

Новый фронтенд в Одноклассниках

Священный грааль: React в Java

Олег Коровин

ok.ru/madcow

vk.com/madcow



Зачем этот доклад?

- Показать переход от старого стека к новому с GraalVM
- Продемонстрировать путь миграции
- Мотивировать разработчиков

Часть 1

Предыстория

A long time ago
in a galaxy far, far away....

2006



[Моя страница](#) | [Мои сообщения](#) | [Поиск людей](#) | [Сейчас на сайте](#) | [Помощь](#) | [Выход](#)

[Основное](#) | [Друзья](#) | [Друзья друзей](#) | [Фотографии](#) | [Фотоальбомы](#) | [Группы](#) | [Сообщества](#) | [Форум](#) | [О себе](#)



Ольга Вайнер
Последний визит: 12.08.2006
👤 **20 лет**, Россия, Москва

 [Написать сообщение](#)
 [Показать переписку](#)
 [Добавить в список друзей](#)
 [Показать общих друзей](#)
 [Добавить в фавориты](#)
 [Пригласить в группу](#)
 [Сделать подарок](#)
 [Пожаловаться на пользователя](#)

 6  1  5+
[Комментарии \(0\)](#)
[Личные фото \(2\)](#)

Лента активности

Нет записей.

Здесь показываются публичные действия пользователя на сайте.
Например: добавление новых друзей, фотографий и фотоальбомов, сообщений в форумах,
комментариев к фотографиям, а также присоединение к группам и сообществам.

Друзья

1 000 000 

Пользователей



www.one.ee
www.one.lv

[latviski](#)
[no-ryccia](#)
[english](#)

tikai savējiem!

Login:

Parole:

[Aizmirsi paroli?](#)

sāc lietot!

Login:

@one.lv

Telefons:

[Pārbaudi! Nosūti ziņu...](#)

MOBILAIS PASTS

Sasniedz draugus un ģimeni dažās sekundēs vienmēr un visur ar SMS palīdzību.

MOBILAIS E-PASTS

Bieži esi prom no datora? Saņem SMS paziņojumus katru reizi, kad pienāk steidzams e-pasts.

MOBILIE KONTAKTI

Glabā visus kontaktus vienā vietā un pievieno tos mobilajam pastam vai e-pastam ar vienu klikšķi.

MOBILAIS KALENĀRS

Plāno savu laiku un nekavē sapulces un nozīmīgus notikumus ar SMS atgādinājumu palīdzību.

MOBILĀS PIEZĪMES

Nepieciešams paņemt svarīgu informāciju līdzi? Pārsūti piezīmes tieši uz mobilo telefonu.

LOGO

	54
	133
	115
	71
	540

[vēl >>>](#)

MELODIJAS

Doom	1227
James Bond 2	680
Katiusha	627
Tatu: Robot	1135
Ja na solishke L	486

[vēl >>>](#)

© 1999-2002 ONE Ltd. All rights reserved.

[Pirmo reizi?](#) | [Lietošanas noteikumi](#)

Jautājumi un komentāri: one@one.lv



Время шло



Олег Коровин

Лента

Друзья 131

Фото 12

Группы 13

Игры 9

Подарки

Товары

Заметки 5

Ещё

Платежи и подписки

- Подписки на видео
- История покупок в ОК
- История денежных переводов
- Настройки платежей

На счёте: 29 ОК

Пополнить счёт

Активировать сертификат

Сбербанк Онлайн

ОКи за СПАСИБО от Сбербанка

Служба поддержки

Денежные переводы

Переводите деньги друзьям и близким на карты Visa, MasterCard и Maestro! Это быстро, просто и безопасно. Переводы осуществляются в 18 стран мира

Отправить перевод

Запросить деньги

Подписки на платные функции

5+

Оценка 5+

Вы сможете ставить на все понравившиеся вам фото самую высокую оценку — пять с плюсом!

Купить

Невидимка

Вы станете невидимкой для других, никто не узнает, что вы были у них в гостях.

Купить

Смайлики и стикеры

Возможность использовать любые смайлики на сайте.

Купить

Музыка без ограничений

Никакой аудиорекламы. Получи доступ к полной коллекции музыки.

Купить

Всё включено

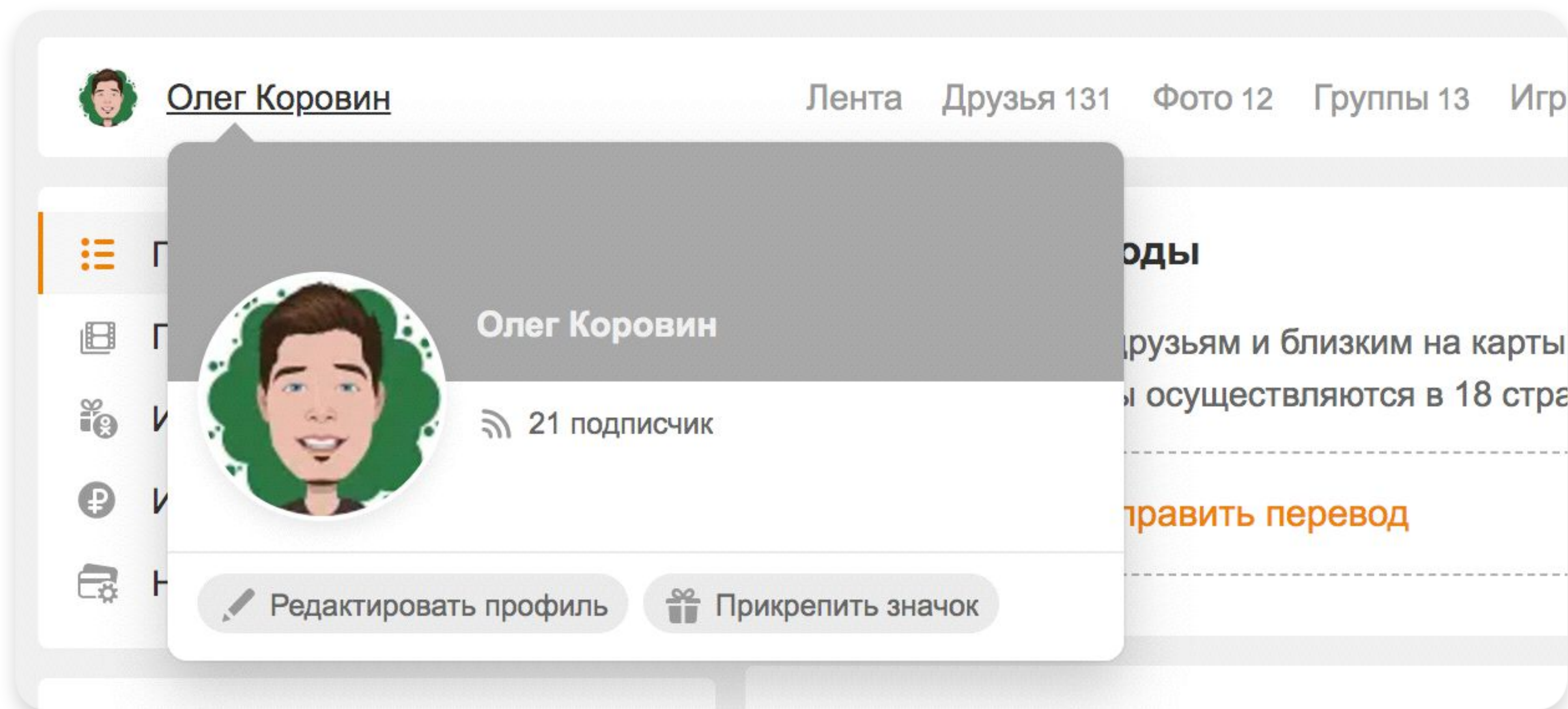
Возможность дарить подарки (кроме премиум и музыкальных) своим друзьям и близким совершенно бесплатно!

