

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение
высшего образования**

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра информационных систем и технологий

Отчет по лабораторной работе «основы работы с Docker».

Дисциплина: «Основы программной инженерии»

Выполнил:

Студент группы ПИЖ-б-о-22-1,

направление подготовки: 09.03.04

«Программная инженерия»

ФИО: Джараян Арег Александрович

Проверил:

Воронкин Р. А.

Ставрополь 2023

Тема: Лабораторная работа 1. Основы работы с Docker.

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

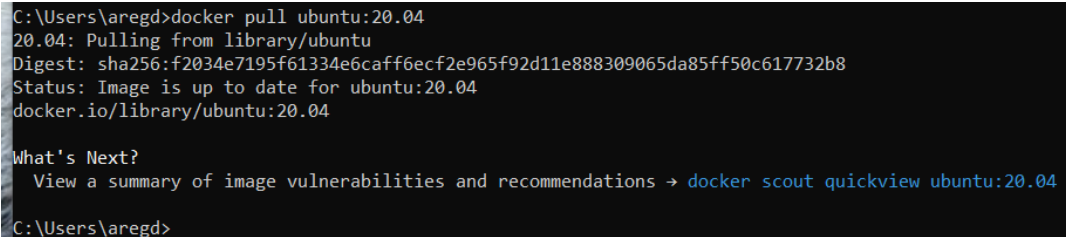
Выполнение работы:

Задача 1: Основы Docker.

Цель: познакомиться с основами Docker и командами для работы с контейнерами.

Задача:

1. Загрузите образ Ubuntu с Docker Hub.



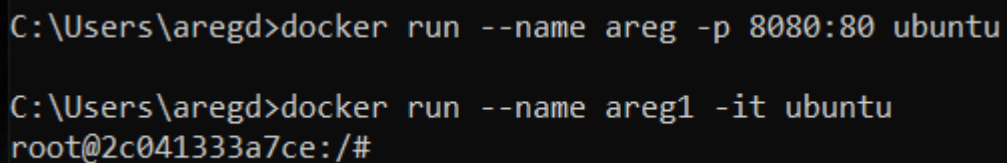
```
C:\Users\aregd>docker pull ubuntu:20.04
20.04: Pulling from library/ubuntu
Digest: sha256:f2034e7195f61334e6caff6ecf2e965f92d11e888309065da85ff50c617732b8
Status: Image is up to date for ubuntu:20.04
docker.io/library/ubuntu:20.04

What's Next?
View a summary of image vulnerabilities and recommendations → docker scout quickview ubuntu:20.04

C:\Users\aregd>
```

Рисунок 1.1 – загрузка образа ubuntu

2. Создайте и запустите контейнер на основе этого образа.

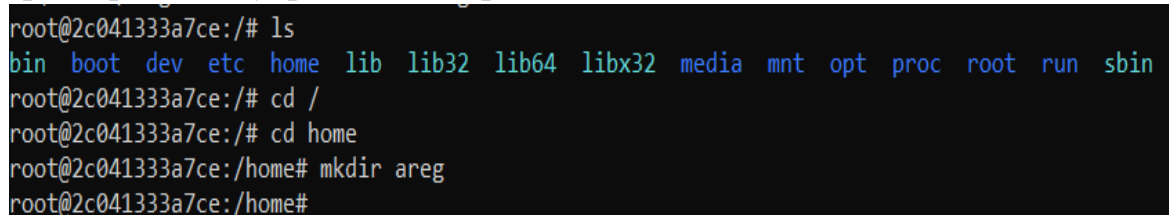


```
C:\Users\aregd>docker run --name areg -p 8080:80 ubuntu

C:\Users\aregd>docker run --name areg1 -it ubuntu
root@2c041333a7ce:/#
```

Рисунок 1.2 – создание и запуск контейнера

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



```
root@2c041333a7ce:/# ls
bin boot dev etc home lib lib32 lib64 libx32 media mnt opt proc root run sbin
root@2c041333a7ce:/# cd /
root@2c041333a7ce:/# cd home
root@2c041333a7ce:/home# mkdir areg
root@2c041333a7ce:/home#
```

Рисунок 1.3 – ос ubuntu

Задача 2: Управление контейнерами и образами.

Цель: освоить команды для управления контейнерами и образами Docker.

