



# Docker

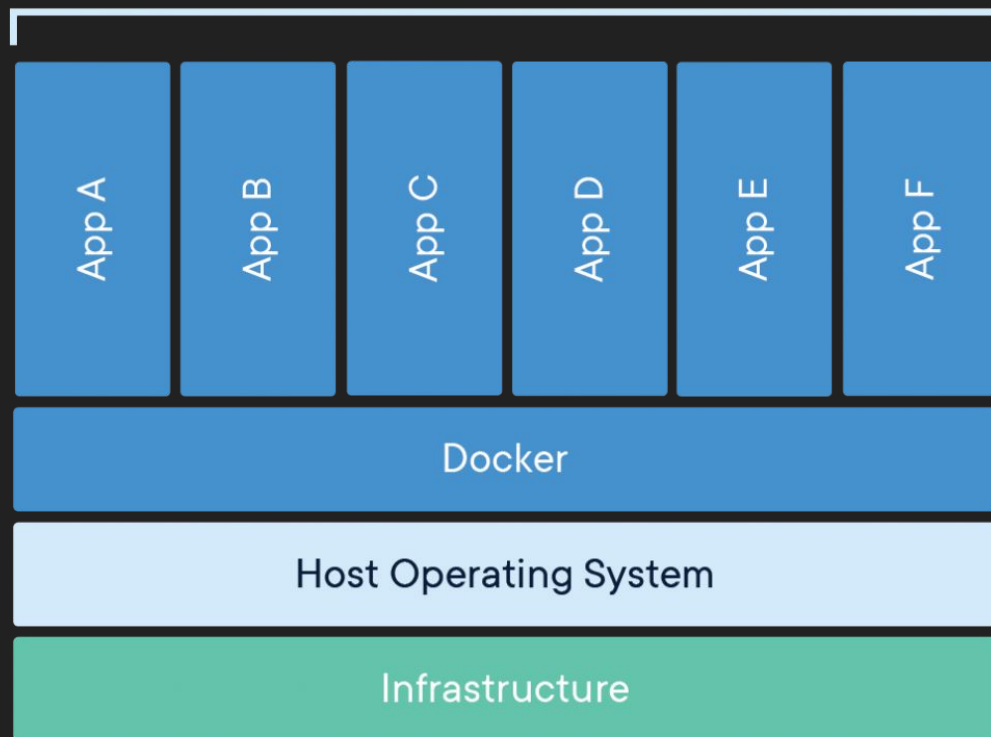
Для кандидата

# Docker... Зачем он мне вообще?

- Ускоряет развертывание тестировочной и отладочной сред
- Понадобится Вам на новой работе
- Может помочь Вам пройти собеседование