Купить

Фоторамки

Вы сможете украшать свое главное фото неограниченное количество раз любыми рамками и украшениями совершенно бесплатно! (кроме VIP-рамок)

Купить



2018



71 млн
пользователей



7 тыс
машин



600 тыс/сек
запросов



На клиенте

- GWT
- jQuery
- DotJs
- RequireJs
- ...



Проблемы

- Старые библиотеки
- Нет единого фреймворка
- Нет изоморфности
- Нет единого структурированного приложения на клиенте
- Плохая отзывчивость
- Недостаточный инструментарий
- Большой порог входа
- ...

Требования к решению

- Изоморфный код для UI
- Плавный переход (миграция)
- Серверный рендеринг
- У нас не должно быть усложнения эксплуатации

Почему серверный рендеринг?

Наш любимый пользователь

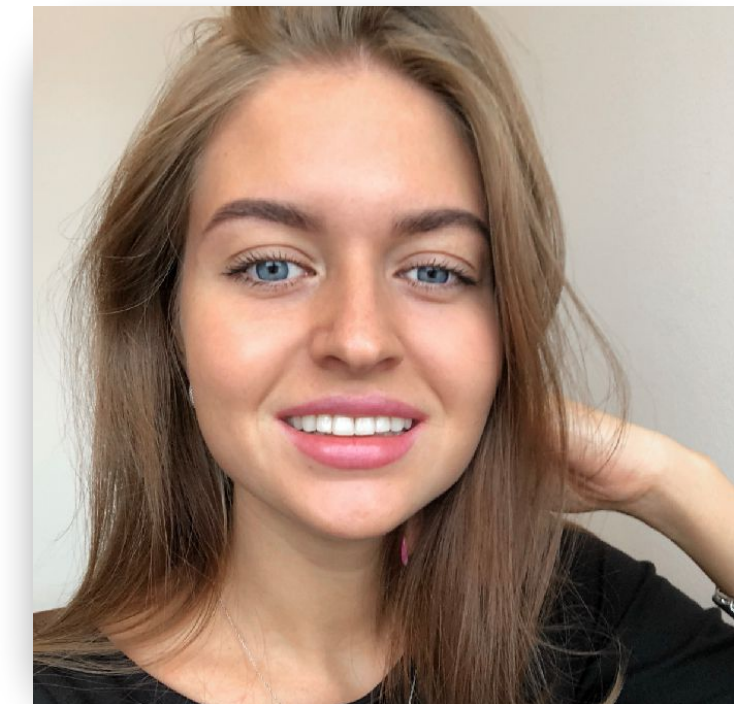


В деревне Кужепятовка

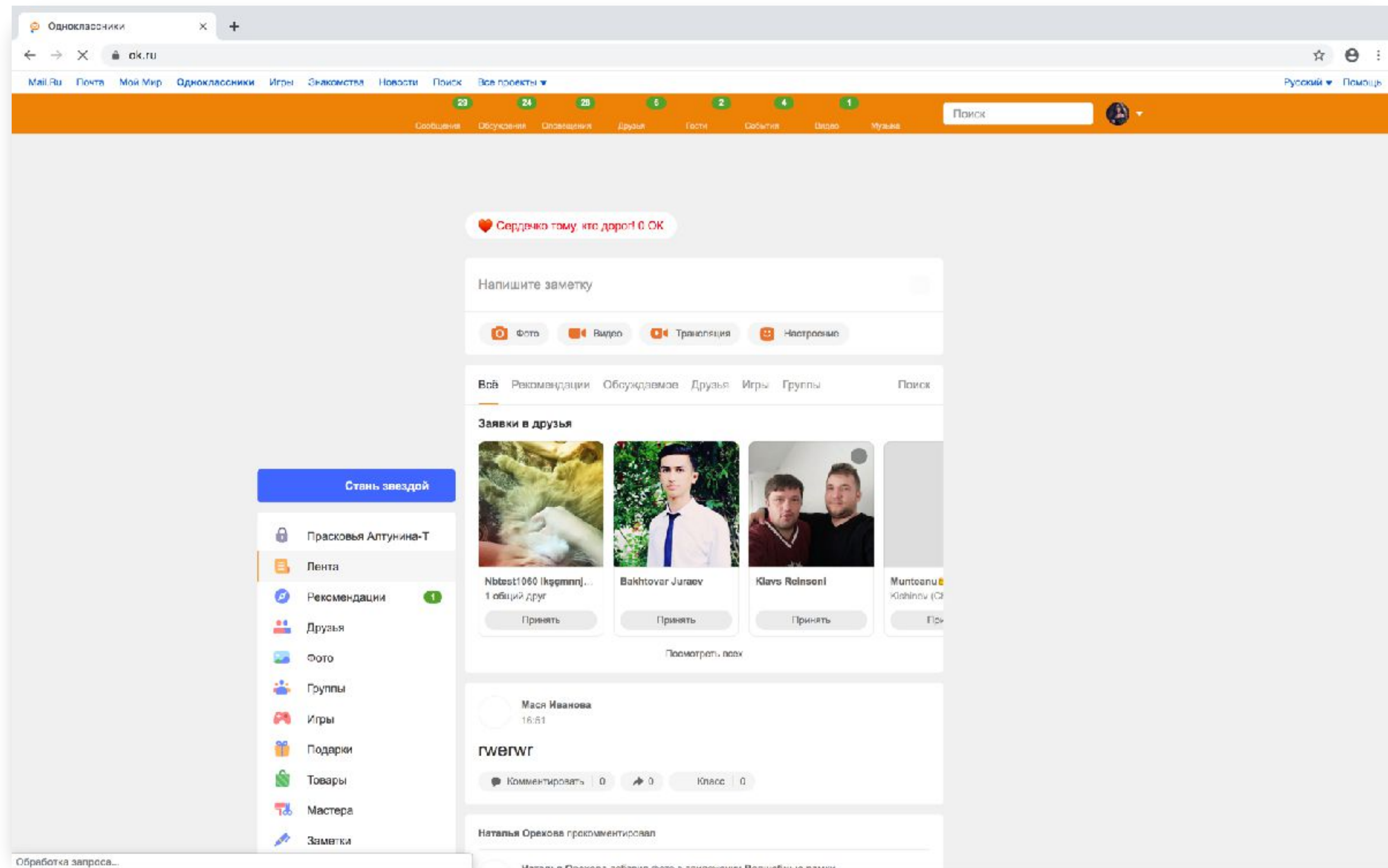
~~4G~~



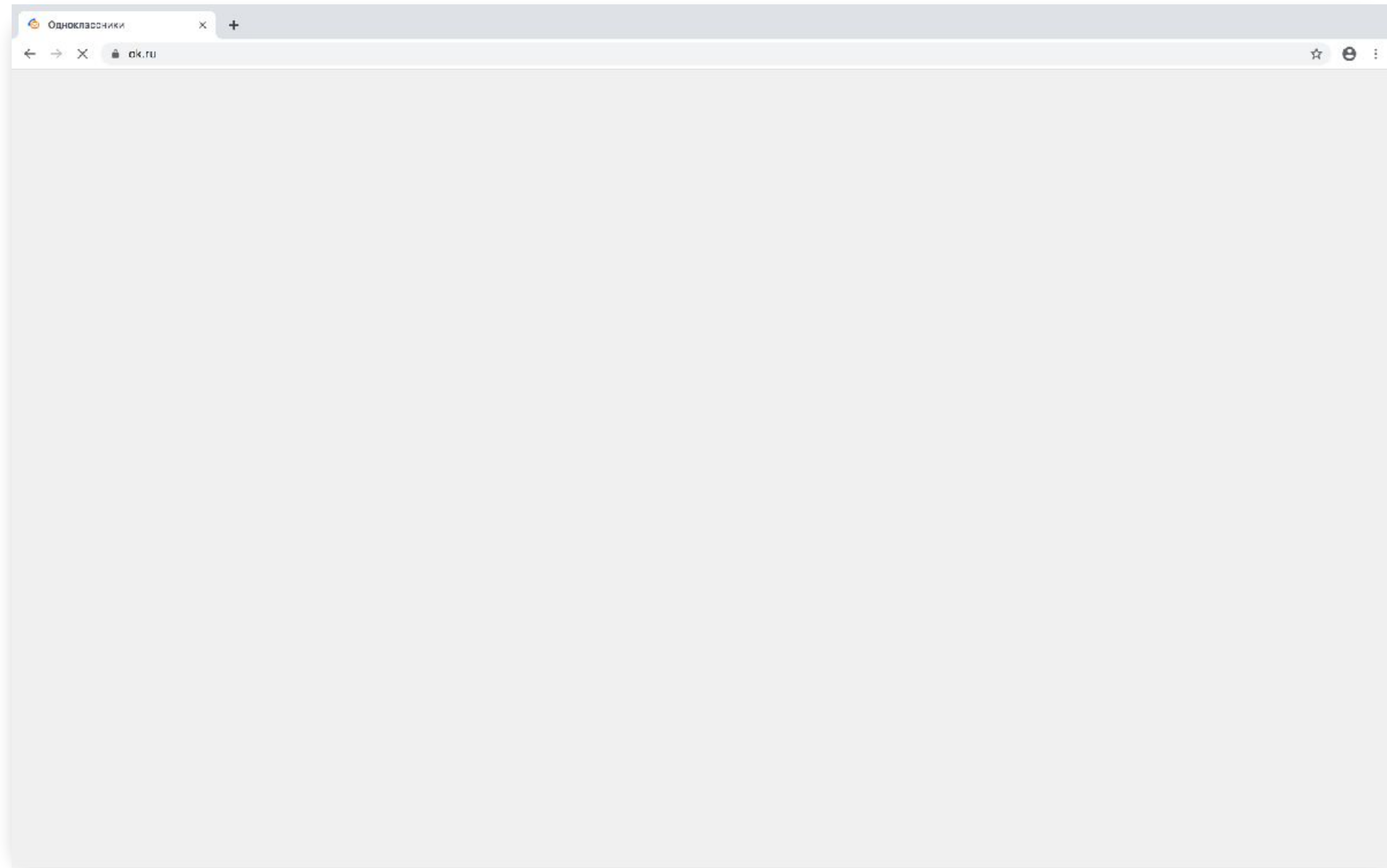
100500км от МКАД



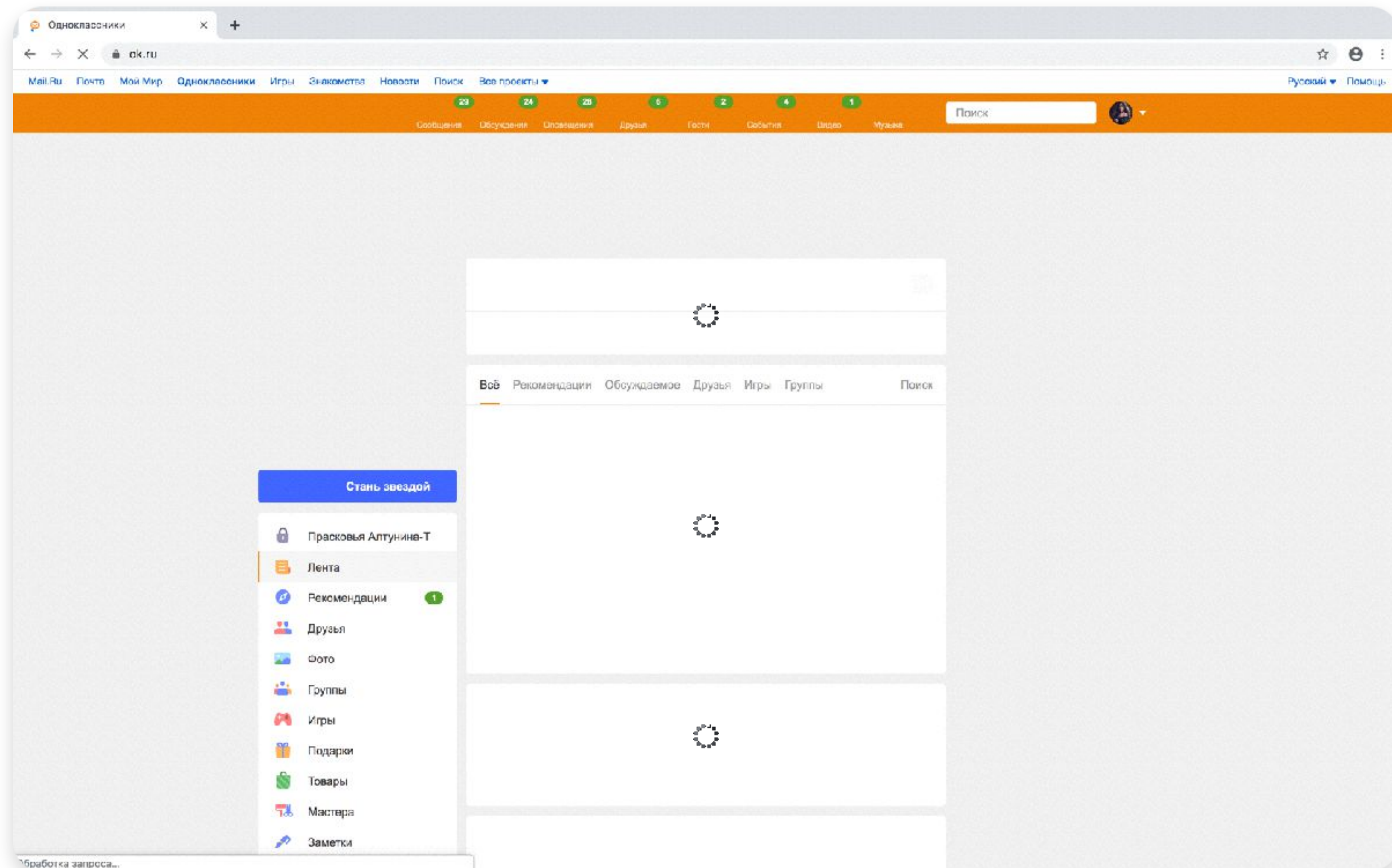
Если будет серверный рендеринг



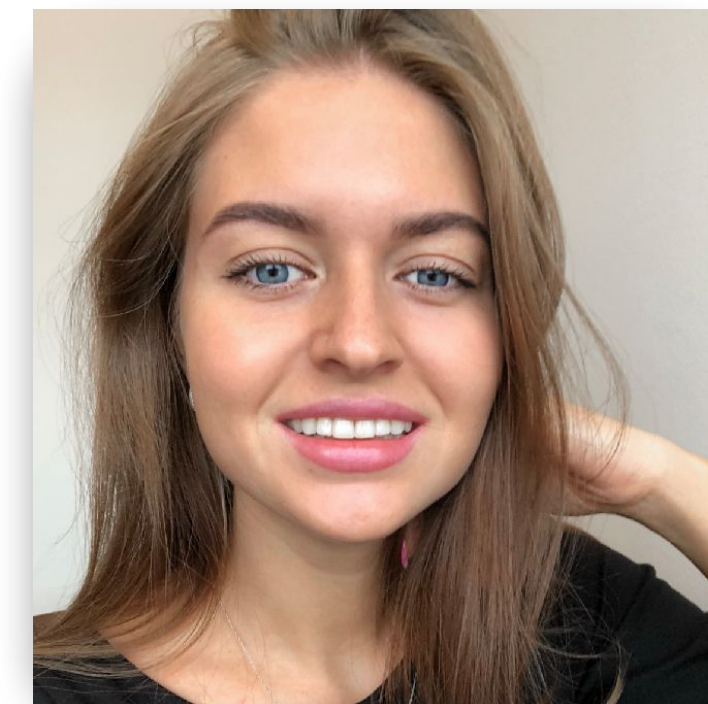
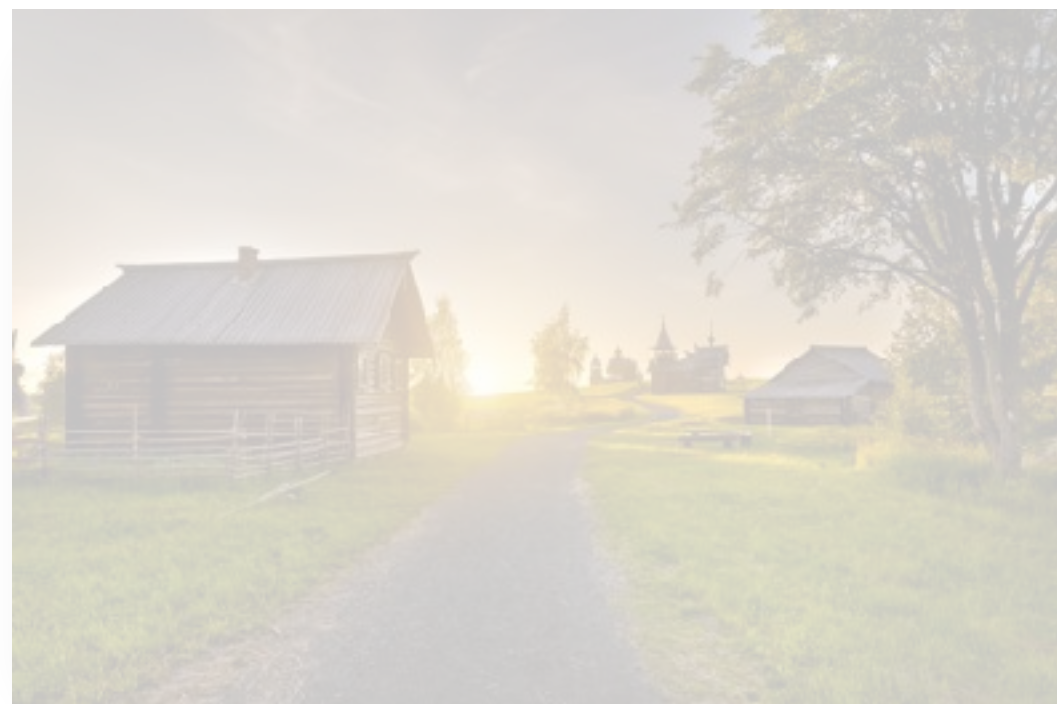
Если не будет серверного рендеринга



