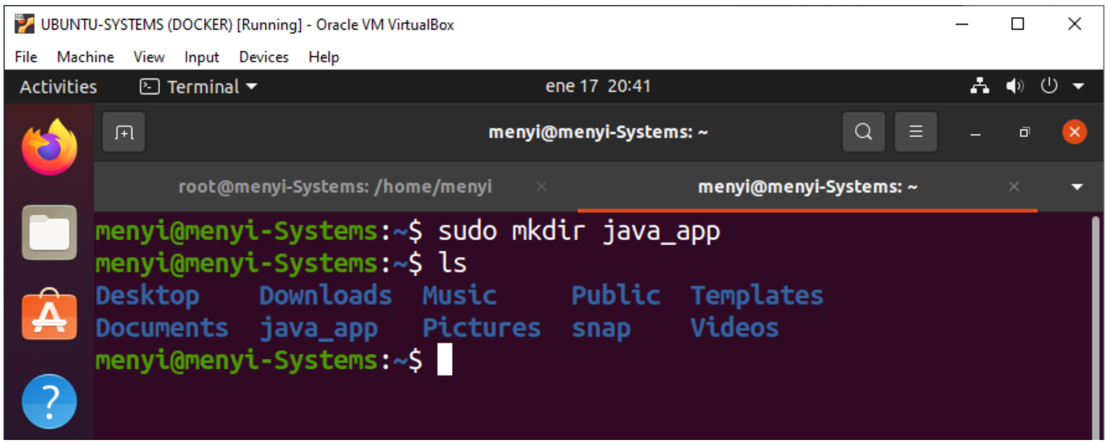
UNIT 04: Basic concepts of operating systems | DOCKER Exercises

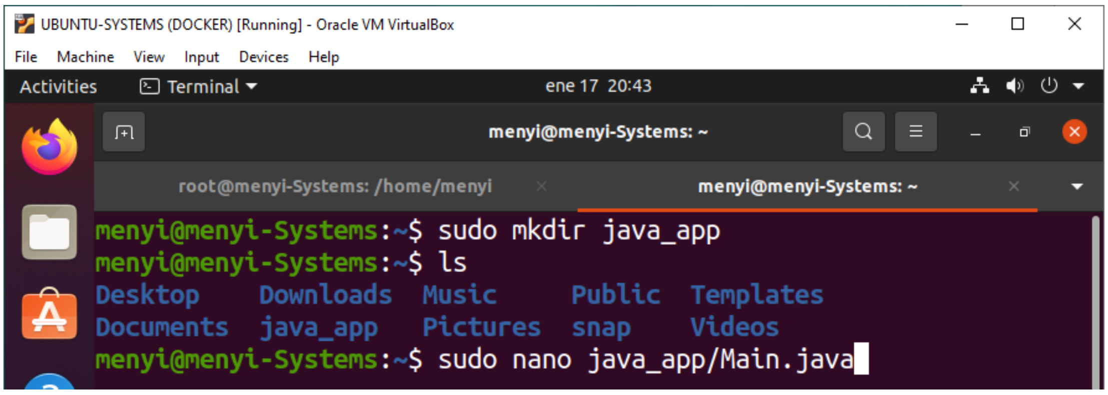
RUNNING A JAVA APPLICATION ON DOCKER

Let's create a folder called ***java_app*** with a single file ***Main.java*** whose content is:

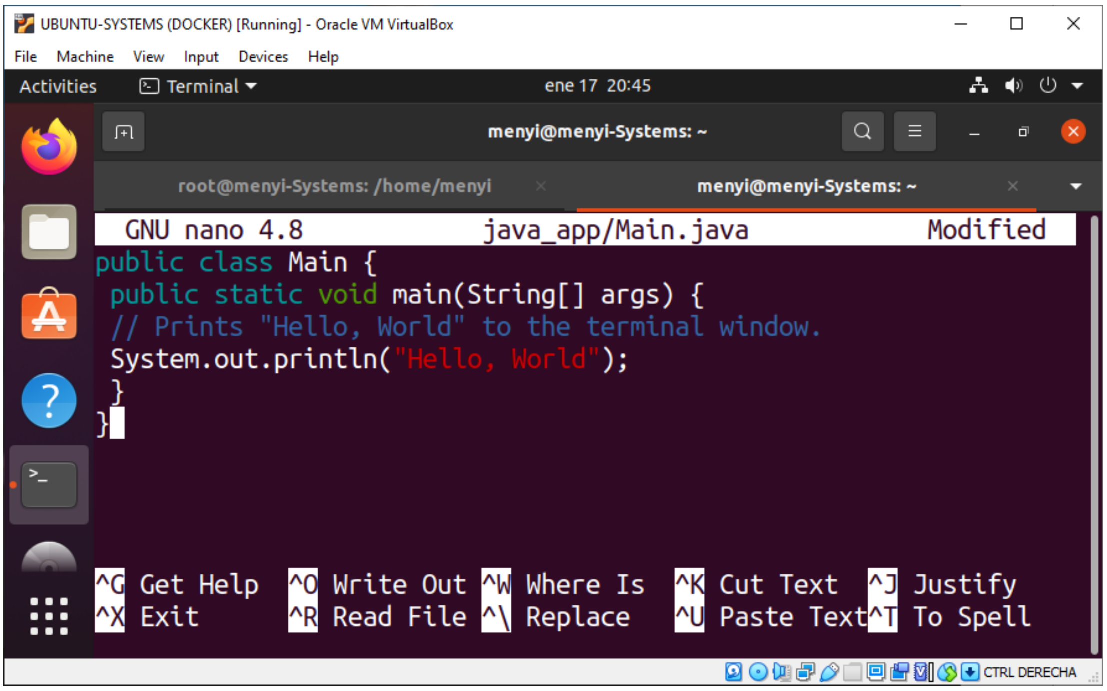
```
public class Main {
   public static void main(String[] args) {
    // Prints "Hello, World" to the terminal window.
   System.out.println("Hello, World");
   }
}
```


UBUNTU-SYSTEMS (DOCKER) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal
ene 17 20:41
menyi@menyi-Systems: ~
root@menyi-Systems: /home/menyi
menyi@menyi-Systems: ~

```
menyi@menyi-Systems:~$ sudo mkdir java_app
menyi@menyi-Systems:~$ ls
Desktop    Downloads  Music     Public  Templates
Documents  java_app   Pictures  snap    Videos
menyi@menyi-Systems:~$
```

