Инструментальные средства информационных систем

Лабораторная работа 3 Hello Docker

Цель:

Познакомиться с один из инструментов контейнеризации – Docker Desktop.

Задачи:

1. Установить Docker Desktop
2. Познакомиться с основными командами
3. Установить несколько контейнеров и проверить их работу

Содержание

Примечание: сохраните все команды, которые выполните в командной строке, в текстовый файл.

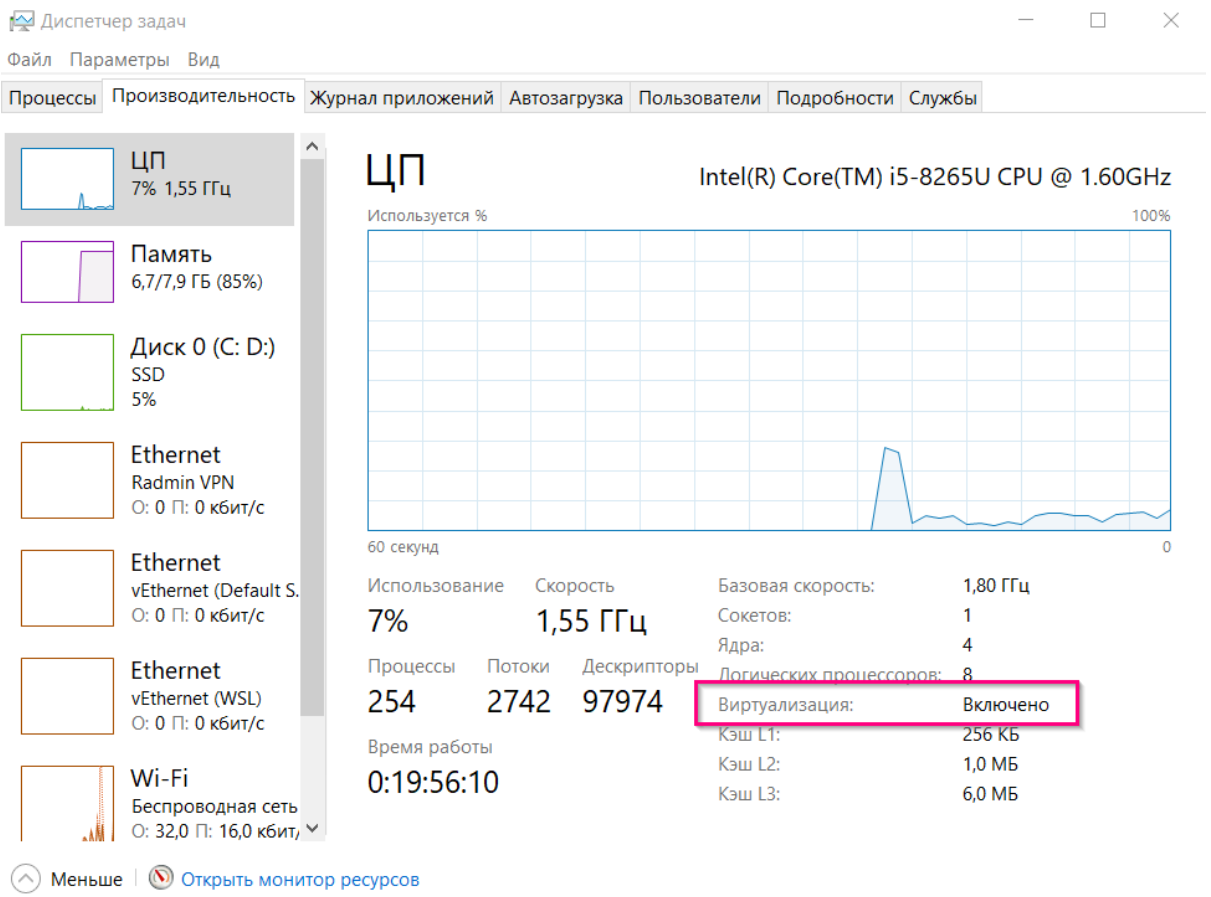
1 Установка Docker Desktop

Сначала немного подготовим систему, затем установим.

1.1 Подготовка ОС Windows 10/11

1) Сначала проверим включена ли **виртуализация в BIOS** компьютера.

Для этого запустите Диспетчер задач, перейдите на вкладку “Производительность” и справа внизу увидите “Виртуализация” = Включено или Выключено.