# Docker... Взгляд с высоты дрона

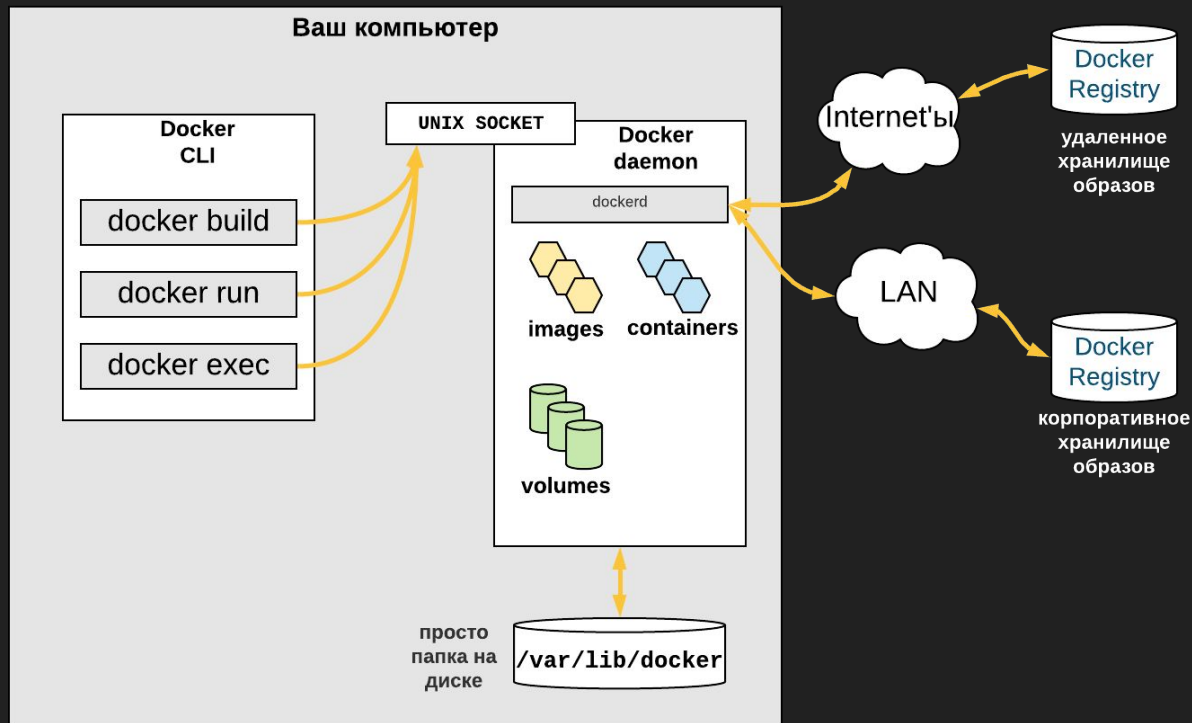
Containerized Applications



# Docker Namespaces

- pid - пространство идентификаторов процесса
- net - сетевой стек, ip netns для управления ими
- uts - информация о системе (hostname, etc.)
- ....

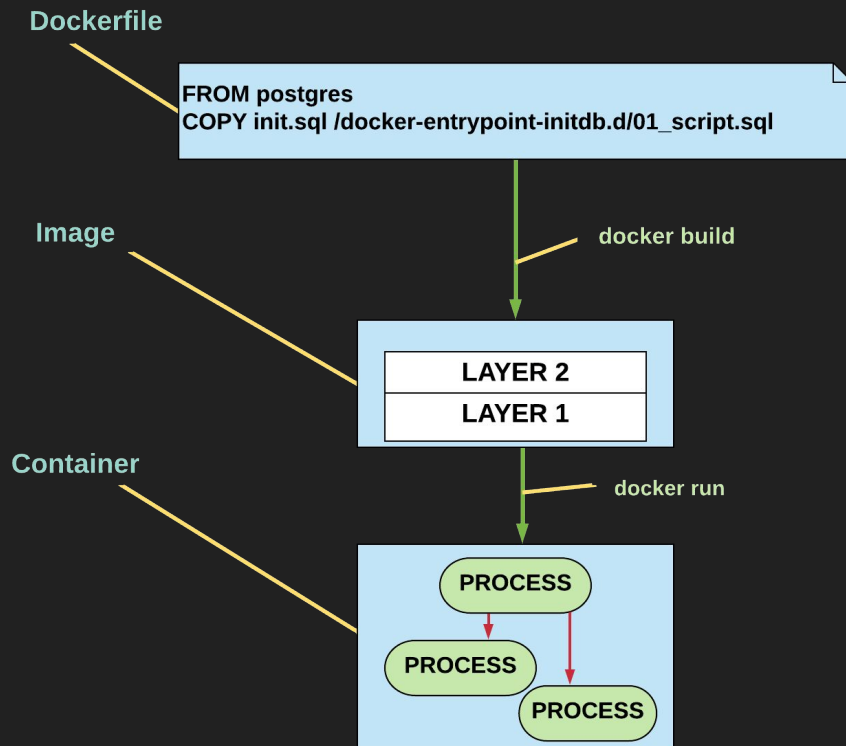
# Docker... Из чего состоит?



# Docker - сущности

- **Images** - это как исполняемый файл .exe. Лежит на диске и ничего не делает, пока не запущен
- **Containers** - это как файл .exe который уже запустили и он исполняется процессором в данный момент
- **Volumes** - как жесткий диск, который можно подключить

