

Урок 4. Dockerfile и слои

Задание: необходимо создать Dockerfile, основанный на любом образе (вы в праве выбрать самостоятельно).

В него необходимо поместить приложение, написанное на любом известном вам языке программирования (Python, Java, C, C#, C++).

При запуске контейнера должно запускаться самостоятельно написанное приложение.

№1) начнем с++, как дань уважения, ниже приведен код приветствия , с выводом даты

```
#include <iostream>
#include <ctime>

int main() {
    // Запрашиваем имя пользователя
    std::cout << "Enter your nickname: ";
    std::string name;
    std::cin >> name;

    // Получаем текущее время и дату
    std::time_t current_time = std::time(0);
    std::tm *local_time = std::localtime(&current_time);

    // Выводим приветствие и текущее время
    std::cout << "HEY, " << name << "!" << std::endl;
    std::cout << "Today is " << (local_time->tm_mday) << " " << (local_time->tm_mon + 1) << " "
    << (local_time->tm_year + 1900) << std::endl;

    return 0;
}
```

1) создадим папку и перейдем в нее

```
sudo mkdir cpp; cd cpp; ls
```

2) создадим cpp file

```
sudo nano hello.cpp
```

также нужно установить компилятор для проверки

```
sudo apt install g++
```

3) скомпилируем, проверим:

```
sudo g++ -o hello hello.cpp
```

4) запустим

```
./hello
```

```
Enter your nickname: FRED1
```

```
HEY, FRED1!
```

```
Today is 10 10 2023
```

проверили работает

5) создадим докер файл

```
sudo nano Dockerfile
```

Установка базового образа Ubuntu
FROM ubuntu:latest

Обновление списка пакетов и установка компилятора C++
RUN apt update && \\\n apt install -y g++

Создание рабочей директории
WORKDIR /app

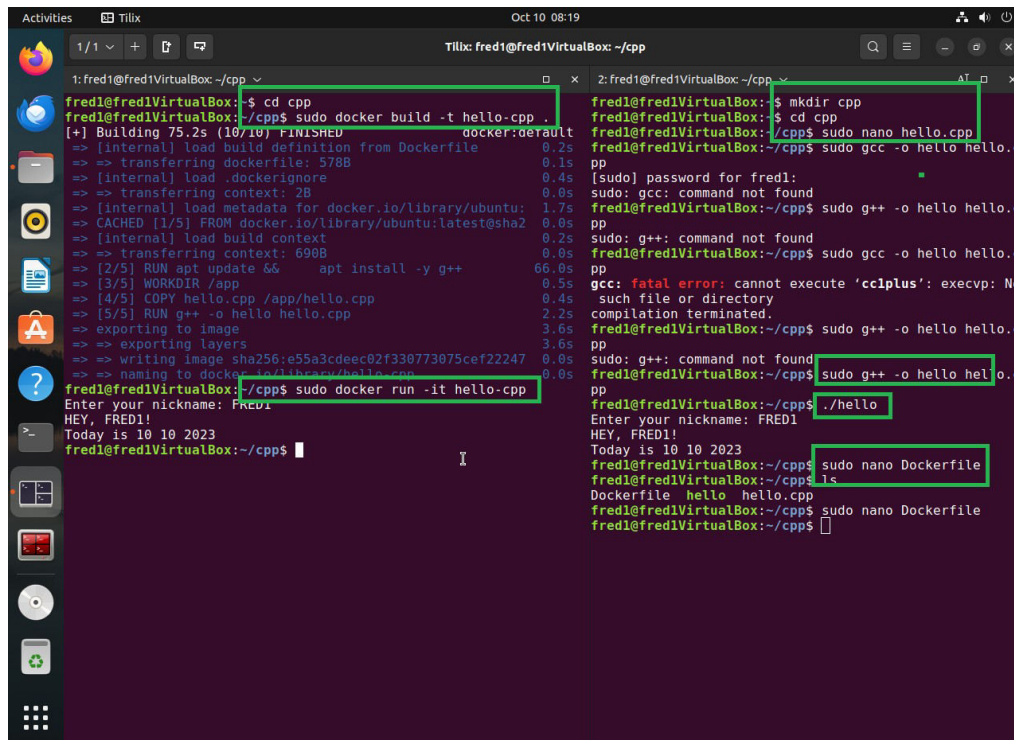
Копирование исходного файла C++ в образ
COPY hello.cpp /app/hello.cpp

Компиляция программы
RUN g++ -o hello hello.cpp

Команда для запуска программы
CMD ["/hello"]

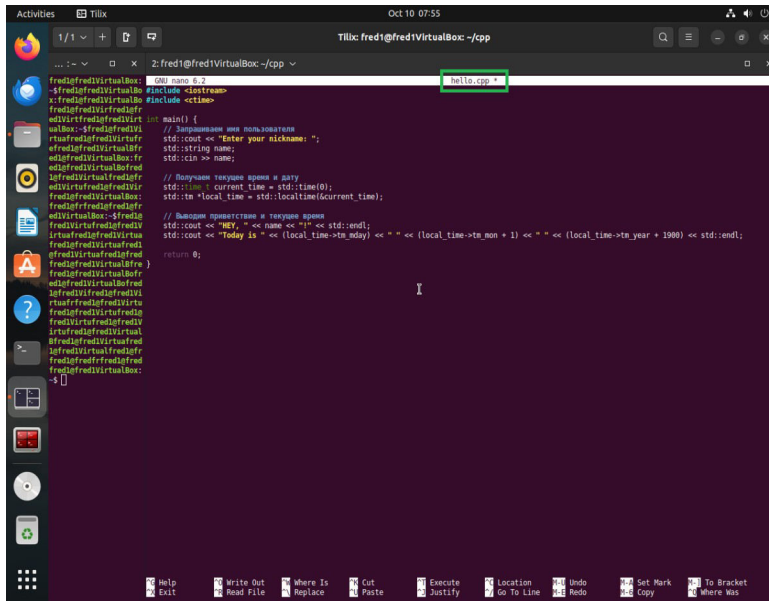
6) остается сбилдить и запустить в интерактивном режиме

sudo docker build -t hello-cpp .
sudo docker run -it hello-cpp



```
1: fred1@fred1VirtualBox: ~/cpp
fred1@fred1VirtualBox:~$ cd cpp
fred1@fred1VirtualBox:~/cpp$ sudo docker build -t hello-cpp
[+] Building 75.2s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> [internal] load .dockerignore
=> [internal] load metadata for docker.io/library/ubuntu:
=> CACHED [1/5] FROM docker.io/library/ubuntu:latest@sha2
=> [internal] load build context
=> transferring context: 690B
=> [2/5] RUN apt update && apt install -y g++
=> [3/5] WORKDIR /app
=> [4/5] COPY hello.cpp /app/hello.cpp
=> [5/5] RUN g++ -o hello hello.cpp
=> exporting to image
=> exporting layers
=> writing image sha256:e55a3cdec92f330773075cef22247
=> naming to docker.io/library/hello-cpp
fred1@fred1VirtualBox:~/cpp$ sudo docker run -it hello-cpp
Enter your nickname: FRED1
HEY, FRED1!
Today is 10 10 2023
fred1@fred1VirtualBox:~/cpp$

2: fred1@fred1VirtualBox: ~/cpp
fred1@fred1VirtualBox:~$ mkdir cpp
fred1@fred1VirtualBox:~$ cd cpp
fred1@fred1VirtualBox:~/cpp$ sudo nano hello.cpp
fred1@fred1VirtualBox:~/cpp$ sudo gcc -o hello hello.c
pp
[sudo] password for fred1:
sudo: gcc: command not found
fred1@fred1VirtualBox:~/cpp$ sudo g++ -o hello hello.c
pp
sudo: g++: command not found
fred1@fred1VirtualBox:~/cpp$ sudo gcc -o hello hello.c
pp
gcc: fatal error: cannot execute 'cc1plus': execvp: No
such file or directory
compilation terminated.
fred1@fred1VirtualBox:~/cpp$ sudo g++ -o hello hello.c
pp
sudo: g++: command not found
fred1@fred1VirtualBox:~/cpp$ sudo g++ -o hello hel o.c
pp
fred1@fred1VirtualBox:~/cpp$ ./hello
Enter your nickname: FRED1
HEY, FRED1!
Today is 10 10 2023
fred1@fred1VirtualBox:~/cpp$ sudo nano Dockerfile
fred1@fred1VirtualBox:~/cpp$ ls
Dockerfile hello hello.cpp
fred1@fred1VirtualBox:~/cpp$ sudo nano Dockerfile
fred1@fred1VirtualBox:~/cpp$
```



