## Демо 1

Итак, на прошлых семинарах мы запускали приложения в изолированном пространстве. И, при этом, у вас в голове могли возникать мысли: зачем вообще мы это все делаем. Так вот, на лекции я это уже показал, а теперь давайте углубимся в практику.

### Пример решения

|  |
| --- |
| docker run -it ubuntu:22.10 bash  ps -aux  не работает:  ip a  ifconfig  apt update  apt install net-tools  ifconfig  hostname  exit  docker ps -a  docker run -it -h GeekBrains ubuntu:22.10 bash  hostname  ifconfig (не работает)  exit  docker ps -a  docker start <name of container>  docker exec -it <name of container> bash  Можно вообще любую команду запускать без входа в контейнер:  docker exec -it elated\_elbakyan hostname  docker ps -a docker stop <name of container>  Показать выход с нулевым кодом, к примеру  docker run -it -h GB --name gb-test ubuntu:22.10  exit  docker ps -a #показать код выхода 0  попытаться запустить еще раз этот же контейнер и получим ошибку  docker run -it -h GB --name gb-test ubuntu:22.10  docker start gb-test  docker exec -it gb-test bash  контейнер не остановится ни Ctrl-D, ни через exit  нет команд shutdown и reboot  Дальше остановить контейнер, запустить с centos  docker run -it centos bash  выйти из контейнера  сказать docker start <имя контейнера>  docker ps -a  docker stop <имя контейнера>  docker ps -a # выход сигнала другой  Это связано с внутренней архитектурой контейнера и зависит от того, что разработчики или мы сами туда вложили.  удалить все, что накопилось  docker rm $(docker ps -a -q)  docker rmi $(docker images -aq)  docker rmi $(docker images -aq) –force |

## Демо 2

Итак, первую половину мы прошли. Теперь мы умеем запускать контейнеры, работать с параметрами и свойствами. Теперь давайте разберемся с хранением данных. Это очень важно т.к. каждому инженеру, который работает с контейнерами необходимо понимать где и как хранятся данные. Сейчас мы рассмотрим несколько вариантов.

|  |
| --- |
| docker run -it -h GB --name gb-test ubuntu:22.10 #для начала запустим контейнер из образа и “провалимся” вовнутрь  ls -l / #посмотрим, что есть в корне  mkdir /example #создадим папку в корне  vi /example/passwords.txt #создадим файл и добавим каких-либо данных. Предположим, что на месте этого файла могут быть данные сайта либо БД. Теперь давайте разбираться: в каких моментах мы сохраним данные, а в каких - нет.  Вопрос! Как записать в файл, если у нас нет редактора?  Ответ ниже  touch /example/passwords.txt  echo "123test" >> /example/passwords.txt |

Первым делом попробуем остановить контейнер и запустить снова.  
Вопрос: как думаете, данные сохранятся? И почему?

Ответ - сохранятся тк мы не пересоздавали контейнер.

|  |
| --- |
| docker start gb-test  docker exec -it gb-test bash  cat /example/passwords.txt |

Вариант два (он все-таки очевиден, но давайте попробуем). Удалим наш контейнер и создадим заново, используя те же команды:

|  |
| --- |
| docker run -it -h GB --name gb-test ubuntu:22.10 |

Очевидно, тут ничего нет.

|  |
| --- |
| docker stop gb-test  docker rm gb-test |

Наиболее интересный вариант: теперь давайте добавим внешнее хранилище. Прежде чем его смонтировать, давайте создадим.

|  |
| --- |
| mkdir /test/folder  docker run -it -h GB --name gb-test -v /test/folder:/otherway ubuntu:22.10  Директория примонтировалась в путь, который мы и указали, хотя изначально в контейнере и не было никаких данных об этом пути.  Теперь давайте добавим данных  echo “$HOSTNAME” >> /otherway.test.txt  etc  Ну и остановим контейнер. Следующий момент: данные доступны с локальной системы:  cat /test/folder/test.txt  Удалим контейнер  docker rm gb-test  и пересоздадим, подмонтировав папку. |

Все работает! Это и есть идеальный вариант хранения данных при использовании контейнеров: не хранить никаких важных данных внутри контейнера!