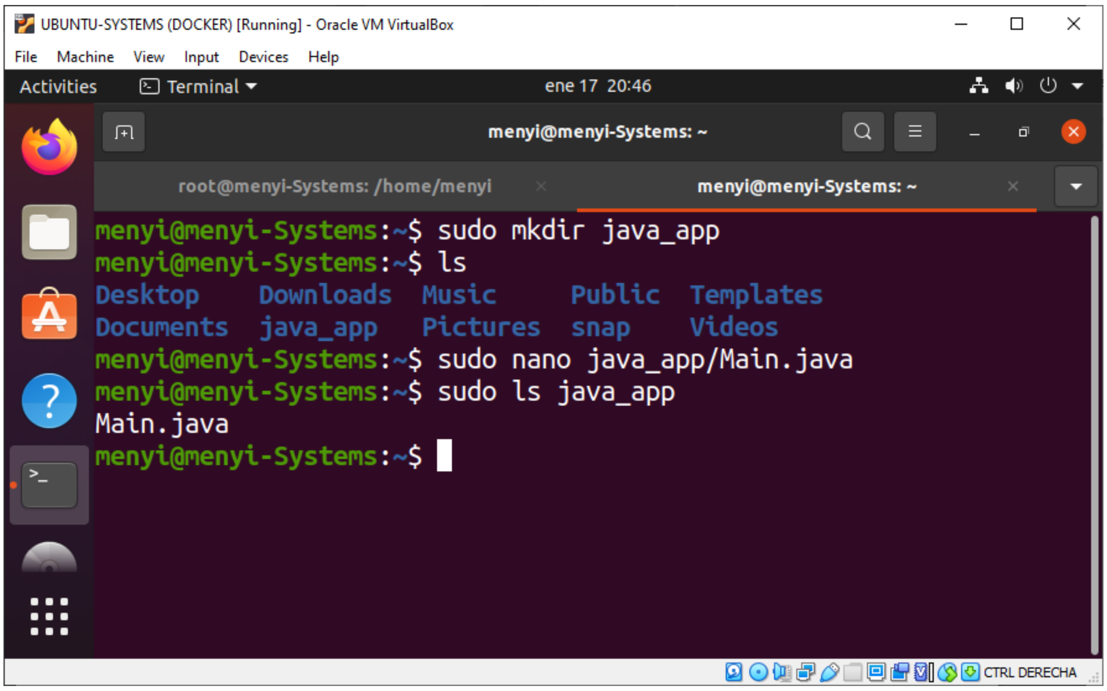

UBUNTU-SYSTEMS (DOCKER) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal
ene 17 20:43
menyi@menyi-Systems: ~
root@menyi-Systems: /home/menyi
menyi@menyi-Systems: ~

```
menyi@menyi-Systems:~$ sudo mkdir java_app
menyi@menyi-Systems:~$ ls
Desktop    Downloads  Music     Public  Templates
Documents  java_app   Pictures  snap    Videos
menyi@menyi-Systems:~$ sudo nano java_app/Main.java
```


UBUNTU-SYSTEMS (DOCKER) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal ene 17 20:45
menyi@menyi-Systems: ~
root@menyi-Systems: /home/menyi
menyi@menyi-Systems: ~
CTRL DERECHA

```
  GNU nano 4.8            java_app/Main.java            Modified
public class Main {
 public static void main(String[] args) {
 // Prints "Hello, World" to the terminal window.
 System.out.println("Hello, World");
 }
}

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell
```


UBUNTU-SYSTEMS (DOCKER) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal ene 17 20:46
menyi@menyi-Systems: ~
root@menyi-Systems: /home/menyi
menyi@menyi-Systems: ~
CTRL DERECHA

```
menyi@menyi-Systems:~$ sudo mkdir java_app
menyi@menyi-Systems:~$ ls
Desktop    Downloads  Music     Public  Templates
Documents  java_app   Pictures  snap    Videos
menyi@menyi-Systems:~$ sudo nano java_app/Main.java
menyi@menyi-Systems:~$ sudo ls java_app
Main.java
menyi@menyi-Systems:~$
```

1. RUN USING A DOCKERFILE

It is possible to create an image which compiles and runs the file.

We will use the last ***openjdk*** version.

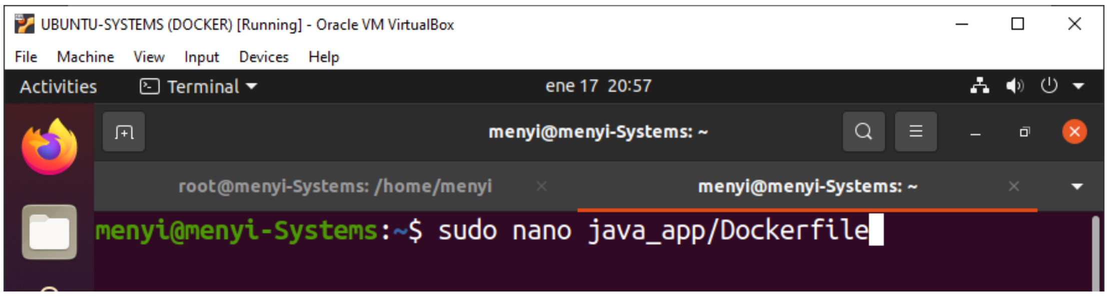
The image will be created in the folder ***java_app***

Create and save file ***Dockerfile*** in ***java_app*** folder with the following code:

