



Docker

Шипицын Кирилл

08.12.2022

Что такое Docker?

Основной принцип работы Docker

Docker — это платформа для:

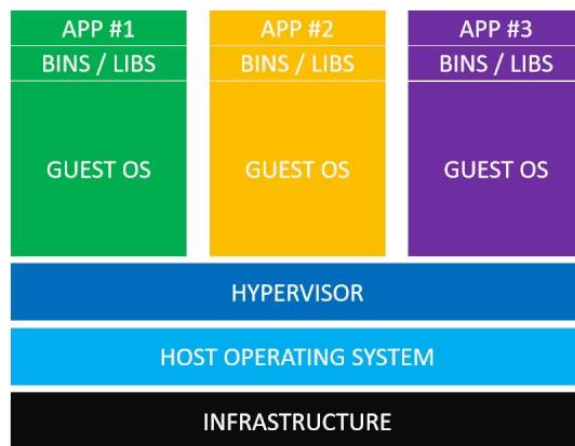
- разработки
- тестирования
- развертывания приложений.

Docker упаковывает ПО в стандартизированные блоки (называются контейнерами). Контейнеры включают все необходимое для работы приложения.

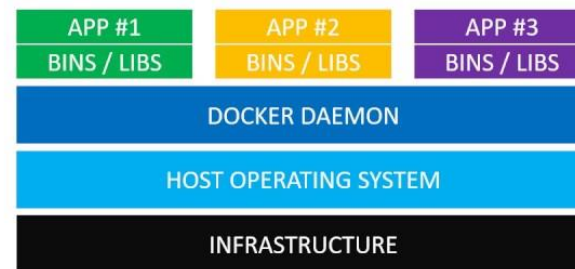
Что такое контейнер?

Контейнер — изолированная среда
(почти как виртуальная машина).

В чем отличие от VM?



Virtual Machines



Docker Containers

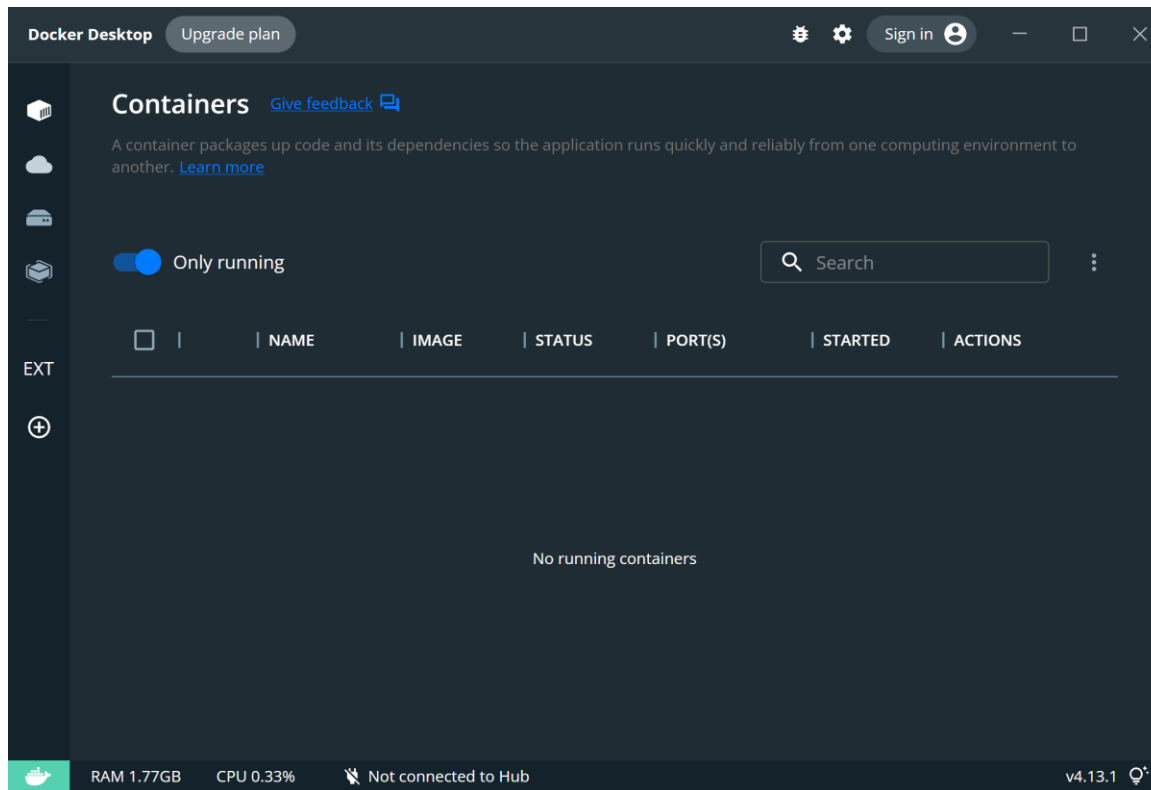
Зачем использовать контейнеры?

