

Урок 7. Запуск веб-приложения из контейнеров.

Домашнее задание:

1. Установить в виртуальную машину (или VDS) Docker, настроить набор контейнеров через docker compose по инструкции.

Часть с настройкой certbot и HTTPS опустить, если у вас нет настоящего домена и белого IP.

2. Запустить два контейнера, связанные одной сетью (используя документацию).

Первый контейнер БД (например, образ mariadb:10.8), второй контейнер — phpmyadmin.

Получить доступ к БД в первом контейнере через второй контейнер (веб-интерфейс phpmyadmin).

Результат:

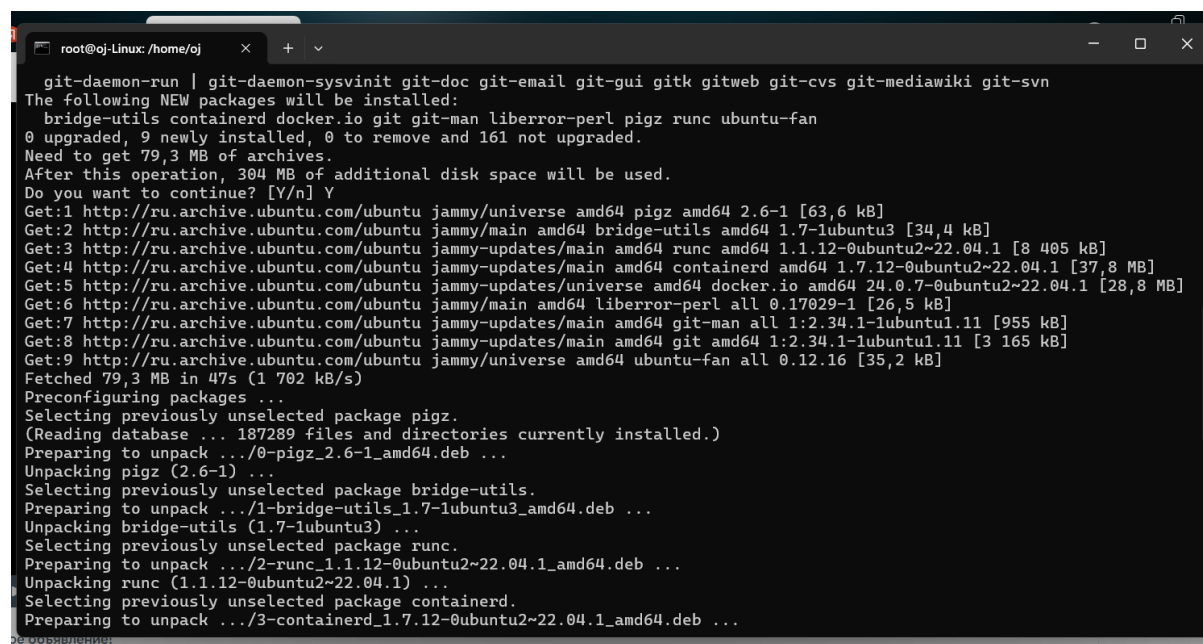
Текст команд, которые применялись при выполнении задания.

При наличии: часть конфигурационных файлов, которые решают задачу.

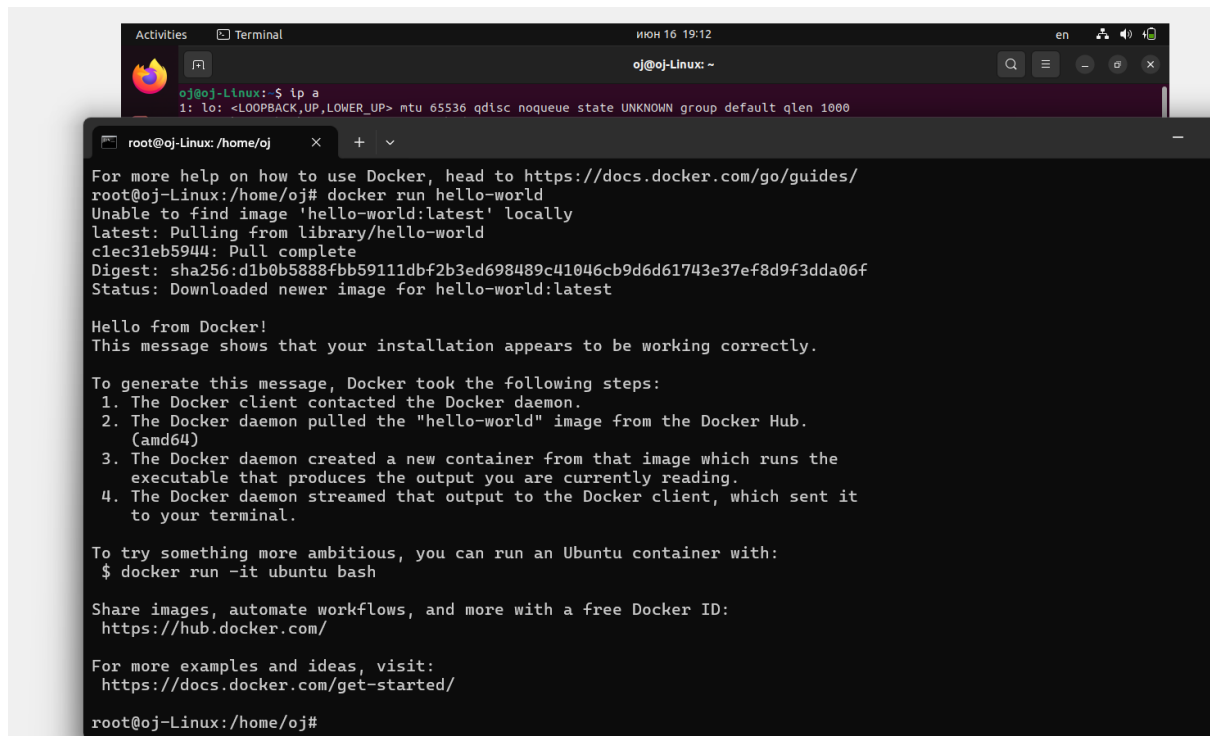
Скриншоты результата запуска контейнеров (веб-интерфейс).