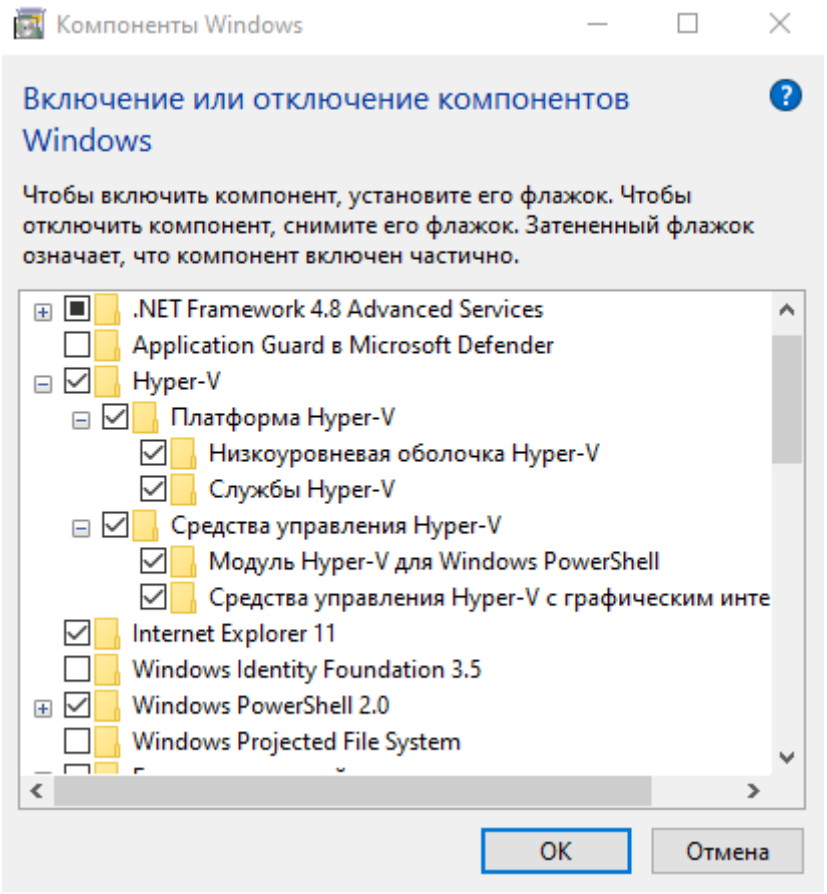


Виртуализация = Включено – всё хорошо.

Виртуализация = Выключено – надо включить. Заходим в BIOS, обычно при перезагрузке надо нажать F2, далее нужно найти пункт который говорит нам о виртуализации, например, для Intel: Intel Virtualization Technology. Ставим – Enabled.

2) Второй этап включаем в Windows **Hyper-V**.

Hyper-V – это технология виртуализации доступная в Windows 10. Включается Hyper-V через настройку компонентов Windows. (Замечание: после включения службы Hyper-V пропадет возможность запускать и создавать x64 виртуальные машины на VirtualBox.)



Выставляем галочки во всех полях и нажимаем OK.

1.2 Устанавливаем Docker Desktop

Нужно скачать с официального сайта дистрибутив:

https://www.docker.com/products/docker-desktop/

Далее проходим установку. Смотрите скриншоты ниже. Следим, чтобы галочка “Use WSL 2 instead of Hyper-V” была установлена, вторая галочка – решайте сами, нужен ли вам ярлык на рабочем столе или нет. Нажимаем Ок, и ждем окончания установки.

Docker Desktop включает в себя:

– Docker Engine – движок Docker.

– Docker CLI client – клиент.

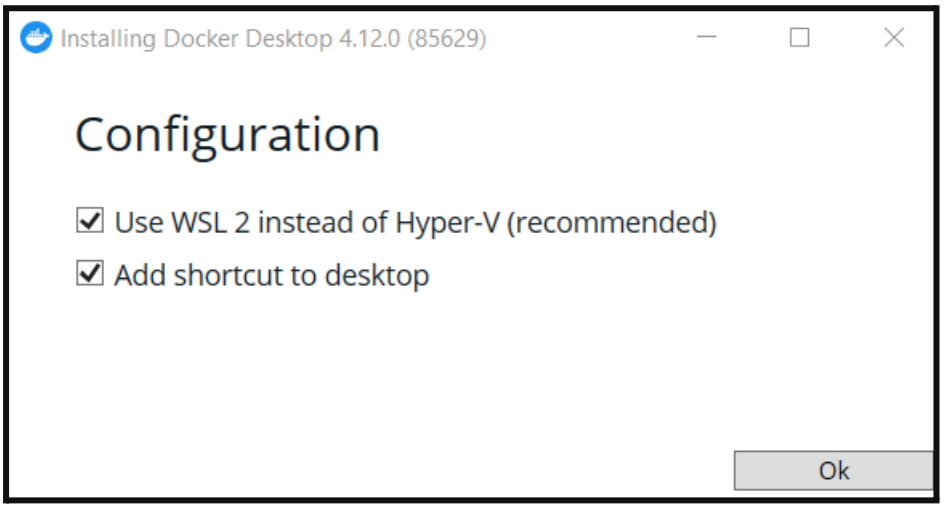
– Docker Build – функция создания контейнеров.

