

Module 11 Pre-Read - Docker Part 1

Sunday, April 7, 2024 19:12

Images vs. containers

Docker image

- source code, libraries, dependencies, tools, and other files needed to run a container.
- a **blueprint** for the environment in which you will execute your analysis.

Docker container

- created when we run an image.
- construct a run-time environment to execute your code.
- or provide an interactive way to execute code.



Image: the "blueprint" for a container

Container: a computer inside a computer

```
docker run -it ubuntu bash
```

Create and run a container

-it is interactive mode

```
apt-get install -y vim
```

The apt-get command is to install software using ubuntu container

-y says "yes" to any prompts following installation e.g. if u need to install another package to use vim

```
exit
```

Will exit the container and it will no longer exist and will be destroyed

We call this as the container being TRANSIENT

DOCKERFILES

Cuz we don't always want the container to be destroyed after exiting, so we create an image so we can always come back to it

```
David Benkeser@DESKTOP-0776JQC MINGW64 ~/Documents/docker_files
$ mkdir my_first_docker_image

David Benkeser@DESKTOP-0776JQC MINGW64 ~/Documents/docker_files
$ cd my_first_docker_image/

David Benkeser@DESKTOP-0776JQC MINGW64 ~/Documents/docker_files/my_first_docker_image
$ touch Dockerfile
```

In the Dockerfile, write all of the instructions to build the image

```
Dockerfile x
1 FROM ubuntu
2
3 RUN apt-get update
4 RUN apt-get install -y pandoc
5 RUN apt-get install -y vim
```

FROM is what image you are building this on top of, e.g. ubuntu

RUN is start from ubuntu image, then run these particular commands e.g. run apt-get install pandoc

You can type comments the same as in R such as # comment

```
docker build -t ubuntu_plus .
```

This builds an image

The -t means you're giving the image a name

ubuntu-plus is the name

. means our current working directory to look for the docker file

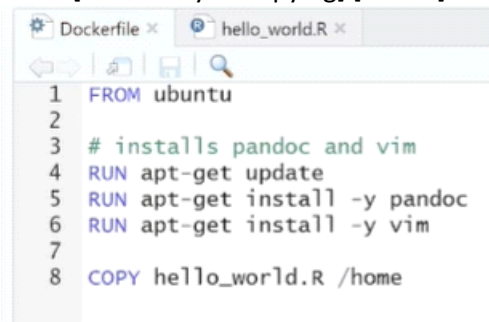
To check that our image is built, type `docker image ls`

```
docker run -it ubuntu_plus bash
```

Enters the image container we just created

Copying files into an image

COPY [what are you copying] [where]



Then in terminal, do `docker build -t ubuntu_plus_file .` if u need a new image for this

To check if it worked:

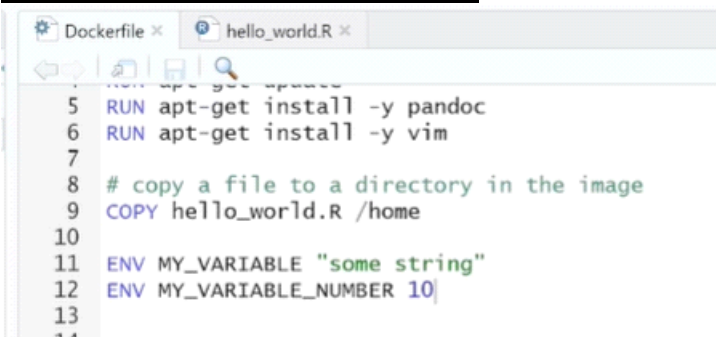
```
docker image ls
```

```
docker run -it ubuntu_plus_file bash
```

```
ls
```

```
cd home
```

ENV variables in an image at build time



ENV sets the value of an environment variable in our image

ENV VARIABLENAME "the value" or number

Then in terminal, do `docker build -t ubuntu_plus_file_env .` if u need a new image for this

To check if it worked:

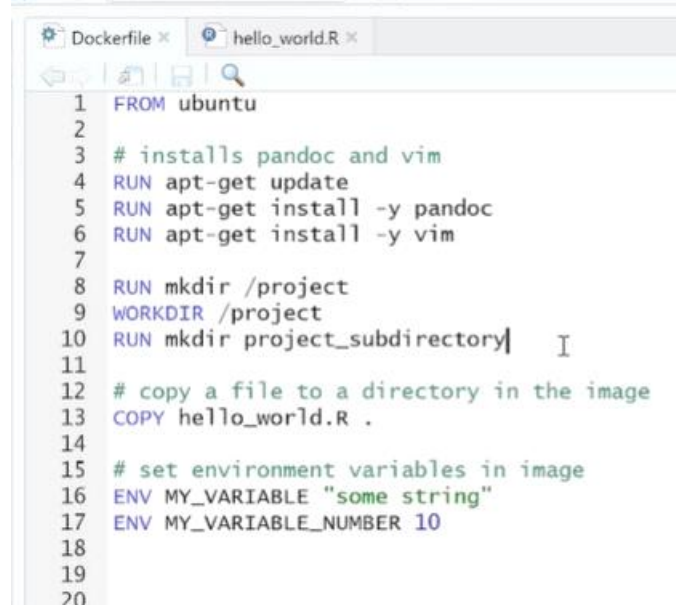
```
docker image ls
```

```
docker run -it ubuntu_plus_file_env bash
```

```
ls
cd home
```

