# 

### Задание

* Необходимо создать 2 контейнера и показать возможность взаимодействия между собой (командой пинг)
* В данном случае необходимо повторить пример из лекции и продемонстрировать линковку контейнеров

### Пример решения

|  |
| --- |
| docker run --name some-mysql -e MYSQL\_ROOT\_PASSWORD=my-secret-pw -d mysql:8.0.31  docker run --name myphp -d --link some-mysql:db -p 8081:80 phpmyadmin/phpmyadmin |

## Задача 2 (обязательно)

### Задание

Создать Docker Compose файл, который бы запускал 2 контейнера. Повтор примера лекции для лучшего понимания процесса

### Пример решения

|  |
| --- |
| version: ‘3.9’  services:  db:  image: mariadb:10.10.2  restart: always  environment:  MYSQL\_ROOT\_PASSWORD: 12345  adminer:  image: adminer:4.8.1  restart: always  ports:  - 6080:8080  docker-compose up  docker container inspect adminer  docker container inspect mariadb |

## Задача 3 (обязательно)

### Задание

Необходимо пройтись по списку доступных команд у ДК с целью лучшего понимания обозначения каждой.

### Пример решения

|  |
| --- |
| docker-compose build - команда позволяет собрать сервисы, описанные в конфигурационных файлах  docker-compose up -d - запускает наш проект. В данном случае проект запустится в фоновом режиме т.к. в команде присутствует флаг -d  docker-compose start - запускает любые остановленные ранее сервисы в соответствии с указанными параметрами  docker-compose down - останавливает наш проект и, что немаловажно, удаляет все сервисы, которые были запущены ранее  docker-compose stop - эта команда просто останавливает все сервисы, описанные в конфигурации. Она не удаляет контейнеры, тома, сети и прочие сущности, описанные в конфигурационном файле  docker-compose logs -f [service name] - с помощью этой команды можно посмотреть логи нашего сервиса  docker-compose ps - выводит на экран список всех доступных контейнеров  docker-compose exec [service name] [command] - с ее помощью можно выполнить команду в сервисе, не заходя при этом в контейнер. Ранее мы рассматривали подобное на уроке Введение в Docker  docker-compose images - позволяет вывести список образов. |

## Задача 4 (обязательно)

### Задание

Создать свой кластер из нод докера. Сделать так, чтобы каждая нода могла управлять кластером (была лидером). Также необходимо добавить каждой ноде по метке: prod, stage, lab. Проверить и убедиться, что метки действительно добавились.

### Пример решения

|  |
| --- |
| docker swarm init  docker swarm join --token SWMTKN-1-3un77cn4m5ok3ijrdouwg3mit69uwmfwx96krc7taua7ovpjha-97y4zk9ppc8hxk2caxlig23xo 192.168.50.90:2377  docker node ls  docker node update --label-add env=prod docker-1  docker node inspect stage |

## Задача 5 (обязательно)

### Задание

Создать простой сервис в количестве трех реплик. Использовать можно nginx:alpine. Попросить проверить: на каких нодах запустился сервис. Второй запуск сервисов запустить второй командой. Объяснить почему во втором случае у нас не указывалось количество реплик.

Показать студентам, как запускается сервис (шаблон):

|  |
| --- |
| docker service create --name <name-service> --replicas <count-replicas> <image-name>  docker service create --mode=global --name <name-service> <image-name> |

### Пример решения

|  |
| --- |
| docker service create --name nginx --replicas 3 nginx:alpine  docker service ps nginx  docker service create --mode=global --name nginx-global nginx:alpine |

## Задача 6 (обязательно)

### Задание

Создать простой сервис в количестве трех реплик. Теперь необходимо запустить тот же сервис, но на разных нодах. Для этого необходимо использовать “селектор”. Добавить необходимо метки нод. Использовать можно nginx:alpine. Попросить проверить: на каких нодах запустился сервис

|  |
| --- |
| docker service create --name <name-service> --label <key>=<value>  --replicas <count-replicas> <image-name> |

### Пример решения

|  |
| --- |
| docker service create --name nginx --label env=lab --replicas 3 nginx:alpine;  docker service ps nginx; |

## Задача 7 (обязательно)

### Задание

Отскейлить созданный ранее сервис. Для этого использовать команду:

|  |
| --- |
| docker service scale <service\_name>=<count> |

### Пример решения

|  |
| --- |
| docker service scale nginx=1  docker service ps nginx |

## 