Задача:

1. Загрузите образ Nginx с Docker Hub.

```
C:\Users\aregd>docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
af107e978371: Pull complete
336ba1f05c3e: Pull complete
8c37d2ff6efa: Pull complete
51d6357098de: Pull complete
782f1ecce57d: Pull complete
5e99d351b073: Pull complete
7b73345df136: Pull complete
Digest: sha256:2bdc49f2f8ae8d8dc50ed00f2ee56d00385c6f8bc8a8b320d0a294d9e3b49026
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview nginx

C:\Users\aregd>
```

Рисунок 2.1 – загрузка образа

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

```
C:\Users\aregd>docker run --name nginx -p 8080:80 nginx
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform con
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/defau
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/def
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/12/28 09:10:10 [notice] 1#1: using the "epoll" event method
2023/12/28 09:10:10 [notice] 1#1: nginx/1.25.3
2023/12/28 09:10:10 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2023/12/28 09:10:10 [notice] 1#1: OS: Linux 5.15.133.1-microsoft-standard-WSL2
2023/12/28 09:10:10 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/12/28 09:10:10 [notice] 1#1: start worker processes
2023/12/28 09:10:10 [notice] 1#1: start worker process 29
2023/12/28 09:10:10 [notice] 1#1: start worker process 30
2023/12/28 09:10:10 [notice] 1#1: start worker process 31
2023/12/28 09:10:10 [notice] 1#1: start worker process 32
2023/12/28 09:10:10 [notice] 1#1: start worker process 33
2023/12/28 09:10:10 [notice] 1#1: start worker process 34
2023/12/28 09:10:10 [notice] 1#1: start worker process 35
2023/12/28 09:10:10 [notice] 1#1: start worker process 36
```

Рисунок 2.2 – создание контейнера на основе образа

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

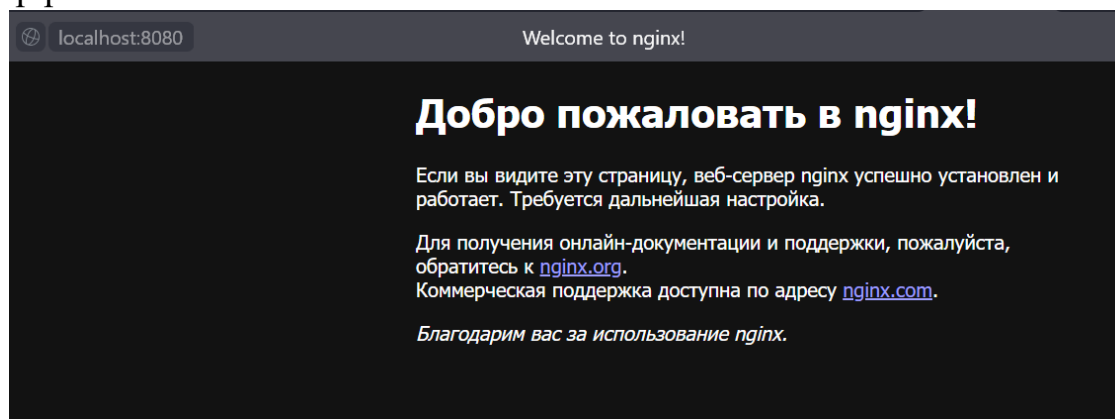


Рисунок 2.3 – localhost:8080

```
C:\Users\aregd>docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
f660db662e28	nginx	"/docker-entrypoint..."	6 minutes ago	Exited (0) 19 seconds ago

Рисунок 2.4 – список активных контейнеров

4. Остановите и удалите контейнер.

```
C:\Users\aregd>docker stop nginx
nginx

C:\Users\aregd>docker rm nginx
nginx

C:\Users\aregd>docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
ubuntu	20.04	f78909c2b360	2 weeks ago	72.8MB
ubuntu	latest	174c8c134b2a	2 weeks ago	77.9MB
docker/welcome-to-docker	latest	c1f619b6477e	7 weeks ago	18.6MB
nginx	latest	d453dd892d93	2 months ago	187MB
openjdk	latest	71260f256d19	10 months ago	470MB
docker/getting-started	latest	3e4394f6b72f	12 months ago	47MB

```
C:\Users\aregd>docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
2c041333a7ce	ubuntu	"/bin/bash"	22 minutes ago	Exited (0) 13 minutes ago
2fc7bcea0ca6	ubuntu	"/bin/bash"	28 minutes ago	Exited (0) 15 minutes ago
96312b576acd	ubuntu	"/bin/bash"	12 hours ago	Exited (255) 32 minutes ago
a869c67ea7a8	openjdk	"jshell"	12 hours ago	Exited (143) 12 hours ago
2924c9e84e44	openjdk	"jshell"	12 hours ago	Exited (0) 12 hours ago
a9dc98323efa	ubuntu	"/bin/bash"	14 hours ago	Exited (137) 14 hours ago

Рисунок 2.5 – остановка и удаление контейнера

Задача 3: Мониторинг и управление контейнерами.

Цель: изучить команды мониторинга и управления контейнерами.

Задача:

1. Запустите контейнер с именем "my_container".