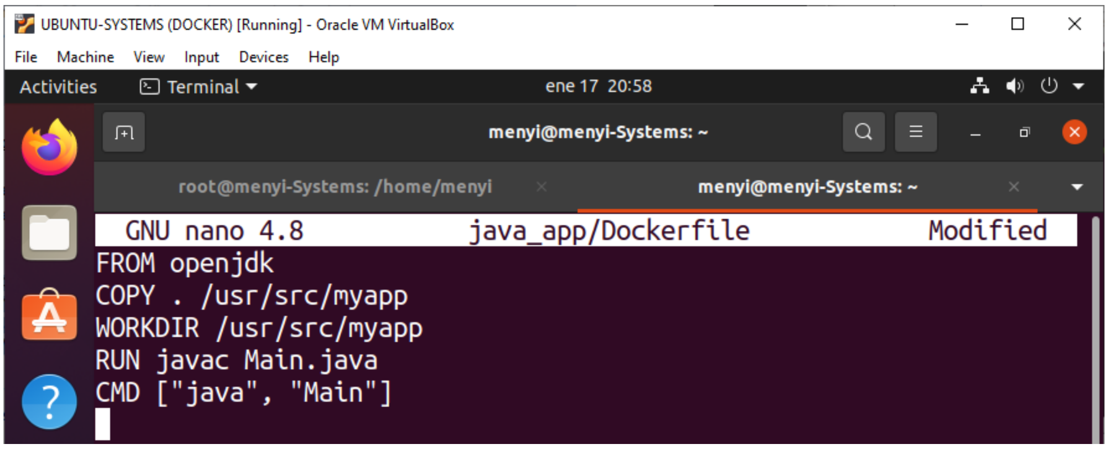
```
    FROM openjdk
    COPY . /usr/src/myapp
    WORKDIR /usr/src/myapp
    RUN javac Main.java
CMD ["java", "Main"]
```

The code basically copies the contents from the current directory into ***/usr/src/myapp.***

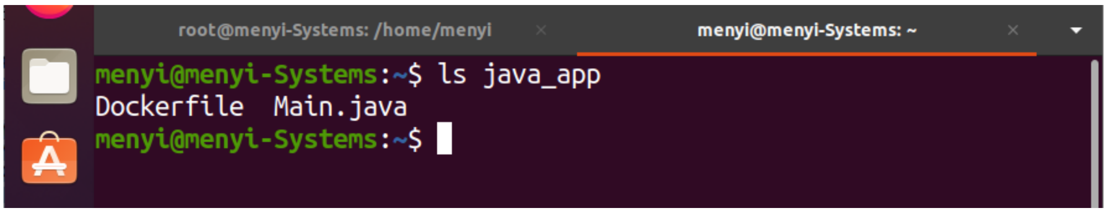
Then, the Java application will be complied and run.


UBUNTU-SYSTEMS (DOCKER) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal ene 17 20:57
menyi@menyi-Systems: ~
root@menyi-Systems: /home/menyi
menyi@menyi-Systems: ~

```
menyi@menyi-Systems:~$ sudo nano java_app/Dockerfile
```

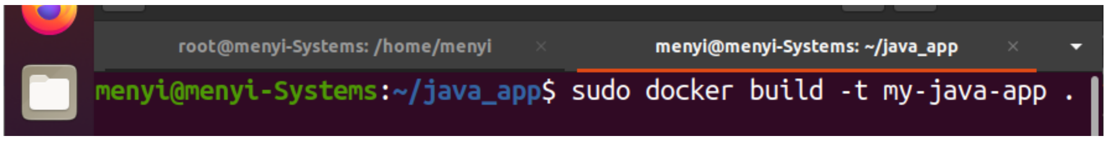

UBUNTU-SYSTEMS (DOCKER) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal ene 17 20:58
menyi@menyi-Systems: ~
root@menyi-Systems: /home/menyi
menyi@menyi-Systems: ~

```
  GNU nano 4.8              java_app/Dockerfile              Modified
FROM openjdk
COPY . /usr/src/myapp
WORKDIR /usr/src/myapp
RUN javac Main.java
CMD ["java", "Main"]
```

