885de8f7a0619ba0ac76e92542bbc
8

Total reclaimed space: 792.6 MB

$ docker images

REPOSITORY          TAG                 IMAGE ID            CREATED
  SIZE
foo                 latest              2f287ac753da        About a minute ago
  3.98 MB
alpine              latest              88e169ea8f46        8 days ago
  3.98 MB
busybox             latest              e02e811dd08f        2 months ago
  1.09 MB
```

The following example removes images with the label `deprecated` :

```
$ docker image prune --filter="label=deprecated"
```

The following example removes images with the label `maintainer` set to `john` :

```
$ docker image prune --filter="label=maintainer=john"
```

This example removes images which have no `maintainer` label:

```
$ docker image prune --filter="label!=maintainer"
```

This example removes images which have a maintainer label not set to `john` :

```
$ docker image prune --filter="label!=maintainer=john"
```

> Note: You are prompted for confirmation before the `prune` removes anything, but you are not shown a list of what will potentially be removed. In addition, `docker image ls` does not support negative filtering, so it difficult to predict what images will actually be removed.