

Департамент образования и науки города Москвы
Государственное автономное образовательное учреждение
высшего образования города Москвы
«Московский городской педагогический университет»
Институт цифрового образования
Департамент информатики, управления и технологий

ОТЧЕТ

по дисциплине «Интеграция и развертывание программного обеспечения с
помощью контейнеров»

Лабораторная работа 1.1.: Установка и настройка Docker. Работа с
контейнерами в Docker

Направление подготовки – 38.03.05 «Бизнес-информатика».
профиль подготовки – «Аналитика данных и эффективное управление»

Выполнила:

студентка группы АДЭУ-211
st_92

Руководитель:

BosenkoTM
Кандидат технических наук, доцент

Цель работы: освоить процесс установки и настройки Docker, научиться работать с контейнерами и образами Docker.

Задачи:

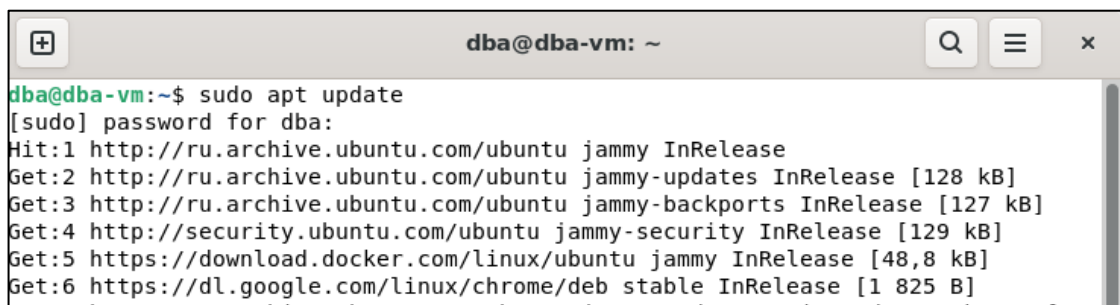
1. Установить Docker на локальный компьютер.
2. Проверить корректность установки Docker.
3. Ознакомиться с основными командами Docker CLI для работы с образами и контейнерами.
4. Выполнить индивидуальное задание.

Ход выполнения работы:

1. Установка Docker.

Выполним следующие команды для установки Docker:

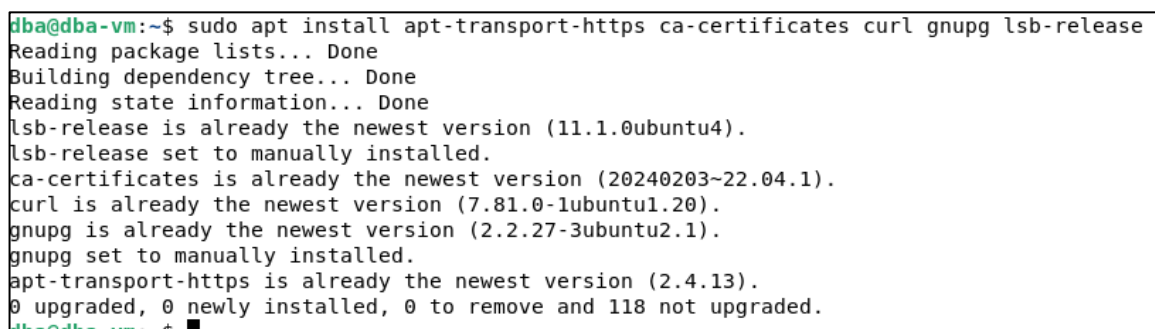
`sudo apt update`



```
dba@dba-vm:~$ sudo apt update
[sudo] password for dba:
Hit:1 http://ru.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://ru.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Get:3 http://ru.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Get:5 https://download.docker.com/linux/ubuntu jammy InRelease [48,8 kB]
Get:6 https://dl.google.com/linux/chrome/deb stable InRelease [1 825 B]
```

Рисунок 1 – Выполнение команды для обновления списка доступных пакетов

`sudo apt install apt-transport-https ca-certificates curl gnupg lsb-release`



```
dba@dba-vm:~$ sudo apt install apt-transport-https ca-certificates curl gnupg lsb-release
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
lsb-release is already the newest version (11.1.0ubuntu4).
lsb-release set to manually installed.
ca-certificates is already the newest version (20240203-22.04.1).
curl is already the newest version (7.81.0-1ubuntu1.20).
gnupg is already the newest version (2.2.27-3ubuntu2.1).
gnupg set to manually installed.
apt-transport-https is already the newest version (2.4.13).
0 upgraded, 0 newly installed, 0 to remove and 118 not upgraded.
dba@dba-vm:~$
```