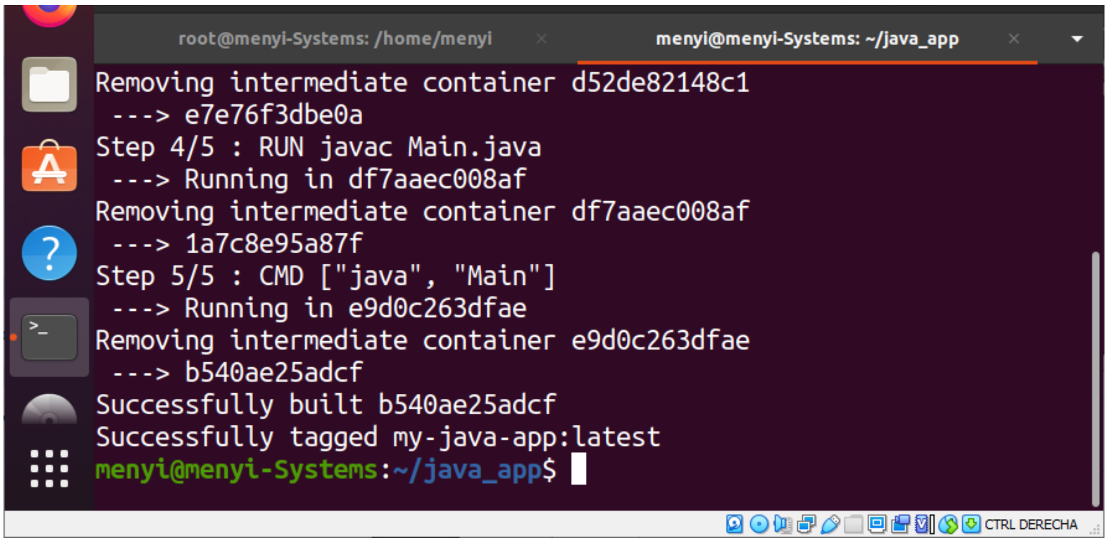


```
menyi@menyi-Systems:~$ ls java_app
Dockerfile  Main.java
menyi@menyi-Systems:~$
```

Now, let’s build the ***image*** from the current folder ***java_app***

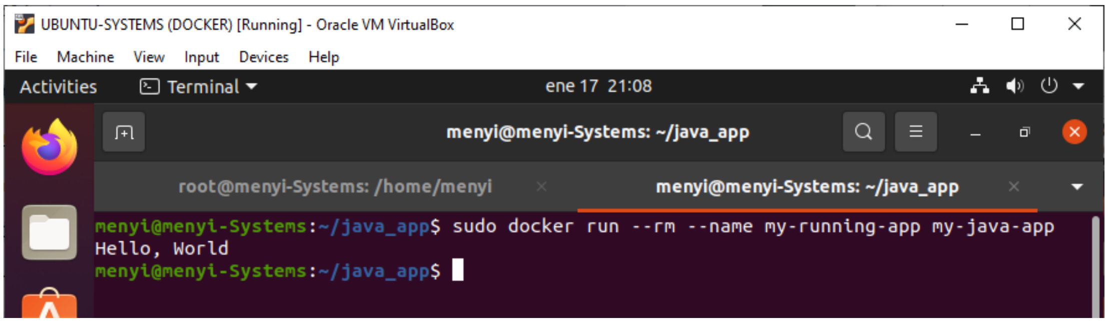

root@menyi-Systems: /home/menyi
menyi@menyi-Systems: ~/java_app

```
menyi@menyi-Systems:~/java_app$ sudo docker build -t my-java-app .
```


root@menyi-Systems: /home/menyi
menyi@menyi-Systems: ~/java_app
CTRL DERECHA

```
Removing intermediate container d52de82148c1
 ---> e7e76f3dbe0a
Step 4/5 : RUN javac Main.java
 ---> Running in df7aaec008af
Removing intermediate container df7aaec008af
 ---> 1a7c8e95a87f
Step 5/5 : CMD ["java", "Main"]
 ---> Running in e9d0c263dfae
Removing intermediate container e9d0c263dfae
 ---> b540ae25adcf
Successfully built b540ae25adcf
Successfully tagged my-java-app:latest
menyi@menyi-Systems:~/java_app$
```

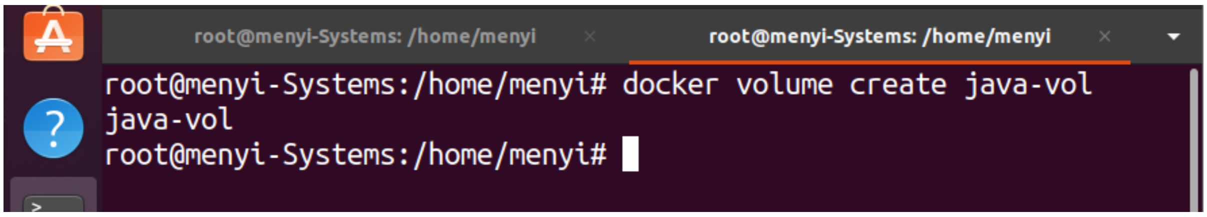
Finally, start a container to run the Java file and see the output


