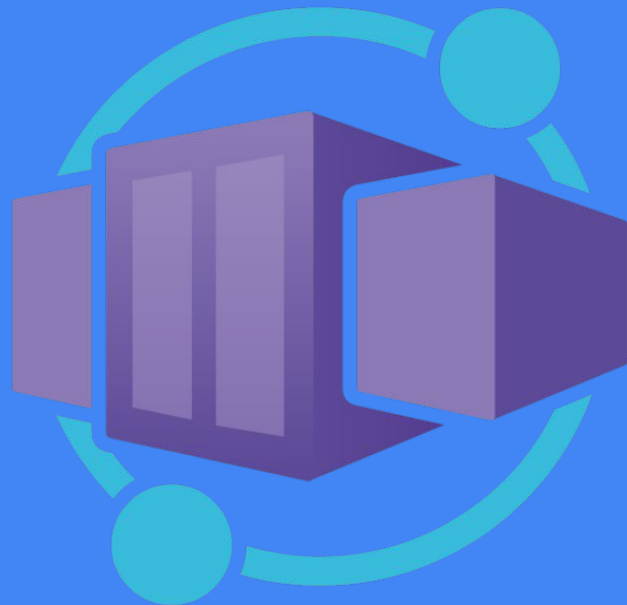


# Dockerfiles и слои

Урок 4

Docker: Dockerfiles, слои, архитектура





# Основы контейнеризации

1

Лекция 1: Механизмы пространства имен

2

Семинар 1: Механизмы пространства имен

3

Лекция 2: Механизмы контрольных групп

4

Семинар 2: Механизмы контрольных групп

5

Лекция 3: Введение в Docker

6

Семинар 3: Введение в Docker

7

Лекция 4: Dockerfiles и слои

8

Семинар 4: Dockerfiles и слои

9





Лекция 5: Docker Compose и Docker Swarm

10

Семинар 5: Docker Compose и Docker Swarm



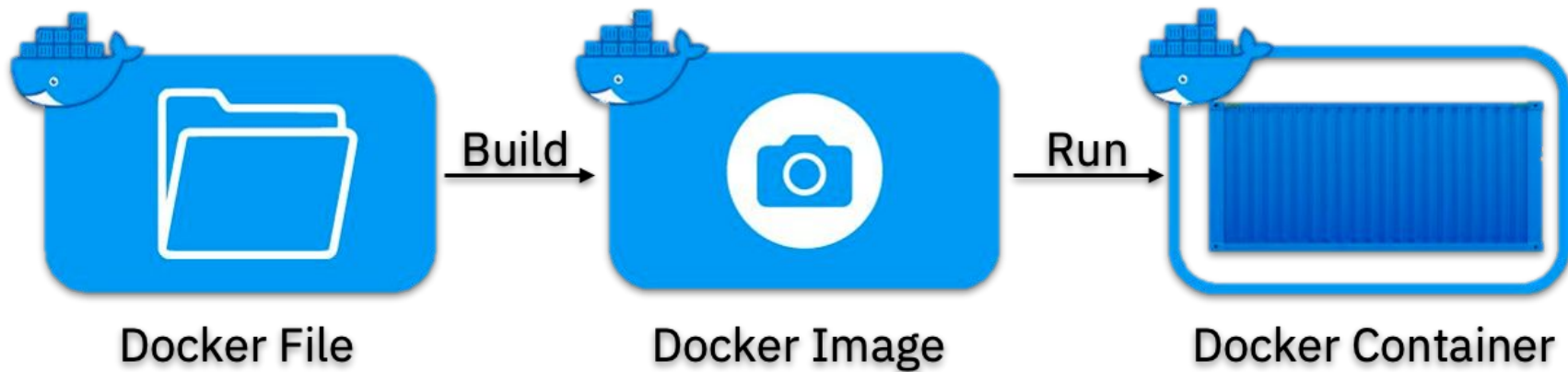
## Что будет на уроке сегодня

-  Понятие “Слои” в образах в контейнерах
-  Архитектура образа
-  Взаимосвязь размера образа и его компонент
-  Основные практики создания образов



