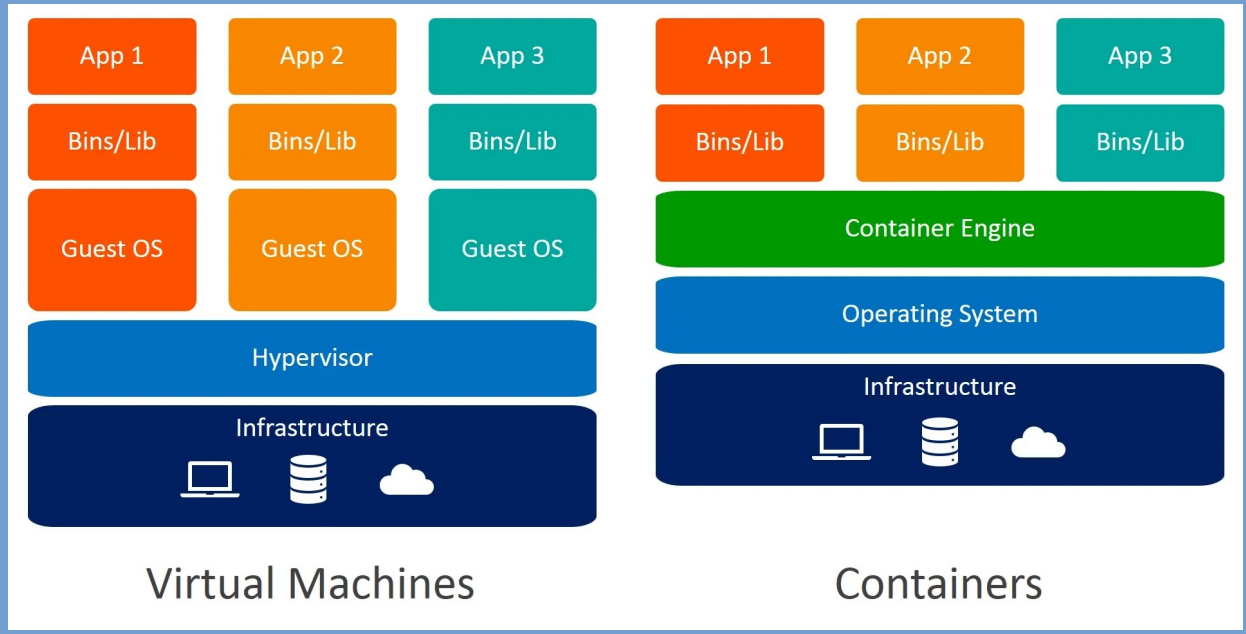
Docker: что это, а главное, зачем

~~Википедия~~ сайт docs.docker.com: “Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code, you can significantly reduce the delay between writing code and running it in production.”

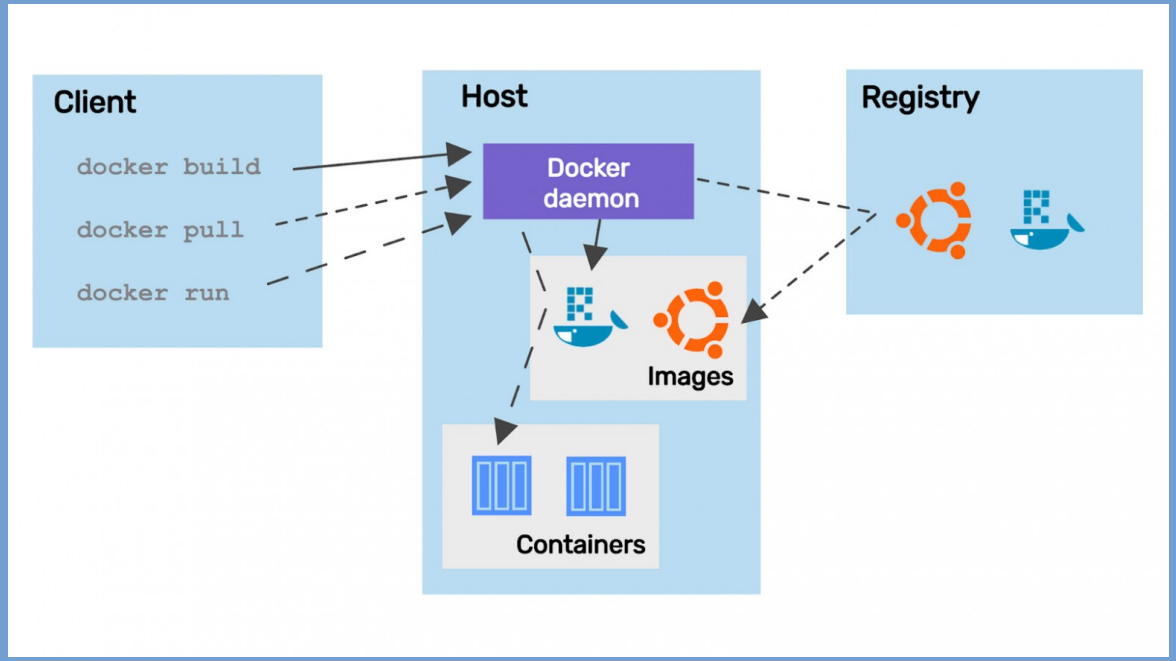
- *Мобильность*
- *Изоляция*
- *Эффективность по ресурсам (в сравнении с VM)*
- *Масштабируемость*