The screenshot shows a Tilix terminal window with a file manager icon in the top-left corner. The terminal is open to a file named 'hello.cpp'. The code inside the file is a C++ program that prompts the user for their name and prints the current date and time. The code is as follows:

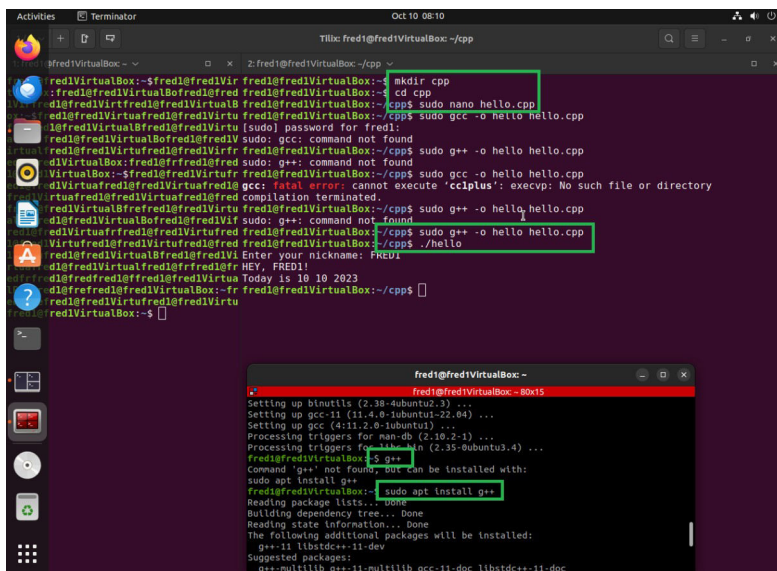
```
#include <iostream>
#include <ctime>

int main() {
    // Запросим имя пользователя
    std::cout << "Enter your nickname: ";
    std::string name;
    std::cin >> name;

    // Выясним текущее время и дату
    std::time_t current_time = std::time(0);
    std::tm *local_time = std::localtime(&current_time);

    // Выведем результат и текущее время
    std::cout << "hey " << name << "! " << std::endl;
    std::cout << "Today is " << (local_time->tm_mon + 1) << " " << (local_time->tm_mon + 1) << " " << (local_time->tm_mon + 1) << " " << std::endl;

    return 0;
}
```



The screenshot shows a Tilix terminal window with a file manager icon in the top-left corner. The terminal is open to a file named 'hello.cpp'. The code inside the file is a C++ program that prompts the user for their name and prints the current date and time. The code is as follows:

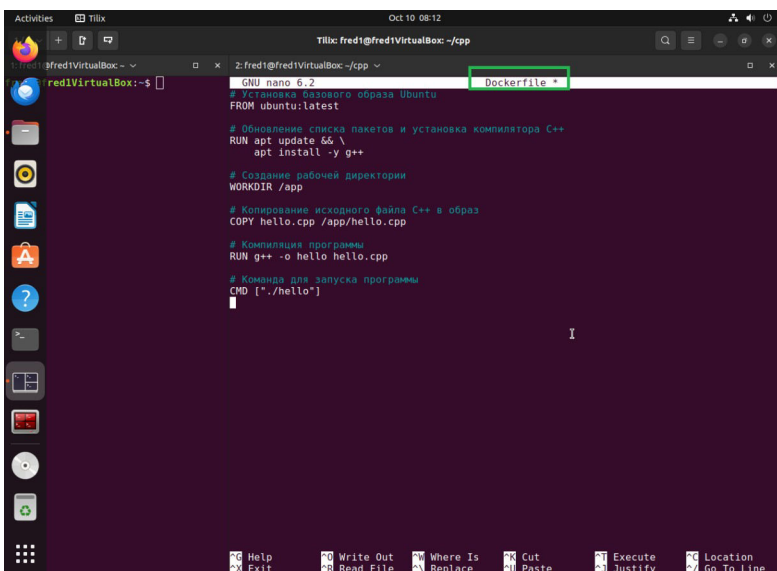
```
#include <iostream>
#include <ctime>

int main() {
    // Запросим имя пользователя
    std::cout << "Enter your nickname: ";
    std::string name;
    std::cin >> name;

    // Выясним текущее время и дату
    std::time_t current_time = std::time(0);
    std::tm *local_time = std::localtime(&current_time);

    // Выведем результат и текущее время
    std::cout << "hey " << name << "! " << std::endl;
    std::cout << "Today is " << (local_time->tm_mon + 1) << " " << (local_time->tm_mon + 1) << " " << (local_time->tm_mon + 1) << " " << std::endl;

    return 0;
}
```



The screenshot shows a Tilix terminal window with a file manager icon in the top-left corner. The terminal is open to a file named 'Dockerfile'. The code inside the file is a Dockerfile that sets up a C++ environment and runs the 'hello.cpp' program. The code is as follows:

```
FROM ubuntu:latest

# Обновление списка пакетов и установка компилятора C++
RUN apt update && \
    apt install -y g++

# Создание рабочей директории
WORKDIR /app

# Копирование исходного файла C++ в образ
COPY hello.cpp /app/hello.cpp

# Компиляция программы
RUN g++ -o hello hello.cpp

# Команда для запуска программы
CMD ["/hello"]
```

№2) тоже самое, но на java

1) создадим папку и перейдем в нее

```
sudo mkdir java; cd java; ls
```

2) создадим cpp file

```
sudo nano Docker.java
```

также нужно установить компилятор для проверки

```
sudo apt update && sudo apt install openjdk-11-jdk && java --version
```

3) скомпилируем, проверим:

```
sudo javac Docker.java
```

4) запустим

```
java Docker
```

5) создадим докер файл

```
sudo nano Dockerfile
```

Установка базового образа Ubuntu

```
FROM ubuntu:latest
```

Обновление списка пакетов и установка OpenJDK

```
RUN apt update && \
    apt install -y openjdk-11-jre
```

Создание рабочей директории

```
WORKDIR /app
```

Копирование скомпилированной Java-программы в образ

```
COPY Docker.class /app/Docker.class
```

