Урок-recap: Docker

Часть 1. Зачем вообще Docker

Проблема до Docker

Факты:

– "У меня работает, у тебя нет"

– Разные версии ОС, библиотек, языков

– Сложно повторить окружение

DevOps-задача:

– Сделать окружение **воспроизводимым**

Что решает Docker

Факты:

– Приложение + зависимости упакованы вместе

– Запускается одинаково везде

Ключевые эффекты:

– Быстрый запуск

– Изоляция

– Повторяемость

– Упрощение CI/CD

Docker простыми словами

Docker — это инструмент для запуска приложений в изолированных контейнерах.

Контейнер:

– Не виртуальная машина

– Использует ядро хоста

– Лёгкий и быстрый

Часть 2. VM vs Containers (важно для собеседования)

Виртуальная машина

– Своя ОС
– Гипервизор
– Тяжёлая

Пример:
– VirtualBox
– VMware

Контейнер

– Общая ОС хоста
– Изоляция на уровне процессов
– Быстрый старт

Пример:
– Docker

Главное отличие (кратко)

VM	Container
Своя ОС	Общая ОС
Медленно	Быстро
Много ресурсов	Мало ресурсов

Часть 3. Архитектура Docker

Основные компоненты

– Docker Client
– Docker Daemon
– Docker Images
– Docker Containers
– Docker Registry

Docker Client

Факт:

– Команда `docker` в терминале

Примеры:

– docker run

– docker build

– docker ps

Docker Daemon

Факт:

– Сервис `dockerd`

Что делает:

– Собирает образы

– Запускает контейнеры

– Управляет сетью и volume

Docker Registry

Факт:

– Хранилище образов

Примеры:

– Docker Hub

– GitLab Registry

– Harbor

Часть 4. Image и Container

Docker Image

Факты:

– Шаблон

– Read-only

– Слои

Пример:

– nginx:1.25

Docker Container

Факты:

– Запущенный image

– Есть lifecycle

– Можно остановить, удалить

Команды:

– docker run

– docker stop

– docker rm

Image vs Container (спросит любой интервьюер)

Image — что запускать

Container — уже запущено

Часть 5. Dockerfile

Что такое Dockerfile

Факт:

– Инструкция для сборки image

Цель:

– Описать окружение как код

Минимальный Dockerfile

```
FROM nginx:alpine
COPY index.html /usr/share/nginx/html/
```

Что происходит:

– Берём базовый образ

– Кладём файл

– Получаем image

Основные инструкции Dockerfile

– FROM
– RUN
– COPY / ADD
– WORKDIR
– CMD
– ENTRYPOINT
– EXPOSE

CMD vs ENTRYPOINT (обязательно знать)

CMD:

– Команда по умолчанию

ENTRYPOINT:

– Основная команда контейнера

Кратко:

– ENTRYPOINT — что запускать

– CMD — аргументы

Часть 6. Docker CLI (минимум для Junior)

Базовые команды

– docker pull
– docker run
– docker ps
– docker images
– docker rm
– docker rmi

docker run — самая важная

Пример:

```
docker run -d -p 8080:80 nginx
```

Что здесь:

– `-d` — background
– `-p` — проброс портов

docker exec

Факт:

– Зайти внутрь контейнера

Пример:

```
docker exec -it nginx bash
```

Часть 7. Volumes

Проблема без volumes

Факт:

– Данные исчезают при удалении контейнера

Docker Volume

Факт:

– Хранение данных вне контейнера

Типы:

– named volume

– bind mount

Пример volume

```
docker run -v data:/var/lib/mysql mysql
```

Часть 8. Docker Networks

Зачем сети

Факт:

– Контейнеры должны общаться

Типы сетей

– bridge (по умолчанию)
– host
– none

Bridge network

Факт:

– Контейнеры видят друг друга по имени

Часть 9. Docker Compose

Проблема без compose

Факт:

– Много контейнеров = много команд

Docker Compose

Факт:

– Описание нескольких сервисов в YAML

Команда:

```
docker-compose up -d
```

Пример docker-compose.yml

```yaml
version: '3'
services:
  web:
    image: nginx
    ports:
      - "8080:80"
```

Часть 10. Docker в работе DevOps

Где используется Docker

– CI/CD
– Локальная разработка
– Kubernetes
– Testing

Docker и CI/CD

Факт:

– Каждый билд в одинаковом окружении

Пример:

– GitLab CI

– GitHub Actions

Часть 11. Частые вопросы на собеседовании

Вопросы

– Чем контейнер отличается от VM?
– Что такое image и container?
– Зачем Dockerfile?
– CMD vs ENTRYPOINT?
– Как сохранить данные?

Как отвечать

Принципы:

– Коротко

– По фактам

– Без заученных фраз

Практика (на уроке)

1. Запустить nginx контейнер
2. Написать простой Dockerfile
3. Собрать image
4. Подключить volume
5. Запустить через docker-compose

Итог урока

Факт:

– Docker — базовый инструмент DevOps

Junior должен:

– Понимать концепции

– Уверенно пользоваться CLI

– Объяснить Docker словами

Домашнее задание (опционально)

– Написать Dockerfile для простого приложения

– Поднять связку nginx + app через docker-compose