```
C:\Users\aregd>docker run --name my_container -d nginx
825ecc06b98f79dd0057ea4d92333478515bef397dd61674a8434c3ee519aa07
```

Рисунок 3.1 – запуск контейнера с именем "my_container"

2. Используя команду docker ps, убедитесь, что контейнер запущен.

```
C:\Users\aregd>docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
825ecc06b98f	nginx	"/docker-entrypoint..."	About a minute ago	Up About a minute

```
C:\Users\aregd>
```

Рисунок 3.2 – запущенные контейнеры

3. Остановите контейнер.

```
C:\Users\aregd>docker stop 825ecc06b98f
825ecc06b98f
```

Рисунок 3.3 – остановка контейнера

4. Проверьте его статус снова и убедитесь, что он остановлен.

```
C:\Users\aregd>docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
C:\Users\aregd>
```

Рисунок 3.4 – список запущенных контейнеров

5. Удалите контейнер.

```
C:\Users\aregd>docker rm my_container
my_container

C:\Users\aregd>docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
2c041333a7ce   ubuntu   "/bin/bash"   31 minutes ago
areg1
2fc7bcea0ca6   ubuntu   "/bin/bash"   37 minutes ago
areg
96312b576acd   ubuntu   "/bin/bash"   12 hours ago
a869c67ea7a8   openjdk   "jshell"      12 hours ago
myjava
2924c9e84e44   openjdk   "jshell"      12 hours ago
nervous_williams
a9dc98323efa   ubuntu   "/bin/bash"   15 hours ago
exciting_leakey
fe3bb70c33b9   ubuntu:20.04   "/bin/bash"   15 hours ago
festive_feynman
da49362a7d93   docker/getting-started   "/docker-entrypoint...."   15 hours ago
compassionate_pike
c8727aa592e0   docker/welcome-to-docker:latest   "/docker-entrypoint...."   15 hours ago
welcome-to-docker

C:\Users\aregd>
```

Рисунок 3.5 – удаление контейнера

Задача 4: Удаление образов и оптимизация дискового пространства.

Цель: освоить команды для удаления образов и оптимизации использования дискового пространства.

Задача:

1. Загрузите образы Ubuntu и Alpine с Docker Hub.

```
C:\Users\aregd>docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
661ff4d9561e: Pull complete
Digest: sha256:51b67269f354137895d43f3b3d810bfacd3945438e94dc5ac55fdac340352f48
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview alpine

C:\Users\aregd>
```

Рисунок 4.1 – установка образа alpine

2. Создайте контейнеры на основе обоих образов.

```
C:\Users\aregd>docker run --name konteiner1 -it alpine
/ #

C:\Users\aregd>docker run --name konteiner2 -it ubuntu
root@6b481f904e42:/#
exit
```

Рисунок 4.2 – создание 2 контейнеров

3. Убедитесь, что контейнеры запущены и работают.

```
C:\Users\aregd>docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
6b481f904e42	ubuntu	"/bin/bash"	48 seconds ago	Up 3 seconds		konteiner2
89b06f48e154	alpine	"/bin/sh"	About a minute ago	Up 9 seconds		konteiner1

Рисунок 4.3 – список активных контейнеров

4. Удалите образ Ubuntu.