Docker vs Virtual Machine

- Виртуализация на уровне ОС, а не аппаратного обеспечения
- Каждый контейнер работает как процесс основной ОС



Архитектура Docker

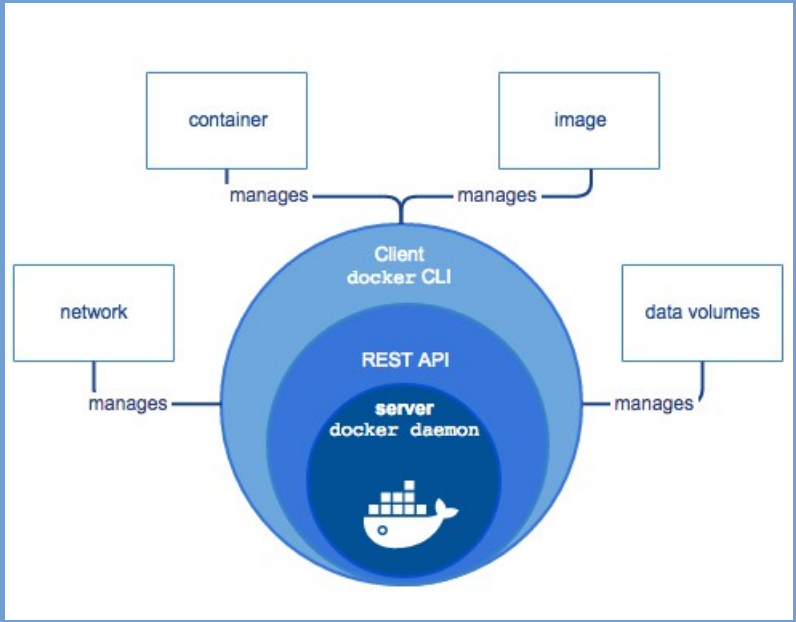


Клиент-сервер + реестр:

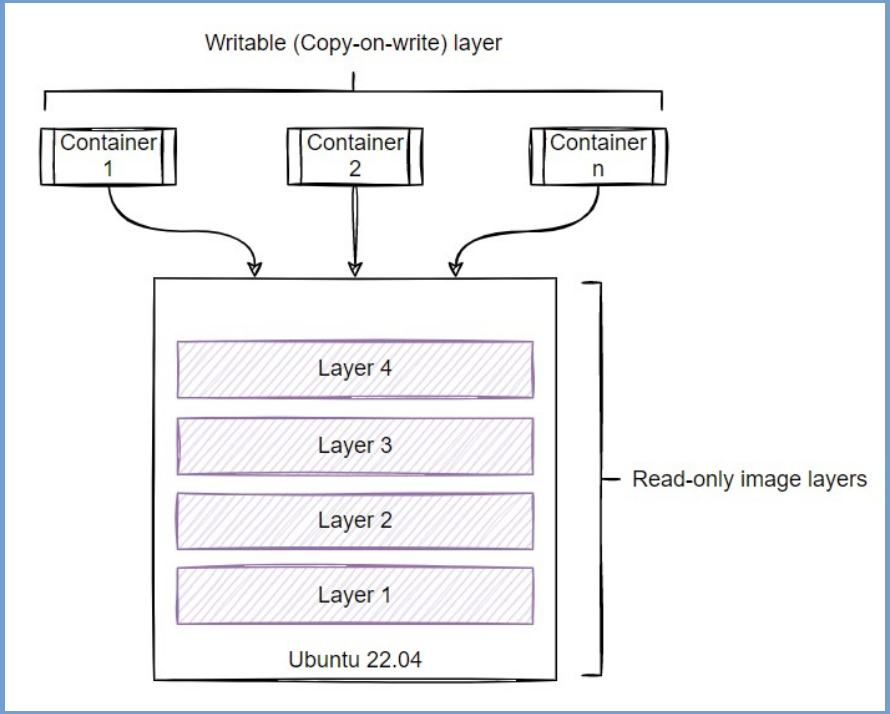
- Клиент
- Сервер (хост)
- Реестр (удаленные хабы с образами)