Рисунок 2 – Выполнение команды для установки пакетов

`curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg`

```
dba@dba-vm:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
dba@dba-vm:~$
```

Рисунок 3 – Выполнение команды взаимодействия с сервером по протоколу

```
echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
dba@dba-vm:~$ echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
dba@dba-vm:~$
```

Рисунок 4 – Выполнение команды для установки Docker

```
sudo apt update
```

```
dba@dba-vm:~$ sudo apt update
Hit:1 https://download.docker.com/linux/ubuntu jammy InRelease
Hit:2 https://dl.google.com/linux/chrome/deb stable InRelease
Hit:3 http://ru.archive.ubuntu.com/ubuntu jammy InRelease
Hit:4 http://ru.archive.ubuntu.com/ubuntu jammy-updates InRelease
Get:5 http://cz.archive.ubuntu.com/ubuntu bionic InRelease [242 kB]
Hit:6 http://ru.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:7 http://security.ubuntu.com/ubuntu jammy-security InRelease
Fetched 242 kB in 2s (137 kB/s)
```

Рисунок 5 – Выполнение команды для обновления пакетов

```
sudo apt install docker-ce docker-ce-cli containerd.io
```

```
dba@dba-vm:~$ sudo apt install docker-ce docker-ce-cli containerd.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  cgroupfs-mount | cgroup-lite
The following packages will be upgraded:
  containerd.io docker-ce docker-ce-cli
3 upgraded, 0 newly installed, 0 to remove and 115 not upgraded.
Need to get 64,5 MB of archives.
After this operation, 21,4 MB disk space will be freed.
Get:1 https://download.docker.com/linux/ubuntu jammy/stable amd64 containerd.io amd64 1.7.25-1 [29,6 MB]
24% [1 containerd.io 19,7 MB/29,6 MB 66%]
```

Рисунок 6 – Выполнение команды для установки Docker

После завершения установки добавим своего пользователя в группу docker, чтобы избежать необходимости использования sudo при выполнении команд Docker:

```
sudo usermod -aG docker $USER
```

```
newgrp docker
```

```
dba@dba-vm:~$ sudo usermod -aG docker $USER
dba@dba-vm:~$ newgrp docker
```

Рисунок 7 – Выполнение команды добавления пользователя в группу docker

2. Проверка установки Docker.

Выполним команду `docker --version`, чтобы убедиться, что Docker установлен правильно. Вывод должен содержать информацию о версии Docker.

```
dba@dba-vm:~$ docker --version
Docker version 28.0.0, build f9ced58
dba@dba-vm:~$
```

Рисунок 8 – Выполнение команды для просмотра версии docker

Выполним команду `docker run hello-world`, чтобы проверить, что Docker запущен и может загружать образы и запускать контейнеры.

```
dba@dba-vm:~$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

Рисунок 9 – Выполнение команды для проверки работы docker

3. Знакомство с основными командами Docker CLI.

Выполним команду `docker images`, чтобы просмотреть список локальных образов Docker.

```
dba@dba-vm:~$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
nginx-static	latest	15f38d0b8bf2	6 days ago	212MB
hello-world	latest	74cc54e27dc4	4 weeks ago	10.1kB
myapp	latest	3e1b602bae81	3 months ago	649MB
postgres	latest	d57ed788c154	4 months ago	434MB

Рисунок 10 – Выполнение команды для просмотра списка локальных образов docker

Выполним команду `docker ps`, чтобы просмотреть список запущенных контейнеров.

```
dba@dba-vm:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
dba@dba-vm:~\$						

Рисунок 11 – Выполнение команды для просмотра списка запущенных контейнеров

Выполним команду `docker ps -a`, чтобы просмотреть список всех контейнеров, включая остановленные.

```
dba@dba-vm:~$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
e895a8aa7888	hello-world	"/hello"	About a minute ago	Exited (0)	About a minute ago	gracious_noether
7f075cfee8b9	hello-world	"/hello"	2 minutes ago	Exited (0)	2 minutes ago	unruffled_mendeleev
a457dbff5f72	myapp	"python -m uvicorn s..."	3 months ago	Exited (255)	4 weeks ago	myapp
fc7c96c36fe7	postgres	"docker-entrypoint.s..."	3 months ago	Exited (255)	4 weeks ago	mydb

Рисунок 12 – Выполнение команды для просмотра всех контейнеров docker

4. Выполнение индивидуального задания

Вариант 5. Загрузить образ `redis`, запустить контейнер, подключиться к Redis CLI внутри контейнера и выполнить несколько команд.

