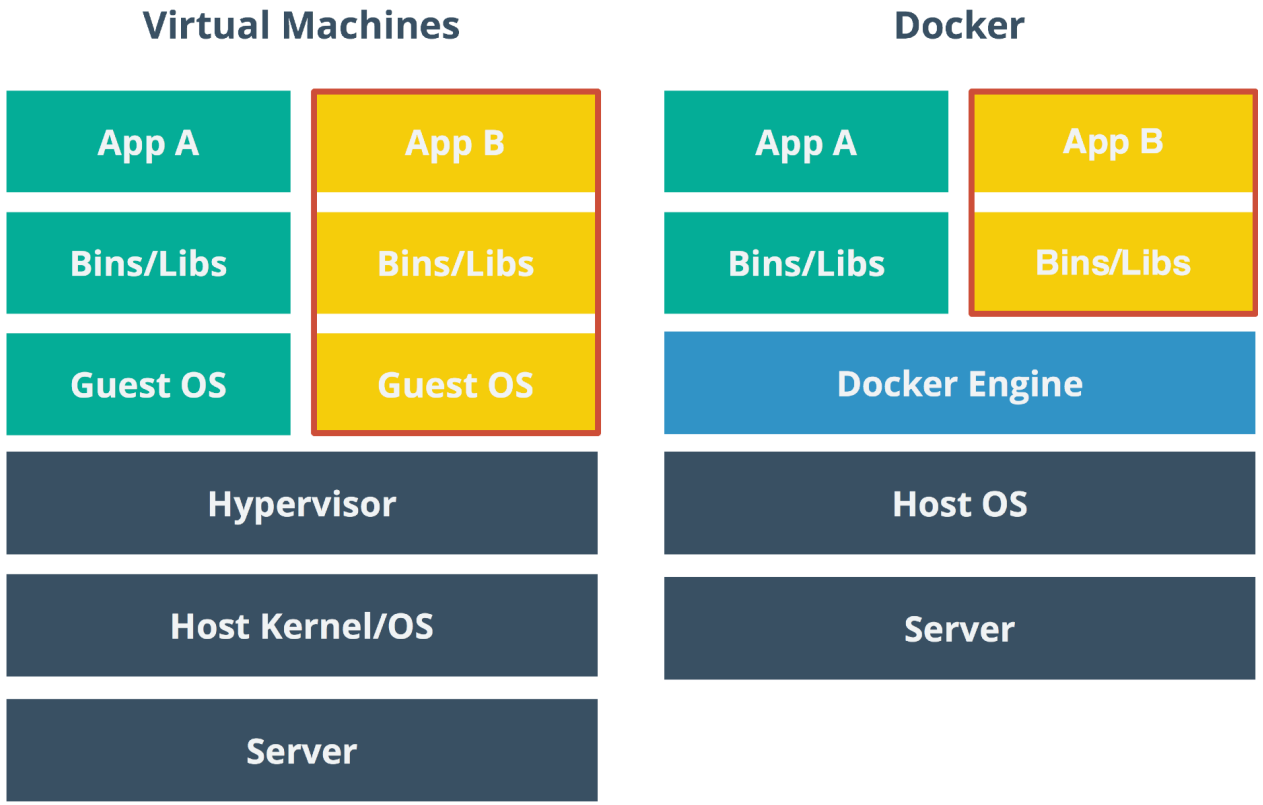
# Технология и методы программирования

Тема 11. Docker

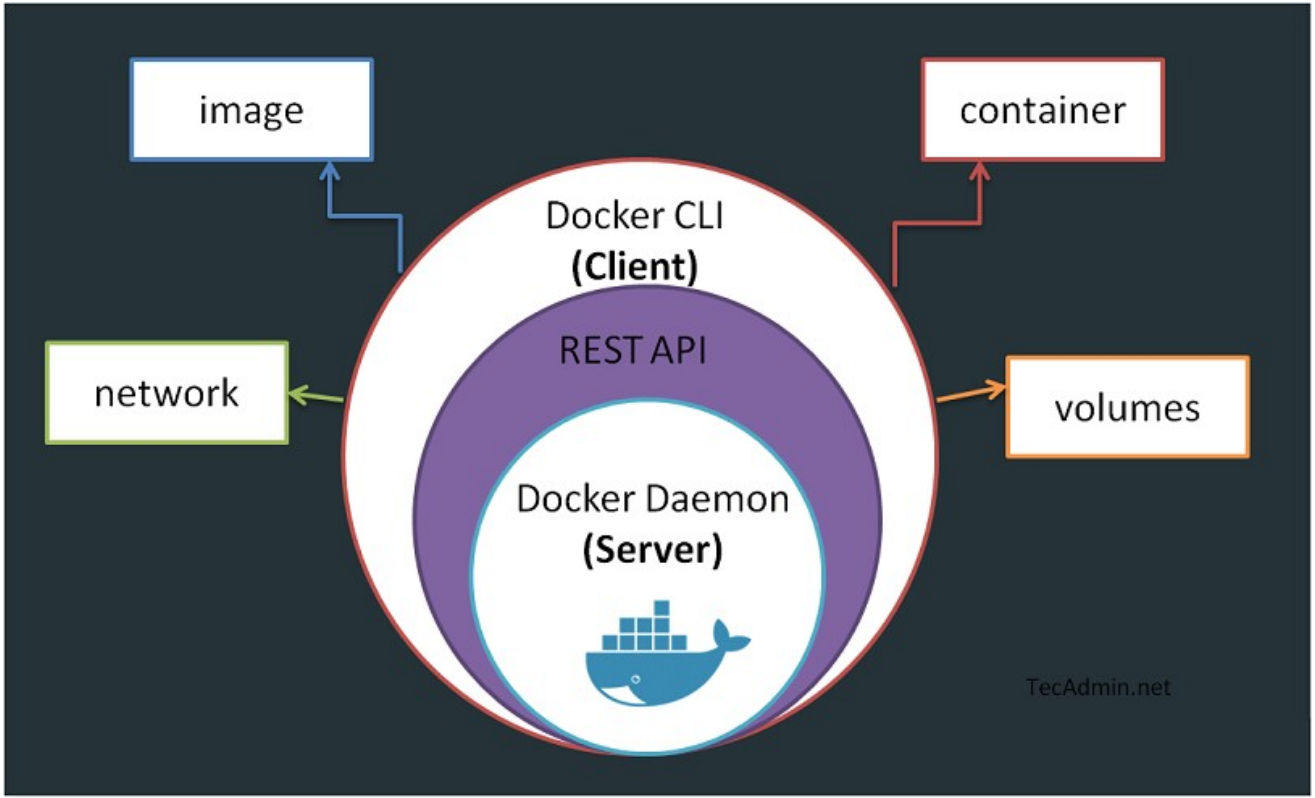
# Docker. Зачем?

- Изолированный запуск приложений в контейнерах
- Упрощение разработки, тестирования и развёртывания приложений
- Отсутствие необходимости конфигурировать среду для запуска (она поставляется вместе с приложением в контейнере)
- Упрощает масштабируемость
- Чтобы запускать **rm -rf** и тебе за это ничего не было