```
C:\Users\aregd>docker rmi -f f78909c2b360
Untagged: ubuntu:20.04
Untagged: ubuntu@sha256:f2034e7195f61334e6caff6ecf2e965f92d11e888309065da85ff
Deleted: sha256:f78909c2b360d866b3220655c0b079838258b8891a12ac25fc670f0cbb542

C:\Users\aregd>docker rmi -f 174c8c134b2a
Untagged: ubuntu:latest
Untagged: ubuntu@sha256:6042500cf4b44023ea1894effe7890666b0c5c7871ed83a97c36c
Deleted: sha256:174c8c134b2a94b5bb0b37d9a2b6ba0663d82d23ebf62bd51f74a2fd45733

C:\Users\aregd>AAAAA
```

Рисунок 4.4 – удаление двух образов ubuntu

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

```
C:\Users\aregd>docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
alpine	latest	f8c20f8bbcb6	2 weeks ago	7.38MB
docker/welcome-to-docker	latest	c1f619b6477e	7 weeks ago	18.6MB
nginx	latest	d453dd892d93	2 months ago	187MB
openjdk	latest	71260f256d19	10 months ago	470MB
docker/getting-started	latest	3e4394f6b72f	12 months ago	47MB

Задача 5: Взаимодействие с контейнером/

Цель: научиться взаимодействовать с работающим контейнером и выполнить команды внутри него.

Задача:

1. Запустите контейнер с именем "my_container" в фоновом режиме.
2. Используя команду `docker exec`, выполните команду `ls -l /app` внутри контейнера.

```
C:\Users\aregd>docker exec my_container ls -l
total 48
lrwxrwxrwx 1 root root 7 Dec 11 14:04 bin -> usr/bin
drwxr-xr-x 2 root root 4096 Apr 18 2022 boot
drwxr-xr-x 5 root root 360 Dec 28 10:08 dev
drwxr-xr-x 1 root root 4096 Dec 28 10:07 etc
drwxr-xr-x 2 root root 4096 Apr 18 2022 home
lrwxrwxrwx 1 root root 7 Dec 11 14:04 lib -> usr/lib
lrwxrwxrwx 1 root root 9 Dec 11 14:04 lib32 -> usr/lib32
lrwxrwxrwx 1 root root 9 Dec 11 14:04 lib64 -> usr/lib64
lrwxrwxrwx 1 root root 10 Dec 11 14:04 libx32 -> usr/libx32
drwxr-xr-x 2 root root 4096 Dec 11 14:04 media
drwxr-xr-x 2 root root 4096 Dec 11 14:04 mnt
drwxr-xr-x 2 root root 4096 Dec 11 14:04 opt
dr-xr-xr-x 250 root root 0 Dec 28 10:08 proc
drwx----- 1 root root 4096 Dec 28 10:08 root
drwxr-xr-x 5 root root 4096 Dec 11 14:08 run
lrwxrwxrwx 1 root root 8 Dec 11 14:04/sbin -> usr/sbin
drwxr-xr-x 2 root root 4096 Dec 11 14:04 srv
dr-xr-xr-x 11 root root 0 Dec 28 10:08 sys
drwxrwxrwt 2 root root 4096 Dec 11 14:08 tmp
drwxr-xr-x 14 root root 4096 Dec 11 14:04 usr
drwxr-xr-x 11 root root 4096 Dec 11 14:08 var
```

Рисунок 5.1 – выполнение команды

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

```
C:\Users\aregd>docker exec my_container ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0  4624  3740 pts/0    Ss+  10:08   0:00 /bin/bash
root        21  0.0  0.0   7060  1564 ?        Rs   10:10   0:00 ps aux
```

Рисунок 5.2 – список запущенных процессов

4. Остановите и удалите контейнер.

```
C:\Users\aregd>docker stop my_container
my_container

C:\Users\aregd>docker rm my_container
my_container

C:\Users\aregd>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS
NAMES
89b06f48e154   alpine    "/bin/sh"               31 minutes ago Up 30 minutes
konteiner1
c64ec3b048c7   alpine    "/bin/sh"               32 minutes ago Up 31 minutes
kont222
C:\Users\aregd>
```

Рисунок 5.3 – остановка и удаление контейнера

Контрольные вопросы:

1. Что делает команда `docker pull` ?

Команда `docker pull` используется для скачивания Docker-образа из репозитория и сохранения его на локальной машине. Эта команда загружает образы, но не создает контейнеры.

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

Синтаксис для загрузки образа с Docker Hub с помощью `docker pull` выглядит следующим образом: `docker pull <имя_репозитория> /<имя_образа> :<тег>` (например, `docker pull nginx:latest`). Если тег не указан, по умолчанию используется тег `latest`.

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

Для просмотра списка всех доступных образов на системе можно воспользоваться командой `docker images`, которая выводит список всех локально доступных Docker-образов.

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

Для просмотра образов в формате таблицы с использованием `docker images`, можно использовать ключ `-a` или `--all`, например: `docker images -a` или `docker images --all`, что позволяет просмотреть все образы, включая те, которые не помечены тегами.