Среда выполнения:

- предсказуемая
- изолированная

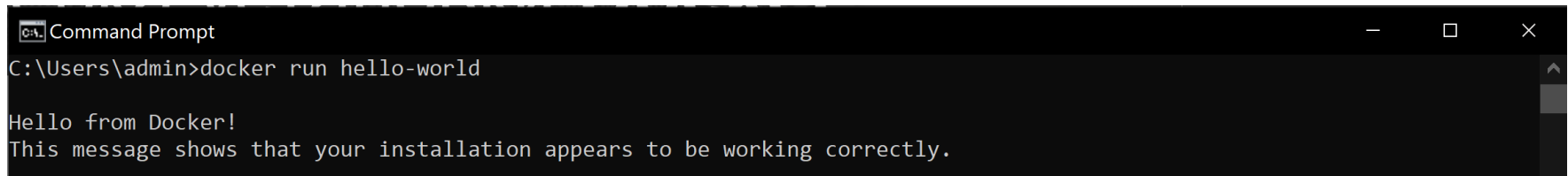
Установка Docker

Docker Desktop: <https://www.docker.com>



Проверка установки Docker

1. Запустите **Docker Desktop**
2. Запустите **Терминал**
3. `$ docker run hello-world`



```
Command Prompt
C:\Users\admin>docker run hello-world

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

\$ docker pull

Image (образ) — исполняемый пакет

\$ docker pull

Docker Hub — регистр Docker-образов

The screenshot displays the Docker Hub interface. At the top, there's a blue header with the Docker Hub logo, a search bar, and links for 'Explore', 'Pricing', 'Sign In', and a 'Register' button. Below the header, the main content area shows search results. On the left, there are filters for 'Products' (Images, Extensions, Plugins) and 'Trusted Content' (Docker Official Image, Verified Publisher, Sponsored OSS). The 'Operating Systems' filter shows 'Linux' and 'Windows'. The search results list '1 - 25 of 10,000 available results.' and a 'Suggested' dropdown. The first result is 'alpine', a Docker Official Image, updated 15 days ago, with 1B+ downloads and 9.5K stars. It is described as 'A minimal Docker image based on Alpine Linux with a complete package index and only 5 MB in size!'. The second result is 'busybox', also a Docker Official Image, updated 6 hours ago, with 1B+ downloads and 2.8K stars, described as 'Busybox base image.'.

dockerhub Search Docker Hub Explore Pricing Sign In Register

Filters

Products

- ☐ Images
- ☐ Extensions
- ☐ Plugins

Trusted Content

- ☐ Docker Official Image
- ☐ Verified Publisher
- ☐ Sponsored OSS

Operating Systems

- ☐ Linux
- ☐ Windows

1 - 25 of 10,000 available results. Suggested

alpine DOCKER OFFICIAL IMAGE 1B+ 9.5K Downloads Stars

Updated 15 days ago

A minimal Docker image based on Alpine Linux with a complete package index and only 5 MB in size!

Linux 386 PowerPC 64 LE IBM Z riscv64 x86-64 ARM ARM 64

busybox DOCKER OFFICIAL IMAGE 1B+ 2.8K Downloads Stars

Updated 6 hours ago

Busybox base image.

Linux riscv64 IBM Z x86-64 ARM ARM 64 386 mips64le PowerPC 64 LE

\$ docker pull

\$ docker pull busybox