# Отличия от ВМ



# Архитектура

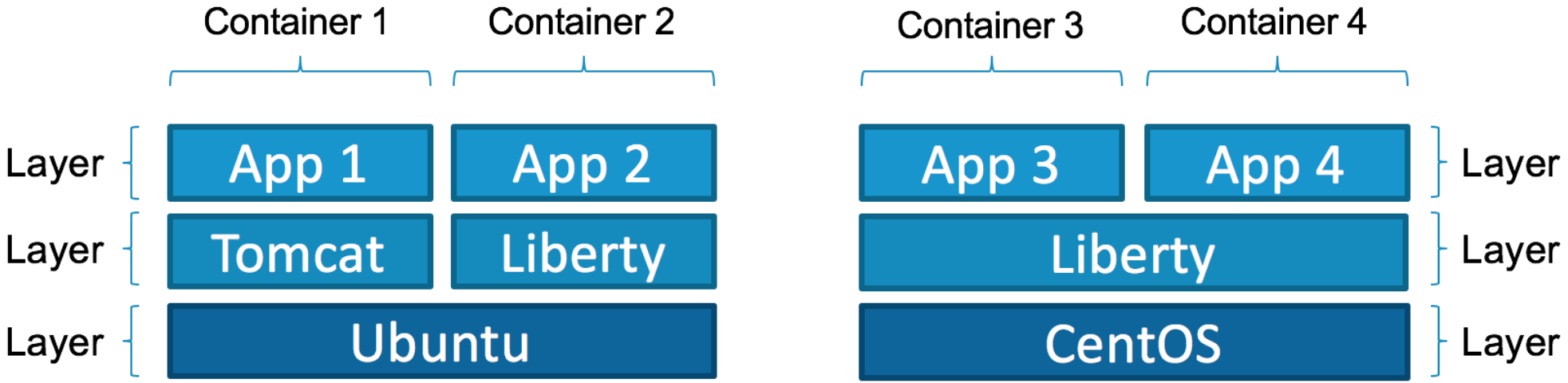


# Архитектура. Терминология

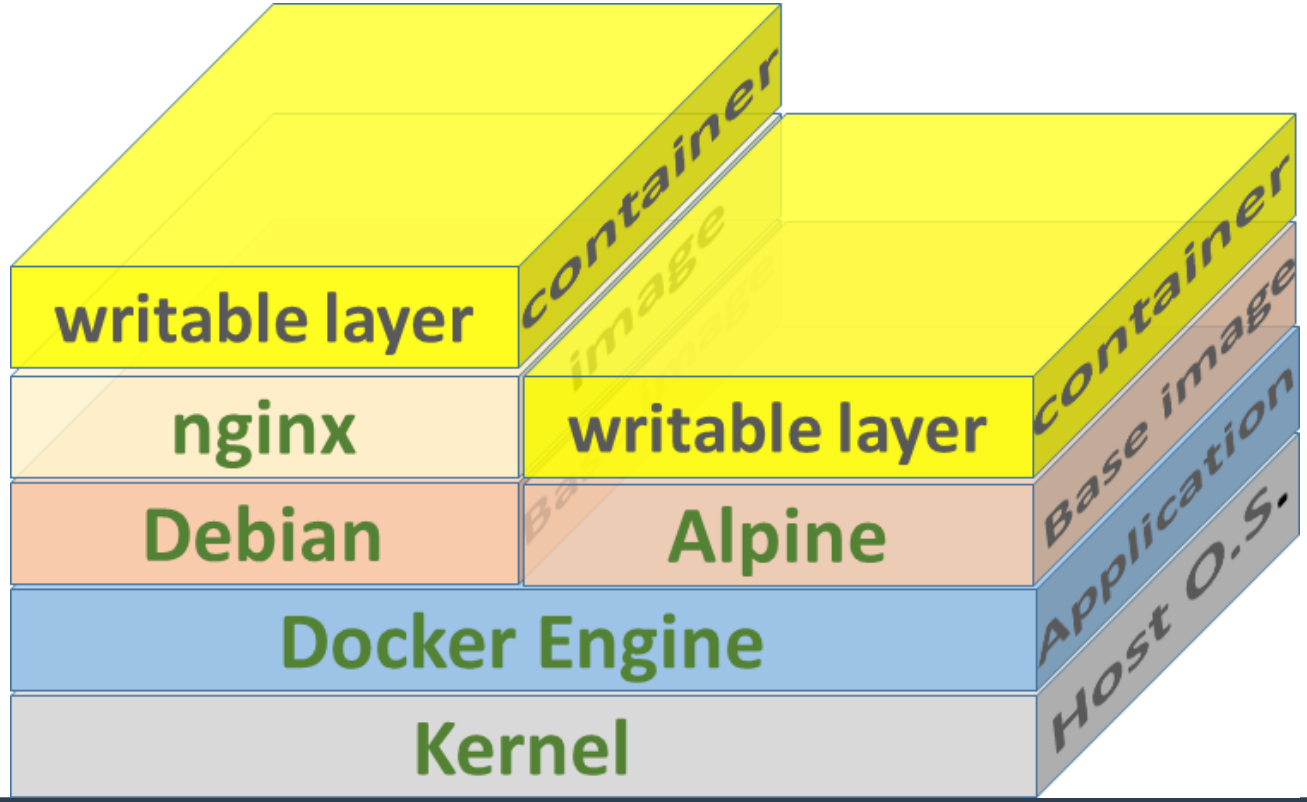
- Image (образ) — полная копия файловой системы и первоначальная конфигурация контейнера
- container (контейнер) — запущенный на выполнение образ (который содержит приложение/набор приложений), изолированный от хостовой ОС
- Docker daemon (демон докера) — служба, запущенная на машине, которая отвечает за создание, запуск и уничтожение контейнеров. Демон — фоновый процесс, с которым взаимодействует клиент
- Docker client (клиент докера) — утилита командной строки, с помощью которой пользователь может взаимодействовать с демоном
- Docker registry (регистр образов) — сервер, хранящий набор образов. Пример https://hub.docker.com/

