

РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

«Основы работы с Docker»

Отчет по лабораторной работе по
дисциплине «Анализ данных»

Выполнил студент группы ИВТ-б-о-21-1

Стригалов Дмитрий Михайлович.

«13» ноября 2023г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2023

Цель работы: научиться использовать основные команды Docker для управления контейнерами и понимать их назначение.

Порядок выполнения работы:

Задача 1: Основы Docker

Загрузите образ Ubuntu с Docker Hub.

```
root@ReviOS:~# docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
aece8493d397: Pull complete
Digest: sha256:2b7412e6465c3c7fc5bb21d3e6f1917c167358449fecac8176c6e496e5c1f05f
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
```

Рисунок 1 – Загрузка образа ubuntu

Создайте и запустите контейнер на основе этого образа.

```
root@ReviOS:~# docker run -it ubuntu
```

Рисунок 2 - Запуск контейнера

Войдите в созданный контейнер и выполните команду ls, чтобы просмотреть файлы внутри контейнера.

```
root@925c7e38aab2:/# ls
bin boot dev etc home lib lib32 lib64 libx32 media mnt opt proc root run sbin srv sys tmp usr var
```

Рисунок 3 - Выполнение команды ls внутри контейнера

Задача 2: Управление контейнерами и образами Загрузите образ Nginx с Docker Hub.

```
root@ReviOS:~# docker pull nginx:latest
latest: Pulling from library/nginx
1f7ce2fa46ab: Pull complete
9b16c94bb686: Pull complete
9a59d19f9c5b: Pull complete
9ea27b074f71: Pull complete
c6edf33e2524: Pull complete
84b1ff10387b: Pull complete
517357831967: Pull complete
Digest: sha256:10d1f5b58f74683ad34eb29287e07dab1e90f10af243f151bb50aa5dbb4d62ee
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
```

Рисунок 4 – Загрузка образа nginx

Создайте контейнер на основе этого образа и пробросьте порт 8080 контейнера на порт 80 хоста.

```
root@ReviOS:~# docker run -p 8080:80 nginx
/docker-entrypoint.sh: /docker-entrypoint.d/
```

Рисунок 5 - Создание контейнера и проброс порта

Посмотрите список активных контейнеров и убедитесь, что ваш контейнер работает.

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
c1bcc99e4004	nginx	"/docker-entrypoint..."	59 seconds ago	Up 3 seconds	0.0.0.0:8080->80/tcp	clever_wiles

Рисунок 6 - Список активных контейнеров

Остановите и удалите контейнер.

```
root@ReviOS:~# docker stop clever_wiles
clever_wiles
root@ReviOS:~# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS    PORTS    NAMES
root@ReviOS:~# |
root@ReviOS:~# docker rm clever_wiles
clever_wiles
```

Рисунок 7 - Остановка и удаление контейнера

Задача 3: Мониторинг и управление контейнерами

Запустите контейнер с именем "my_container".

```
root@ReviOS:~# docker run --name my_container -d nginx
bcd9af3be9ef54d4424b224ae5432d77cf3e0e92771f206b28255c5b5570db9c
root@ReviOS:~# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS    PORTS    NAMES
bcd9af3be9ef   nginx     "/docker-entrypoint..."  8 seconds ago  Up 7 seconds  80/tcp    my_container
```

Рисунок 8 – Запуск контейнера

```
root@ReviOS:~# docker stop my_container
my_container
root@ReviOS:~# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS    PORTS    NAMES
root@ReviOS:~# docker rm my_container
my_container
```

Рисунок 10 - Остановка и удаление контейнера

Задача 4: Удаление образов и оптимизация дискового пространства.

Загрузите образы Ubuntu и Alpine с Docker Hub.

```
root@ReviOS:~# docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
96526aa774ef: Pull complete
Digest: sha256:eece025e432126ce23f223450a0326fbebde39cdf496a85d8c016293fc851978
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
```

Рисунок 13 - Загрузка образа alpine

Создайте контейнеры на основе обоих образов.

```
root@ReviOS:~# docker run --name container_1 -d ubuntu
42d6551cd6a32112e82ec9da61b30e5c36ce2d3592437b0c332582054b7c99c5
root@ReviOS:~# docker run --name container_2 -d alpine
95675f609ba24f31863b685d249f8b9ec83bb851bd86215d65421249ace5ed08
```

Рисунок 14 - Создание контейнеров

Убедитесь, что контейнеры запущены и работают.

```
root@ReviOS:~# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
95675f609ba2	alpine	"/bin/sh"	28 seconds ago	Exited (0) 27 seconds ago		container_2
42d6551cd6a3	ubuntu	"/bin/bash"	38 seconds ago	Exited (0) 38 seconds ago		container_1

Рисунок 15 - Список контейнеров

Удалите образ Ubuntu.