Слои — это результат создания образов

## Workflow

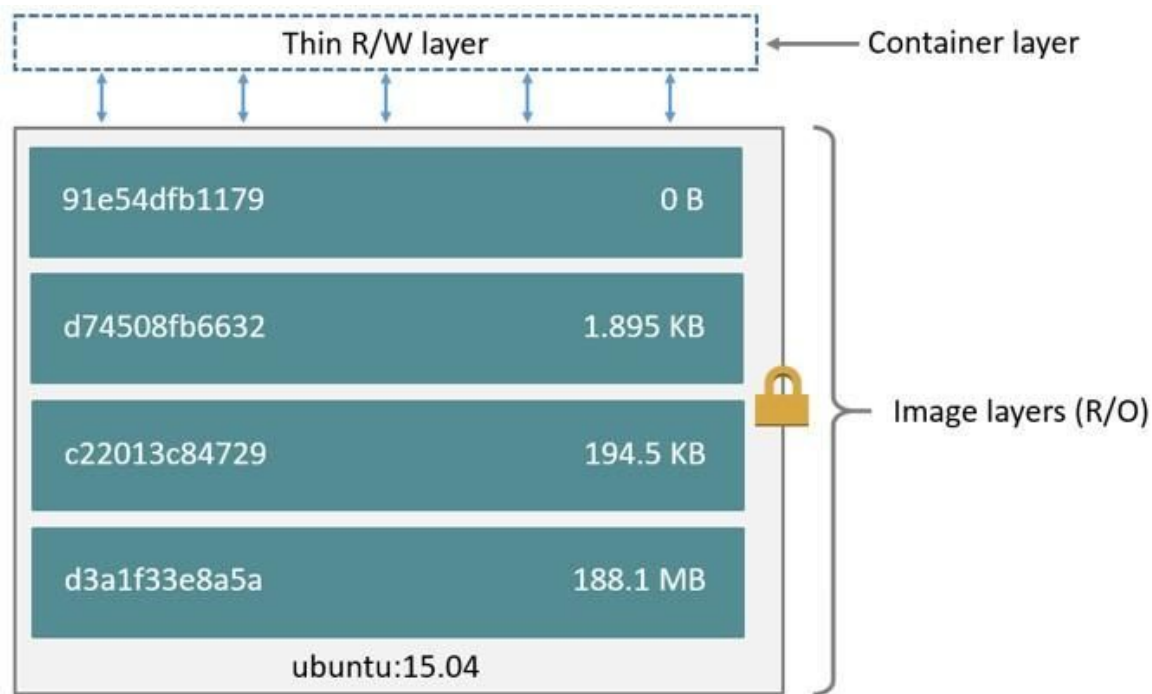




# Dockerfile



## Слои образа



Container  
(based on ubuntu:15.04 image)



## Слои образа

```
root@testVM:~# docker run ubuntu:15.04
Unable to find image 'ubuntu:15.04' locally
15.04: Pulling from library/ubuntu
Image docker.io/library/ubuntu:15.04 uses outdated schema1 manifest format. Please upgrade to
mpatibility. More information at https://docs.docker.com/registry/spec/deprecated-schema-v1/
9502adfb7f1: Pull complete
4332ffb06e4b: Pull complete
2f937cc07b5f: Pull complete
a3ed95caeb02: Pull complete
Digest: sha256:2fb27e433b3ecccea2a14e794875b086711f5d49953ef173d8a03e8707f1510f
Status: Downloaded newer image for ubuntu:15.04
```



