

# Контейнеризация

Урок 10

Кухтичев Антон



**Не забудьте отметить!**

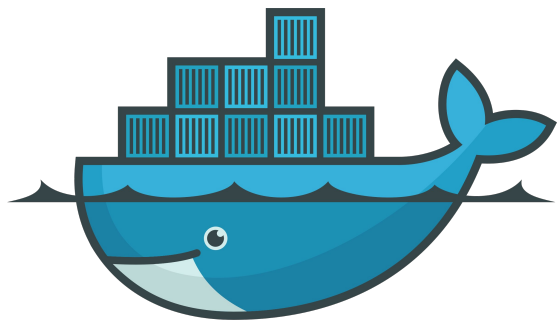
- Понятие контейнера и образа;
- Использование docker;
- Написание Dockerfile-ов;
- Проброс портов и volume;
- Использование Docker-Compose.

# Понятие контейнера и образа

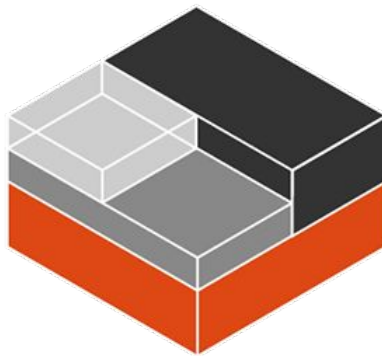
- Изоляция процессов
- Ограничение ресурсов
  - CPU
  - RSS
  - I/O
  - Disk usage
- Экосистема образов

- + Легковесность;
- + Почти нет накладных расходов;
- + Готовые образы, инфраструктура доставки;
- ОС / Ядро фиксированы;
- Худшая безопасность.

- Повышение утилизации железа;
- Гибкое управление зависимостями;
- Способ доставки ПО на сервера;
- Простое разворачивание тестовых сред;
- (\*) Декларативное описание структуры проекта



docker



**LXC**



OpenVZ



## **namespaces**

Механизм изоляции: PID, NET, MNT, USER, ...

## **cgroups**

Механизм ограничения ресурсов процесса

```
mkdir /sys/fs/cgroup/memory/new_group  
echo 655360 > /sys/fs/cgroup/memory/new1/memory.limit_in_bytes  
echo `pidof project` > /sys/fs/cgroup/memory/new1/tasks
```

# Docker

```
https://docs.docker.com/install/linux/docker-ce/ubuntu/
```

Ha 27.11.2019

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo  
apt-key add -
```

```
sudo add-apt-repository \  
"deb [arch=amd64] https://download.docker.com/linux/ubuntu \  
$(lsb_release -cs) stable"
```