Если не будет серверного рендеринга



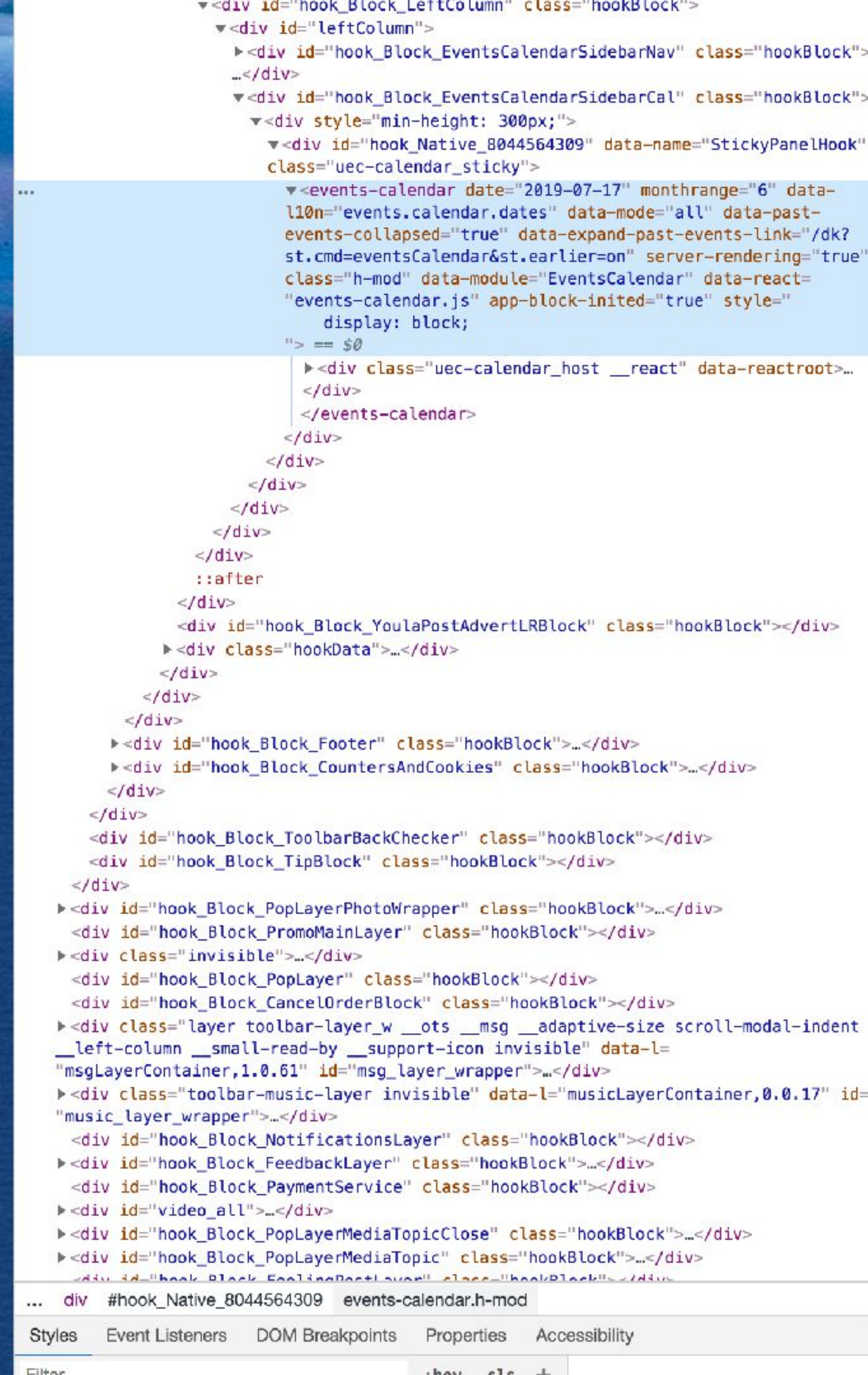
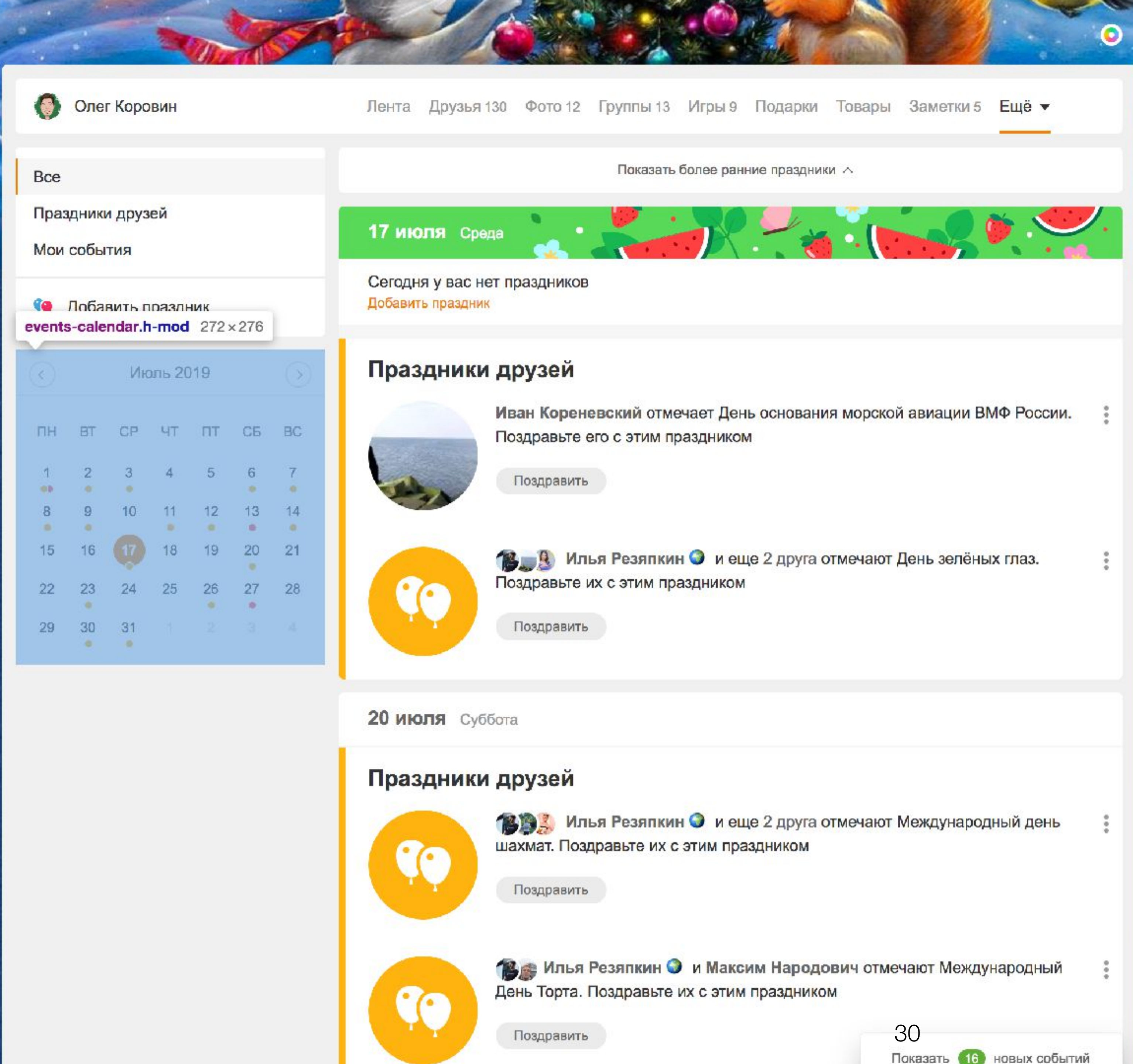
Пользователи в печали



