Build your own image

Estimated reading time: 9 minutes

The `whalesay` image could be improved. It would be nice if you didn't have to think of something to say and you didn't need to type as much to get `whalesay` to talk.

```
docker run docker/whalesay cowsay boo-boo
```

In this next section, you will improve the `whalesay` image by building a new version that "talks on its own" and requires fewer words to run.

Step 1: Write a Dockerfile

In this step, you use a text editor to write a short Dockerfile (https://docs.docker.com/engine/reference/builder/). A Dockerfile is a recipe which describes the files, environment, and commands that make up an image. Your recipe is going to be very short.

You run these steps in a terminal window on Linux or macOS, or a command prompt on Windows. Remember that if you are using macOS or Windows, you are still creating an image which runs on Linux.

1. Make a new directory. If you're on Windows, use `md` instead of `mkdir`.

   ```
   $ mkdir mydockerbuild
   ```

   This directory will contain all the things you need to build your image. Right now, it's empty.

2. Change to your new directory. Whether you're on Linux, macOS, or Windows, the `cd` command is the same.

```
$ cd mydockerbuild
```

3. Edit a new text file named `Dockerfile` in the current directory, using a text editor such as `nano` or `vi` on Linux or Mac, or `notepad` on Windows.

   Linux or Mac:

   ```
   $ nano Dockerfile
   ```

   Windows:

   ```
   C:\> notepad Dockerfile.
   ```

4. Add a `FROM` statement by copying the following line into the file:

   ```
   FROM docker/whalesay:latest
   ```

   The `FROM` keyword tells Docker which image your image is based on. Whalesay is cute and has the `cowsay` program already, so we'll start there.

5. Add a `RUN` statement which will install the `fortunes` program into the image.

   ```
   RUN apt-get -y update && apt-get install -y fortunes
   ```

   The `whalesay` image is based on Ubuntu, which uses `apt-get` to install packages. These two commands refresh the list of packages available to the image and install the `fortunes` program into it. The `fortunes` program prints out wise sayings for our whale to say.

6. Add a `CMD` statement, which tells the image the final command to run after its environment is set up. This command runs `fortune -a` and sends its output to the `cowsay` command.

```
CMD /usr/games/fortune -a | cowsay
```

7. Check your work. Your file should look just like this:

```
FROM docker/whalesay:latest
RUN apt-get -y update && apt-get install -y fortunes
CMD /usr/games/fortune -a | cowsay
```

8. Save the file and close the text editor. At this point, your software recipe is described in the `Dockerfile` file. You are ready to build a new image.

Step 2: Build an image from your Dockerfile

While you are in the `mydockerbuild` directory, build the image using the `docker build` command. The `-t` parameter gives your image a tag, so you can run it more easily later. Don't forget the `.` command, which tells the `docker build` command to look in the current directory for a file called `Dockerfile`. This command works the same in Linux, macOS, or Windows.

```
$ docker build -t docker-whale .

Sending build context to Docker daemon 2.048 kB
...snip...
Removing intermediate container cb53c9d09f3b
Successfully built c2c3152907b5
```

The command takes several seconds to run and reports its outcome. Keep reading to learn a little more about what the output means and what happens when you build an image.

Step 3: Learn about the build process

The `docker build -t docker-whale .` command reads the `Dockerfile` in the current directory and processes its instructions one by one to build an image called `docker-whale` on your local machine. The command takes about a minute and its output looks really long and complex. In this section, you learn what each message means.

1. Docker checks to make sure it has everything it needs to build. This generates this message:

   ```
   Sending build context to Docker daemon 2.048 kB
   ```

2. Docker checks to see whether it already has the `whalesay` image locally and pulls it from Docker hub if not. In this case, the image already exists locally because you pulled it in a previous task. This corresponds to `FROM` statement in the Dockerfile, and generates this message:

   ```
   Step 1 : FROM docker/whalesay:latest
    ---> 6b362a9f73eb
   ```

   At the end of each step, an ID is printed. This is the ID of the layer that was created by this step. Each line in a Dockerfile corresponds to a layer in the image. Your ID will be different.

3. Docker starts up a temporary container running the `whalesay` image (the `Running in` line below). In the temporary container, Docker runs the next command in the Dockerfile, which is the `RUN` command, which installs the `fortune` command. This generates a lot of lines of output, just like you would see if you were manually running the `apt-get` commands on an Ubuntu host.