```
sudo apt-get update
```

```
sudo apt-get install docker-ce
```

```
sudo usermod -aG docker `id -un`
```

```
sudo systemctl start docker
```

Подробная инструкция [тут](#)

**Контейнер / Container** - группа процессов работающих в изолированном окружении, в своей файловой системе, возможно, с ограничением ресурсов.

Контейнер может содержать как одну запущенную программу (например Nginx), так и целое окружение (init, bash, и т.д.)

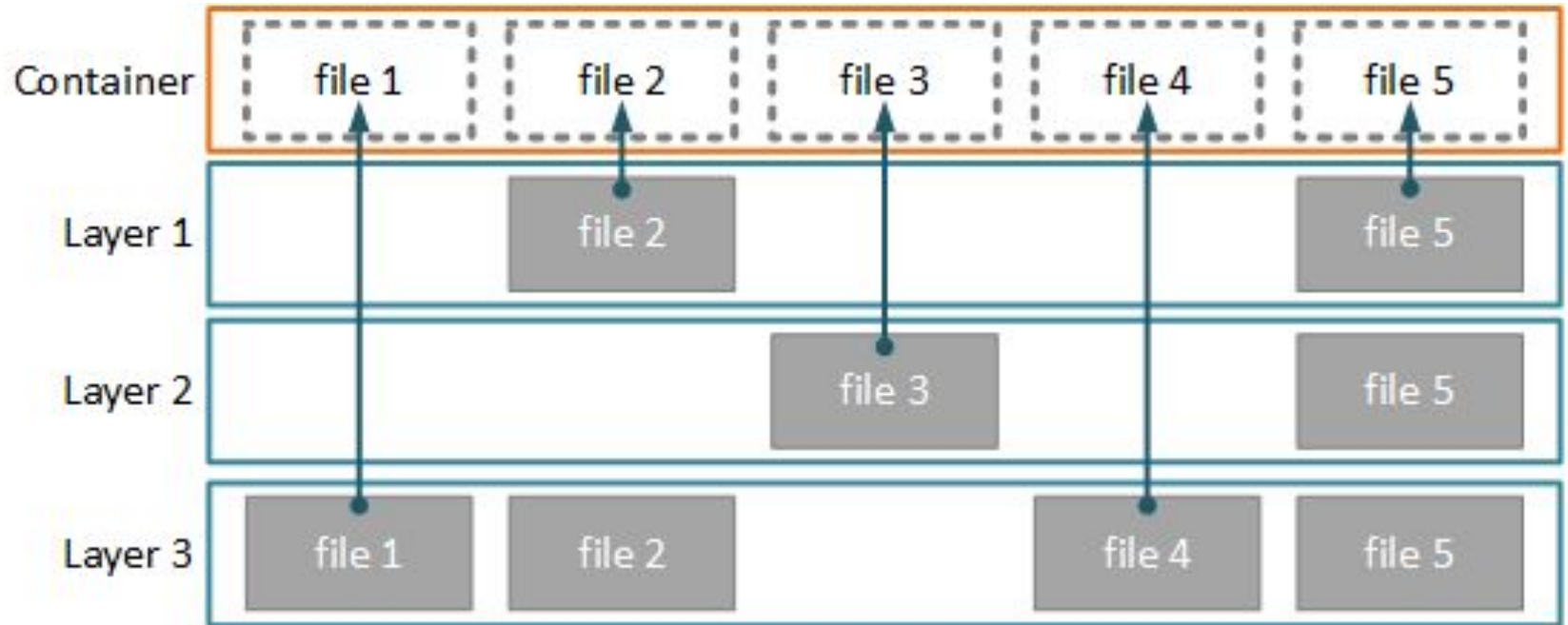
```
docker run -d nginx      # запустить контейнер
docker ps                # список контейнеров
docker ps -a             # список всех контейнеров
docker logs 5a592c       # посмотреть логи
docker exec -it 5a592c bash # "подключиться"
docker stop 5a592c       # остановить контейнер
docker rm 5a592c         # удалить контейнер
docker inspect 5a592c    # информация о контейнер
```

**Образ / Image** - образец (шаблон) файловой системы для контейнера. Образ содержит все необходимые образцу программы и файлы настроек, но не содержит пользовательских данных.

Образы могут наслаиваться друг на друга.

```
docker pull nginx      # скачать образ из registry
docker images          # список образов
docker rmi nginx       # удалить образ
docker run -d nginx    # запустить контейнер
                        # на основе образа
docker push my_proj:v2 # загрузить образ в registry
```





Ок, а как использовать nginx ?