Объекты в Docker

- Образ — набор файлов, содержащих все необходимое для запуска контейнера
- Контейнер — запущенное приложение на основе образа
- Том (Volume) — способ хранения данных в докер (default /var/lib/docker/volumes)
- Сеть (Network) — виртуальная локальная сеть



Слои (Layers)



Контейнер, как и образ, состоит из слоев, на нижних слоях всегда лежит ОС, далее наслаиваются ваши приложения. При запуске контейнера запускается самый верхний образ, и только он имеет возможность записи.
Благодаря такой архитектуре один и тот же слой может быть многократно использован как в разных контейнерах, так и в разных образах, что существенно экономит память.

Установка Docker

1) Добавить репозиторий в соответствии с вашим дистрибутивом (см. [4])

2) Установить docker-ce (пример: `$ sudo apt install docker-ce`)

3) Для работы с мультиконтейнерными приложениями:

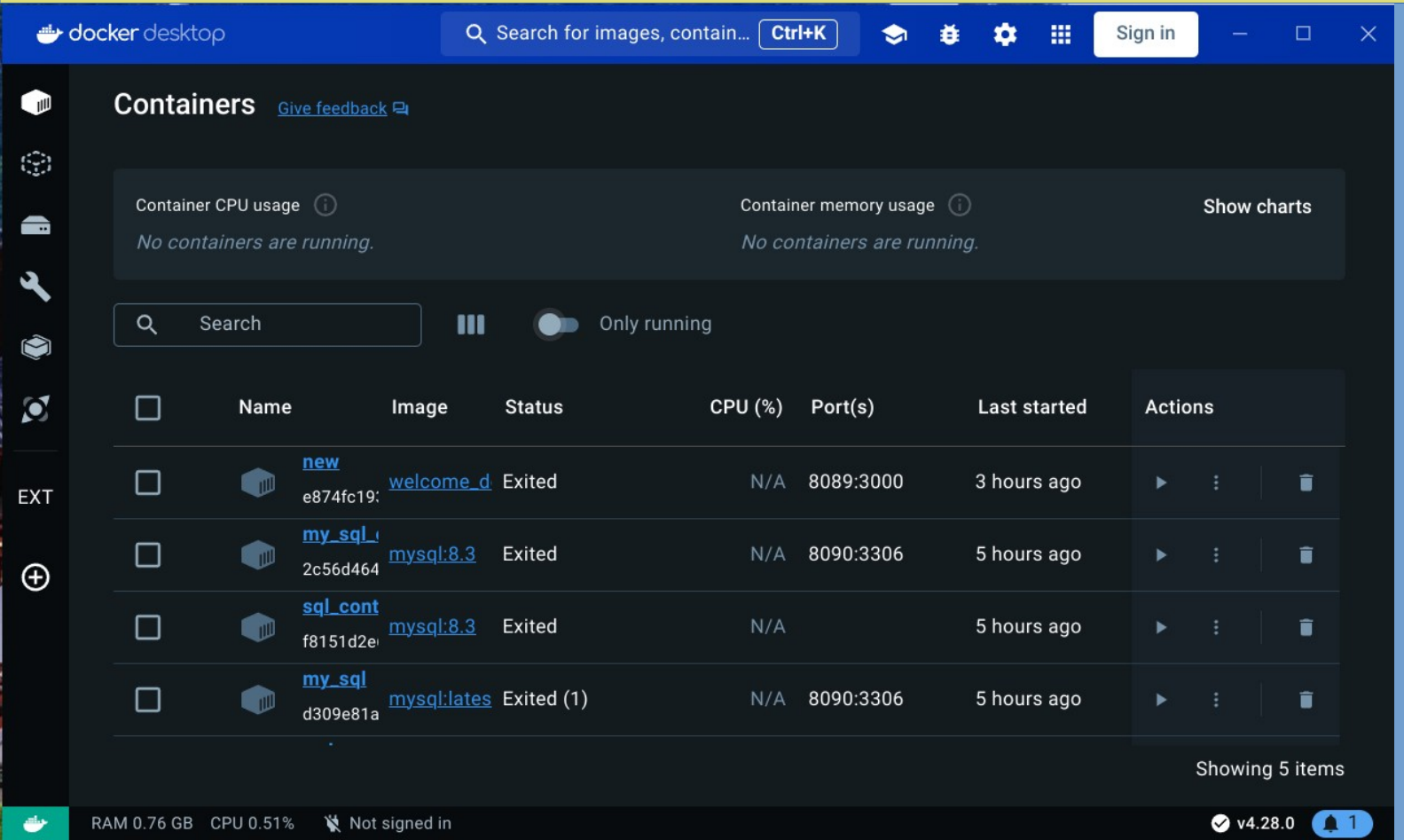
   docker-compose

4) Проверить, что docker-daemon запущен:

   `$ sudo systemctl status docker`

   если нет: `$ sudo systemctl start docker`

Docker desktop



Он есть,

но цель его неясна

Docker Command Line Interface

```
~> docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Common Commands:
  run         Create and run a new container from an image
  exec        Execute a command in a running container
  ps          List containers
  build       Build an image from a Dockerfile
  pull        Download an image from a registry
  push        Upload an image to a registry
  images      List images
  login       Log in to a registry
  logout      Log out from a registry
  search      Search Docker Hub for images
  version     Show the Docker version information
  info        Display system-wide information
```

- $ docker — выведет все команды
- $ docker COMMAND --help

  — вся информация по команде, например:

  $ docker build --help

Первый контейнер

1) Создать Dockerfile, поместить все нужные файлы в ту же директорию
2) Собрать образ `$ sudo docker build .`
3) Найти image id, и запустить с его помощью контейнер

Dockerfile

```
FROM python:latest

COPY . /py_script

WORKDIR /py_script

CMD [ "python3", "hello.py"]
```

```
user@localhost:~/docker_files$ sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED              SIZE
<none>       <none>    7952ef4a455f   About an hour ago    1.02GB
user@localhost:~/docker_files$ sudo docker run 7952ef4a455f
hello world!
```

```
user@localhost:~/docker_files$ sudo docker start 7b7d592128ce
7b7d592128ce
user@localhost:~/docker_files$ sudo docker ps -a
CONTAINER ID   IMAGE          COMMAND              CREATED          STATUS                      PORTS     NAMES
7b7d592128ce   7952ef4a455f   "python3 hello.py"   27 minutes ago   Exited (0) 6 seconds ago              mystifying_nightingale
```

Больше опций Богу опций

Запустим Убунту и свяжем ее консоль с нашей флагом -it (интерактивный режим)

```
~> sudo docker run -it ubuntu:22.04 /bin/bash
Unable to find image 'ubuntu:22.04' locally
22.04: Pulling from library/ubuntu
Digest: sha256:77906da86b60585ce12215807090eb327e
Status: Downloaded newer image for ubuntu:22.04
root@701897df55f0:/#
```

Заметим, что мы автоматически зашли от root'а

Пробросить порт в контейнер:

```
FROM php:7.2-apache
# Указываем рабочую папку
WORKDIR /var/www/html
# Копируем все файлы проекта в контейнер
COPY . /var/www/html
EXPOSE 80
```

```
docker run -p <HOST_PORT>:<CONTAINER_PORT>
```

Частый usecase: контейнер должен быть удален после остановки

```
~> sudo docker run --help | grep -i remove
      --rm                                Automatically remove the container when it exits
```