# Docker - сущности

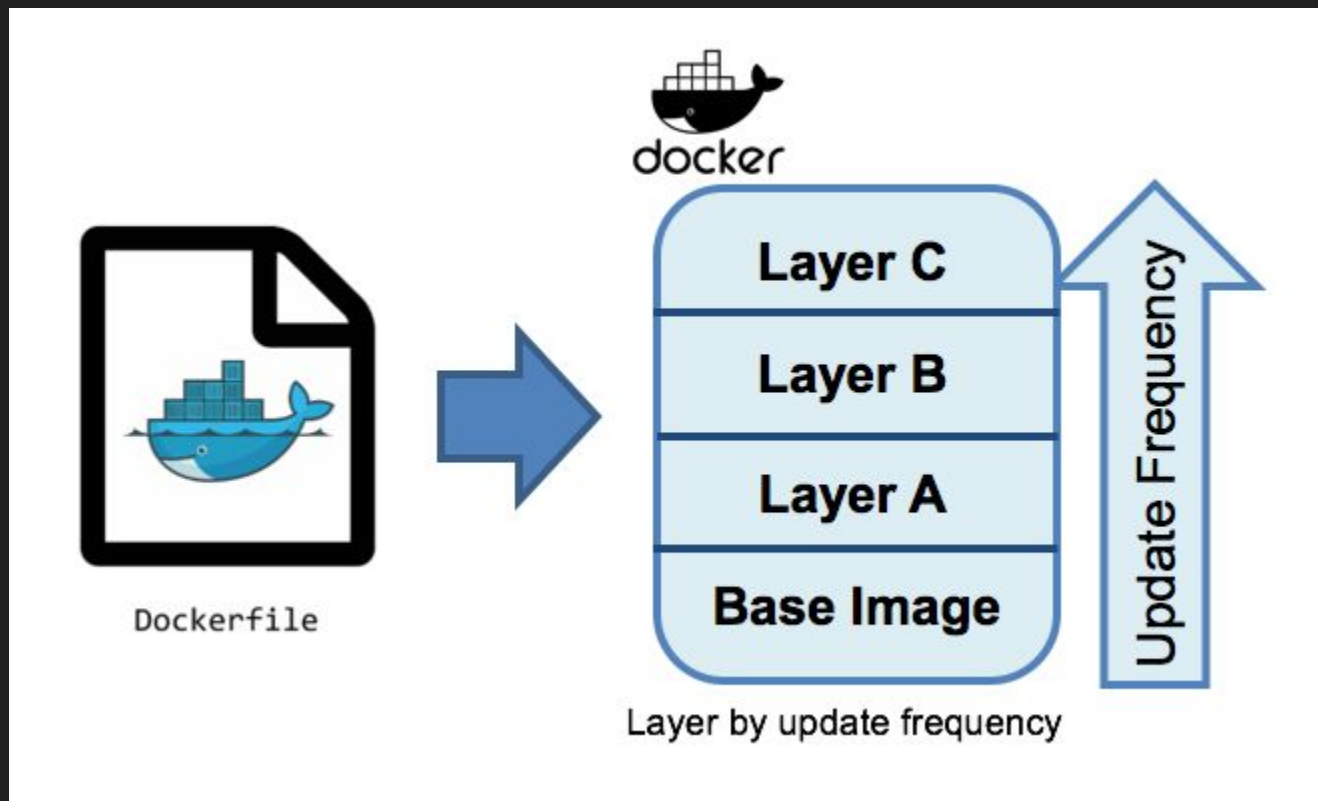


# Docker Image

- Скомпилировать image с тегом **myfirstimage**, информацию об image взять из текущей папки ( точка . в команде ниже это *текущая* папка) :
  - **docker build . -t myfirstimage**
- вывести на экран список скомпилированных образов
  - **docker image ls**



# Docker Image



# Docker Image Security

- Минимизация размера образа (alpine, etc.)
- Как можно меньше прав пользователю процесса
- Подписывать и верифицировать образы (Notary, etc.)
- Сканирование образов на безопасность (Snyk)
- Линтеры (Hadolint, etc.)
- Etc., etc.

# Docker Container

- `docker run -it myfirstimage` - запустить контейнер из image
- `docker ps` - посмотреть список запущенных контейнеров
- `docker exec -it CONTAINER_NAME bash` - зайти внутрь контейнера, как будто залогиниться на удаленный хост через `ssh/telnet`
- `docker stop CONTAINER_NAME` - остановить контейнер
- `docker start CONTAINER_NAME` - запуститься контейнер
- `docker diff CONTAINER_NAME` - посмотреть историю изменений в контейнере после его запуска