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

Для создания и запуска контейнера с использованием `docker run` используйте синтаксис: `docker run <опции> <имя_образа>` (например, `docker run -it ubuntu`).

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

Для проброса порта при запуске контейнера используйте опцию `-p` или `--publish`, например: `docker run -p 8080:80 nginx`, что пробросит порт 8080 на хосте на порт 80 внутри контейнера.

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

Изменение имени контейнера при его создании с помощью `docker run` осуществляется с использованием опции `--name`, например: `docker run --name my_container nginx`.

8. Как создать контейнер в фоновом режиме с `docker run` ?

Для создания контейнера в фоновом режиме с использованием `docker run` используйте опцию `-d` или `--detach`, например: `docker run -d nginx`.

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

Для просмотра активных контейнеров на системе используйте команду `docker ps`, которая показывает список активных (запущенных) контейнеров.

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

Для отображения остановленных контейнеров с использованием `docker ps`, можно использовать опции `-a` или `--all`, например: `docker ps -a` или `docker ps --all`, что выводит список всех контейнеров, включая остановленные.

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

Для просмотра всех контейнеров, включая остановленные, используйте команду: `docker ps -a` или `docker ps --all`.

12. Что делает команда `docker start` ?

Команда `docker start` запускает остановленный контейнер.

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

Синтаксис: `docker start <идентификатор_или_имя_контейнера>`.

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

Команда `docker start` не позволяет указать опцию для фонового режима. Если контейнер был запущен в фоновом режиме, он будет запущен в том же режиме.

15. Что делает команда `docker stop` ?

Команда `docker stop` останавливает работающий контейнер.

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

Синтаксис: `docker stop <имя_контейнера>`.

17. Как принудительно остановить контейнер с `docker stop` ?

Команда `docker stop` по умолчанию пытается остановить контейнер деликатно. Для принудительной остановки используйте `docker stop -f` или `docker stop --force`.

18. Что делает команда `docker rm` ?

Команда `docker rm` удаляет контейнеры.

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

`docker rm <ID_контейнера>`.

20. Как удалить несколько контейнеров сразу с docker rm ?

`docker rm <ID_контейнера1> <ID_контейнера2>`

21. Что делает команда docker rmi ?

Команда `docker rmi` удаляет Docker-образ

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

`docker rmi <имя_образа>:<тег>`

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

`docker rmi <имя_образа1>:<тег1> <имя_образа2>:<тег2>`

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

`docker exec <ID_или_имя_контейнера> <команда>`

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

Используйте опцию `-it`, например: `docker exec -it my_container /bin/bash`

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

Используйте опцию `--user`, например: `docker exec --user=username my_container ls`

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

Используйте опцию `-d`, например: `docker exec -d my_container <команда>`

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

`docker exec <имя_контейнера> <команда>`

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

Аргументы могут быть переданы после имени или ID контейнера и команды, например: `docker exec <имя_или_ID_контейнера> <команда> <аргумент1> <аргумент2>`.

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

Для просмотра справки и доступных опций можно воспользоваться командой `docker exec --help`.

31 Как передать переменную окружения в контейнер при его запуске?

Используйте опцию `-e` при запуске контейнера с командой `docker run`.

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

Используйте опцию `-d` или `--detach`, например: `docker run -d <имя_образа>`.

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

`docker ps` покажет только активные (запущенные) контейнеры. Для просмотра всех контейнеров, включая остановленные, используйте опцию `-a` или `--all`, например: `docker ps -a`.

34. Как завершить выполнение контейнера без его удаления?

Используйте команду `docker stop <имя_или_ID_контейнера>` для остановки контейнера.

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

`docker rm $(docker ps -q -f status=exited)` удаляет все остановленные контейнеры.

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

Опция `-a` или `--all` выводит список всех контейнеров, включая остановленные.

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

Опция `-q` выводит только ID контейнеров (quiet mode), полезно, например, при использовании в комбинации с другими командами.

38. Как принудительно удалить контейнер с флагом `-f` ?

Используйте `docker rm -f <имя_или_ID_контейнера>`.

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

Например, `docker run --name postgres-container -e POSTGRES_PASSWORD=mysecretpassword -d postgres:latest` создаст и запустит контейнер PostgreSQL с паролем.

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

Используйте опции `-it`, например: `docker exec -it <имя_или_ID_контейнера> <команда>`.

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

Используйте опцию `--user`, например: `docker exec --user=user_ID <имя_или_ID_контейнера> <команда>`.