Команда для запуска Java-программы

```
CMD ["java", "Docker"]
```

6) остается сбилдить и запустить в интерактивном режиме

```
sudo docker build -t hello-java .
```

```
sudo docker run -it hello-java
```

```
Activities 1/1 Tilda Oct 10 07:44
Tilda: fred1@fred1VirtualBox: ~/java

fred1@fred1VirtualBox:~/java$ docker build -t hello-java .
[+] Building 3.4s (9/9) FINISHED
=> [internal] load .dockerignore
=> [internal] load build definition from Dockerfile
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [1/4] FROM docker.io/library/ubuntu:latest@sha256:9b8d0c3b7938b0c8f7e75d8d5e9e0e
=> [internal] load build context
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [2/4] RUN apt update && apt install -y openjdk-11-jre
=> [3/4] WORKDIR /app
=> [4/4] COPY Docker.class /app/Docker.class
=> exporting image
=> writing image sha256:15bfa8887229797d889d88bf7e75307a7c
=> naming to docker.io/library/hello-java
fred1@fred1VirtualBox:~/java$ docker run -it hello-java
docker: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://x2fvar2f2run2f2docker.sock/v1.24/containers/create": dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.

fred1@fred1VirtualBox:~/java$ sudo docker run -it hello-java
????????, ?????? ????: Fred1
???????, Fred1
???????? 10 10 2023 11:41
fred1@fred1VirtualBox:~/java$ sudo docker build -t hello-java2 .
[+] Building 3.4s (9/9) FINISHED
=> [internal] load .dockerignore
=> [internal] load build definition from Dockerfile
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [1/4] FROM docker.io/library/ubuntu:latest@sha256:9b8d0c3b7938b0c8f7e75d8d5e9e0e
=> [internal] load build context
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [2/4] RUN apt update && apt install -y openjdk-11-jre
=> [3/4] WORKDIR /app
=> [4/4] COPY Docker.class /app/Docker.class
=> exporting image
=> writing image sha256:a219f6e3e10-d6d1e579c4e6b21f434528ed1e47e61bdc02b62
=> naming to docker.io/library/hello-java2
fred1@fred1VirtualBox:~/java$ sudo docker run -it hello-java2
Enter your name: FRED1
Welcome, FRED1!
Today is 10 10 2023 11:44
fred1@fred1VirtualBox:~/java$
```

```
Activities 1/1 Tilda Oct 10 07:41
Tilda: fred1@fred1VirtualBox: ~/java

fred1@fred1VirtualBox:~/java$ cd java
fred1@fred1VirtualBox:~/java/java$ docker build -t hello-java .
ERROR: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://x2fvar2f2run2f2docker.sock/v1.24/containers/create": dial unix /var/run/docker.sock: connect: permission denied

fred1@fred1VirtualBox:~/java/java$ sudo docker build -t hello-java .
[sudo] password for fred1:
[+] Building 141.6s (9/9) FINISHED
=> [internal] load .dockerignore
=> [internal] load build definition from Dockerfile
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [1/4] FROM docker.io/library/ubuntu:latest@sha256:9b8d0c3b7938b0c8f7e75d8d5e9e0e
=> [internal] load build context
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [2/4] RUN apt update && apt install -y openjdk-11-jre
=> [3/4] WORKDIR /app
=> [4/4] COPY Docker.class /app/Docker.class
=> exporting image
=> writing image sha256:15bfa8887229797d889d88bf7e75307a7c
=> naming to docker.io/library/hello-java
fred1@fred1VirtualBox:~/java/java$ docker run -it hello-java
docker: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post "http://x2fvar2f2run2f2docker.sock/v1.24/containers/create": dial unix /var/run/docker.sock: connect: permission denied.

fred1@fred1VirtualBox:~/java/java$ sudo docker run -it hello-java
????????, ?????? ????: Fred1
???????, Fred1
???????? 10 10 2023 11:41
fred1@fred1VirtualBox:~/java/java$
```

```
Activities 1/1 Tilda Oct 10 07:36
Tilda: fred1@fred1VirtualBox: ~/java

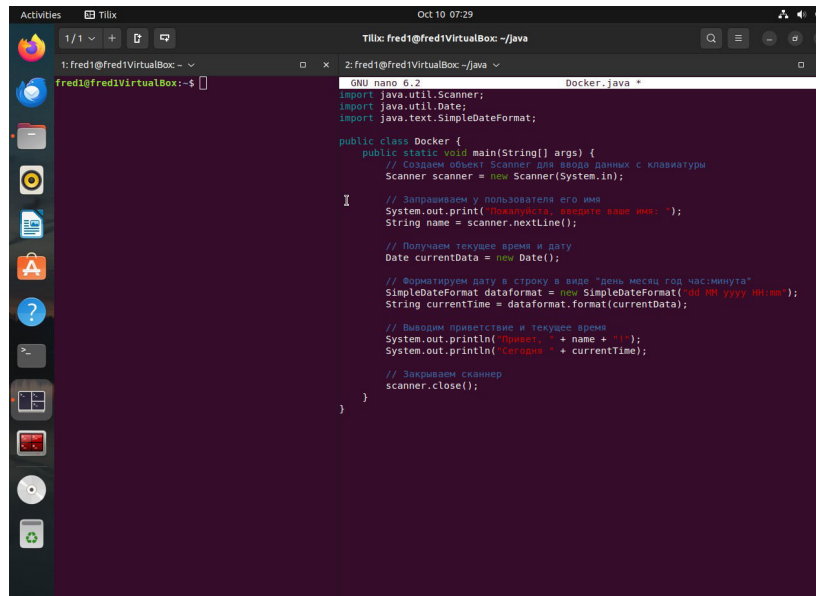
fred1@fred1VirtualBox:~/java$ nano Dockerfile
GNU nano 6.2 Dockerfile
FROM ubuntu:latest

# Обновление списка пакетов и установка OpenJDK
RUN apt update && \
    apt install -y openjdk-11-jre

# Создание рабочей директории
WORKDIR /app

# Копирование скомпилированной Java-программы в образ
COPY Docker.class /app/Docker.class

# Команда для запуска Java-программы
CMD ["java", "Docker"]
```



```
GNU nano 6.2 Docker.java
import java.util.Scanner;
import java.util.Date;
import java.text.SimpleDateFormat;

public class Docker {
    public static void main(String[] args) {
        // Создаем объект Scanner для ввода данных с клавиатуры
        Scanner scanner = new Scanner(System.in);

        // Запрашиваем у пользователя его имя
        System.out.print("Введите ваше имя: ");
        String name = scanner.nextLine();

        // Получаем текущую дату и время
        Date currentDate = new Date();

        // Форматируем дату в строку в виде "день, месяц, год час:минута"
        SimpleDateFormat dateFormat = new SimpleDateFormat("dd MM yyyy HH:mm");
        String currentTime = dateFormat.format(currentDate);

        // Выводим приветствие и текущее время
        System.out.println("Добро! " + name + "!");
        System.out.println("Сегодня " + currentTime);

        // Закрываем сканнер
        scanner.close();
    }
}
```

№2) тоже самое, но на python

1) создадим папку и перейдем в нее

```
sudo mkdir python; cd python; ls
```

2) создадим python.py file

```
sudo nano python.py
```

```
sudo python3 --version
```

3) запускаем, проверим:

```
sudo python3 python.py
```

4) создадим докер файл

```
sudo nano Dockerfile
```

```
# Используем официальный образ Ubuntu
FROM ubuntu:latest
```

```
# Обновляем список пакетов и устанавливаем Python
```

```
RUN apt update && \
    apt install -y python3
```

```
# Создаем рабочую директорию
```

```
WORKDIR /app
```

```
# Копируем файл hello.py внутрь образа
```

```
COPY python.py /app/python.py
```

```
# Команда для запуска программы
```

```
CMD ["python3", "python.py"]
```

либо вообще без ОС :

```
# Используем официальный образ Python
FROM python:latest
```

```
# Создаем рабочую директорию
WORKDIR /app
```

```
# Копируем файл hello.py внутрь образа
COPY python.py /app/python.py
```

```
# Команда для запуска программы
CMD ["python", "python.py"]
```

6) остается сбилдить и запустить в интерактивном режиме

```
sudo docker build -t hello-os-python .
sudo docker run -it hello-os-python
```

7) ошибся при создании Dockerfile, пришлось удалить образ:
`sudo docker rmi 82d883b128b7 --force`