Часть 2

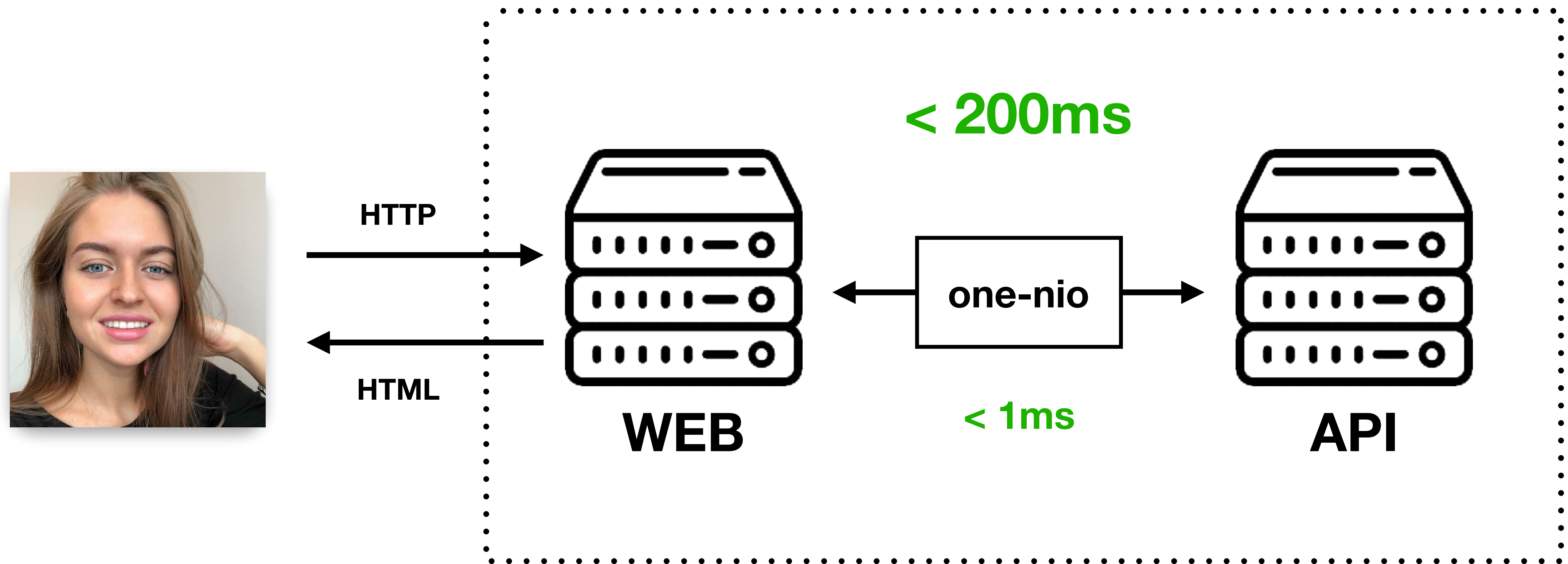
Поиск решения

Концепция решения - виджеты



Как работает сервер сейчас

Как работает сервер сейчас



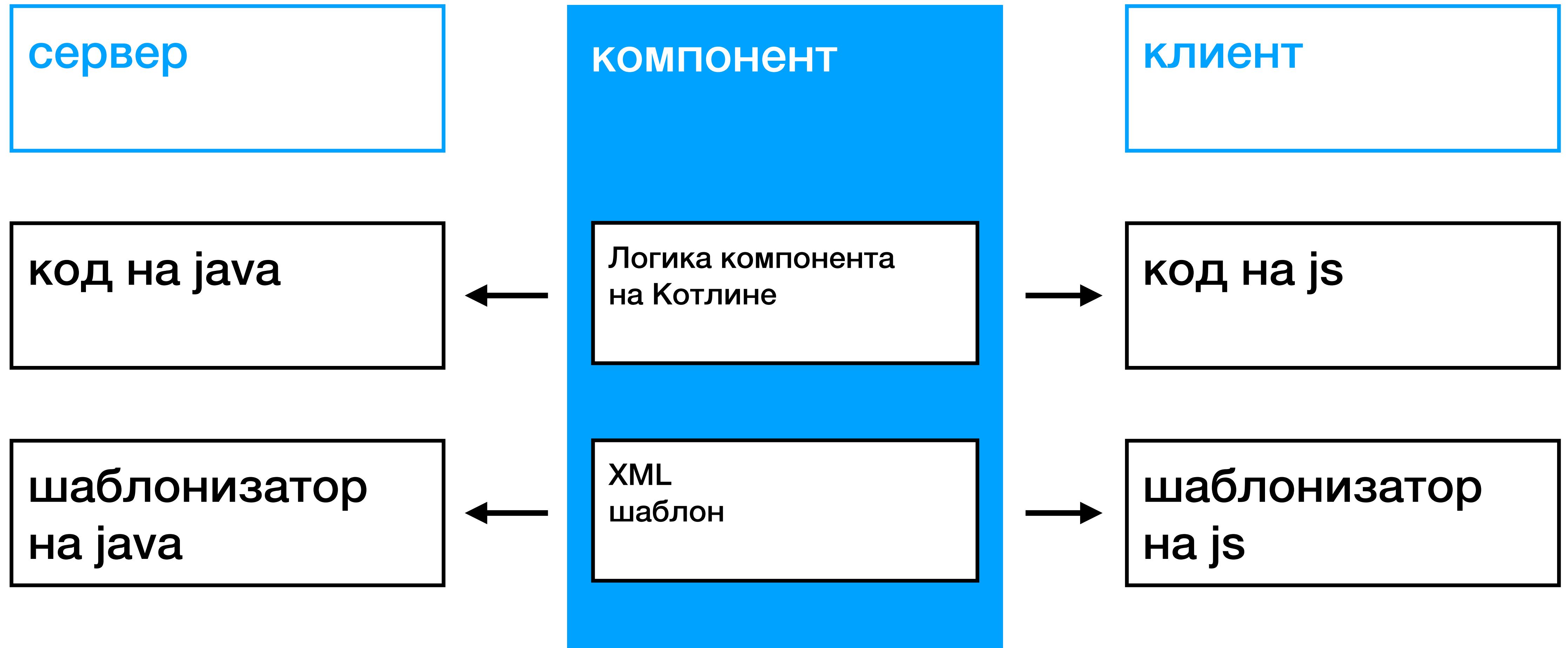
one-nio - бинарный протокол
<https://github.com/odnoklassniki/one-nio>

Что делать?





Kotlin - концепт



Почему не Kotlin

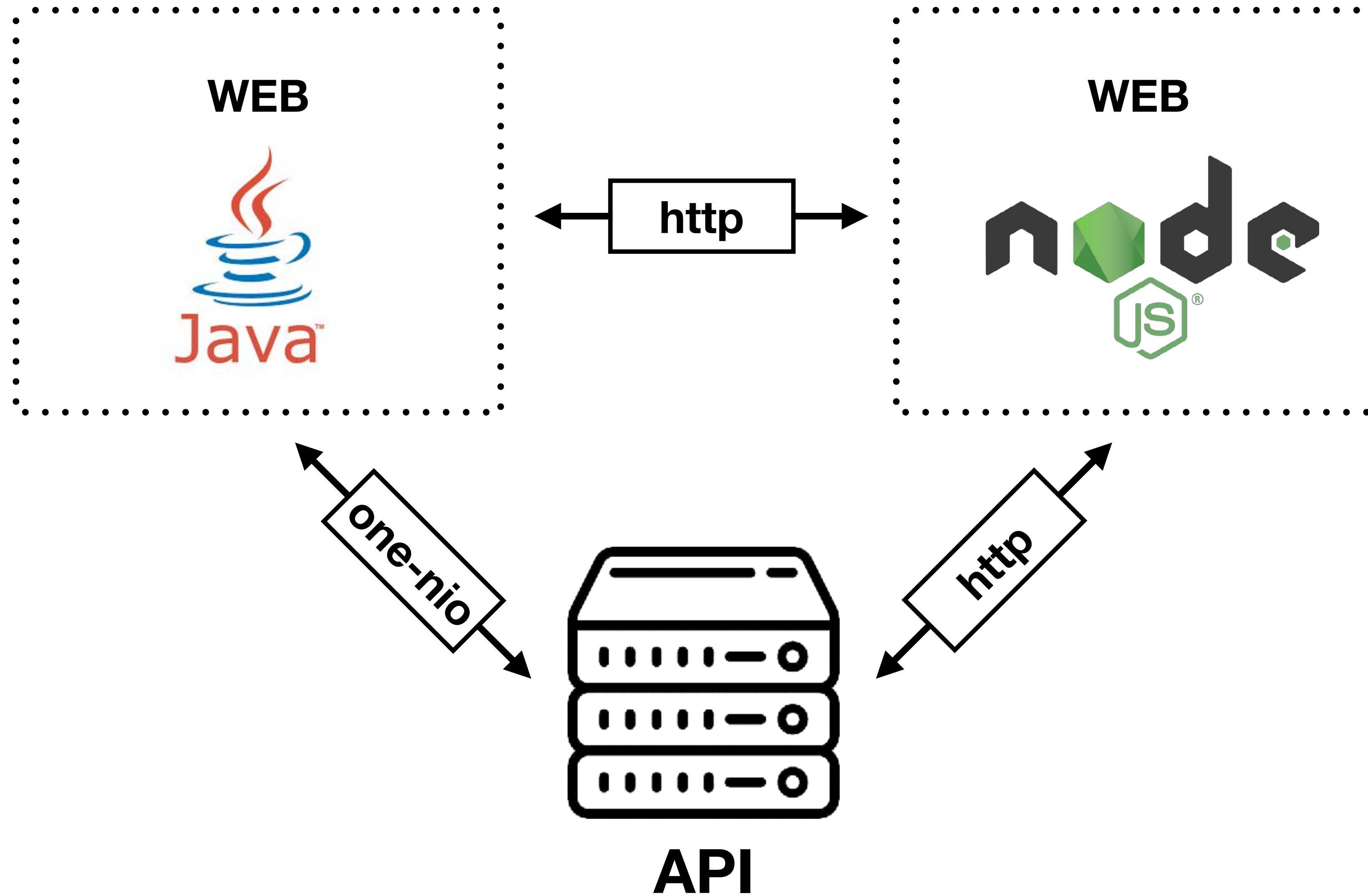
- еще 1 язык в системе
- примитивный компилятор в JS
- поддержка 2х рантаймов
- высокий порог входа
- возможные сложности с наймом людей







