docker image pull

Description	Download an image from a registry
Usage	`docker image pull [OPTIONS] NAME[:TAG|@DIGEST]`
Aliases	`docker pull`

Description

Most of your images will be created on top of a base image from the Docker Hub registry.

Docker Hub contains many pre-built images that you can `pull` and try without needing to define and configure your own.

To download a particular image, or set of images (i.e., a repository), use `docker pull`.

Proxy configuration

If you are behind an HTTP proxy server, for example in corporate settings, before open a connect to registry, you may need to configure the Docker daemon's proxy settings, refer to the dockerd command-line reference for details.

Concurrent downloads

By default the Docker daemon will pull three layers of an image at a time. If you are on a low bandwidth connection this may cause timeout issues and you may want to lower this via the `--max-concurrent-downloads` daemon option. See the daemon documentation for more details.

Options

Option	Default	Description
`-a, --all-tags`		Download all tagged images in the repository
`--disable-content-trust`	`true`	Skip image verification
`--platform`		API 1.32+ Set platform if server is multi-platform capable
`-q, --quiet`		Suppress verbose output

Examples

Pull an image from Docker Hub

To download a particular image, or set of images (i.e., a repository), use `docker image pull` (or the `docker pull` shorthand). If no tag is provided, Docker Engine uses the `:latest` tag as a default. This example pulls the `debian:latest` image:

```
$ docker image pull debian

Using default tag: latest
latest: Pulling from library/debian
e756f3fdd6a3: Pull complete
Digest: sha256:3f1d6c17773a45c97bd8f158d665c9709d7b29ed7917ac934086ad96f92e4510
Status: Downloaded newer image for debian:latest
docker.io/library/debian:latest
```

Docker images can consist of multiple layers. In the example above, the image consists of a single layer, `e756f3fdd6a3`.

Layers can be reused by images. For example, the `debian:bookworm` image shares its layer with the `debian:latest`. Pulling the `debian:bookworm` image therefore only pulls its metadata, but not its layers, because the layer is already present locally:

```
$ docker image pull debian:bookworm

bookworm: Pulling from library/debian
Digest: sha256:3f1d6c17773a45c97bd8f158d665c9709d7b29ed7917ac934086ad96f92e4510
Status: Downloaded newer image for debian:bookworm
docker.io/library/debian:bookworm
```

To see which images are present locally, use the `docker images` command:

```
$ docker images

REPOSITORY   TAG        IMAGE ID       CREATED       SIZE
debian       bookworm   4eaceab30377a  8 days ago    124MB
debian       latest     4eaceab30377a  8 days ago    124MB
```

Docker uses a content-addressable image store, and the image ID is a SHA256 digest covering the image's configuration and layers. In the example above, `debian:bookworm` and `debian:latest` have the same image ID because they are the same image tagged with different names. Because they are the same image, their layers are stored only once and do not consume extra disk space.

For more information about images, layers, and the content-addressable store, refer to understand images, containers, and storage drivers.

Pull an image by digest (immutable identifier)

So far, you've pulled images by their name (and "tag"). Using names and tags is a convenient way to work with images. When using tags, you can `docker pull` an image again to make sure you have the most up-to-date version of that image. For example, `docker pull ubuntu:24.04` pulls the latest version of the Ubuntu 24.04 image.

In some cases you don't want images to be updated to newer versions, but prefer to use a fixed version of an image. Docker enables you to pull an image by its digest. When pulling an image by digest, you specify exactly which version of an image to pull. Doing so, allows you to "pin" an image to that version, and guarantee that the image you're using is always the same.

To know the digest of an image, pull the image first. Let's pull the latest `ubuntu:24.04` image from Docker Hub:

```
$ docker pull ubuntu:24.04

24.04: Pulling from library/ubuntu
125a6e411906: Pull complete
Digest: sha256:2e863c44b718727c860746568e1d54afd13b2fa71b160f5cd9058fc436217b30
Status: Downloaded newer image for ubuntu:24.04
docker.io/library/ubuntu:24.04
```

Docker prints the digest of the image after the pull has finished. In the example above, the digest of the image is:

```
sha256:2e863c44b718727c860746568e1d54afd13b2fa71b160f5cd9058fc436217b30
```

Docker also prints the digest of an image when pushing to a registry. This may be useful if you want to pin to a version of the image you just pushed.

