Solution

The solution to the first exercise

Before looking at the solution, I want you to try the exercise yourself. If you are able to solve the exercise, verify your answer with the solution. This solution is a set of suggested commands. It's not necessary that you follow the exact steps to get the desired output.

By any chance, if you are not able to solve the exercise, please go through the lessons one more time and try again. The exercise only tests what you have learned in this section.

Let's go through the commands to get the solution.

1. We have to pull a Python 3.8 image.

Python is the base image and 3.8 is the version. There are different versions of Python 3.8 images such as alpine and rc-buster. You can pull any of that by looking at the exact name on Docker Hub. For this solution, I'll use `python:3.8` only.

`docker pull python:3.8`

```
root@educative:/# docker pull python:3.8
3.8: Pulling from library/python
90fe46dd8199: Pull complete
35a4f1977689: Pull complete
bbc37f14aded: Pull complete
74e27dc593d4: Pull complete
4352dcff7819: Pull complete
deb569b08de6: Pull complete
98fd06fa8c53: Pull complete
7b9cc4fdefe6: Pull complete
e8e1fd64f499: Pull complete
Digest: sha256:adcfb73e4ca83b126cc3275f3851c73aecca20e59a48782e9ddebb3a88e57f9
6
Status: Downloaded newer image for python:3.8
```

Here, you can see the number of layers it has pulled.

2. In this step, we have to create a volume named app_files.

Use the `docker volume create <volume_name>` command like so:

```
root@educative:/# docker volume create app_files
app_files
root@educative:/# docker volume ls
DRIVER              VOLUME NAME
local               app_files
```

3. Next, we will run the container with a created volume attached to the container named "first_container".

Type `docker inspect app_files` to know the path of the volume to mount it on the container filesystem.

Use `docker run -it --name <name-of-the-container> -v <volume>:<container-path> <image> <command>`

- `-it` this is used to get an interactive terminal from the container
- `--name`: you can specify the name of a container using this option
- `-v`: this is used to bind volumes to the container filesystem.

```
root@educative:/# docker run -it --name "first_container" -v /usercode/app_fil
es:/app_files python:3.8 bash
root@266b0d1999cf:/# ls
app_files  boot  etc   lib    media  opt   root  sbin  sys  usr
bin        dev   home  lib64  mnt    proc  run   srv   tmp  var
root@266b0d1999cf:/# cd app_files
root@266b0d1999cf:/app_files# pwd
/app_files
```

4. We will write a program in the current_time.py file to print current time.

`cd app_files && touch current_time.py`

> You will need to install an editor inside the container to write the code to your file.

Type and run `apt update` and `apt install nano`. This will install a basic editor in the container. Now, edit the file using `nano current_time.py`

> If you have printed time without a timezone, that's fine. You can print a whole timestamp as it is. The idea is to have a script that we can modify from the second container.

I will be printing time with a timestamp, for that, I need to install a library called 'pytz'.

`pip install pytz`

```
import datetime, pytz

current_time = datetime.datetime.now(pytz.timezone('Asia/Kolkata')).strftime("%H:%M:%S %z")

print(current_time)
```

The output of the code above will be:

```
# python current_time.py
16:46:10 +0530
```

Now, exit the container and proceed to the next step.

5. Create a second container and modify the script to print only the date.

Run the exact command you used for creating the first container. The only change is the container name. Change it to "second_container".

```
root@educative:/# docker run -it --name "second_container" -v /usercode/app_fi
les:/app_files python:3.8 bash
```

Install nano here as well and change the script to print the current date.

```
import datetime

current_date = datetime.datetime.now().strftime("%m-%d-%Y")

print(current_date)
```

Once the changes are done, try to run the script using `python current_time.py`.

```
# python current_time.py
```

```
# python current_time.py
04-26-2020
```

Well, the name file does not go with the output but we will let it be because we completed the solution.

Hurray!! You have done it. It was a long lesson, but you made it. You can also commit the container and push the image from Docker Hub into your repository.

So, take a break. Have a coffee. Once you absorb what you have done, proceed to the next section.