   ```
   Step 2 : RUN apt-get -y update && apt-get install -y fortunes
    ---> Running in 05d4eda04526
   Ign http://archive.ubuntu.com trusty InRelease
   Get:1 http://archive.ubuntu.com trusty-updates InRelease [65.9 kB]
   Get:2 http://archive.ubuntu.com trusty-security InRelease [65.9 kB]
   Hit http://archive.ubuntu.com trusty Release.gpg
   Hit http://archive.ubuntu.com trusty Release
   Get:3 http://archive.ubuntu.com trusty-updates/main Sources [480 kB]
   Get:4 http://archive.ubuntu.com trusty-updates/restricted Sources [5
   921 B]
   Get:5 http://archive.ubuntu.com trusty-updates/universe Sources [214
    kB]
   Get:6 http://archive.ubuntu.com trusty-updates/main amd64 Packages [
   1160 kB]
   Get:7 http://archive.ubuntu.com trusty-updates/restricted amd64 Pack
   ages [20.4 kB]
   Get:8 http://archive.ubuntu.com trusty-updates/universe amd64 Packag
   ```

```
Get:8 http://archive.ubuntu.com trusty-updates/universe amd64 Packag
es [505 kB]
Get:9 http://archive.ubuntu.com trusty-security/main Sources [157 kB
]
Get:10 http://archive.ubuntu.com trusty-security/restricted Sources
[4621 B]
Get:11 http://archive.ubuntu.com trusty-security/universe Sources [5
4.5 kB]
Get:12 http://archive.ubuntu.com trusty-security/main amd64 Packages
 [700 kB]
Get:13 http://archive.ubuntu.com trusty-security/restricted amd64 Pa
ckages [17.0 kB]
Get:14 http://archive.ubuntu.com trusty-security/universe amd64 Pack
ages [191 kB]
Hit http://archive.ubuntu.com trusty/main Sources
Hit http://archive.ubuntu.com trusty/restricted Sources
Hit http://archive.ubuntu.com trusty/universe Sources
Hit http://archive.ubuntu.com trusty/main amd64 Packages
Hit http://archive.ubuntu.com trusty/restricted amd64 Packages
Hit http://archive.ubuntu.com trusty/universe amd64 Packages
Fetched 3640 kB in 11s (329 kB/s)
Reading package lists...
Reading package lists...
Building dependency tree...
Reading state information...
The following extra packages will be installed:
  fortune-mod fortunes-min librecode0
Suggested packages:
  x11-utils bsdmainutils
The following NEW packages will be installed:
  fortune-mod fortunes fortunes-min librecode0
0 upgraded, 4 newly installed, 0 to remove and 92 not upgraded.
Need to get 1961 kB of archives.
After this operation, 4817 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu/ trusty/main librecode0 amd64
 3.6-21 [771 kB]
Get:2 http://archive.ubuntu.com/ubuntu/ trusty/universe fortune-mod
amd64 1:1.99.1-7 [39.5 kB]
Get:3 http://archive.ubuntu.com/ubuntu/ trusty/universe fortunes-min
 all 1:1.99.1-7 [61.8 kB]
Get:4 http://archive.ubuntu.com/ubuntu/ trusty/universe fortunes all
 1:1.99.1-7 [1089 kB]
debconf: unable to initialize frontend: Dialog
debconf: (TERM is not set, so the dialog frontend is not usable.)
debconf: falling back to frontend: Readline
debconf: unable to initialize frontend: Readline
debconf: (This frontend requires a controlling tty.)
debconf: falling back to frontend: Teletype
dpkg-preconfigure: unable to re-open stdin:
Fetched 1961 kB in 19s (101 kB/s)
Selecting previously unselected package librecode0:amd64.
(Reading database ... 13116 files and directories currently install
(Reading database ... 13116 files and directories currently installe
d.)
Preparing to unpack .../librecode0_3.6-21_amd64.deb ...
Unpacking librecode0:amd64 (3.6-21) ...
Selecting previously unselected package fortune-mod.
Preparing to unpack .../fortune-mod_1%3a1.99.1-7_amd64.deb ...
Unpacking fortune-mod (1:1.99.1-7) ...
Selecting previously unselected package fortunes-min.
Preparing to unpack .../fortunes-min_1%3a1.99.1-7_all.deb ...
Unpacking fortunes-min (1:1.99.1-7) ...
Selecting previously unselected package fortunes.
Preparing to unpack .../fortunes_1%3a1.99.1-7_all.deb ...
Unpacking fortunes (1:1.99.1-7) ...
Setting up librecode0:amd64 (3.6-21) ...
Setting up fortune-mod (1:1.99.1-7) ...
Setting up fortunes-min (1:1.99.1-7) ...
Setting up fortunes (1:1.99.1-7) ...
Processing triggers for libc-bin (2.19-0ubuntu6.6) ...
 ---> dfaf993d4a2e
Removing intermediate container 05d4eda04526
```

When the `RUN` command finishes, a new layer is created and the intermediate container is removed.