# Docker image

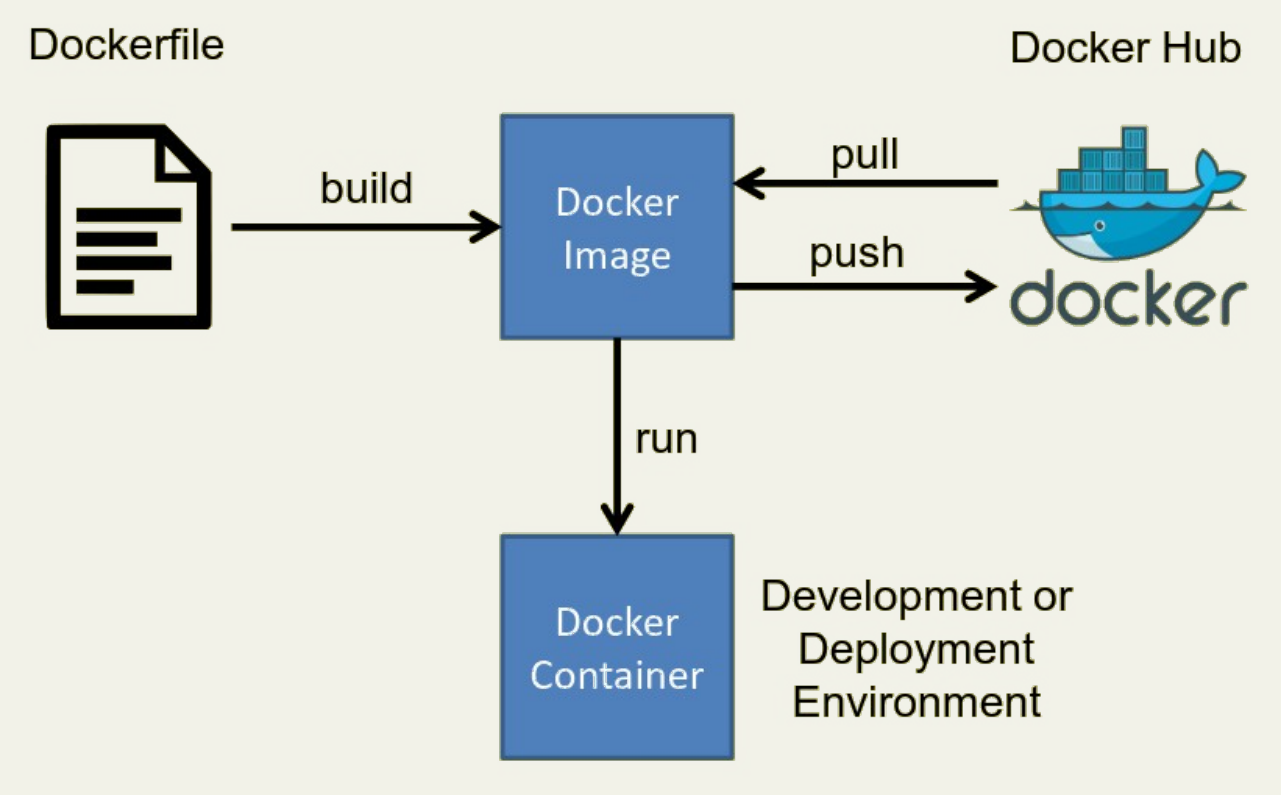
- Read-only шаблон с набором инструкций, предназначенных для создания контейнера
- Образ стоит из неизменяемых слоёв, каждый из которых может изменять ряд файлов из предыдущего слоя
- Из-за того, что слои readonly — они могут использоваться в нескольких образах