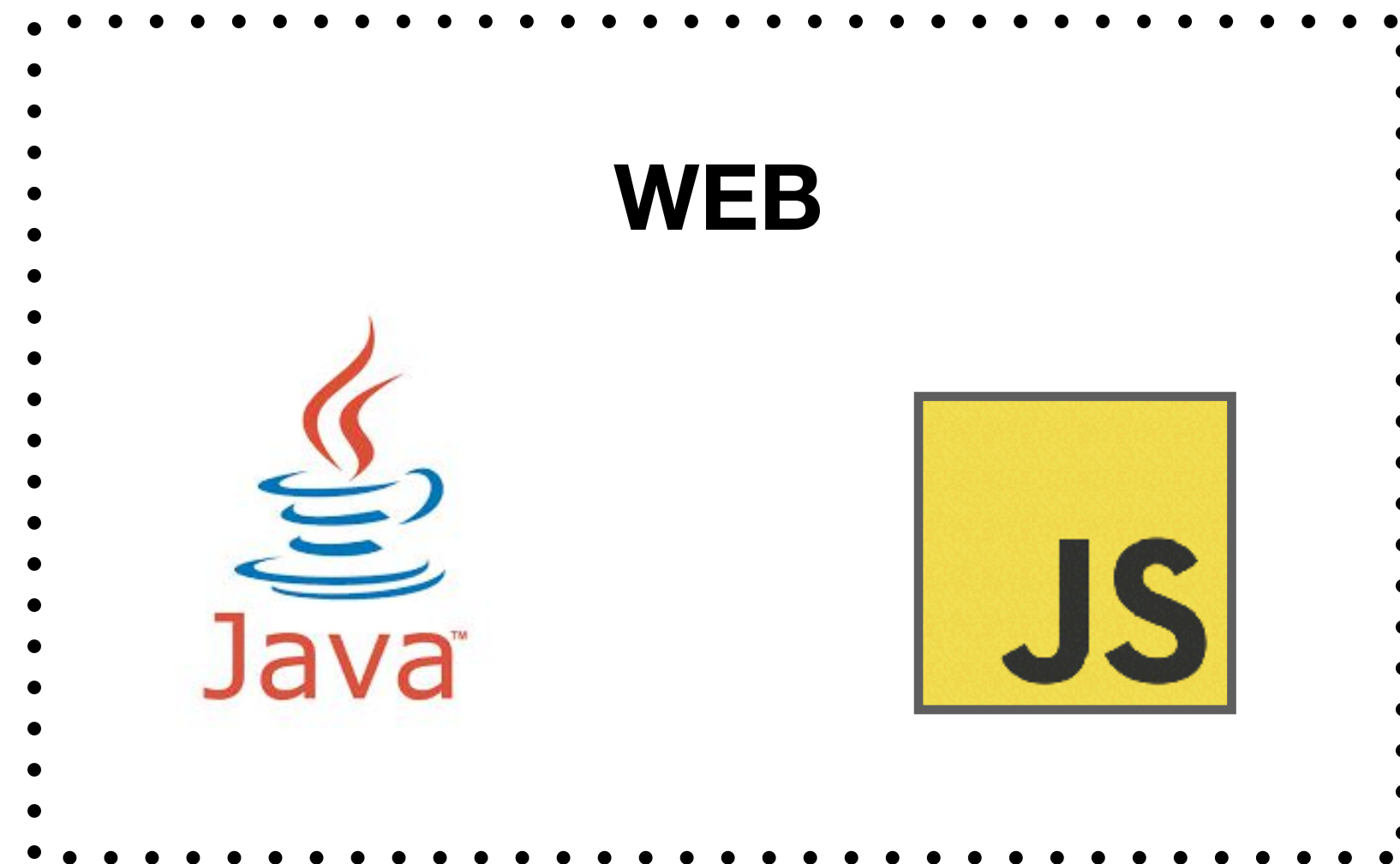
NodeJS



Почему не NodeJS

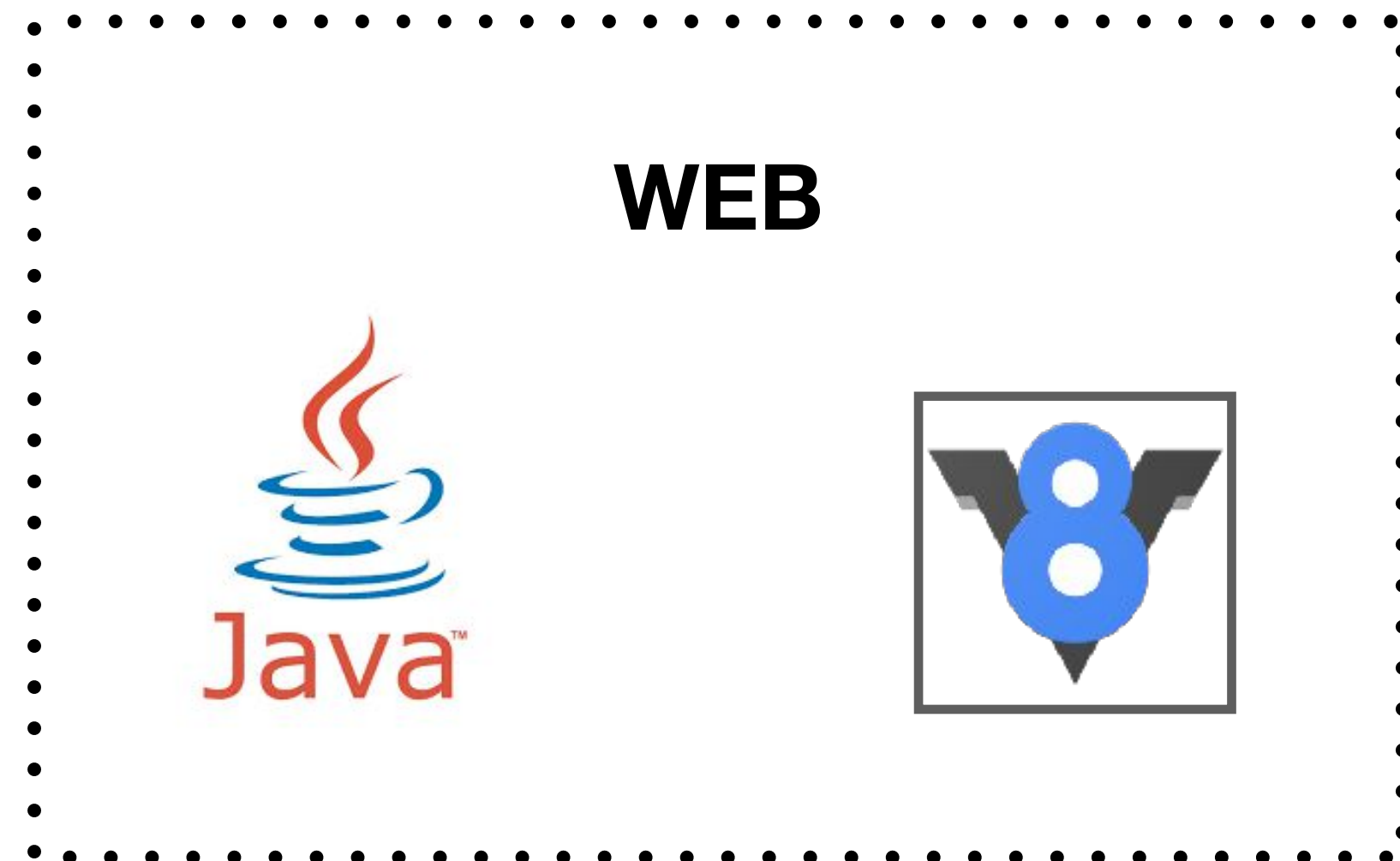
- HTTP транспорт - медленно
- Сериализация/десериализация - доп нагрузка, задержка, сложность
- Еще один компонент - риски эксплуатации

А можно ли запустить JS в JVM?



- Простая инфраструктура
- Нет накладных расходов

Почему не V8



- Сложность эксплуатации



