GeekBrains

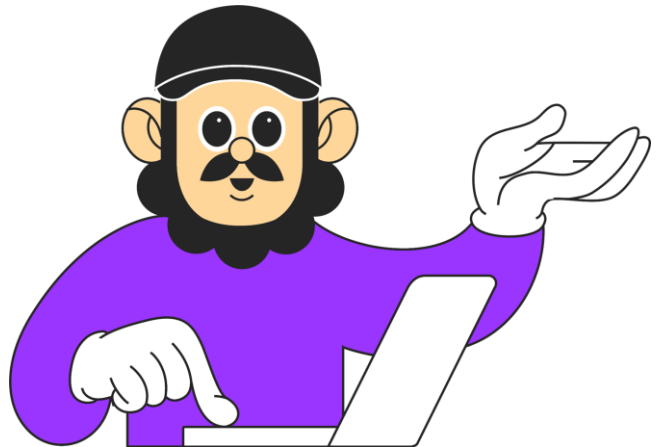
Запуск веб-приложения из контейнеров

Семинар 7

Курс “Операционные системы и виртуализация”

План на сегодня

- Виртуализация и контейнеризация
- Обзор docker
- Установка docker
- Управление образами и контейнерами
- Управление сетями в docker
- Обзор docker-compose



Эволюция виртуализации

Контейнеризация — метод виртуализации, при котором ядро операционной системы поддерживает несколько изолированных экземпляров пространства пользователя вместо одного.

Эти экземпляры с точки зрения пользователя полностью идентичны отдельному экземпляру операционной системы. Простыми словами, контейнеризация позволяет виртуализировать процесс.

Docker — программное обеспечение для автоматизации развёртывания и управления приложениями в средах с поддержкой контейнеризации

TEMA

Подключаем репозиторий

01

GeekBrains

1. apt-get install apt-transport-https ca-certificates curl gnupg-agent software-properties-common -y
2. curl -fsSL https://download.docker.com/linux/ubuntu/gpg | apt-key add -
3. add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