```
Command Prompt
C:\Users\admin>docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
Digest: sha256:fcd85228d7a25feb59f101ac3a955d27c80df4ad824d65f5757a954831450185
Status: Image is up to date for busybox:latest
docker.io/library/busybox:latest
```

\$ docker images

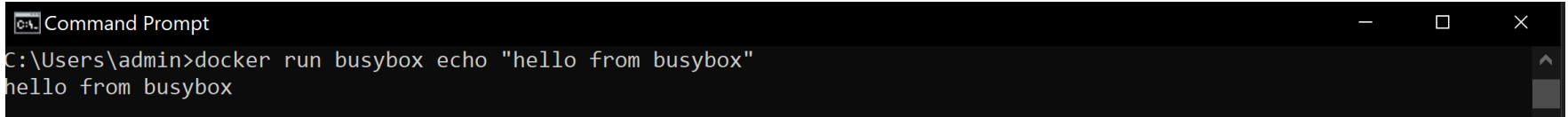
Command Prompt

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

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
busybox	latest	9d5226e6ce3f	5 days ago	1.24MB
hello-world	latest	feb5d9fea6a5	14 months ago	13.3kB

\$ docker run

\$ docker run busybox echo "hello from busybox"



```
Command Prompt
C:\Users\admin>docker run busybox echo "hello from busybox"
hello from busybox
```

\$ docker ps

Command Prompt

C:\Users\admin>docker ps

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
--------------	-------	---------	---------	--------	-------	-------

\$ docker ps -a

```
Command Prompt
C:\Users\admin>docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
3d3101e2c7b3	busybox	"echo 'hello from bu..."	59 seconds ago	Exited (0) 58 seconds ago		relaxed_dijkstra
3fd89551c396	hello-world	"/hello"	About a minute ago	Exited (0) About a minute ago		crazy_haslett

\$ docker run -it

\$ docker run -it busybox sh

```
Command Prompt - docker run -it busybox sh
C:\Users\admin>docker run -it busybox sh
/ #

Command Prompt
C:\Users\admin>docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
af35039f8c4b	busybox	"sh"	27 seconds ago	Up 27 seconds		dazzling_jennings

\$ docker rm

Command Prompt

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

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
9f00967cccca	busybox	"sh"	37 seconds ago	Exited (0) 31 seconds ago		relaxed_johnson
3d3101e2c7b3	busybox	"echo 'hello from bu..."	2 minutes ago	Exited (0) 2 minutes ago		relaxed_dijkstra
3fd89551c396	hello-world	"/hello"	2 minutes ago	Exited (0) 2 minutes ago		crazy_haslett

```
C:\Users\admin>docker rm 9f00967cccca 3d3101e2c7b3 3fd89551c396
```

```
9f00967cccca
```

```
3d3101e2c7b3
```

```
3fd89551c396
```

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

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
--------------	-------	---------	---------	--------	-------	-------

Термины

- **Docker Hub** — регистр Докер-образов
- **Image** — исполняемый пакет
- **Container** — изолированная среда




WEB

```
$ docker pull yeasy/simple-web
```


WEB

```
$ docker run -it yeasy/simple-web
```



```
Command Prompt - docker run -it yeasy/simple-web
C:\Users\admin>docker run -it yeasy/simple-web
Serving HTTP on 0.0.0.0 port 80 ...
```

WEB

📘 https://127.0.0.1



Unable to connect


An error occurred during a connection to 127.0.0.1.

- The site could be temporarily unavailable or too busy. Try again in a few moments.
- If you are unable to load any pages, check your computer's network connection.
- If your computer or network is protected by a firewall or proxy, make sure that Firefox is permitted to access the web.

Try Again

WEB

```
$ docker run -it -p 8080:80 yeasy/simple-web
```



Command Prompt - docker run -it -p 8080:80 yeasy/simple-web

```
C:\Users\admin>docker run -it -p 8080:80 yeasy/simple-web  
Serving HTTP on 0.0.0.0 port 80 ...
```

WEB

🛡️ 📄 127.0.0.1:8080



Real Visit Results



WEB

```
$ docker run --net=host ...
```


Dockerfile

Dockerfile — описывает инструкции для создания Image (образа)

Пример (requirements.txt)

```
flask==2.2.2
```

Пример (app.py)

```
from flask import Flask
```

```
app = Flask(__name__)
```

```
@app.route("/")
```

```
def file():
```

```
    return __file__
```

```
if __name__ == '__main__':
```

```
    app.run(debug=True, host='0.0.0.0')
```

Пример (Dockerfile)