Setting a WORKDIR during build process

e.g. if you need to copy a file into an image, it might be hard to remember what's there and stuff



```
Dockerfile x hello_world.R x
1 FROM ubuntu
2
3 # installs pandoc and vim
4 RUN apt-get update
5 RUN apt-get install -y pandoc
6 RUN apt-get install -y vim
7
8 RUN mkdir /project
9 WORKDIR /project
10 RUN mkdir project_subdirectory
11
12 # copy a file to a directory in the image
13 COPY hello_world.R .
14
15 # set environment variables in image
16 ENV MY_VARIABLE "some string"
17 ENV MY_VARIABLE_NUMBER 10
18
19
20
```

`RUN mkdir /project`

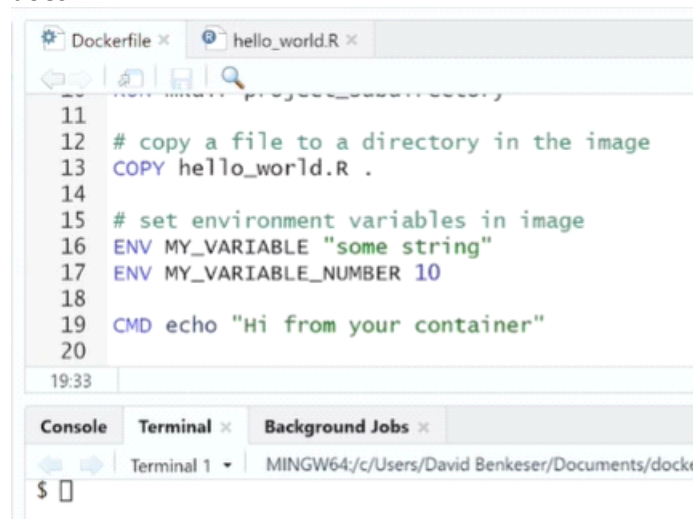
Makes a new directory within the root directory called `/project`

`WORKDIR /project`

makes it the project directory

Setting and overriding an entry point for a container

CMD command in docker file defines the entry point in container, aka what your container automatically does



```
Dockerfile x hello_world.R x
11
12 # copy a file to a directory in the image
13 COPY hello_world.R .
14
15 # set environment variables in image
16 ENV MY_VARIABLE "some string"
17 ENV MY_VARIABLE_NUMBER 10
18
19 CMD echo "Hi from your container"
20
19:33
Console Terminal x Background Jobs x
Terminal 1 MINGW64/c/Users/David Benkeser/Documents/docke
$
```

To check if this worked, do `docker build -t ubuntu_plus_cmd .`

`docker run ubuntu_plus_cmd` (leave off `-it` and `bash`)

The default thing you put in should run

Now to override it, do `docker run -it ubuntu_plus_cmd bash`

`-it` means running in interactive mode, terminal is `bash` and means override our cmd and we want a bash terminal

Removing images

In terminal,

docker image ls to list the images we have

docker image rm a4asdlfkj948r

After the rm put the image id

```
terminal 1  ~/Documents/docker_files/my_first_docker_image
$ docker image ls
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
ubuntu_plus_cmd     latest             6d28581532c8       25 minutes ago     333MB
ubuntu_workdir      latest             416d0f9d657d       27 minutes ago     333MB
ubuntu_plus_file    latest             72cd18500a9a       38 minutes ago     333MB
ubuntu_plus_file_env latest             29139e24572c       38 minutes ago     333MB
ubuntu_plus         latest             f6b65d4fbb35       53 minutes ago     333MB
ubuntu              latest             cdb68b455a14       8 days ago         77.8MB
hello-world         latest             feb5d9fea6a5       13 months ago      13.3kB

David Benkeser@DESKTOP-0776JQC MINGW64 ~/Documents/docker_files/my_first_docker_i
$ docker image rm -f 6d28581532c8 416d0f9d657d
Untagged: ubuntu_plus_cmd:latest
Deleted: sha256:6d28581532c820b66e1d8ddf41f59b3011
Untagged: ubuntu_workdir:latest
Deleted: sha256:416d0f9d657dd8bdba941ddb22acfa6032

David Benkeser@DESKTOP-0776JQC MINGW64 ~/Documents
$ docker image ls
REPOSITORY          TAG                 IMAGE ID            CR
ubuntu_plus_file    latest             72cd18500a9a       39
ubuntu_plus_file_env latest             29139e24572c       39
ubuntu_plus         latest             f6b65d4fbb35       53
```



To remove ALL images (dangerous!)

docker system prune -a