Загрузим образ `redis`, выполнив команду `docker pull redis`

```
dba@dba-vm:~$ docker pull redis
Using default tag: latest
latest: Pulling from library/redis
c29f5b76f736: Already exists
5de2dd3ff2ef: Pull complete
6c334acf232e: Pull complete
3090e1a50a6c: Pull complete
f5bc47c37726: Pull complete
20eea55b3ebb: Pull complete
4f4fb700ef54: Pull complete
d128ccd842a6: Pull complete
Digest: sha256:93a8d83b707d0d6a1b9186edecca2e37f83722ae0e398aee4eea0ff17c2fad0e
Status: Downloaded newer image for redis:latest
docker.io/library/redis:latest
```

Рисунок 13 – Выполнение команды для загрузки образа `redis`

Redis по умолчанию использует порт `6379`.

Запустим контейнер `redis` с маршрутизацией портов `6379`:

`docker run --name my-redis -p 6379:6379 -d redis`

```
dba@dba-vm:~$ docker run --name my-redis -p 6379:6379 -d redis
00d095b29d64d64955e6a76b73d1feab15bc95838c162d31c1caaf0be7bae1d1
```

```
dba@dba-vm:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
00d095b29d64	redis	"docker-entrypoint.s..."	3 seconds ago	Up 2 seconds	0.0.0.0:6379->6379/tcp, [::]:6379->6379/tcp	my-redis

Рисунок 14 – Выполним команду для запуска контейнера `redis`

Подключимся к Redis CLI внутри контейнера

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

```
dba@dba-vm:~$ docker exec -it my-redis redis-cli
127.0.0.1:6379> ping
PONG
127.0.0.1:6379> █
```

Рисунок 15 – Выполнение команды для подключения к Redis CLI

Выполним команды:

set mytest "hello, world" – устанавливаем пару ключ-значение

get mytest – получаем значение ключа

rpush mylist "First sentence" – добавляем элемент в список

rpush mylist "Second sentence"

rpush mylist "Third sentence"

lrange mylist 0 2 – выводим элементы списка

```
127.0.0.1:6379> set mytest "hello, world"
OK
127.0.0.1:6379> get mytest
"hello, world"
127.0.0.1:6379> append mytest "!"
(integer) 13
127.0.0.1:6379> get mytest
"hello, world!"
127.0.0.1:6379> rpush mylist "First sentence"
(integer) 1
127.0.0.1:6379> rpush mylist "Second sentence"
(integer) 2
127.0.0.1:6379> rpush mylist "Third sentence"
(integer) 3
127.0.0.1:6379> lrange mylist 0 2
1) "First sentence"
2) "Second sentence"
3) "Third sentence"
127.0.0.1:6379>
```

Рисунок 16 – Выполнение команд в Redis CLI

Остановим контейнер с помощью команды `docker stop my-redis`

```
dba@dba-vm:~$ docker stop my-redis
my-redis
dba@dba-vm:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
dba@dba-vm:~$
```

Рисунок 17 – Выполнение команды остановки контейнера

Выводы:

В ходе выполнения работы была установлена платформа Docker, проверена корректность ее установки. Также мы ознакомились с

основными командами Docker CLI для работы с образами и контейнерами. При выполнении индивидуального задания мы загрузили образ redis, запустили контейнер redis, подключились к Redis CLI внутри контейнера и выполнили несколько команд. Цель работы была достигнута.

Контрольные вопросы:

Что такое Docker и для чего он используется?

Docker — программное обеспечение для автоматизации развертывания и управления приложениями в средах с поддержкой контейнеризации, контейнеризатор приложений.

Какие преимущества дает использование контейнеров Docker по сравнению с виртуальными машинами?

Виртуальная машина виртуализирует аппаратные ресурсы, такие как процессор, память, а контейнеры docker виртуализируют только ОС.

Преимущества docker контейнеров:

- содержат только запущенное в контейнере приложение;
- обеспечивают быстрый старт приложения;
- имеют маленький размер.

Что такое образ Docker и как он связан с контейнерами?

Docker образы – это шаблоны готового к запуску приложения со всем нужным для работы окружением.

Контейнер - это запущенный образ, находящийся в процессе выполнения

Какие основные команды Docker CLI вы узнали в ходе выполнения лабораторной работы?

Мы узнали такие команды как:

- `docker images` выполняется, чтобы просмотреть список локальных образов Docker
- `docker ps` выполняется, чтобы просмотреть список запущенных контейнеров
- `docker ps -a` выполняется, чтобы просмотреть список всех контейнеров, включая остановленные

Как можно настроить маршрутизацию портов при запуске контейнера Docker?

Настройка маршрутизации портов при запуске контейнера Docker реализуется с помощью флага `-p` команды `docker run`.