– Docker Compose – это инструмент для определения и запуска многоконтейнерных приложений.

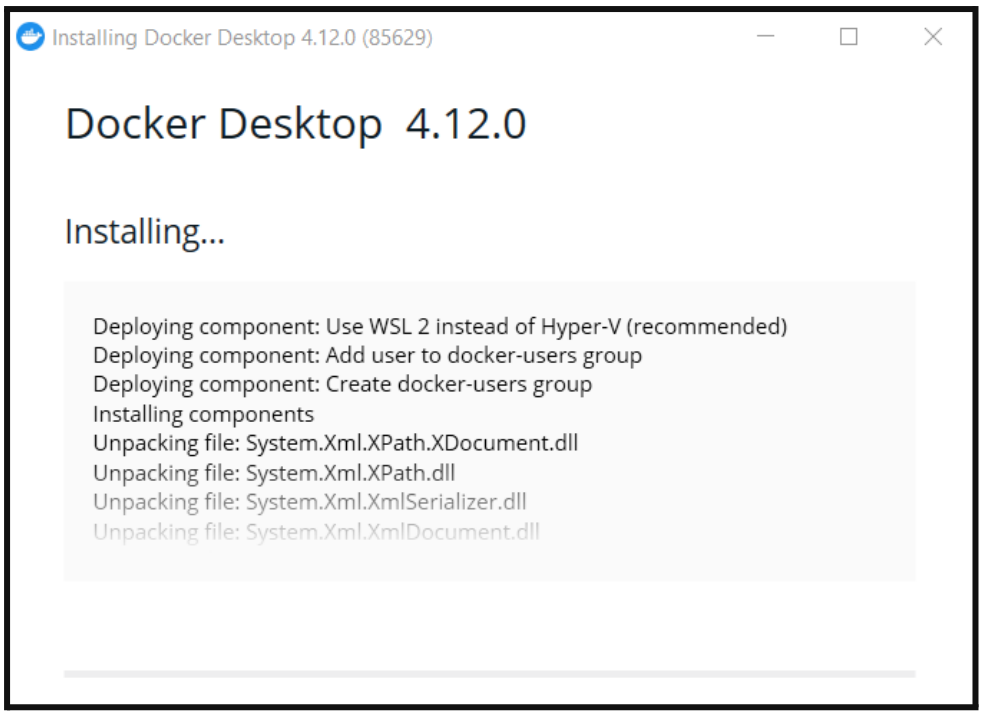
– Docker Content Trust – предоставляет возможность использовать цифровые подписи для данных.

– Kubernetes

– Credential Helper



Процесс установки выглядит так:



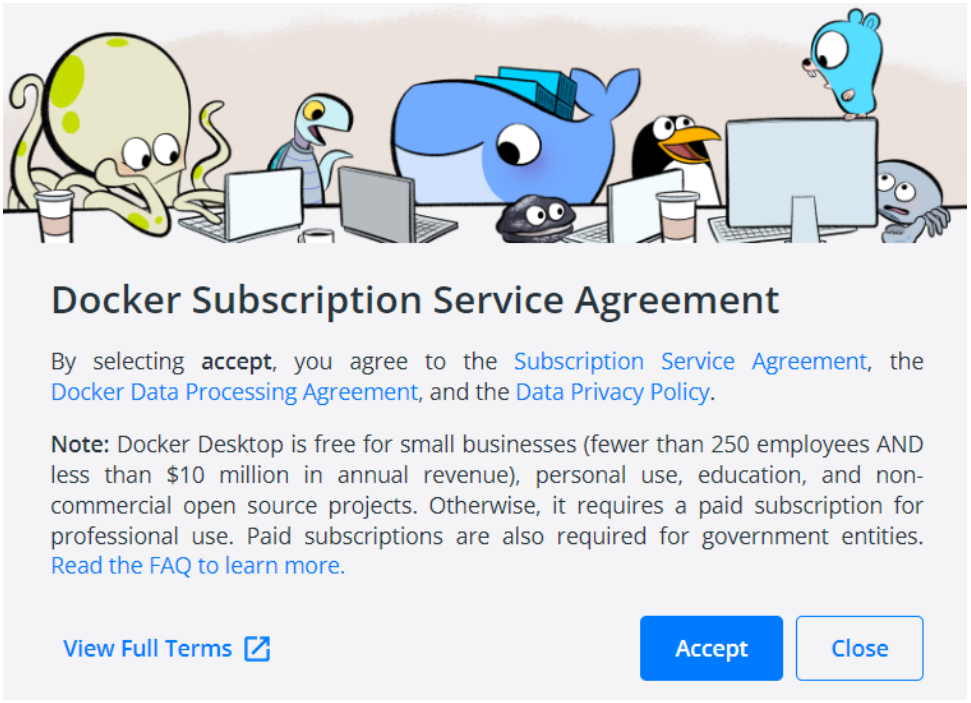
После окончания установки нажимаем Close. Возможно потребуется перезагрузка.