A digest takes the place of the tag when pulling an image, for example, to pull the above image by digest, run the following command:

```
$ docker pull ubuntu@sha256:2e863c44b718727c860746568e1d54afd13b2fa71b160f5cd9058fc436217b30

docker.io/library/ubuntu@sha256:2e863c44b718727c860746568e1d54afd13b2fa71b160f5cd9058fc436217b30: Pulling
Digest: sha256:2e863c44b718727c860746568e1d54afd13b2fa71b160f5cd9058fc436217b30
Status: Image is up to date for ubuntu@sha256:2e863c44b718727c860746568e1d54afd13b2fa71b160f5cd9058fc4362
docker.io/library/ubuntu@sha256:2e863c44b718727c860746568e1d54afd13b2fa71b160f5cd9058fc436217b30
```

Digest can also be used in the `FROM` of a Dockerfile, for example:

```
FROM ubuntu@sha256:2e863c44b718727c860746568e1d54afd13b2fa71b160f5cd9058fc436217b30
LABEL org.opencontainers.image.authors="some maintainer <maintainer@example.com>"
```

> **Note**
>
> Using this feature "pins" an image to a specific version in time. Docker does therefore not pull updated versions of an image, which may include security updates. If you want to pull an updated image, you need to change the digest accordingly.

Pull from a different registry

By default, `docker pull` pulls images from Docker Hub. It is also possible to manually specify the path of a registry to pull from. For example, if you have set up a local registry, you can specify its path to pull from it. A registry path is similar to a URL, but does not contain a protocol specifier (`https://`).

The following command pulls the `testing/test-image` image from a local registry listening on port 5000 (`myregistry.local:5000`):

```
$ docker image pull myregistry.local:5000/testing/test-image
```

Registry credentials are managed by docker login.

Docker uses the `https://` protocol to communicate with a registry, unless the registry is allowed to be accessed over an insecure connection. Refer to the insecure registries section for more information.

Pull a repository with multiple images (-a, --all-tags)

By default, `docker pull` pulls a single image from the registry. A repository can contain multiple images. To pull all images from a repository, provide the `-a` (or `--all-tags`) option when using `docker pull`.

This command pulls all images from the `ubuntu` repository:

```
$ docker image pull --all-tags ubuntu

Pulling repository ubuntu
ad57ef8d78d7: Download complete
105182bb5e8b: Download complete
511136ea3c5a: Download complete
73bd853d2ea5: Download complete
....

Status: Downloaded newer image for ubuntu
```

After the pull has completed use the `docker image ls` command (or the `docker images` shorthand) to see the images that were pulled. The example below shows all the `ubuntu` images that are present locally:

```
$ docker image ls --filter reference=ubuntu
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       22.04     8a3cdc4d1ad3   3 weeks ago   77.9MB
ubuntu       jammy     8a3cdc4d1ad3   3 weeks ago   77.9MB
ubuntu       24.04     35a888802559d  6 weeks ago   78.1MB
ubuntu       latest    35a888802559d  6 weeks ago   78.1MB
ubuntu       noble     35a888802559d  6 weeks ago   78.1MB
```

Cancel a pull

Killing the `docker pull` process, for example by pressing `CTRL-c` while it is running in a terminal, will terminate the pull operation.

```
$ docker pull ubuntu

Using default tag: latest
latest: Pulling from library/ubuntu
a3ed95caeb02: Pulling fs layer
236608c7b546: Pulling fs layer
^C
```

The Engine terminates a pull operation when the connection between the daemon and the client (initiating the pull) is cut or lost for any reason or the command is manually terminated.