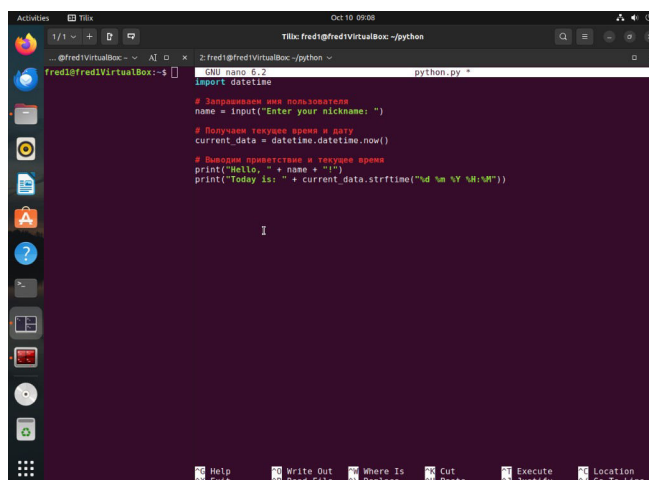
8) пересобрал все ок, теперь нужно запушить на docker.hub
`docker login`

```
docker tag hello-os-python farid555/hello-os-python
```

```
docker push farid555/hello-os-python
```

9) Радуетмся :)

<https://hub.docker.com/repository/docker/farid555/hello-os-python/general>



```
Yllix:fred@fredVirtualBox:~/python
fred@fredVirtualBox:~$ python.py
import datetime

# Запрашиваем имя пользователя
name = input("Enter your nickname: ")

# Получаем текущее время и дату
current_data = datetime.datetime.now()

# Выводим приветствие и текущее время
print("Hello, " + name + "!")
print("Today is: " + current_data.strftime("%d %m %Y %H:%M"))

I
```



```
Activities  Tilix  Oct 10 09:38
Tilix: fred1@fred1VirtualBox: ~/python

fred1@fred1VirtualBox:~/python$ ls
Dockerfile  python.py
fred1@fred1VirtualBox:~/python$ sudo docker build -t hello-os-python
[+] Building 28.9s (9/9) FINISHED
=> [internal] load .dockerignore
=> [internal] load build definition from Dockerfile
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [1/4] FROM docker.io/library/ubuntu:latest@sha256:9b8dec3
=> [internal] load build context
=> [2/4] RUN apt update && apt install -y python3
=> [3/4] WORKDIR /app
=> [4/4] COPY python.py /app/python.py
=> exporting to image
=> writing image sha256:82d883b128b7721447f85e0be7c451be51384e
=> naming to docker.io/library/hello-os-python

fred1@fred1VirtualBox:~/python$ docker run -it hello-os-python
docker: permission denied while trying to connect to the Docker daemon
n socket at unix:///var/run/docker.sock: Post "http://v2Fvark2Frunk2F
docker.sock/v1.24/containers/create": dial unix /var/run/docker.sock:
connect: permission denied.
See 'docker run --help'.
fred1@fred1VirtualBox:~/python$ sudo docker run -it hello-os-python
```

```
Activities  Tilix  Oct 10 09:51
Tilix: fred1@fred1VirtualBox: ~/python

fred1@fred1VirtualBox:~/python$ sudo docker build -t hello-os-python
[+] Building 3.3s (9/9) FINISHED
=> [internal] load .dockerignore
=> [internal] load build definition from Dockerfile
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [1/4] FROM docker.io/library/ubuntu:latest@sha256:9b8dec3b938b
=> [internal] load build context
=> [2/4] RUN apt update && apt install -y python3
=> [3/4] WORKDIR /app
=> [4/4] COPY python.py /app/python.py
=> exporting to image
=> writing image sha256:06026e78ec88fdd731afc29ae9415efb0f9e6e
=> naming to docker.io/library/hello-os-python

fred1@fred1VirtualBox:~/python$ sudo docker run -it hello-os-python
Enter your nickname: FRED1
Hello, FRED1!
Today is: 10 10 2023 13:51
fred1@fred1VirtualBox:~/python$
```

```
Activities  Tilix  Oct 10 09:58
Tilix: fred1@fred1VirtualBox: ~/python

fred1@fred1VirtualBox:~/python$ sudo docker build -t hello-os-python
[+] Building 3.3s (9/9) FINISHED
=> [internal] load .dockerignore
=> [internal] load build definition from Dockerfile
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [1/4] FROM docker.io/library/ubuntu:latest@sha256:9b8dec3b938b
=> [internal] load build context
=> [2/4] RUN apt update && apt install -y python3
=> [3/4] WORKDIR /app
=> [4/4] COPY python.py /app/python.py
=> exporting to image
=> writing image sha256:06026e78ec88fdd731afc29ae9415efb0f9e6e
=> naming to docker.io/library/hello-os-python

fred1@fred1VirtualBox:~/python$ sudo docker run -it hello-os-python
Enter your nickname: FRED1
Hello, FRED1!
Today is: 10 10 2023 13:51
fred1@fred1VirtualBox:~/python$ sudo docker login
Log in with your Docker ID or email address to push and pull images from
Docker Hub. If you don't have a Docker ID, head over to https://hub.docke
r.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using
a limited-scope PAT grants better security and is required for organizat
ions using SSO. Learn more at https://docs.docker.com/go/access-tokens/

Username: farid555
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config
.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-s
tore

Login Succeeded
fred1@fred1VirtualBox:~/python$
```



```
Activities 1/1 TILIX Oct 10 10:10
TILIX: fred1@fred1VirtualBox: ~/python

1:fred1@fred1VirtualBox:~/python$
=> transferring context: 31B 0.0s
=> CACHED [2/4] RUN apt update && apt install -y python3 0.0s
=> CACHED [3/4] WORKDIR /app 0.0s
=> [4/4] COPY python.py /app/python.py 0.3s
=> exporting to image 0.3s
=> exporting layers 0.2s
=> writing image sha256:06026e78ec888fd4f31afc29ae9415efb0f9e6e 0.0s
=> naming to docker.io/library/hello-os-python 0.0s
fred1@fred1VirtualBox:~/python$ sudo docker run -it hello-os-python
Enter your nickname: FRED1
Hello, FRED1!
Today is: 10 10 2023 13:51
fred1@fred1VirtualBox:~/python$ sudo docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head
over to https://hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and
is required for organizations using SSO. Learn more at https://docs.docker.com/go/access-tokens/

Username: farid555
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

fred1@fred1VirtualBox:~/python$ docker tag hello-os-python farid555/hello-os-python
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post "http://%2Fva
r%2Frun%2Fdocker.sock/v1.24/images/hello-os-python/tag?repo=farid555%2Fhello-os-python&tag=latest": dial unix /var/run/d
ocker.sock: connect: permission denied
fred1@fred1VirtualBox:~/python$ sudo docker tag hello-os-python farid555/hello-os-python
fred1@fred1VirtualBox:~/python$ docker push farid555/hello-os-python
Using default tag: latest
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post "http://%2Fva
r%2Frun%2Fdocker.sock/v1.24/images/farid555/hello-os-python/push?tag=latest": dial unix /var/run/docker.sock: connect: p
ermission denied
fred1@fred1VirtualBox:~/python$ sudo docker push farid555/hello-os-python
Using default tag: latest
The push refers to repository [docker.io/farid555/hello-os-python]
2420b71057d3: Pushed
17482c15438e: Pushed
5f2c963ddac2: Pushed
01d4e4b4f381: Mounted from library/ubuntu
latest: digest: sha256:b60ea802d59f68efd48de93fdf7f73ff6a62c88660c521012e7f8dcd7e1eaf6 size: 1154
fred1@fred1VirtualBox:~/python$
```