```
root@ReviOS:~# docker rmi -f ubuntu
Untagged: ubuntu:latest
Untagged: ubuntu@sha256:2b7412e6465c3c7fc5bb21d3e6f1917c167358449fecac8176c6e496e5c1f05f
Deleted: sha256:e4c58958181a5925816faa528ce959e487632f4cfd192f8132f71b32df2744b4
```

Рисунок 16 - Удаление образа ubuntu

Проверьте, что образ Ubuntu больше не существует, но образ Alpine остался на системе.

```
root@ReviOS:~# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
95675f609ba2	alpine	"/bin/sh"	About a minute ago	Exited (0) About a minute ago		container_2
42d6551cd6a3	e4c58958181a	"/bin/bash"	About a minute ago	Exited (0) About a minute ago		container_1

Рисунок 17 - Просмотр контейнеров

Задача 5: Взаимодействие с контейнером

Запустите контейнер с именем "my_container" в фоновом режиме.

```
root@ReviOS:~# docker run --name my_container -d nginx
892bd7057fe6093b37b7797c4026cde6ede6c247a256c8abe1141e139fa98933
```

Рисунок 18 – Запуск контейнера

Используя команду `docker exec`, выполните команду `ls -l /app` внутри контейнера.

```

root@ReviOS:~# docker exec my_container ls -l /app
ls: cannot access '/app': No such file or directory
root@ReviOS:~# docker exec my_container ls -l
total 64
lrwxrwxrwx    1 root root    7 Nov 20 00:00 bin -> usr/bin
drwxr-xr-x    2 root root 4096 Sep 29 20:04 boot
drwxr-xr-x    5 root root  340 Nov 25 11:47 dev
drwxr-xr-x    1 root root 4096 Nov 21 09:05 docker-entrypoint.d
-rwxrwxr-x    1 root root 1620 Nov 21 09:05 docker-entrypoint.sh
drwxr-xr-x    1 root root 4096 Nov 25 11:47 etc
drwxr-xr-x    2 root root 4096 Sep 29 20:04 home
lrwxrwxrwx    1 root root    7 Nov 20 00:00 lib -> usr/lib
lrwxrwxrwx    1 root root    9 Nov 20 00:00 lib32 -> usr/lib32
lrwxrwxrwx    1 root root    9 Nov 20 00:00 lib64 -> usr/lib64
lrwxrwxrwx    1 root root   10 Nov 20 00:00 libx32 -> usr/libx32
drwxr-xr-x    2 root root 4096 Nov 20 00:00 media
drwxr-xr-x    2 root root 4096 Nov 20 00:00 mnt
drwxr-xr-x    2 root root 4096 Nov 20 00:00 opt
dr-xr-xr-x   339 root root    0 Nov 25 11:47 proc
drwx-----   2 root root 4096 Nov 20 00:00 root
drwxr-xr-x    1 root root 4096 Nov 25 11:47 run
lrwxrwxrwx    1 root root    8 Nov 20 00:00 sbin -> usr/sbin
drwxr-xr-x    2 root root 4096 Nov 20 00:00 srv
dr-xr-xr-x   11 root root    0 Nov 25 11:47 sys
drwxrwxrwt    1 root root 4096 Nov 21 09:05 tmp
drwxr-xr-x    1 root root 4096 Nov 20 00:00 usr
drwxr-xr-x    1 root root 4096 Nov 20 00:00 var

```

Рисунок 19 - Выполнение команды ls -l

Выполните команду ps aux внутри контейнера, чтобы увидеть список запущенных процессов.

```

root@293de6cca461:/# ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.3  0.0  4624  3612 pts/0    Ss   19:19   0:00 /bin/bash
root         9  0.0  0.0   7060  1668 pts/0    R+   19:20   0:00 ps aux

```

Рисунок 20 - Выполнение команды ps aux

Остановите и удалите контейнер.

```

root@ReviOS:~# docker stop my_container
my_container
root@ReviOS:~# docker rm my_container
my_container

```

Рисунок 21 - Остановка и удаление контейнера

Контрольные вопросы:

1. Что делает команда docker pull?

Команда docker pull в Docker используется для загрузки образа контейнера с Docker Hub или другого репозитория.

2. Какой синтаксис используется для загрузки образа с Docker Hub с помощью docker pull?

docker pull <имя_образа>:<тег>

3. Как можно просмотреть список всех доступных образов на системе с помощью docker images?