Присылаем в формате текстового документа: задание и команды для решения (без вывода).

Формат — PDF (один файл на все задания).



```
root@ej-Linux: /home/ej
git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  bridge-utils containerd docker.io git git-man liberror-perl pigz runc ubuntu-fan
0 upgraded, 9 newly installed, 0 to remove and 161 not upgraded.
Need to get 79,3 MB of archives.
After this operation, 304 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://ru.archive.ubuntu.com/ubuntu jammy/universe amd64 pigz amd64 2.6-1 [63,6 kB]
Get:2 http://ru.archive.ubuntu.com/ubuntu jammy/main amd64 bridge-utils amd64 1.7-1ubuntu3 [34,4 kB]
Get:3 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 runc amd64 1.1.12-0ubuntu2~22.04.1 [8 405 kB]
Get:4 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 containerd amd64 1.7.12-0ubuntu2~22.04.1 [37,8 MB]
Get:5 http://ru.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 docker.io amd64 24.0.7-0ubuntu2~22.04.1 [28,8 MB]
Get:6 http://ru.archive.ubuntu.com/ubuntu jammy/main amd64 liberror-perl all 0.17029-1 [26,5 kB]
Get:7 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git-man all 1:2.34.1-1ubuntu1.11 [955 kB]
Get:8 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git amd64 1:2.34.1-1ubuntu1.11 [3 165 kB]
Get:9 http://ru.archive.ubuntu.com/ubuntu jammy/universe amd64 ubuntu-fan all 0.12.16 [35,2 kB]
Fetched 79,3 MB in 47s (1 702 kB/s)
Preconfiguring packages ...
Selecting previously unselected package pigz.
(Reading database ... 187289 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.6-1_amd64.deb ...
Unpacking pigz (2.6-1) ...
Selecting previously unselected package bridge-utils.
Preparing to unpack .../1-bridge-utils_1.7-1ubuntu3_amd64.deb ...
Unpacking bridge-utils (1.7-1ubuntu3) ...
Selecting previously unselected package runc.
Preparing to unpack .../2-runc_1.1.12-0ubuntu2~22.04.1_amd64.deb ...
Unpacking runc (1.1.12-0ubuntu2~22.04.1) ...
Selecting previously unselected package containerd.
Preparing to unpack .../3-containerd_1.7.12-0ubuntu2~22.04.1_amd64.deb ...
```



The screenshot shows a terminal window on a Linux system. The user has run the command `docker run hello-world`. The output indicates that the 'hello-world:latest' image was pulled from the Docker Hub. The container executed the 'hello-world' program, which printed 'Hello from Docker!' and a message stating that the installation appears to be working correctly. It also lists the steps Docker took to generate this message: contacting the daemon, pulling the image, creating a container, and streaming the output. Finally, it provides a link to the Docker Hub and a reference to the Docker documentation for more examples.

```
root@oj-Linux: /home/oj# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:d1b0b5888fbb59111dbf2b3ed698489c41046cb9d6d61743e37ef8d9f3dda06f
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

root@oj-Linux: /home/oj#
```

Решение:

Установка Docker

echo seminare7

sudo apt install docker.io docker-compose -y

sudo service apache2 stop

sudo service nginx stop

Просмотр

ss -ntlp

sudo docker ps -a sudo docker ps

Удаление

sudo docker -rm 37e9f84a9b64

Просмотр образов

sudo docker images

Поиск и скачивание контейнера

```
sudo docker search nginx
```

```
sudo docker pull nginx
```

```
sudo docker ps
```

```
sudo docker ps -a
```

Запуск контейнера(одного)

```
sudo docker run -d -p 8888:80 --name my_nginx -v /var/www/html:/usr/share/nginx/html --restart always nginx
```

```
sudo docker ps -a
```

```
sudo docker ps
```

```
localhost:8888
```

Войти в контейнер

```
sudo docker exec -ti my_nginx bash
```

Остановка, удаление активного, удаление истории, удаление образа

```
sudo docker ps
```

```
sudo docker ps -a
```

```
sudo docker images
```

```
sudo docker stop my_nginx
```

```
sudo docker rm 37e9f84a9b64
```

```
sudo docker rmi nginx
```

```
sudo docker ps
```

```
sudo docker ps -a
```

WORDPRESS

<https://www.digitalocean.com/community/tutorials/how-to-install-wordpress-with-docker-compose-ru>

```
mkdir wordpress && cd wordpress
```

```
mkdir nginx-conf
```

```
nano nginx-conf/nginx.conf
```

```
nano .env
```

```
nano .dockerignore
```

```
nano docker-compose.yml
```

```
nano docker-compose -d
```

```
nano docker-compose up -d
```

```
sudo docker-compose up -d
```

```
sudo docker ps
```