Удобная команда

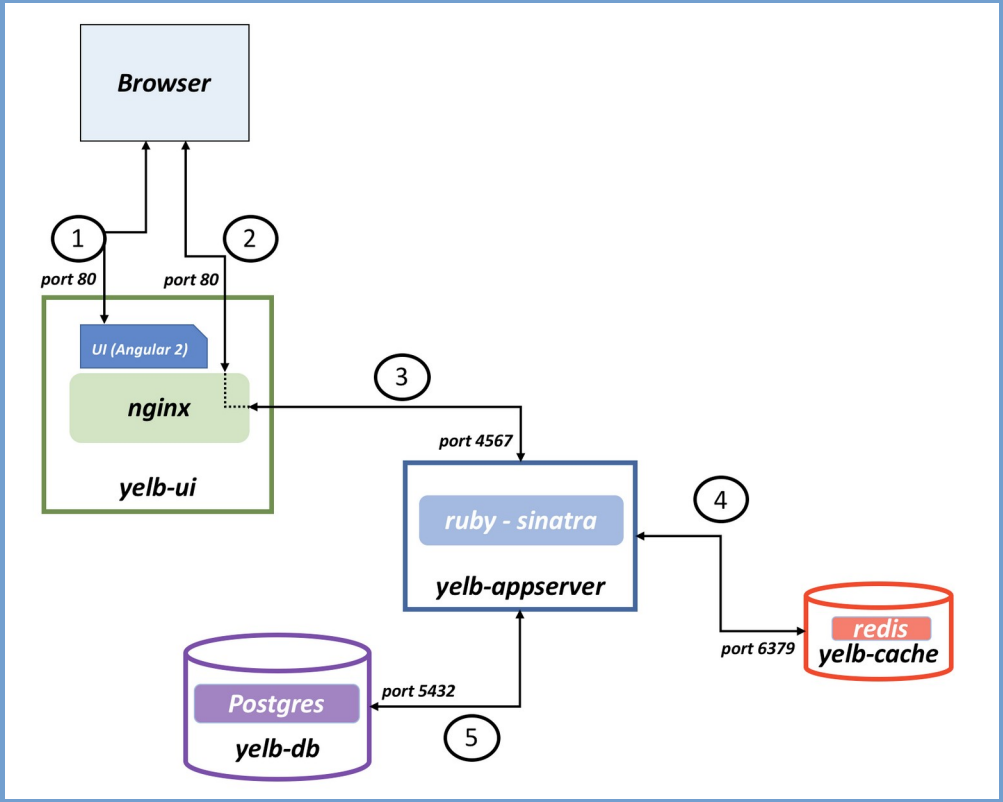
Хочу удалить все контейнеры:

```
~> sudo docker rm $(sudo docker ps -a -q)
```

Хочу удалить все образы:

```
~> sudo docker rmi $(sudo docker images -q)
```

Мультиконтейнерные приложения: Docker Compose



Docker compose позволяет развернуть мультиконтейнерное приложение, он выполнит сборку образов, запустит контейнеры в нужном порядке, свяжет порты.

docker-compose.yml

- version: "3"
- services — список контейнеров с их описанием
- networks — список виртуальных сетей докера (могут быть сети между контейнерами или между контейнером и хостом)
- volumes — список томов, например, в качестве разделяемой памяти

```yaml
services:
  proxy:
    image: nginx
    volumes:
      - type: bind
        source: ./proxy/nginx.conf
        target: /etc/nginx/conf.d/default.conf
        read_only: true
    ports:
      - 80:80
    depends_on:
      - backend

  backend:
    build:
      context: backend
      target: builder
```

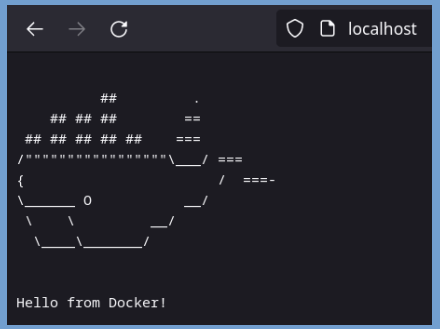
[3] → awesome-compose/nginx-golang

Запуск приложения с Docker Compose

Создаем образы и запускаем контейнеры одной командой:

```
nginx-golang> sudo docker compose up -d
[+] Running 2/3
 ⠿ Network nginx-golang_default      Created
 ✔ Container nginx-golang-backend-1  Started
 ✔ Container nginx-golang-proxy-1    Started
```

```
nginx-golang> sudo docker compose ps
NAME                    IMAGE                   COMMAND                  SERVICE   CREATED         STATUS         PORTS
nginx-golang-backend-1  nginx-golang-backend    "/code/bin/backend"      backend   2 minutes ago   Up 2 minutes
nginx-golang-proxy-1    nginx                   "/docker-entrypoint.…"   proxy     2 minutes ago   Up 2 minutes   0.0.0.0:80->80/tcp
```



Остановим и удалим:

```
nginx-golang> sudo docker compose down
[+] Running 3/3
 ✔ Container nginx-golang-proxy-1    Removed
 ✔ Container nginx-golang-backend-1  Removed
 ✔ Network nginx-golang_default      Removed
```

Ссылотьки=3

[1]Документация докера
[2]Прикольная статья с примерами для начинающих
[3]Репозиторий с примерами использования docker compose
[4]Страница с информацией о репозиториях (просто скопируйте баш для добавления репы, не ставьте докер десктоп, как там написано!!!)
[5]Способы хранения данных (для продвинутых)
[6]О слоях в образе (для очень продвинутых)
[7]Подробнее о compose.yaml (для продвинутых)
[8]По-простому о VM и Докере, различия
[9]Docker hub

В целом на [1] много информации, полистайте, попрактикуйтесь