# Layers - Building Blocks in Docker

Layers - speed up the build and pull process of an image by breaking down the components into minor pieces.

**We'll cover the following**

## Layers #

In the last chapter, we saw that when we pulled the Docker hello-world image, it pulled a layer. The hello-world image is a very small-sized image. So, Docker pulled only one layer. If you pull a bigger image, say, Nginx we get something like this:

```
venkateshs-air:~ venkateshachintalwar$ docker pull nginx
Using the default tag: latest
latest: Pulling from library/nginx
b8f262c62ec6: Pull complete
e9218e8f93b1: Pull complete
7acba7289aa3: Pull complete
Digest: sha256:aeded0f2a861747f43a01cf1018cf9efe2bdd02afd57d2b11fcc7fcadc16ccd
1
```

Docker pulled three layers. If you think a little bit, you asked the Docker daemon to pull one image, which is Nginx. Why is it pulling three?

The reason is whenever the Docker pulls an image, it takes up a lot of space on the device. If you type

```
venkateshs-air:~ venkateshachintalwar$ docker image ls nginx
REPOSITORY          TAG                 IMAGE ID            CREATED
             SIZE
nginx               latest              f949e7d76d63        8 days ag
o          126MB
```

You can see above that the Nginx image took 126 MB of space. If images were built as a standalone image with no component sharing, every image would have taken a .iso size memory, which is not efficient.

As every image is built on top of Linux kernel, it has some common dependencies that can be reused by other images. Docker bundles these dependencies in one stack and these stacks are called layers.

> Only the instructions RUN, COPY, and ADD create layers. Other instructions create temporary intermediate images and do not increase the size of the build. We will see in the next section what these instructions are.

Docker caches these intermediate layers to speed up the image building process. We will see that in the upcoming lessons.

Before moving to the next lesson, try pulling a Python 3.5 image using `$ sudo docker pull python:3.5` and see how many layers Docker fetches.