## Задача 1

### Задание

Необходимо:

* Создать папку, которую мы будем готовы смонтировать в контейнер
* В этой папке создать файл test.txt и наполнить данными
* В домашней директории создать файл test.txt, который также необходимо будет смонтировать в контейнер и наполнить совершенно другими данными
* Создать контейнер из образа ubuntu:22.10
* Задать ему имя
* Задать hostname
* Смонтировать созданную ранее папку с хоста в контейнер
* Смонтировать созданный ранее текстовый файл внутрь смонтированной папки, чтобы он пересекался с созданным ранее файлом в этой папке. Просмотреть этот файл.
* Обсудить, как это возможно и почему случилось?
* Что произошло с изначальным файлом?

### Пример решения

|  |
| --- |
| vi /test/folder/test.txt  vi /root/test.txt  docker run -it -h GB --name gb-test -v /test/folder:/otherway -v /root/test.txt:/otherway/test.txt ubuntu:22.10  root@GB:/# cat /otherway/test.txt |

1. Задача 2

### Задание

Необходимо запустить в контейнере базу данных MariaDB версии 10.10.2, используя инструкции, представленные в dockerhub: <https://hub.docker.com/_/mariadb>

### Пример решения

Это примерный вариант решения, главное – чтобы логика была верной.

|  |
| --- |
| docker run --name test-mariadb -e MARIADB\_ROOT\_PASSWORD=test123 -d mariadb:10.10.2  docker ps #проверить, что работает  docker exec -it test-mariadb bash  mariadb -u root -p  show databases; |

## Задача 3

### Задание

Разделить учеников на группы (в группе 2-5 человек).

Далее необходимо запустить второй контейнер. Теперь запускаем контейнер phpmyadmin, используя инструкции: <https://hub.docker.com/_/phpmyadmin>

В данном случае, при запуске этого контейнера необходимо передать данные контейнера mariadb: имя контейнера, которое вы дали ранее.

ВАЖНО! В данном случае необходимо иметь доступ к машине с Linux, чтобы можно было запустить web-интерфейс phpmyadmin и увидеть подключенную БД.

Пример решения

Это примерный вариант решения, главное – чтобы логика была верной.

|  |
| --- |
| docker run --name my-phpmyadmin -d --link test-mariadb:db -p 8081:80 phpmyadmin/phpmyadmin |

### 

В конце выполнения предложить удалить все, что было сделано:

|  |
| --- |
| docker rm test-mariadb  docker rm my-phpmyadmin  docker ps -a |

## Задача 5

### Задание

Разделить учеников на группы (в группе 2-5 человек).

Предлагается использовать контейнер с БД mariaDB. Необходимо создать контейнер, смонтировав вовнутрь папку таким образом, чтобы БД хранилась локально на машине с Linux.

Последовательность действий:

* Войти в контейнер
* Создать БД через интерфейс mariaDB
* Удалить контейнер
* Инициализировать новый
* Проверить, что созданные ранее данные доступны

Для решения задачи потребуются инструкции с сайта: <https://hub.docker.com/_/mariadb>

Также потребуется этот мануал, если вдруг ученики не работали с БД ранее: <https://mariadb.com/kb/en/create-database/>

### Пример решения

Это примерный вариант решения, главное – чтобы логика была верной.

|  |
| --- |
| mkdir /test-db  docker run --name test-mariadb -e MARIADB\_ROOT\_PASSWORD=test123 -v /test-db:/var/lib/mysql -d mariadb:10.10.2  docker ps #проверить, что работает  docker exec -it test-mariadb bash  mariadb -u root -p  CREATE DATABASE test;  show databases;  ll /test-db  docker stop test-mariadb  docker rm test-mariadb  docker ps #проверить, что контейнер удален  #создать заново контейнер с БД, проверить, что созданная ранее БД доступна и с ней все ОК |