```
docker run -d --name ngx1 nginx
```

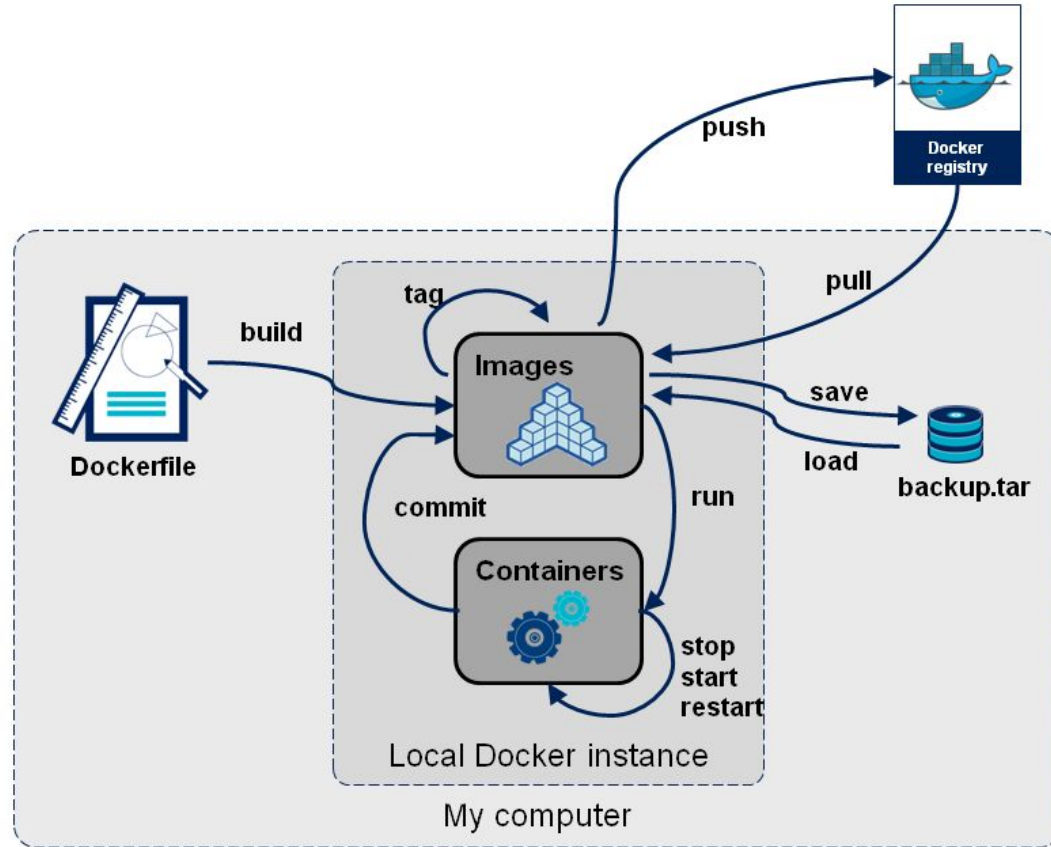
```
docker inspect -f '{{.NetworkSettings.IPAddress}}' ngx1  
# 172.18.0.2
```

Проверяем

```
http://172.18.0.2/
```

```
docker run -d \  
  -p 8080:80 \  
  -v /home/user/proj:/usr/share/nginx/html:ro \  
  -e NGINX_HOST=foobar.com \  
  --name ngx1 \  
  nginx
```

- p local\_port:container\_port - проброс порта
- v local\_dir:container\_dir - проброс директории (volume)
- e NAME=val - установка переменной окружения



# Dockerfile

```
/path/to/project      # BuildDir
├── ask
├── askme
├── templates
├── static
├── manage.py
├── Dockerfile          # Сборка
├── docker-compose.yml  # Оркестрация
├── .db_data            # Volume для базы
└── requirements.txt
```

```
FROM ubuntu:18.04
ADD . /app
RUN apt-get update
RUN apt-get install -y python3.6 python3-pip
RUN pip3 install -r /app/requirements.txt
EXPOSE 8000
USER nobody
WORKDIR /app
CMD /usr/local/bin/gunicorn askme.wsgi
```

**FROM** - базовый образ

**ADD** - добавить файлы из сборочной директории

**RUN** - запустить команду при сборке образа

**EXPOSE** - информация о том какой порт прослушивается

**CMD** - команда, которая будет запущена при старте контейнера

**USER** - пользователь под которым будет запущена **CMD**

**WORKDIR** - директория в которой будет запущена **CMD**