# Docker Inspect

docker inspect postgres

```
docker inspect postgres
[
  {
    "Id": "ebddaacfbd86ce07109173594839e67598dc7cf44c4a8095ce827d8501269407",
    "Created": "2020-08-15T19:36:03.6383807Z",
    "Path": "docker-entrypoint.sh",
    "Args": [
      "postgres"
    ],
    "State": {
      "Status": "running",
      "Running": true,
      "Paused": false,
      "Restarting": false,
      "OOMKilled": false,
      "Dead": false,
      "Pid": 3447,
      "ExitCode": 0,
      "Error": "",
      "StartedAt": "2020-08-15T19:36:05.1915947Z",
      "FinishedAt": "0001-01-01T00:00:00Z"
    },
    "Image": "sha256:3b339fc0fdc04bd35b7002e39858fbeeab2c4deb6558240f6f8c0d38be5debdb",
    "ResolvConfPath":
"/var/lib/docker/containers/ebddaacfbd86ce07109173594839e67598dc7cf44c4a8095ce827d8501269407/resolv.conf",
    "HostnamePath":
"/var/lib/docker/containers/ebddaacfbd86ce07109173594839e67598dc7cf44c4a8095ce827d8501269407/hostname",
    "HostsPath":
"/var/lib/docker/containers/ebddaacfbd86ce07109173594839e67598dc7cf44c4a8095ce827d8501269407/hosts",
    "LogPath":
"/var/lib/docker/containers/ebddaacfbd86ce07109173594839e67598dc7cf44c4a8095ce827d8501269407/ebddaacfbd86ce07109173594839e67598dc7cf44c4a8095ce827d8501269407-json.log",
    "Name": "/postgres",
  }
]
```

# Demo

# Docker Compose

- А теперь можно забыть все предыдущие слайды
- Docker Compose позволяет не выполнять docker команд
- Декларативное описание структуры container'ов и volume'ов в конфиг файле `docker-compose.yml`

# Docker Compose

Запустить docker-compose.yml конфиг:

```
$ docker-compose up -d --build
```

Остановить docker-compose.yml конфиг:

```
$ docker-compose down
```

```
version: '3.3'
```

```
services:
```

```
  web:
```

```
    image: nginx
```

```
    port:
```

```
      - 80:80
```

```
  app:
```

```
    image: my-app
```

```
    environment:
```

```
      - DB_HOST=db
```

```
  db:
```

```
    image: postgres
```

# “Бизнес требования”

Олег, привет.

Нужно разработать систему учета счетов пользователей.

Пополнение, списание. Ну ты понял ;)

И аналитику прикрутить! Пользователей - пара миллионов, не больше.

Можете начинать с ребятами разработку. Пишите на чем хотите.

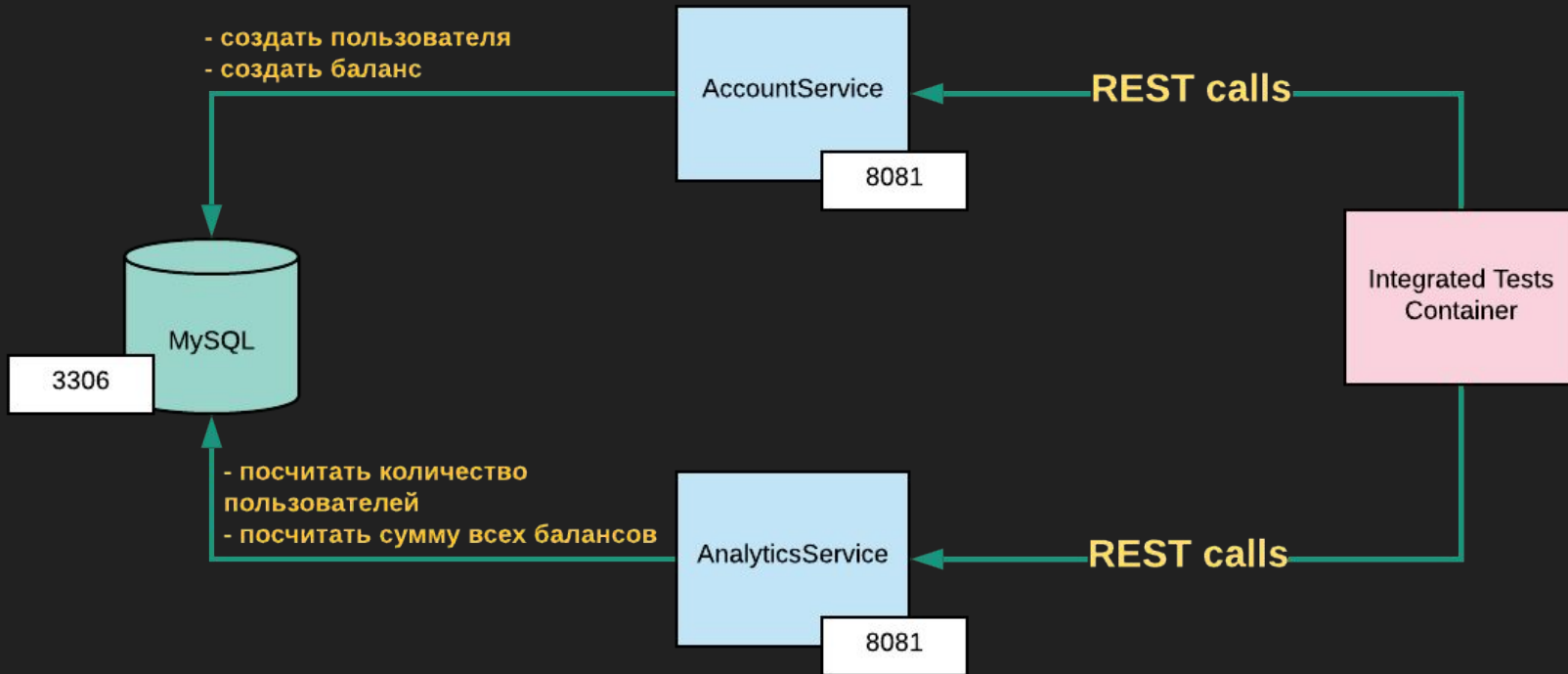
В конце следующей недели покажите, что получилось!