`docker images`

Эта команда выведет список всех образов, которые находятся на вашей системе, включая их имена, теги, размер и ID.

4. Какой ключ используется для просмотра образов в формате таблицы с docker images?

`docker images --format "table {{.Repository}}\t{{.Tag}}\t{{.Size}}"`

5. Как создать и запустить контейнер с использованием docker run?

`docker run [опции] <имя_образа> [команда] [аргументы]`

6. Как пробросить порт при запуске контейнера с docker run?

`docker run -p 8080:80 nginx`

7. Как изменить имя контейнера при его создании с помощью docker run?

`docker run --name my_container -d nginx`

8. Как создать контейнер в фоновом режиме с docker run? docker run -d nginx

9. Какая команда используется для просмотра активных контейнеров на системе?

`docker ps`

10. Какие опции могут использоваться с docker ps для отображения остановленных контейнеров?

`docker ps -a`

11. Как можно просмотреть список всех контейнеров, включая остановленные, с docker ps?

`docker ps -a`

12. Что делает команда docker start?

Команда `docker start` в Docker используется для запуска остановленных контейнеров.

13. Какой синтаксис используется для запуска остановленного контейнера с docker start?

`docker start [опции] <имя_или_ID_контейнера>`

14. Как запустить контейнер в фоновом режиме с docker start?

`docker start -d my_container`

15. Что делает команда docker stop?

Команда `docker stop` в Docker используется для остановки работающего контейнера.

16. Как остановить контейнер по его имени с помощью docker stop? `docker stop my_container`

17. Как принудительно остановить контейнер с docker stop? `docker stop -f my_container`

18. Что делает команда docker rm?

Команда `docker rm` в Docker используется для удаления контейнера, который был остановлен.

19. Как удалить контейнер по его ID с использованием docker rm?
`docker rm 1234567890`

20. Как удалить несколько контейнеров сразу с docker rm? `docker rm container1 container2`

21. Что делает команда docker rmi?

Команда `docker rmi` в Docker используется для удаления образов контейнеров с вашей системы.

22. Как удалить Docker-образ по его имени и тегу с помощью docker rmi?

`docker rmi ubuntu:20.04`

23. Как удалить несколько Docker-образов сразу с docker rmi?

`docker rmi image1 image2`

24. Как выполнить команду внутри работающего контейнера с docker exec?

`docker exec [опции] <имя_или_ID_контейнера> <команда> [аргументы]`

25. Как выполнить команду внутри контейнера в интерактивном режиме с docker exec?

```
docker exec -it my_container /bin/bash
```

26. Как выполнить команду с использованием определенного пользователя внутри контейнера с docker exec?

```
docker exec -u 1000 my_container whoami
```

27. Какой ключ используется для запуска команды в фоновом режиме с docker exec?

```
docker exec -d my_container my_command
```

28. Как выполнить команду внутри контейнера с именем вместо ID с docker exec?

```
docker exec -it $(docker ps -q -f "name=my_container") /bin/bash
```

29. Как передать аргументы при выполнении команды с docker exec?

```
docker exec [опции] <имя_или_ID_контейнера> <команда> [аргументы]
```

30. Как проверить список доступных команд и опций для docker exec?

```
docker exec --help
```

31. Как передать переменную окружения в контейнер при его запуске?

```
docker run -e MYSQL_ROOT_PASSWORD=my-secret-pw mysql
```

32. Какой ключ используется для запуска контейнера в фоновом режиме с командой docker run?

```
docker run -d nginx
```

33. Как проверить статус выполнения контейнеров на системе с помощью docker ps?

```
docker ps -s
```

34. Как завершить выполнение контейнера без его удаления?

```
docker stop my_container
```


35. Каким образом можно удалить все остановленные контейнеры с системы?

```
docker rm $(docker ps -aq)
```

36. Что делает опция -a при использовании docker ps?

Добавление опции -a позволяет просматривать все контейнеры, включая те, которые были остановлены.

37. Что означает опция -q при выполнении docker ps?

Добавление опции -q выводит только ID контейнеров.

38. Как принудительно удалить контейнер с флагом -f? docker rm -f my_container

39. Какой Docker-образ и какую команду можно использовать для создания контейнера с базой данных PostgreSQL?

```
docker run --name postgres_container postgres
```

40. Какой ключ используется для выполнения команды внутри контейнера в интерактивном режиме?

```
docker exec -it my_container <команда>
```

41. Какой ключ можно использовать для передачи ID пользователя при выполнении команды внутри контейнера?

С опцией -u мы указываем ID пользователя, от имени которого будет выполнена команда.

Вывод: были изучены основные команды Docker для управления контейнерами.