8.2) теперь пушим в docker.hub, hello-cpp
docker login

sudo docker tag hello-cpp farid555/hello-cpp

docker push farid555/hello-cpp

<https://hub.docker.com/repository/docker/farid555/hello-cpp/general>

```
Activities 1/1 TILIX Oct 10 10:25
TILIX: fred1@fred1VirtualBox: ~

fred1@fred1VirtualBox:~$ sudo docker run -it hello-cpp
Enter your nickname: FRED1
HEY, FRED1!
Today is: 10 10 2023
fred1@fred1VirtualBox:~$ docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head
over to https://hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and
is required for organizations using SSO. Learn more at https://docs.docker.com/go/access-tokens/

Username: farid555
Password:
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post "http://%2Fva
r%2Frun%2Fdocker.sock/v1.24/auth?service=registry.docker.io": dial unix /var/run/docker.sock: connect: permission denied
fred1@fred1VirtualBox:~$ sudo docker login
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
fred1@fred1VirtualBox:~$ sudo docker tag hello-cpp farid555/hello-cpp
fred1@fred1VirtualBox:~$ docker push farid555/hello-cpp
Using default tag: latest
The push refers to repository [docker.io/farid555/hello-cpp]
98f0e5301860: Pushed
50eea2c851e0: Pushed
589d5c46bc8f: Pushed
f523629744a: Pushed
01d4e4b4f381: Mounted from farid555/hello-os-python
latest: digest: sha256:dca2c1e2723828db6347389cd62d146af2c23c7a2f65b9ebd99e3876afd65 size: 1363
fred1@fred1VirtualBox:~$
```

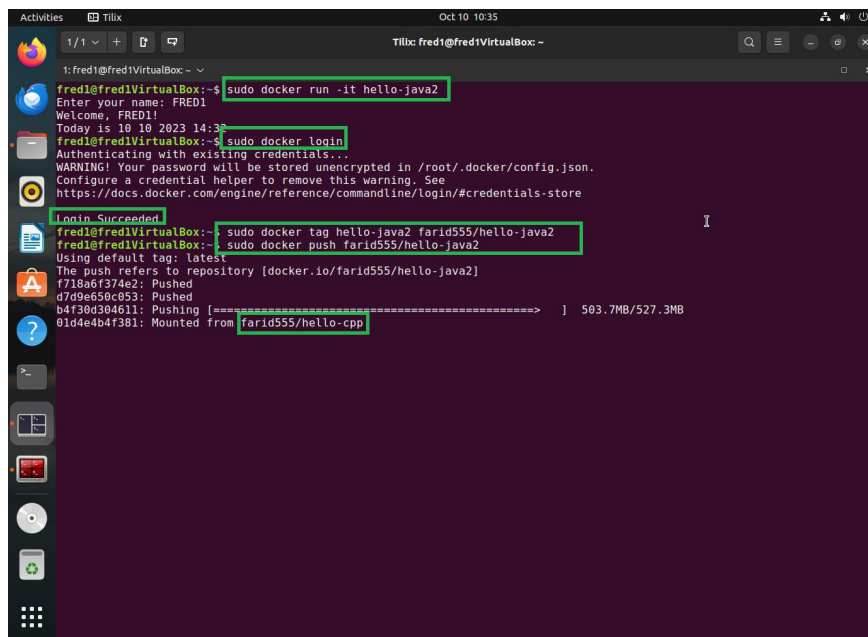
8.3) и наконец java, hello-java2

docker login

sudo docker tag hello-java2 farid555/hello-java2

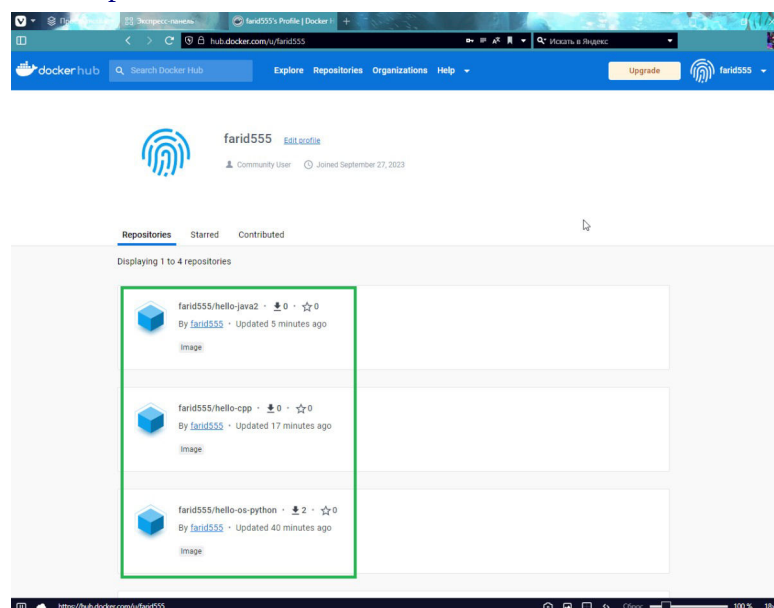
docker push farid555/hello-java2

<https://hub.docker.com/repository/docker/farid555/hello-java2/general>

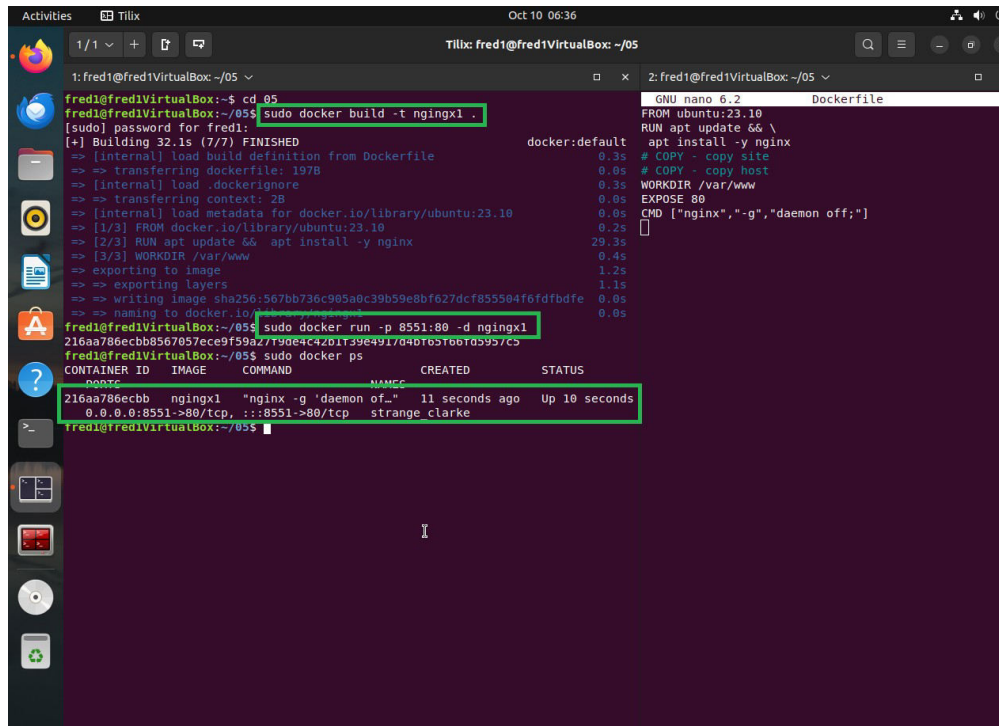


```
1:fred1@fred1VirtualBox:~$ sudo docker run -it hello-java2
fred1@fred1VirtualBox:~$ sudo docker login
Enter your name: FRED1
Welcome, FRED1!
Today is 10 10 2023 14:32
fred1@fred1VirtualBox:~$ sudo docker tag hello-java2 farid555/hello-java2
fred1@fred1VirtualBox:~$ sudo docker push farid555/hello-java2
Using default tag: latest
The push refers to repository [docker.io/farid555/hello-java2]
f718a6f374e2: Pushed
d7d9e650c053: Pushed
b4f30d304611: Pushing [=====] 503.7MB/527.3MB
01d4e4b4f381: Mounted from farid555/hello-cpp
```