Yours, Sr. Vice President,

Sukhbir Dashikira



# Архитектура проекта



# Demo

# Docker. Вопросы на собеседовании

1. Простые
2. Посложнее
3. Для devops...

# Docker. Простые вопросы на собеседовании

- Как залогиниться в контейнер?
  - `docker exec -it postgres bash`

# Docker. Простые вопросы на собеседовании

- Как обеспечить автоматический перезапуск контейнера если он упал?

```
services:  
  
  web:  
  
    image: apache  
  
    restart: always
```

# Docker. Продвинутые вопросы на собеседовании

- Как связать контейнеры так, чтобы один запустился после другого

Возможный вариант решения:

1. `depends_on`
2. `health check`

```
depends_on:  
  db:  
    condition: service_healthy
```

```
healthcheck:  
  test: ["CMD", "mysqladmin", "ping", "-h", "localhost"]  
  timeout: 20s  
  retries: 10
```

# Docker. Продвинутые вопросы на собеседовании

- Как запустить более 1 демон процесса в контейнере?
  - supervisord

```
[supervisord]  
nodaemon=true
```

```
[program:h2server]  
command=/usr/lib/jvm/oracle-java8-jdk-amd64/bin/java  
-cp /opt/myapp.jar org.h2.tools.Console -baseDir  
/opt/openremote_resources/database -web  
autorestart=true
```

```
[program:controller]  
command=/docker/startup.sh  
priority=900  
autorestart=true
```

# Docker. Тактика на собеседовании. Правило **ОПС!**

1. **О**предели задачи команды. Текущие или будущие.
2. **П**одумай, где твои навыки могут быть применены к задачам из пункта 1
3. **С**кажи об этом!



# Docker. Тактика на собеседовании. Правило **ОПС!**

- **О**предели задачи команды где можно применить docker
  - *“Расскажите пожалуйста о том как у вас устроен CI/CD”*
  - *“Расскажите о том как вы тестируете ваши сервисы”*
- **П**одумай, где твои навыки могут быть применены
- **С**кажи об этом!
  - *“Я докеризовал python, spring и PostgreSQL на предыдущем месте работы и настроил среду разработки и тестирования для своих коллег”*

# Что дальше?

- Для углубления в докер:

- <https://www.slideshare.net/kerneltlv/namespaces-and-cgroups-the-basis-of-linux-containers>

- По всем докер вопросам:

- Java: [https://t.me/ihunt\\_io\\_java](https://t.me/ihunt_io_java)
- Python: [https://t.me/ihunt\\_io\\_python](https://t.me/ihunt_io_python)
- QA: [https://t.me/ihunt\\_io\\_qa](https://t.me/ihunt_io_qa)

# Спасибо!