UBUNTU-SYSTEMS (DOCKER) [Running] - Oracle VM VirtualBox
File
Machine
View
Input
Devices
Help
Activities
Terminal
ene 17 21:08
menyi@menyi-Systems: ~/java_app
root@menyi-Systems: /home/menyi
menyi@menyi-Systems: ~/java_app

```
menyi@menyi-Systems:~/java_app$ sudo docker run --rm --name my-running-app my-java-app
Hello, World
menyi@menyi-Systems:~/java_app$
```

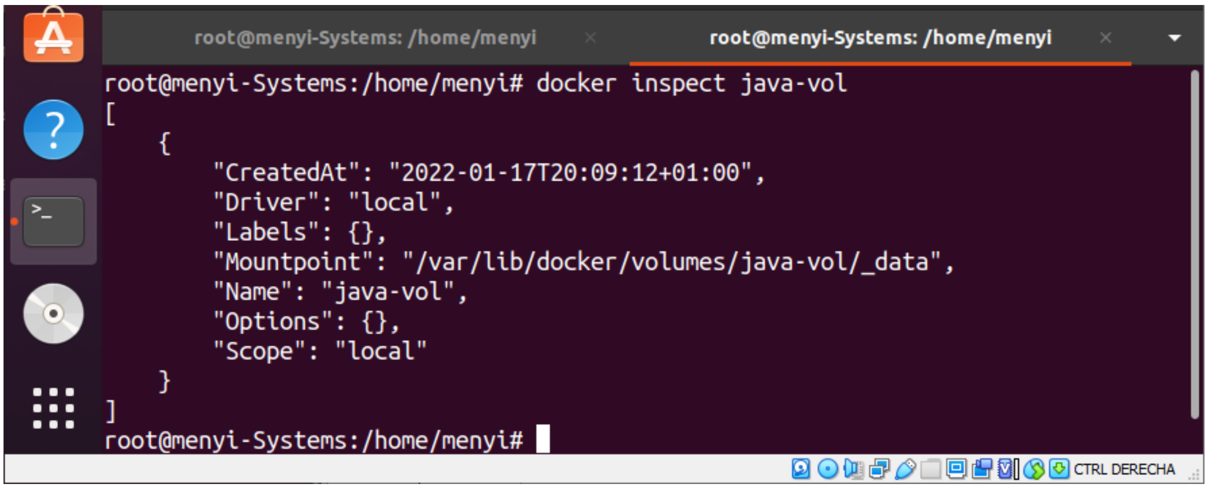
2 - RUN USING VOLUMES NON-INTERACTIVELY

First, create a **volume** named ***java-vol*** to save the Java application



```
root@menyi-Systems:/home/menyi# docker volume create java-vol
java-vol
root@menyi-Systems:/home/menyi#
```

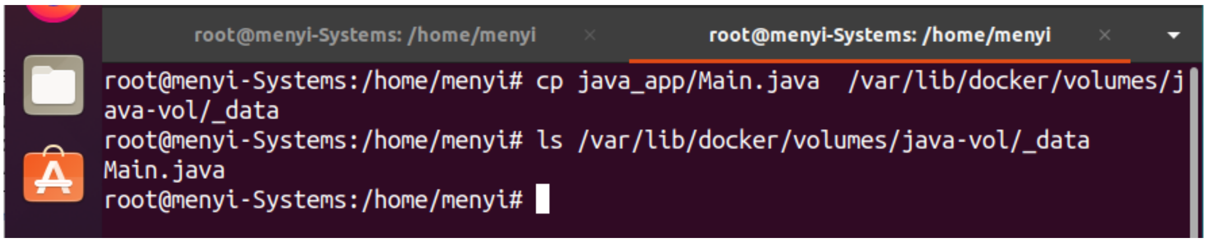
Before running the ***container***, we should check where it is really located with ***docker inspect***



```
root@menyi-Systems:/home/menyi# docker inspect java-vol
[
    {
        "CreatedAt": "2022-01-17T20:09:12+01:00",
        "Driver": "local",
        "Labels": {},
        "Mountpoint": "/var/lib/docker/volumes/java-vol/_data",
        "Name": "java-vol",
        "Options": {},
        "Scope": "local"
    }
]
root@menyi-Systems:/home/menyi#
```