4. A new intermediate container is created, and Docker adds a layer for the `CMD` line in the Dockerfile, and removes the intermediate container.

```
Step 3 : CMD /usr/games/fortune -a | cowsay
 ---> Running in a8e6faa88df3
 ---> 7d9495d03763
Removing intermediate container a8e6faa88df3
Successfully built 7d9495d03763
```

You have now built an image called `docker-whale`.

Step 4: Run your new docker-whale

Now you can verify that the new image is on your computer and you can run it.

1. In a terminal window or command prompt, type `docker images`, which lists the images you have locally.

```
$ docker images

REPOSITORY           TAG          IMAGE ID          CREATED
     SIZE
docker-whale         latest       c2c3152907b5      4 minutes ago
     275.1 MB
docker/whalesay      latest       fb434121fc77      4 hours ago
     247 MB
hello-world          latest       91c95931e552      5 weeks ago
     910 B
```

2. Run your new image by typing `docker run docker-whale`.

```
$ docker run docker-whale

 ______________________________________
< You will be successful in your work. >
 --------------------------------------
    \
     \
      \
                    ##        .
              ## ## ##       ==
           ## ## ## ##      ===
       /""""""""""""""""___/ ===
  ~~~ {~~ ~~~~ ~~~ ~~~~ ~~ ~ /  ===- ~~~
       \______ o          __/
        \    \        __/
          \____\______/
```

The whale is now a lot smarter. It comes up with its own thoughts and it takes less typing to run. You may also notice that Docker didn’t have to download anything, because you built the image locally.

3. Just for fun, run it again.

```
 _________________________________________
/ If everything seems to be going well, \
| you have obviously overlooked         |
\ something.                            /
 -----------------------------------------
    \
     \
      \
                    ##        .
              ## ## ##       ==
           ## ## ## ##      ===
       /""""""""""""""""___/ ===
  ~~~ {~~ ~~~~ ~~~ ~~~~ ~~ ~ /  ===- ~~~
       \______ o          __/
        \    \        __/
          \____\______/
```

Your whale now has a mind of its own, but if you don't like what it says, just run it again!

Where to go next

Now that you have learned to build an image and run it, you can learn to share the image by creating a Docker Hub account (https://docs.docker.com/engine/getstarted/step_five/).

What is Docker (https://www.docker.com/what-docker)

What is a Container (https://www.docker.com/what-container)

Use Cases (https://www.docker.com/use-cases)

Customers (https://www.docker.com/customers)

Partners (https://www.docker.com/partners/partner-program)

For Government (https://www.docker.com/industry-government)

About Docker (https://www.docker.com/company)

Management (https://www.docker.com/company/management)

Press & News (https://www.docker.com/company/news-and-press)

Careers (https://www.docker.com/careers)

Product (https://www.docker.com/products/overview)

Pricing (https://www.docker.com/pricing)

Community Edition (https://www.docker.com/docker-community)

Enterprise Edition (https://www.docker.com/enterprise)

Docker Datacenter (https://www.docker.com/products/docker-datacenter)

Docker Cloud (https://cloud.docker.com/)

Docker Store (https://store.docker.com/)

Docker for Mac (https://www.docker.com/docker-mac)

Docker for Windows (PC) (https://www.docker.com/docker-windows)

Docker for AWS (https://www.docker.com/docker-aws)

Docker for Azure (https://www.docker.com/docker-microsoft-azure)

Docker for Windows Server (https://www.docker.com/docker-windows-server)

Docker for CentOS distribution (https://www.docker.com/docker-centos)

Docker for Debian (https://www.docker.com/docker-debian)

Docker for Fedora® (https://www.docker.com/docker-fedora)

Docker for Oracle Enterprise Linux (https://www.docker.com/docker-oracle-linux)

Docker for RHEL (https://www.docker.com/docker-rhel)