<https://hub.docker.com/u/farid555>



№4 — повторяем семинар, устанавливаем nginx

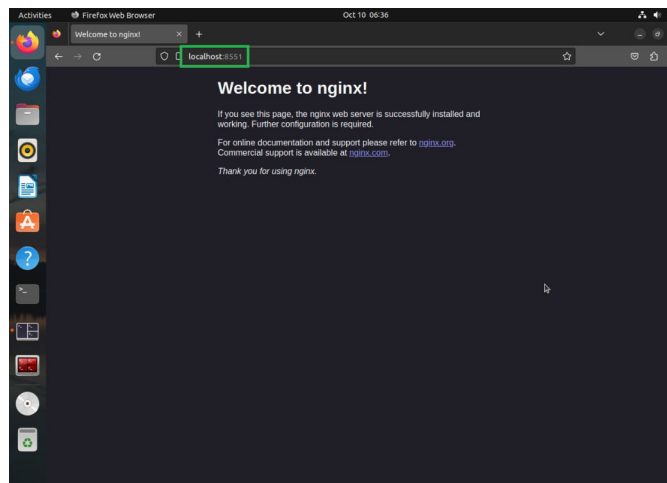


```
1: fred1@fred1VirtualBox: ~/05
fred1@fred1VirtualBox:~$ cd 05
fred1@fred1VirtualBox:~/05$ sudo docker build -t nginx1 .
[sudo] password for fred1:
[+] Building 32.1s (7/7) FINISHED
=> [internal] load build definition from Dockerfile
=> transferring dockerfile: 197B
=> [internal] load .dockerignore
=> transferring context: 2B
=> [internal] load metadata for docker.io/library/ubuntu:23.10
=> [1/3] FROM docker.io/library/ubuntu:23.10
=> [2/3] RUN apt update && apt install -y nginx
=> [3/3] WORKDIR /var/www
=> exporting to image
=> writing image sha256:567bb736c985a0c39b59e8bf627dcf855504f6dfdbdfc
=> naming to docker.io/nginx1
fred1@fred1VirtualBox:~/05$ sudo docker run -p 8551:80 -d nginx1
216aa786ecbb8567057ece9f59a27f9de4c4201739e4917d40f65f667d5957c5
fred1@fred1VirtualBox:~/05$ sudo docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS
216aa786ecbb   nginx1     "nginx -g 'daemon off;'" 11 seconds ago Up 10 seconds
0.0.0.0:8551->80/tcp, :::8551->80/tcp   strange.clarke

Dockerfile
FROM ubuntu:23.10
RUN apt update && \
    apt install -y nginx
# COPY - copy site
# COPY - copy host
WORKDIR /var/www
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

```
FROM ubuntu:23.10
RUN apt update && \
    apt install -y nginx
# COPY - copy site
# COPY - copy host
WORKDIR /var/www
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

```
sudo docker build -t nginx1
sudo docker run -p 8551:80 -d nginx1
sudo docker ps
```



Далее необходимо собрать образ и запустить из него контейнер:
Основой образа должна быть alpine.
Установить необходимо mariadb.
Также не забудьте об уменьшении размера образа. Способ обсуждался на лекции.
Необходимо открыть порт для коммуникации с другими сущностями.
Для проверки решения необходимо подключить к такому контейнеру phpmyadmin.
Необходимо, чтобы в нем вы увидели
данные из вашей БД.
Также при запуске необходимо смонтировать внешнюю папку для хранения данных БД вне
контейнера.

1) Dockerfile

```
FROM alpine:3.14
```

```
RUN apk --update add mariadb mariadb-client && \  
rm -f /var/cache/apk/*
```

```
# Копируем файл конфигурации MariaDB  
COPY my.cnf /etc/mysql/my.cnf
```

```
# Создаем директорию для хранения данных БД  
RUN mkdir -p /var/lib/mysql && chown -R mysql:mysql /var/lib/mysql
```

```
# Открываем порт для коммуникации с другими сущностями  
EXPOSE 3306
```

```
# Скрипт для запуска MariaDB  
CMD ["mysqld"]
```

2) my.cnf – рядом с Dockerfile-лом

```
[mysqld]  
user = mysql  
datadir = /var/lib/mysql  
bind-address = 0.0.0.0
```

2*) создать директорию /home/fred1/mydb_data – общая папка

3)
sudo docker build -t mariadb-alpine .

4)

```
sudo docker run -d --name my-mariadb -p 3306:3306 -v /home/fred1/mydb_data:/var/lib/mysql  
mariadb-alpine
```

```
sudo docker run -d --name mariadb-alpine -p 3306:3306 -v /home/fred2/db1:/var/lib/mysql  
mariadb-alpine
```

```
docker run -d --name my-mariadb-container -e MYSQL_ROOT_PASSWORD=123 -p 3306:3306 -  
v /home/fred2/db2:/var/lib/mysql my-mariadb-ubuntu5
```

```
sudo docker run -d --name my-mariadb-container5 -e MYSQL_ROOT_PASSWORD=123 -p  
3377:3306 my-mariadb-ubuntu5
```

5) `sudo docker run -d --name phpmyadmin-container --link my-mariadb:db -p 8555:80
phpmyadmin/phpmyadmin`

6) <https://localhost:8555>

1) Dockerfile

FROM alpine:3.14

```
RUN apk --update add mariadb mariadb-client && \  
rm -f /var/cache/apk/*
```

```
# Копируем файл конфигурации MariaDB  
#COPY my.cnf /etc/mysql/my.cnf
```

```
# Создаем директорию для хранения данных БД  
#RUN mkdir -p /var/lib/mysql && chown -R mysql:mysql /var/lib/mysql
```

```
# Открываем порт для коммуникации с другими сущностями  
EXPOSE 3306
```

```
# Копируем init.sql в директорию инициализации  
COPY init.sql /docker-entrypoint-initdb.d/init.sql
```

```
# Скрипт для запуска MariaDB в режиме демона с инициализацией БД  
CMD ["mysqld", "--user=mysql", "--init-file=/docker-entrypoint-initdb.d/init.sql"]
```

2) init.sql в той же папки

```
CREATE DATABASE mydb;
```

```
USE mydb;
```

```
CREATE TABLE users (id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(255));
```

```
INSERT INTO users (name) VALUES ('User1'), ('User2'), ('User3');
```

3) дальше тоже самое... так не вышло пришлось использовать ubuntu

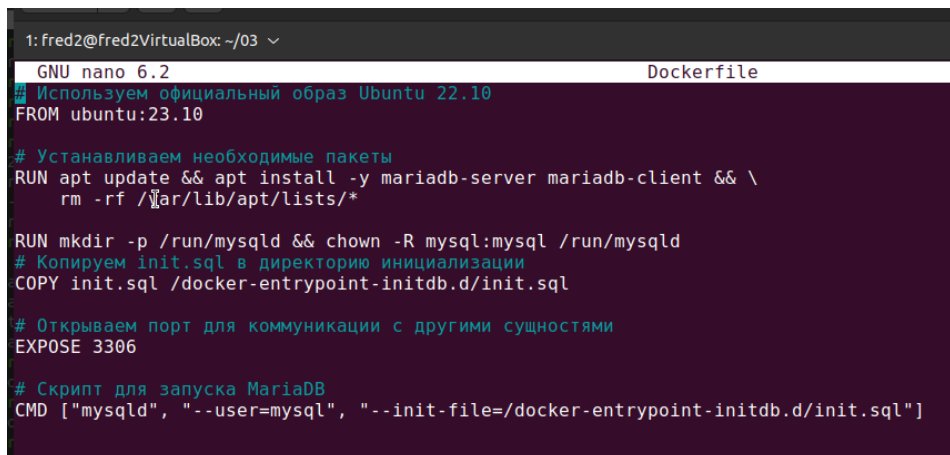
1)# Используем официальный образ Ubuntu 22.10
FROM ubuntu:23.10