## Слои образа





## Инструкции Dockerfiles



### **FROM**

Задаёт базовый образ



### **LABEL**

Добавление метаданных в образ



### **ENV**

Задаёт постоянные переменные образа



### **ARG**

Задание переменных во время сборки образа



### **RUN**

Выполнение команды Linux



### **COPY**

Копирование файлов и папок



## Инструкции Dockerfiles



### **ADD**

Копирование файлов и папок + распаковка архивов



### **WORKDIR**

Задание рабочей директории



### **CMD**

Задание команды, выполняющейся при запуске образа



### **ENTRYPOINT**

Задание команды, выполняющейся при запуске образа



### **EXPOSE**

Открытие порта в образ/контейнер



### **VOLUME**

Создание точки монтирования



# Инструкции Dockerfiles



## Инструкция FROM



Каждый Dockerfile начинается с нее



Определяет базовый образ, на основе которого будет строиться образ



Примеры:

```
1 FROM alpine
2 FROM ubuntu:22.10
3 FROM scratch
```



## Инструкция LABEL



Добавляет в образ различные метаданные



Не замедляет процесс сборки образа и не увеличивает его размер



Примеры:

```
1 LABEL maintainer="Vladlen Melnik"  
2 LABEL mt_telegram="@mazzahaker"  
3 LABEL mt_email="hack-me@list.ru"
```



## Инструкция ENV



Создает постоянные переменные в образе



Подходит при необходимости использования констант внутри



Примеры:

```
1 ENV ADMIN="mazzahaker"  
2 ENV DB_name="GBcontainers"  
3 ENV ROOT_PW="testpass123"
```



## Инструкция RUN



Выполняет Linux-команду во время сборки образа



В процессе выполнения создается отдельный слой



Примеры:

```
1 RUN apt update
2 RUN apt upgrade
3 RUN apt install openssh
```





## Инструкция COPY



Копирует файлы и папки в процессе сборки образа



В случае отсутствия директории, создает ее



Примеры:

```
1 COPY ./test.sh
2 COPY /example/site/test
3 copy ./example_local_dir/
```



## Инструкция ADD



- Может решать те же задачи, что и COPY
- Добавление файлов из удаленных источников
- Распаковка tar-файлов



Примеры:

```
1 ADD https://github.com/mazzahaker/test-lesson-  
jenkins/blob/main/pipeline /testdir123  
2  
3 ADD ftp://testsrv/testfile /testdir123/testfile
```



## Инструкция WORKDIR



Изменяет рабочую директорию контейнера



Лучше указывать абсолютные пути при обращении к файлам



В случае отсутствия директории, происходит автоматическое создание



Примеры:

```
1 WORKDIR /usr/src/test_directory
2 WORKDIR /test123
3 WORKDIR ../dir/lesson
```



## Инструкция ARG

💡 задает переменную, значение которой передается во время выполнения

💡 Имеет значение по умолчанию

💡 Примеры:

```
1 ARG test_var=script.py
2 ARG DB_name="example_db"
3 ARG DB_pass="pass123"
```



## Инструкция CMD



Описывает команду, которая выполнится при запуске контейнера  
Можно переопределить аргументы команды, передав при запуске



Результат не добавляется в образ во время сборки!



Примеры:

```
1 CMD ["python", "./testscript.py"]  
2 CMD ["cowsay", "Hello from Docker!"]  
3 CMD ["bash", "uname -a"]
```



## Инструкция ENTRYPOINT



Описывает команду и аргументы, которые будут выполнены сразу после запуска контейнера



Важное отличие от CMD: команду и аргументы нельзя переопределить



Примеры:

```
1 ENTRYPOINT ["python", "test_var"]  
2 ENTRYPOINT ["bash", "sleep 600"]  
3 ENTRYPOINT ["bash", "uname -a"]
```



## ENTRYPOINT vs CMD



Выполнение одной и той же команды раз за разом



Если контейнер используется в роли приложения



Если постоянно необходимо передавать в контейнер различные аргументы

```
1 CMD ["python", "test_var"]
```



## Инструкции EXPOSE и VOLUME



### Инструкция EXPOSE

Указывает на необходимость открытия портов для связи

```
1 EXPOSE 8080  
2 EXPOSE 2319
```



### Инструкция VOLUME

Указывает место в локальной системе, которое контейнер будет использовать для хранения данных

```
1 VOLUME /my_volume  
2 VOLUME /test/volume123
```









# Итоги занятия



## Подведем итоги

-  Познакомились с понятием “Слой” в образах в контейнерах
-  Изучили архитектуру образа
-  Определили на практике взаимосвязь размера образа и его компонент
-  Познакомились с некоторыми основными практиками создания образов