# Docker container



# Архитектура. Операции с образом



# Установка docker

Общее описание тут https://docs.docker.com/engine/install/

Для linux (debian)

apt install docker.io

# Основные команды

- **docker ps —** показать список запущенных контейнеров
- **docker ps -a** — список всех контейнеров в системе
- **docker run** *IMAGE* — создать контейнер из образа
- **docker stop** *CONTAINER* — остановить контейнер
- **docker container prune —** удалить неиспользуемые контейнеры

# Что делат docker run при запуске?

docker run busybox:latest

- скачивает указанный образ
- создаёт контейнер
- инициализирует ФС и монтирует readonly образ
- инициализирует сеть
- запускает указанный процесс (точку входа образа)
- Обрабатывает ввод/вывод пользователя

**DEMO | OMED | DEMO | OMED | DEMO | OMED**

# Dockerfile

```
1  FROM python:2.7-alpine
2  WORKDIR /
3  COPY requirements.txt .
4  RUN pip install -r requirements.txt
5  COPY . /
6  ENTRYPOINT ["python", "noisy.py"]
7  CMD ["--config", "config.json"]
```

# docker build

Сборка образа из dockerfile:

```
docker build .  -t image_name
```

доп инфа тут https://docs.docker.com/engine/reference/builder/

# docker-compose

Специальный файл в формате YAML. Название по умолчанию docker-compose.yml содержит информацию о группе docker образов, которые должны работать вместе

Документация по docker-compose
https://docs.docker.com/compose/

```yaml
version: "3"

networks:
  loki:

services:
  loki:
    image: grafana/loki:2.5.0
    ports:
      - "3100:3100"
    command: -config.file=/etc/loki/local-config.yaml
    networks:
      - loki

  promtail:
    image: grafana/promtail:2.5.0
    volumes:
      - /var/log:/var/log
    command: -config.file=/etc/promtail/config.yml
    networks:
      - loki

  grafana:
    image: grafana/grafana:latest
    ports:
      - "3000:3000"
    networks:
      - loki
```

# Задания

Первое:

- Поставить docker на ВМ
- Скачать репозиторий https://github.com/1tayH/noisy
- Собрать из него образ Docker, протестировать запуск контейнера

Второе:

- Найти на docker hub’е образ с apache2 и скачать его
- Прокинуть tcp-порт 80 в контейнер
- Создать свою index.html, смонтировать в контейнер директорию с ней, чтобы заменить файл по умолчанию
- Убедиться что в браузере видна новая созданная вами страничка

# Дополнительная информация

- youtube видео. Докер за 10 минут (для ленивых)
- Habr. Полное практическое руководство по docker
- Docker. getting started
- Docker compose. Документация