Устанавливаем необходимые пакеты
RUN apt update && apt install -y mariadb-server mariadb-client && \
rm -rf /var/lib/apt/lists/*

RUN mkdir -p /run/mysqld && chown -R mysql:mysql /run/mysqld
Копируем init.sql в директорию инициализации
COPY init.sql /docker-entrypoint-initdb.d/init.sql

Открываем порт для коммуникации с другими сущностями
EXPOSE 3306

Скрипт для запуска MariaDB
CMD ["mysqld", "--user=mysql", "--init-file=/docker-entrypoint-initdb.d/init.sql"]

A screenshot of a terminal window with a dark background. The title bar shows '1: fred2@fred2VirtualBox: ~/03'. The terminal is running GNU nano 6.2, editing a file named 'Dockerfile'. The content of the file is the Dockerfile code from the previous blocks, including the FROM statement, package installation, directory creation, file copying, port exposure, and the CMD instruction for starting MariaDB.

```
1: fred2@fred2VirtualBox: ~/03
GNU nano 6.2 Dockerfile
Используем официальный образ Ubuntu 22.10
FROM ubuntu:23.10

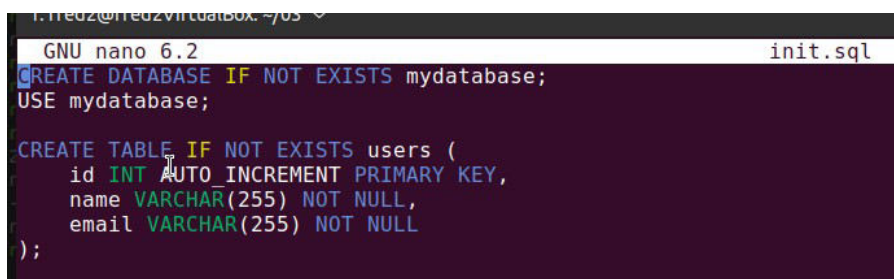
# Устанавливаем необходимые пакеты
RUN apt update && apt install -y mariadb-server mariadb-client && \
rm -rf /var/lib/apt/lists/*

RUN mkdir -p /run/mysqld && chown -R mysql:mysql /run/mysqld
# Копируем init.sql в директорию инициализации
COPY init.sql /docker-entrypoint-initdb.d/init.sql

# Открываем порт для коммуникации с другими сущностями
EXPOSE 3306

# Скрипт для запуска MariaDB
CMD ["mysqld", "--user=mysql", "--init-file=/docker-entrypoint-initdb.d/init.sql"]
```

2) Создадим init.sql
CREATE DATABASE IF NOT EXISTS mydatabase;
USE mydatabase;
CREATE TABLE IF NOT EXISTS users (
id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(255) NOT NULL,
email VARCHAR(255) NOT NULL
);

A screenshot of a terminal window with a dark background. The title bar shows '1: fred2@fred2VirtualBox: ~/03'. The terminal is running GNU nano 6.2, editing a file named 'init.sql'. The content of the file is the SQL code from the previous block, creating a database and a table.

```
1: fred2@fred2VirtualBox: ~/03
GNU nano 6.2 init.sql
CREATE DATABASE IF NOT EXISTS mydatabase;
USE mydatabase;

CREATE TABLE IF NOT EXISTS users (
id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(255) NOT NULL,
email VARCHAR(255) NOT NULL
);
```


3) Соберем Docker-образ:

```
sudo docker build -t my-mariadb-ubuntu5 .
```

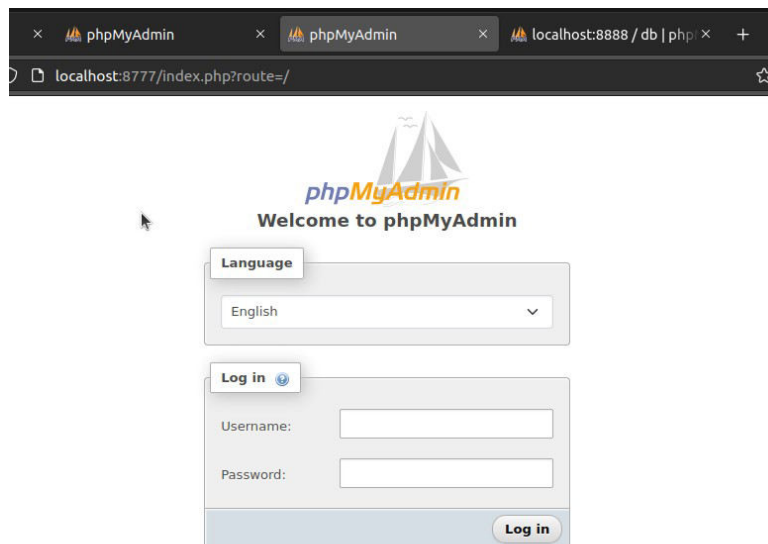
4) Запустим образ

```
sudo docker run -d --name my-mariadb-container5 -e MYSQL_ROOT_PASSWORD=123  
-p 3377:3306 my-mariadb-ubuntu5
```

```
fred2@fred2VirtualBox:~/03$ sudo docker build -t my-mariadb-ubuntu5 .  
[+] Building 2.3s (9/9) FINISHED                                docker:default  
=> [internal] load .dockerignore                                0.0s  
=> transferring context: 28                                     0.0s  
=> [internal] load build definition from Dockerfile             0.1s  
=> transferring dockerfile: 729B                                0.0s  
=> [internal] load metadata for docker.io/library/ubuntu:23.10 1.8s  
=> [1/4] FROM docker.io/library/ubuntu:23.10@sha256:282510723f2be541c2facce0f7e918641bedd90936f8a76f6f38b7110e29 0.0s  
=> [internal] load build context                                0.1s  
=> transferring context: 30B                                     0.0s  
=> CACHED [2/4] RUN apt update && apt install -y mariadb-server mariadb-client && rm -rf /var/lib/apt/lists/ 0.0s  
=> CACHED [3/4] RUN mkdir -p /run/mysqld && chown -R mysql:mysql /run/mysqld 0.0s  
=> CACHED [4/4] COPY init.sql /docker-entrypoint-initdb.d/init.sql 0.0s  
=> exporting to image                                           0.0s  
=> exporting layers                                             0.0s  
=> writing image sha256:73e324fb5715892364d09cce95fd783dbb79e82a85e9a5648dfaf5aa93ed01a8 0.0s  
=> naming to docker.io/library/my-mariadb-ubuntu5             0.0s  
fred2@fred2VirtualBox:~/03$ sudo docker run -d --name my-mariadb-container5 -e MYSQL_ROOT_PASSWORD=123 -p 3377:3306 my-mariadb-ubuntu5  
640b1f9077123577fa75919e2ad85a0a60b97e5c2567a16ed2ce5f80942cfacbf  
fred2@fred2VirtualBox:~/03$ sudo docker ps  
CONTAINER ID   IMAGE          NAMES                COMMAND                CREATED        STATUS        PORTS  
640b1f907712   my-mariadb-ubuntu5   my-mariadb-container5   "mysqld --user=mysql..."   8 seconds ago   Up 6 seconds   0.0.0.0:3377->3306/tcp,  
:::3377->3306/tcp  
a9a036a4d4dd   phpmyadmin/phpmyadmin   "/docker-entrypoint..."   13 minutes ago   Up 13 minutes   0.0.0.0:8555->80/tcp, :  
:::8555->80/tcp  
a00df449d517   my-mariadb-ubuntu4   "mysqld --user=mysql..."   24 minutes ago   Up 24 minutes   0.0.0.0:3366->3306/tcp,  
:::3366->3306/tcp  
fred2@fred2VirtualBox:~/03$ sudo docker run -d --name phpmyadmin-container2 --link my-mariadb-container5:db -p 8777:80 phpmyadmin/phpmyadmin  
0cc2008f03b2697725bacbad5a9ff46d949159abba3d8b69bbbfdbcfb7b9ecf0c
```