TEMA

Установка

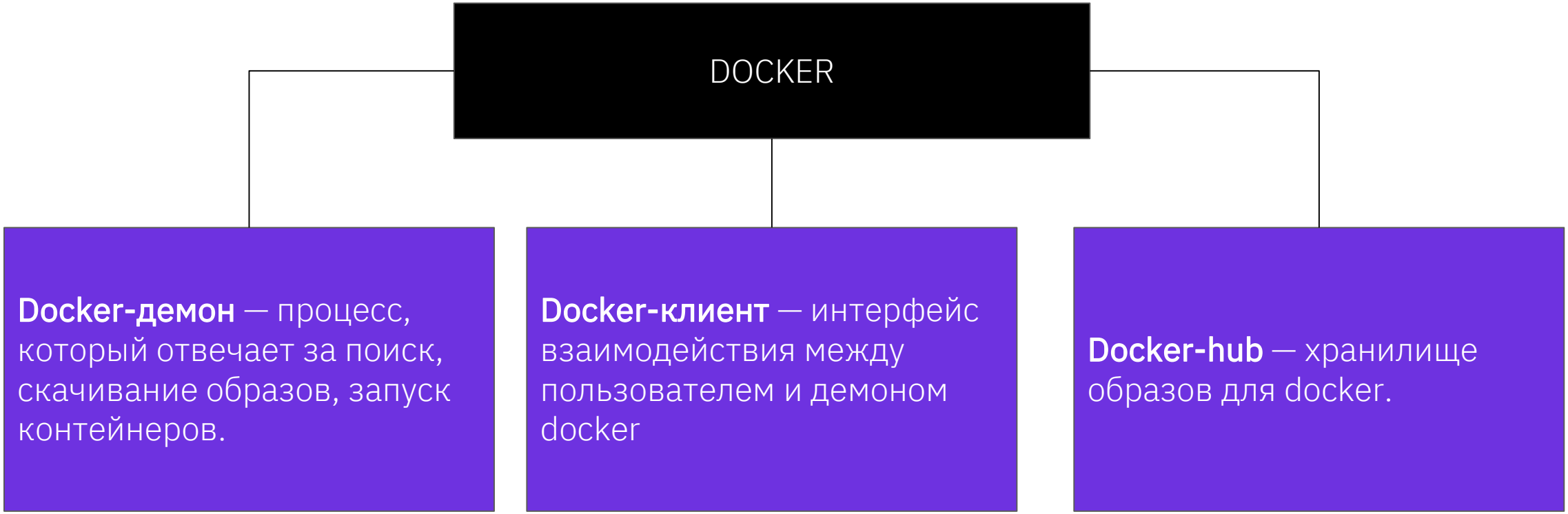
02

GeekBrains

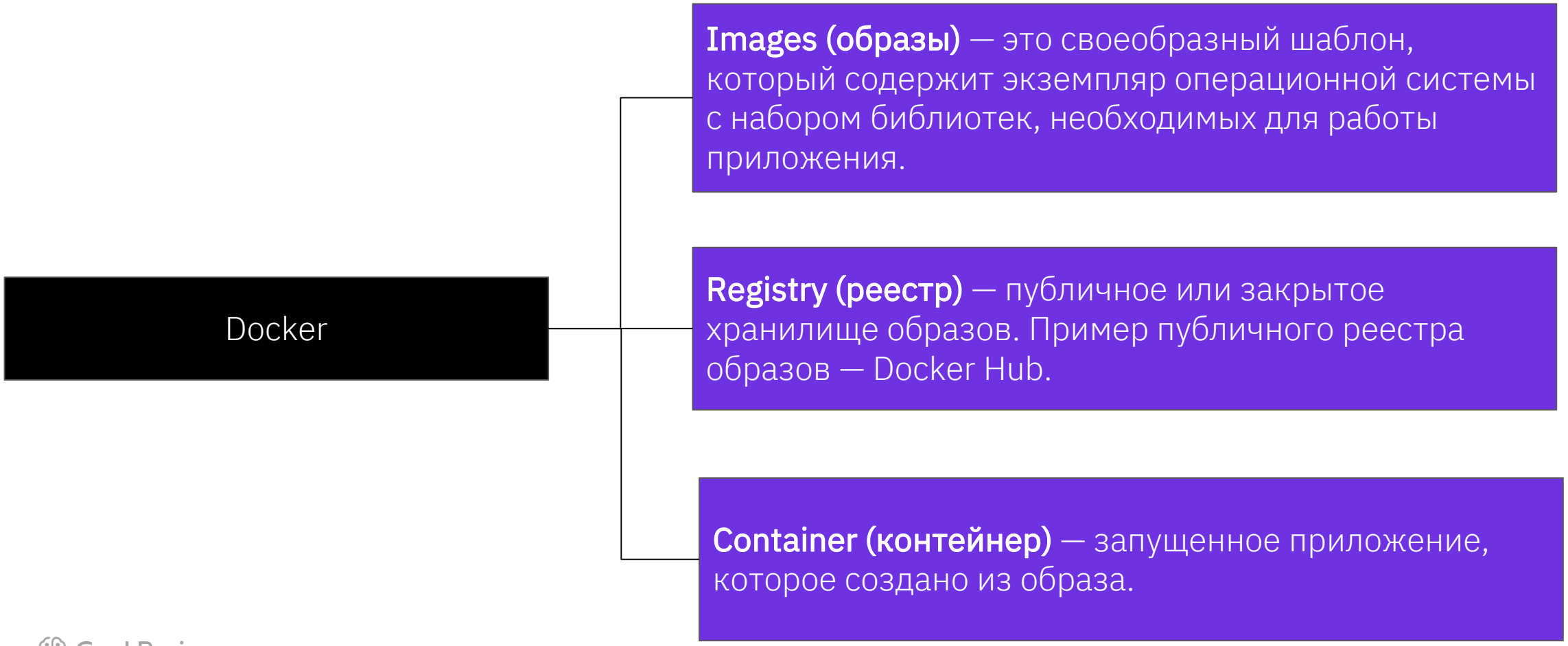
```
apt update;
apt install docker -y
apt install docker.io –y
apt install docker-compose -y
```