Возможно после установки Docker Desktop потребуется установить WSL 2. Переходим по ссылке, которую предложит установщик, скачиваем пакет и устанавливаем.

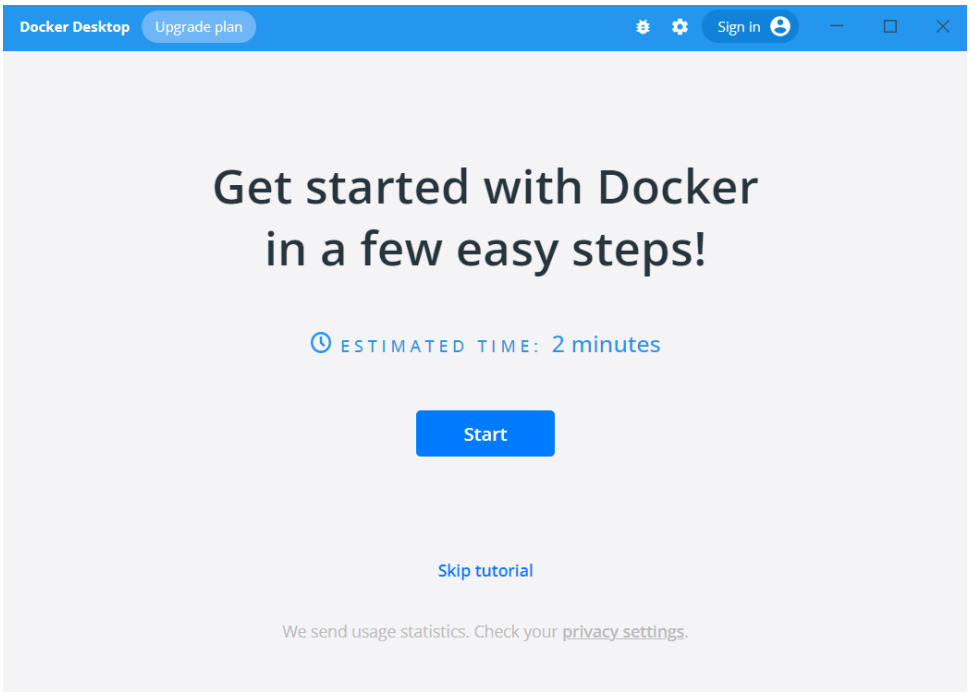
WSL – Windows Subsystem for Linux – это подсистема, обеспечивающая совместимость приложений Linux с Windows.

Можно обойтись и без этого, т.е. использовать только Hyper-V. Тогда при установке можно было бы не устанавливать галочку “Use WSL…”. Но мы тогда были бы ограничены выбором контейнеров. Если у нас ОС Windows, то использовать можно было бы контейнеры только с приложениями Windows, а их значительно меньше.

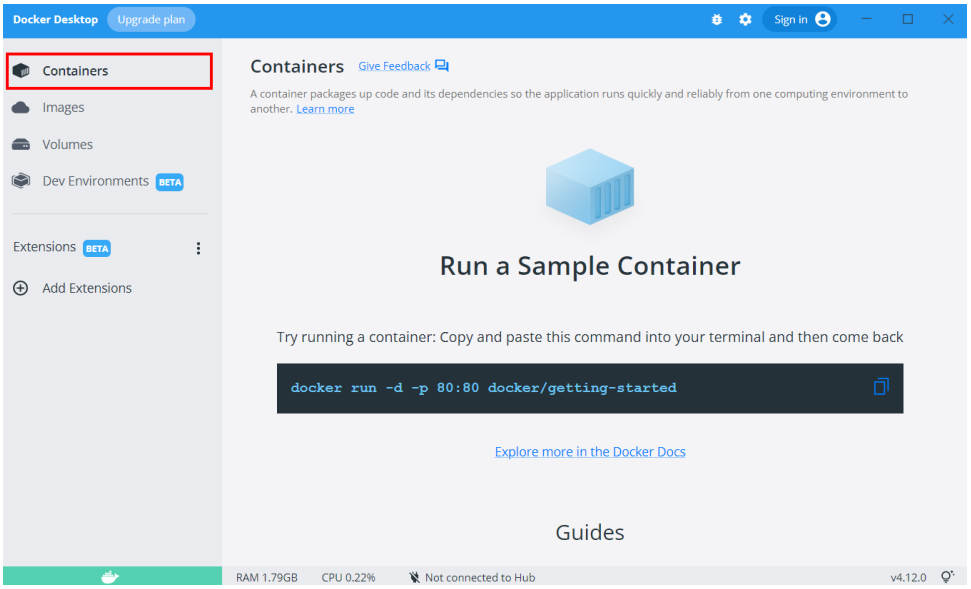
Запускаем Docker Desktop. Принимаем соглашение, в котором написано, что Docker Desktop бесплатен в том числе и для образовательных целей, нажимаем Accept.