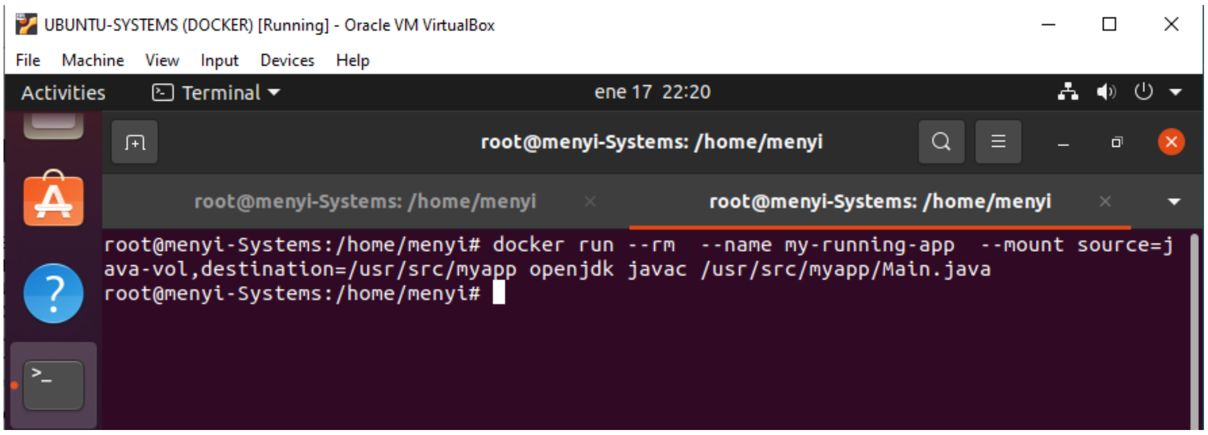
For example, in this case, the real content location will be ***/var/lib/docker/volumes/javavol/_data***

So, just copy the Java file ***Main.java*** into the volume folder and check it



```
root@menyi-Systems:/home/menyi# cp java_app/Main.java  /var/lib/docker/volumes/j
ava-vol/_data
root@menyi-Systems:/home/menyi# ls /var/lib/docker/volumes/java-vol/_data
Main.java
root@menyi-Systems:/home/menyi#
```

At this point we can ***compile*** the Java file ***Main.java*** associating the volume location.

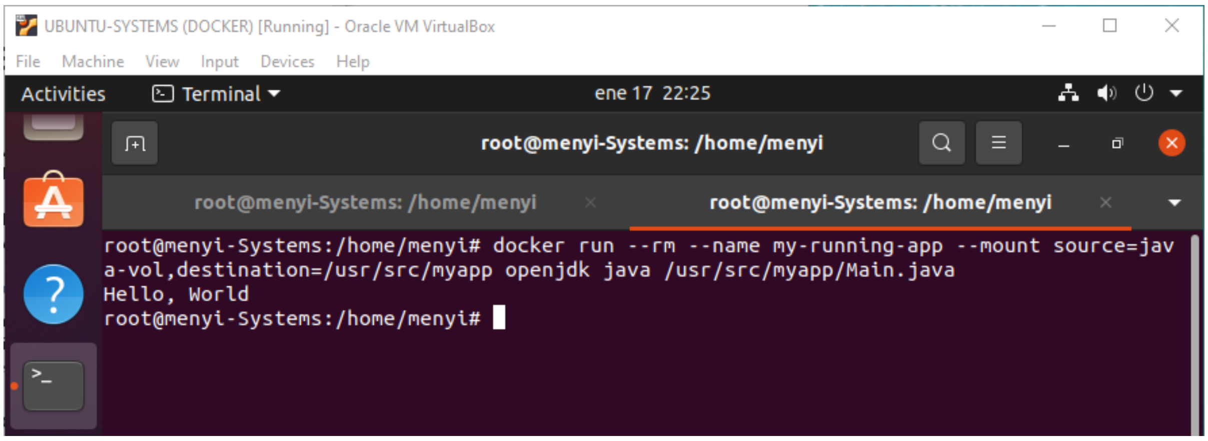

UBUNTU-SYSTEMS (DOCKER) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal ene 17 22:20
root@menyi-Systems: /home/menyi

```
root@menyi-Systems:/home/menyi# docker run --rm  --name my-running-app  --mount source=j
ava-vol,destination=/usr/src/myapp openjdk javac /usr/src/myapp/Main.java
root@menyi-Systems:/home/menyi# 
```

It is not necessary to create an image, but the main disadvantage is that we need to create two different containers to compile and run.