Обзор docker

Docker с точки зрения ОС



Docker с точки зрения архитектуры



Управление образами и контейнерами

Управление образами и контейнерами

docker search image_name — поиск образа в реестре. Например, docker search nginx найдёт все образы, которые содержат веб-сервис nginx.

docker pull image_name скачает диск из реестра, например, docker pull nginx.

docker run --name container_name image_name запустить контейнер из скачанного образа

docker rm container_name удалит контейнер

Dockerfile

Dockerfile — сценарий, в котором будут описаны все шаги по сборке нашего приложения.

Структура Dockerfile:

FROM — определит базовый образ, из которого будет собираться контейнер.

MAINTAINER — сообщит контейнеру имя автора создаваемого образа.

RUN — запустит команду внутри образа.

ADD — берёт файлы с хоста и кладёт внутрь образа.

VOLUME — директория, которая будет подключена в контейнер.

EXPOSE — задаст порт, через который контейнер будет общаться с внешним миром.

CMD — команда, которая будет запущена при старте контейнера из образа.

Контейнер

docker build -t image_name . - сборка образа

docker run --name container_name image_name - запуск контейнера из созданного образа

Управление сетями в docker

Bridge — сети по умолчанию, аналог типа подключения NAT в VirtualBox. Связь устанавливается через Bridge-интерфейс

Host — с помощью этого драйвера контейнер получает доступ к собственному интерфейсу хоста. Аналог подключения «Мост» в VirtualBox.

Macvlan - даёт контейнерам прямой доступ к интерфейсу и суб-интерфейсу (VLAN) хоста.

Overlay - позволяет строить сети на нескольких хостах с Docker.

Управление сетями в docker

1. **docker network ls** - просмотреть доступные сети
2. **docker network inspect network_name** - просмотреть участников сети
3. Доступ к приложениям, запущенным в контейнере, осуществляется через **iptables**

Обзор docker-compose

Обзор docker-compose

docker-compose повторяет весь функционал Docker, за исключением одного: если Docker применяется для управления одним конкретным сервисом, то docker-compose позволяет управлять несколькими контейнерами, входящими в состав приложения.

Файл docker-compose.yml:

```yaml
version: '3'
services:
  nginx:
    image: nginx:latest
    ports:
      - 80:80
    volumes:
      - /var/www/html

```

1. **version** '3' говорит об использовании третьей версии формата файлов для docker-compose.
2. Директива **service** описывает службу, которую мы будем запускать, дальше идёт имя nginx.
3. Собираем контейнер из последней стабильной версии nginx, доступной на Docker Hub: **image**: nginx:latest и пробрасываем 80-й порт хост-машины и каталог /var/www/html, используя директивы **ports** и **volumes**.

Обзор docker-compose

1. **version** '3' говорит об использовании третьей версии формата файлов для docker-compose.
2. Директива **service** описывает службу, которую мы будем запускать, дальше идёт имя nginx.
3. Собираем контейнер из последней стабильной версии nginx, доступной на Docker Hub: **image**: nginx:latest и пробрасываем 80-й порт хост-машины и каталог /var/www/html, используя директивы **ports** и **volumes**.

Обзор docker-compose

4. Обратите внимание, что файл **docker-compose.yml** для каждого контейнера должен лежать в отдельной папке.
5. Запускаем наш проект, используя команду **docker-compose up -d --build**, говорим, что docker-compose должен запустить контейнер в оперативной памяти, выполнив сборку из образа.

Спасибо!

Каждый день вы становитесь лучше :)