Ждём запуска Docker, который, если всё прошло хорошо, выдаст приветственную страницу, на которой предложит начать знакомство. Можно посмотреть небольшую инструкцию, нажав Start. Можно пропустить это, нажав Skip tutorial.



После откроется вкладка по умолчанию – Containers. Так как у нас еще нет ни одного контейнера, Docker предложит попробовать контейнер для примера. Не будем его пробовать, начнём скачивать нужные нам образы.



Официальная документация находится тут: https://docs.docker.com/desktop/

2 Основные команды

Все команды Docker можно посмотреть в встроенной справке. Мы воспользуемся всего несколькими из них. Будем использовать уже готовые контейнеры.

2.1 Команды проверки состояния

Давайте проверим работает ли наш Docker. Попробуйте выполнить указанные команды.

1 – Запустите командную строку Windows

2 – Введите **docker** и нажмите Enter

Если Docker работает, мы должны получит справку по всем командам:

```
C:\>docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers
...
```

3 – Теперь проверим версию Docker, опцией:

```
C:\>docker -v
Docker version 20.10.17, build 100c701
```

4 – Получим список запущенных контейнеров, команда:

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

Получили почти ничего. Нет ни одного запущенного контейнера. Применив опцию **`--all`** (показать все) тоже получим пустой список.

Задание 1:

1) Напишите команду получения справки по командам **pull**, **run** и **exec**.
2) Что делают команды **pull**, **run** и **exec**?

2.2 Команды запуска контейнеры

Теперь скачаем и установим наш первый контейнер. Скачивание и установку контейнера можно выполнить одновременно одной командой:

```
docker run [OPTIONS] IMAGE [COMMAND] [ARG...]
```

Или можно сделать это в два этапа `pull+run`. Сначала скачать из реестра:

```
docker pull [OPTIONS] NAME[:TAG|@DIGEST]
```

Затем стартуем контейнер:

```
docker run [OPTIONS] IMAGE [COMMAND] [ARG...]
```

Пример 1: run.

Попробуем получить и запустить контейнер с MySQL, при этом применим следующие опции:

–p 3306:3306 – сразу пропишем порты для удалённого доступа к серверу MySQL.

--name *my-mysql* – присваиваем контейнеру удобное нам имя.

–e MYSQL_ROOT_PASSWORD=pass – указываем одну из переменных окружения, а именно пароль для пользователя root, т.е. для администратора.

–d – пусть контейнер запустится в фоновом режиме.

mysql – это имя образа (IMAGE), который docker должен развернуть в контейнере.

Получим такую команду (она должны быть написана в одну строку):

```
docker run -p 3306:3306 --name my-mysql -e
MYSQL_ROOT_PASSWORD=pass -d mysql
```

Нажимаем **Enter**.

Кажется, что программа зависла, но это не так, просто Docker ищет этот IMAGE на локальном компьютере и если не найдёт, скачает его из репозитория. Развернет и запустит. Ждем окончания этого процесса.

Получим:

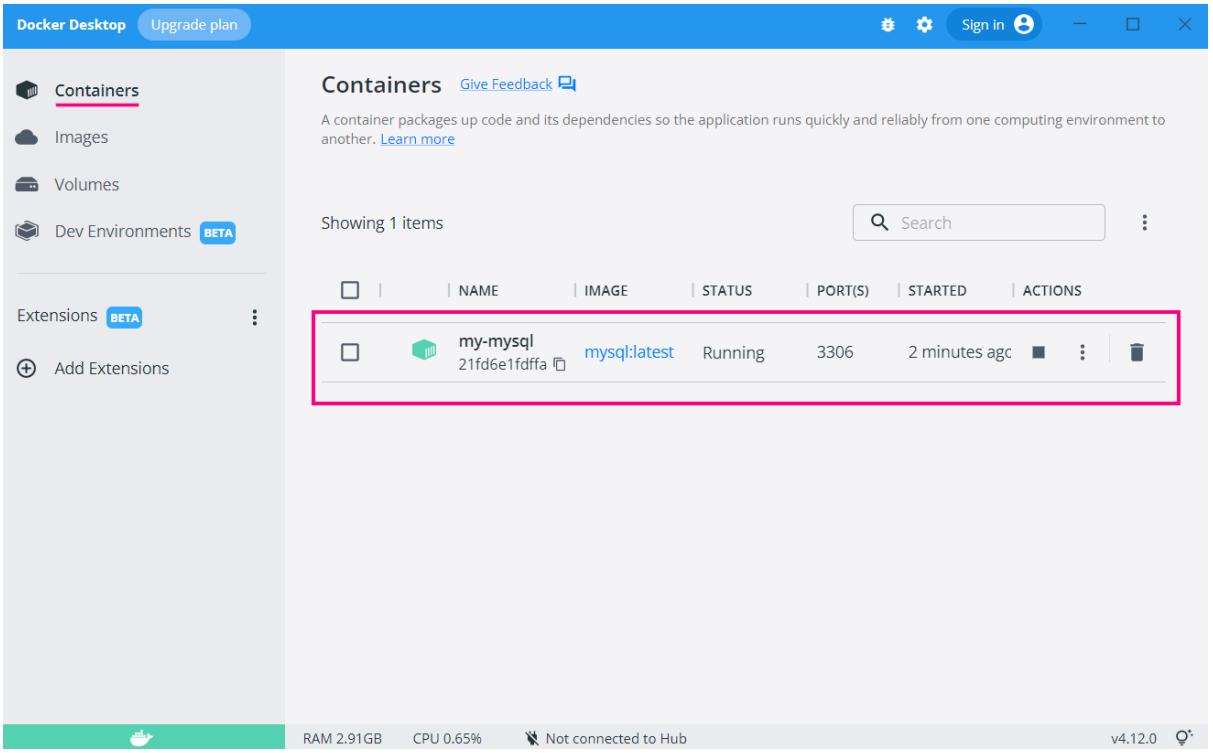
```
C:\>docker run -p 3306:3306 --name my-mysql -e
MYSQL_ROOT_PASSWORD=pass -d mysql
Unable to find image 'mysql:latest' locally
latest: Pulling from library/mysql
051f419db9dd: Pull complete
7627573fa82a: Pull complete
a44b358d7796: Pull complete
95753aff4b95: Pull complete
a1fa3bee53f4: Pull complete
f5227e0d612c: Pull complete
b4b4368b1983: Pull complete
f26212810c32: Pull complete
d803d4215f95: Pull complete
d5358a7f7d07: Pull complete
435e8908cd69: Pull complete
Digest:
sha256:b9532b1edea72b6cee12d9f5a78547bd3812ea5db842566e17f8b33291e
d2921
```

```
Status: Downloaded newer image for mysql:latest
21fd6e1fdffaf7883915e3bef4fc6e1e7f29d3ab299ea740e3dabe452531507f

C:\>
```

Docker должен был скачать образ, развернуть в контейнер и запустить это контейнер. Проверим теперь, какие контейнеры у нас запущены:

```
C:\>docker ps
CONTAINER ID   IMAGE   COMMAND   CREATED         STATUS   PORTS   NAMES
21fd6e1fdffa   mysql "dock..." 5 minutes ago Up 5 minutes
0.0.0.0:3306->3306/tcp, 33060/tcp   my-mysql

C:\>
```

Посмотрите окно приложения Docker Desktop, в нём тоже появился контейнер my-mysql и он запущен.