```
docker build -t askme:v2 /path/to/project
```

askme:v2 - название (и возможно тэг) образа  
/path/to/project - путь к директории с Dockerfile

```
В Dockerfile указываем точку монтирования  
FROM ubuntu:16.04
```

```
...  
VOLUME /app  
...
```

При запуске образа монтируем директорию с проектом

```
docker run -d -v /path/to/project:/app askme
```

# Docker Compose

Для запуска нескольких взаимодействующих контейнеров нужно согласовать:

- IP адреса / имена хостов
- Логины и пароли
- Порядок запуска
- Проверка работоспособности

Это нужно сделать **воспроизводимым**.

```
https://docs.docker.com/compose/install/
```

```
Ha 09.04.2019
```

```
sudo curl -L \
```

```
"https://github.com/docker/compose/releases/download/"\
```

```
"1.22.0/docker-compose-$(uname -s)-$(uname -m)" \
```

```
-o /usr/local/bin/docker-compose
```

```
sudo chmod +x /usr/local/bin/docker-compose
```

```
version: "2.1"
services:
  serviceA:
    image: postgres:10
  serviceB:
    image: askme
  volumes:
    - host_dir:container_dir
    - host_port:container_port
```

Формат файла - YAML

```
docker-compose build    # пересобрать все образы
docker-compose create   # создать все контейнеры
docker-compose start    # запустить контейнеры
docker-compose stop     # остановить контейнеры
docker-compose rm       # удалить контейнеры
docker-compose logs     # посмотреть логи
docker-compose up       # build, create, start, logs -f
```

```
# часто можно указать конкретный контейнер
docker-compose restart webapp
```

# Makefile



```
up:
    docker-compose up

test: up
    docker-compose exec webapp python3 /app/manage.py test

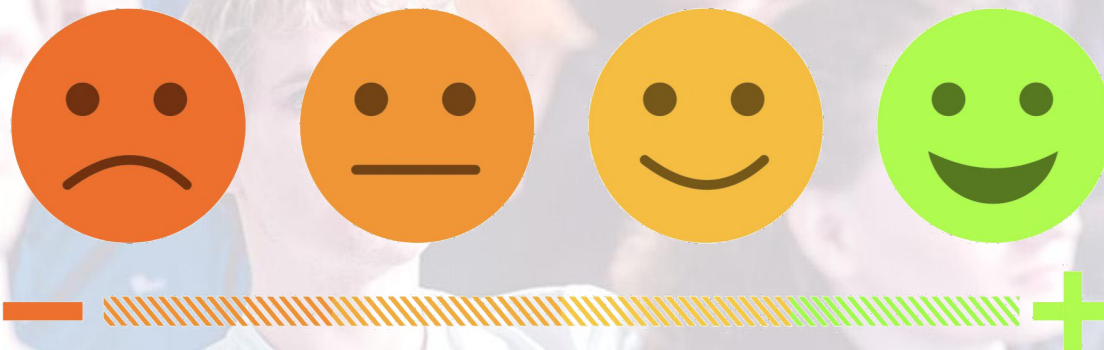
migrate: up
    docker-compose exec webapp python3 /app/manage.py
migrate
```

```
make migrate
```

1. Установить docker и docker-compose (1 балл);
2. Создание Dockerfile для Django приложения (5 баллов);
3. Создание docker-compose для проекта (4 балла);
4. Создание Makefile для проекта (2 балла);

Преподаватель должен иметь возможность, имея установленными только git, docker и docker-compose клонировать проект, выполнить команды ``make migrate`` и увидеть успешную миграцию.

**Не забудьте оценить занятие!**



Спасибо  
за внимание!

Антон Кухтичев

[a.kukhtichev@corp.mail.ru](mailto:a.kukhtichev@corp.mail.ru)