```
# базовый образ  
FROM python:3.8
```

Пример (Dockerfile)

```
# базовый образ
FROM python:3.8
# смена рабочей директории
WORKDIR /app
```

Пример (Dockerfile)

```
# базовый образ
FROM python:3.8
# смена рабочей директории
WORKDIR /app
# копирование всего из текущей папки в образ
COPY . .
```


Пример (Dockerfile)

```
# базовый образ
FROM python:3.8
# смена рабочей директории
WORKDIR /app
# копирование всего из текущей папки в образ
COPY . .
# установить зависимостей
RUN pip install -r requirements.txt
```

Пример (Dockerfile)

```
# базовый образ
FROM python:3.8
# смена рабочей директории
WORKDIR /app
# копирование всего из текущей папки в образ
COPY . .
# установить зависимостей
RUN pip install -r requirements.txt
# команда запускающая приложение
ENTRYPOINT ["python"]
```

Пример (Dockerfile)

```
# базовый образ
FROM python:3.8
# смена рабочей директории
WORKDIR /app
# копирование всего из текущей папки в образ
COPY . .
# установить зависимостей
RUN pip install -r requirements.txt
# команда запускающая приложение
ENTRYPOINT ["python"]
# добавляет параметры к EntryPoint
CMD ["app.py"]
```

Пример (Structure)

- requirements.txt
- app.py
- Dockerfile

Пример (build)

```
$ docker image build -t flask_docker .
```

Пример (run)

```
$ docker run -p 5000:5000 -it flask_docker
```

```
Command Prompt - docker run -p 5000:5000 -it flask_docker
C:\Users\admin\Desktop\sample>docker run -p 5000:5000 -it flask_docker
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://172.17.0.2:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 141-340-788
```



/app/app.py

Пример #2 (app.py)

```
from flask import Flask
import logging

logging.basicConfig(filename='/app_output/record.log',
                    level=logging.DEBUG)
app = Flask(__name__)

@app.route("/<query_param>")
def query(query_param):
    app.logger.info(f"/{query_param}")
    return f"done!"

if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0')
```

Пример #2 (run)

```
$ docker run -p 5000:5000 -it \  
--mount \  
type=bind, \  
source=D:\flask_docker_output, \  
target=/app_output \  
flask_docker
```

Docker Volume (Том)

Docker Volume — рекомендуемый способ хранения данных. Создаются и управляются средствами Docker.

Docker Volume (Create)

```
$ docker volume create flask_docker_volume
```

Docker Volume (Run)

```
$ docker run -p 5000:5000 -it \  
--mount \  
source=flask_docker_volume, \  
target=/app_output \  
flask_docker
```

\$ docker volume inspect

\$ docker volume inspect flask_docker_volume

```
Command Prompt
C:\Users\admin>docker volume inspect flask_docker_volume
[
  {
    "CreatedAt": "2022-11-26T11:41:33Z",
    "Driver": "local",
    "Labels": {},
    "Mountpoint": "/var/lib/docker/volumes/flask_docker_volume/_data",
    "Name": "flask_docker_volume",
    "Options": {},
    "Scope": "local"
  }
]
```


Docker Compose

Docker Compose — инструмент для определения и запуска много-контейнерных приложений

Docker Compose (docker-compose.yml)

services:

- namespace1:

 - image: image_name

- namespace2:

 - build: path_to_proj

Docker Compose (docker-compose.yml)

```
services:
```

```
  namespace1:
```

```
    image: image_name
```

```
    ports:
```

```
      - 8080:80
```

```
    volumes:
```

```
      - volume_name:inner_path
```

```
  namespace2:
```

```
    ...
```

```
volumes:
```

```
  volume_name:inner_path
```

Docker Compose (docker-compose.yml)

```
services:
```

```
    namespace1:
```

```
        ...
```

```
    namespace2:
```

```
        build: path_to_proj
```

```
        network_mode: host
```

```
        depends_on:
```

```
            - namespace1
```

```
        environment:
```

```
            ENV_NAME:ENV_VALUE
```

Docker Compose (build)

```
$ docker compose build
```

Docker Compose (up)

\$ docker compose up

Q&A