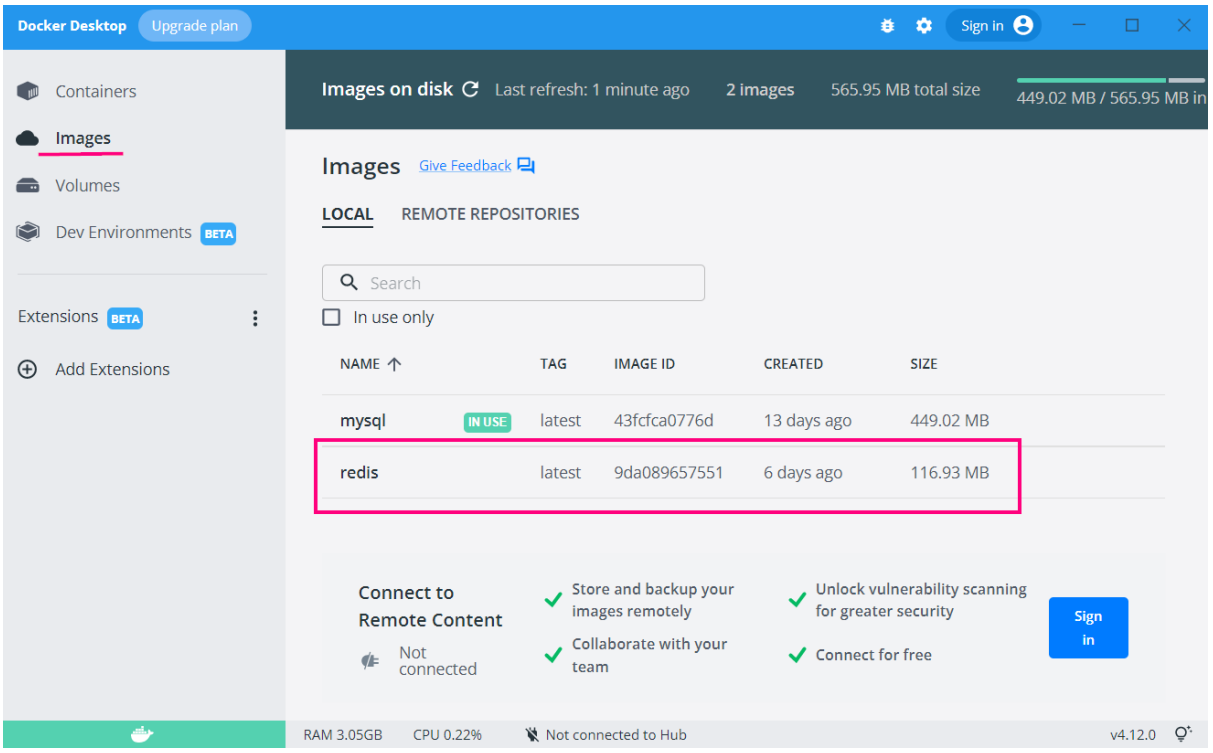


Пример 2: pull+run.

Параметры простые, смотрите код ниже. Имя IMAGE – `redis`. Обратите внимание на `latest` – это мы указываем, что нам нужна последняя версия.

```
C:\>docker pull redis:latest
latest: Pulling from library/redis
31b3f1ad4ce1: Pull complete
ff29a33e56fb: Pull complete
b230e0fd0bf5: Pull complete
9469c4ab3de7: Pull complete
6bd1cefcc7a5: Pull complete
610e362ffa50: Pull complete
Digest:
sha256:b4e56cd71c74e379198b66b3db4dc415f42e8151a18da68d1b61f55fcc7
af3e0
Status: Downloaded newer image for redis:latest
docker.io/library/redis:latest

C:\>
```

Если образ скачан успешно, то мы сможем увидеть его в списке Images в Docker Desktop, но в Containers его еще нет:



Затем стартанем контейнер, указываем:

– имя будущего контейнера – my-redis,

– имя образа – redis.

Docker выведет на экран ID контейнера:

```
C:\>docker run --name my-redis -d redis
d714d44e4f66e32c5935712975ee9ed68a498d7ceab5ea11c7f96fa90519daf8

C:\>
```

После этого контейнер появится в списке Containers в Docker Desktop.

Задание 2

1) Напишите команду, которая:
   Скачает и установит контейнер MongoDB версии 5.0 (если не будет запускаться, вместо 5.0, можно написать latest, чтобы установить последнюю версию)
   Назовет контейнер my-mongo.

2.3 Выполнение команды внутри контейнера

Мы научились скачивать и устанавливать контейнеры, теперь попробуем взаимодействовать с внутренним содержимым контейнера.

Чтобы выполнить команду внутри запущенного контейнера используется команда `docker exec`:

```
docker exec [OPTIONS] CONTAINER COMMAND [ARG...]
```

Пример 3: вход/выход в клиент REDIS.

Запустим клиент Redis

```
docker exec -it my-redis redis-cli
```

`-it` – опция, включающая интерактивный режим работы,

`my-redis` – имя нашего контейнера,

`redis-cli` – команда для Redis, запустить клиент.

Выполнив команду, мы увидим приглашение Redis, IP адрес и порт:

```
127.0.0.1:6379>
```

Теперь мы взаимодействуем с самим Redis, попробуйте ввести команду:

```
127.0.0.1:6379> ping
```

Выход осуществляется командой:

```
127.0.0.1:6379> exit
```

Пример 4: вход/выход в консоль MySQL.