5) Затем запустите контейнер PHPMyAdmin, связанный с контейнером MariaDB

```
sudo docker run -d --name phpmyadmin-container2 --link my-mariadb-container5:db -p 8777:80  
phpmyadmin/phpmyadmin
```

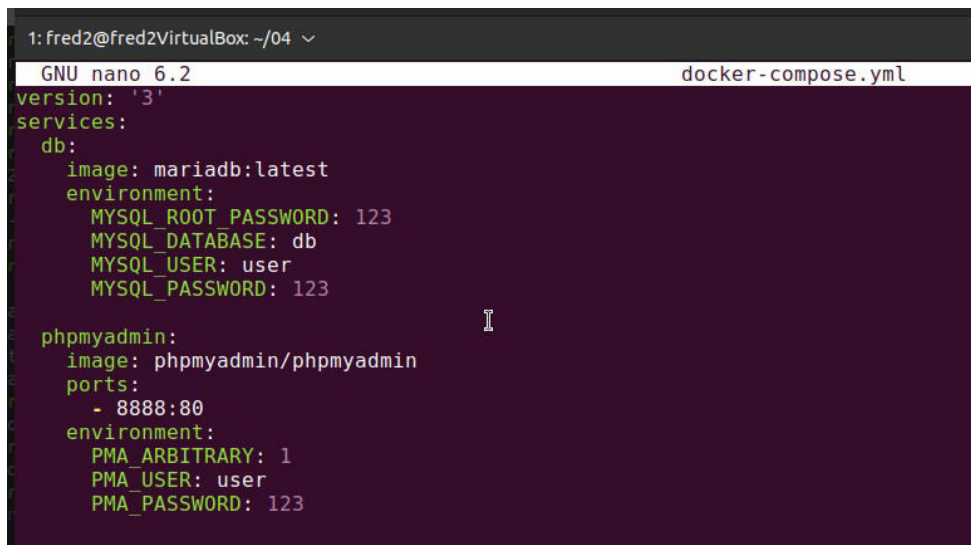


легче это было сделать с помощью docker-compose

1) Создаем файл `docker-compose.yml`

```
version: '3'
services:
  db:
    image: mariadb:latest
    environment:
      MYSQL_ROOT_PASSWORD: 123
      MYSQL_DATABASE: db
      MYSQL_USER: user
      MYSQL_PASSWORD: 123
```

```
  phpmyadmin:
    image: phpmyadmin/phpmyadmin
    ports:
      - 8888:80
    environment:
      PMA_ARBITRARY: 1
      PMA_USER: user
      PMA_PASSWORD: 123
```

A screenshot of a terminal window with a dark background. The title bar shows '1: fred2@fred2VirtualBox: ~/04'. The terminal shows the nano editor interface with 'GNU nano 6.2' and 'docker-compose.yml' in the top bar. The file content is displayed in a light green monospace font, matching the text in the previous blocks. A cursor is visible on the line 'PMA_PASSWORD: 123'.

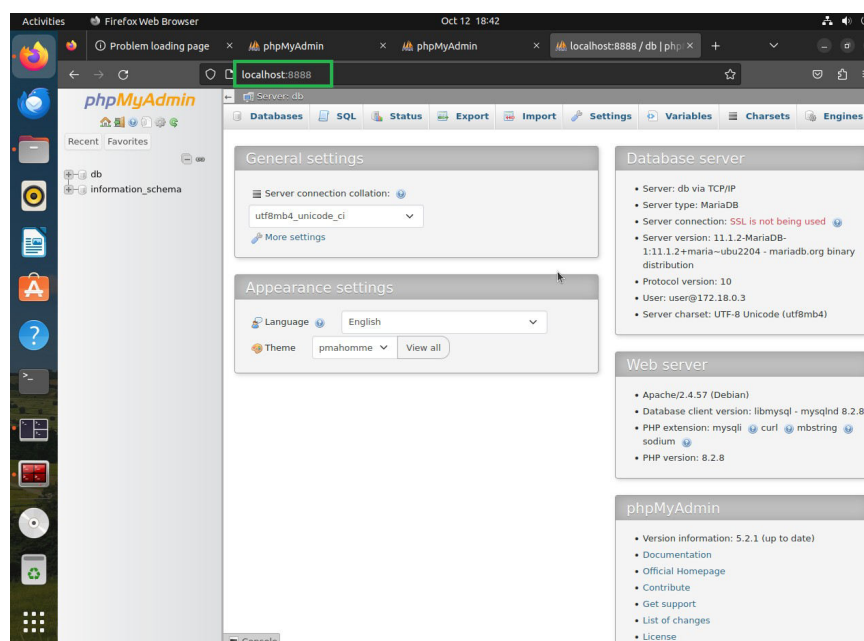
```
1: fred2@fred2VirtualBox: ~/04
GNU nano 6.2 docker-compose.yml
version: '3'
services:
  db:
    image: mariadb:latest
    environment:
      MYSQL_ROOT_PASSWORD: 123
      MYSQL_DATABASE: db
      MYSQL_USER: user
      MYSQL_PASSWORD: 123

  phpmyadmin:
    image: phpmyadmin/phpmyadmin
    ports:
      - 8888:80
    environment:
      PMA_ARBITRARY: 1
      PMA_USER: user
      PMA_PASSWORD: 123
```

2) Запускаем yaml файл в режиме демона
`docker-compose up -d`

```
Activities Terminator Oct 12 18:40
fred2@fred2VirtualBox: ~/04
fred2@fred2VirtualBox:~$ pwd
/home/fred2
fred2@fred2VirtualBox:~$ mkdir 04
fred2@fred2VirtualBox:~$ cd 04
fred2@fred2VirtualBox:~/04$ sudo nano docker-compose.yml
fred2@fred2VirtualBox:~/04$ ll
total 12
drwxrwxr-x 2 fred2 fred2 4096 Oct 12 18:03 ./
drwxr-x--- 20 fred2 fred2 4096 Oct 12 17:59 ../
-rw-r--r-- 1 root root 337 Oct 12 18:03 docker-compose.yml
fred2@fred2VirtualBox:~/04$ docker-compose up -d
Command 'docker-compose' not found, but can be installed with:
sudo snap install docker # version 20.10.24, or
sudo apt install docker # version 20.10.24
fred2@fred2VirtualBox:~/04$ sudo docker-compose up -d
sudo: docker-compose: command not found
fred2@fred2VirtualBox:~/04$ sudo docker-compose up -d
sudo: docker-compose: command not found
fred2@fred2VirtualBox:~/04$ sudo docker compose up -d
[+] Running 9/9
✔ db 8 layers [██████████] 08/08 Pulled 72.6s
✔ 707e32e9fc56 Pull complete 16.4s
✔ 4e342cc0fc32 Pull complete 0.8s
✔ 8c036f0ad2b Pull complete 5.1s
✔ bb0246c52237 Pull complete 2.4s
✔ 0b97cd52094f Pull complete 4.4s
✔ cab045a2ab07 Pull complete 34.9s
✔ 17f95f76179d Pull complete 6.0s
✔ 3e8199e03ff8 Pull complete 7.4s
[+] Running 3/3
✔ Network 04_default Created 0.5s
✔ Container 04-phpmyadmin-1 Started 2.4s
✔ Container 04-db-1 Started 2.4s
fred2@fred2VirtualBox:~/04$
```

3) проверим localhost:8888



Спасибо, всего хорошего :)