The first container should have created the ***Main.class*** in the ***volume*** folder.

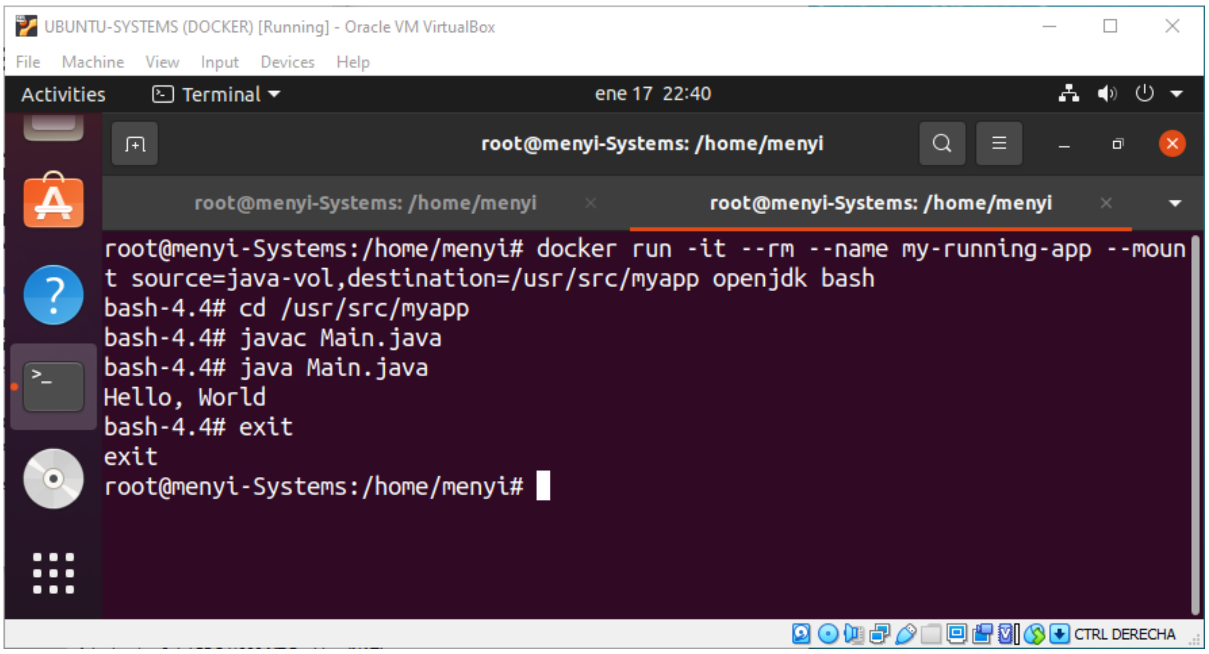
Finally, we can ***run*** the compiled file.


UBUNTU-SYSTEMS (DOCKER) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal ene 17 22:25
root@menyi-Systems: /home/menyi

```
root@menyi-Systems:/home/menyi# docker run --rm --name my-running-app --mount source=jav
a-vol,destination=/usr/src/myapp openjdk java /usr/src/myapp/Main.java
Hello, World
root@menyi-Systems:/home/menyi# 
```

3 - RUN USING VOLUMES INTERACTIVELY

Using the ***same volume as in Part 2***, it is possible to run a container with an ***interactive bash*** from an ***openjdk image*** (default option without bash opens a *jshell*, and we have not studied)


UBUNTU-SYSTEMS (DOCKER) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
ene 17 22:40
root@menyi-Systems: /home/menyi
root@menyi-Systems: /home/menyi
root@menyi-Systems: /home/menyi
CTRL DERECHA

```
root@menyi-Systems:/home/menyi# docker run -it --rm --name my-running-app --moun
t source=java-vol,destination=/usr/src/myapp openjdk bash
bash-4.4# cd /usr/src/myapp
bash-4.4# javac Main.java
bash-4.4# java Main.java
Hello, World
bash-4.4# exit
exit
root@menyi-Systems:/home/menyi# 
```