Открыть консоль MySQL можно следующим образом:

```
docker exec -it my-mysql mysql -uroot -ppass
```

Разбираем строку команды:

`-it` – опция, включающая интерактивный режим работы,

`my-mysql` – имя нашего контейнера,

`mysql -uroot -ppass` – команда самого MySQL, запускает консоль MySQL, с опциями –u и –p, логин и пароль соответственно.

Выполнив команду, увидим приветствие MySQL:

```
C:\>docker exec -it my-mysql mysql -uroot -ppass
mysql: [Warning] Using a password on the command line interface
can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.27 MySQL Community Server - GPL

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current
input statement.

mysql>
```

Выполните команду `help`, ознакомьтесь со справкой.

Чтобы выйти из консоли MySQL выполните команду `quit`.

Задание 3

1) Запустите консоль MongoDB.
2) Узнайте версию своего MongoDB.
3) Найдите команду или функцию выхода из консоли.

2.4 Команды остановки и старта контейнеров

Чтобы остановить работу какого-то контейнера выполняется команда:

```
docker stop [OPTIONS] CONTAINER [CONTAINER...]
```

Останавливает один и более контейнеров, которые указываются через пробел. Например:

```
docker stop my-mysql my-redis my-mongo
```

Останавливает все перечисленные контейнеры.

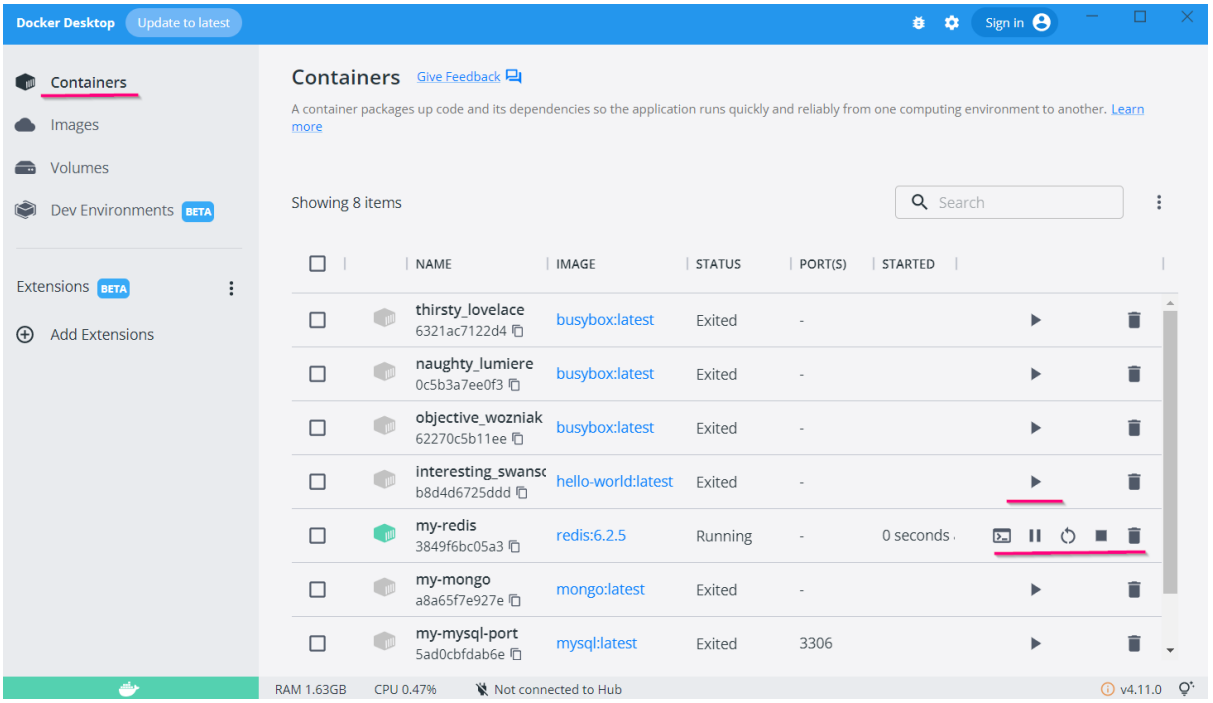
Чтобы снова запустить существующий контейнер или контейнеры выполняется команда:

```
docker start [OPTIONS] CONTAINER [CONTAINER...]
```

Например:

```
docker start my-mysql
```

Управлять контейнерами, конечно, можно через Docker Desktop, напротив каждого контейнера есть элементы управления и вызова консоли:



Если щелкнуть по самому контейнеру, то откроется окно лога контейнера, которое также имеет элементы управления контейнером и вызова консоли.