JS рантаймы в JVM

Nashorn

Rhino



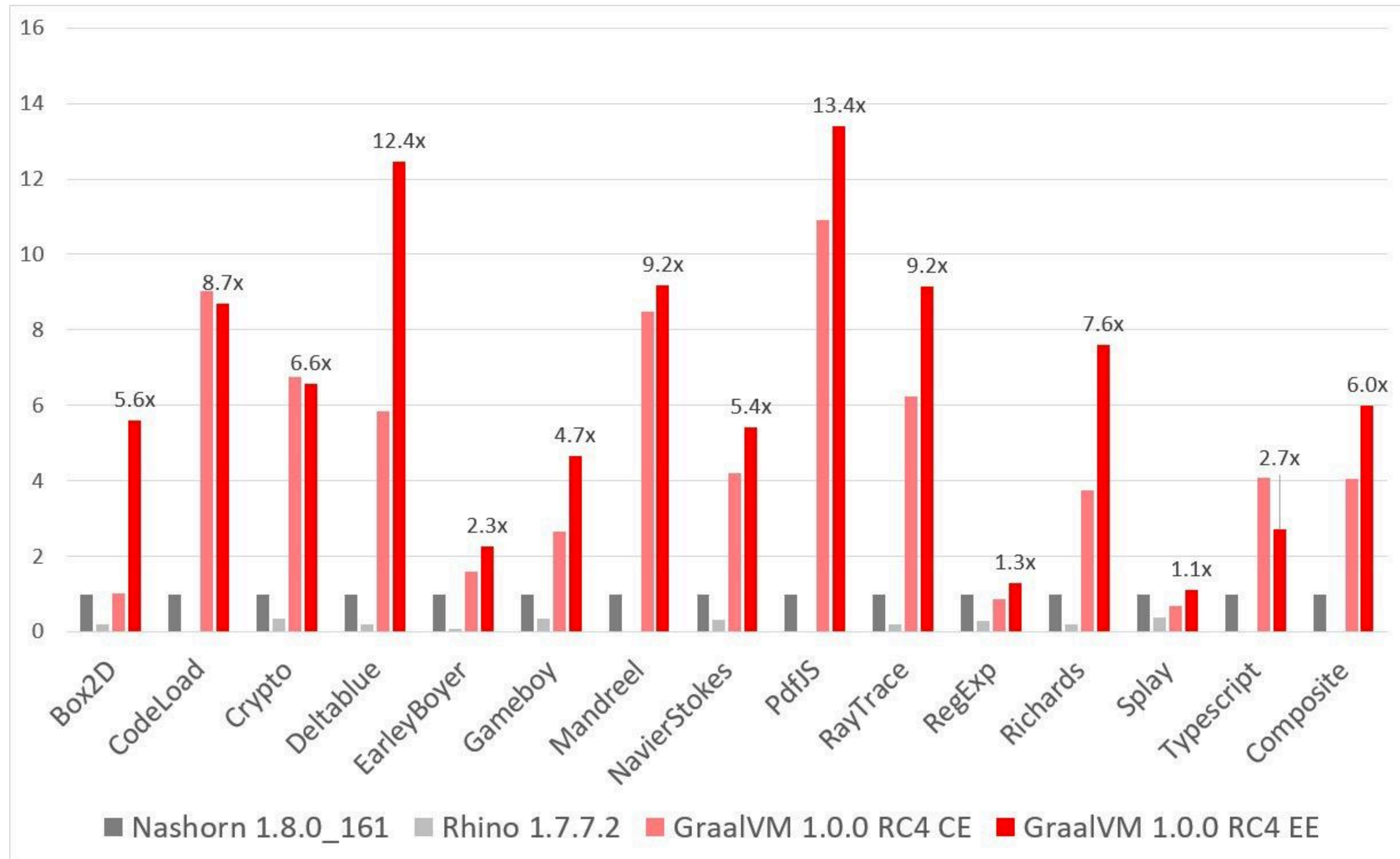
- Всё на JVM
- Java-JS взаимодействие
- Безопасный рантайм
- Компилятор на Java

GraalVM

GraalVM™



Nashorn, Rhino, GraalVM



GraalVM™

<https://www.graalvm.org>

 Kotlin
 Scala

 Java™

 JS

 Ruby

 R

 python™

 C
 C++


Automatic transformation of interpreters to compilers

GraalVM™

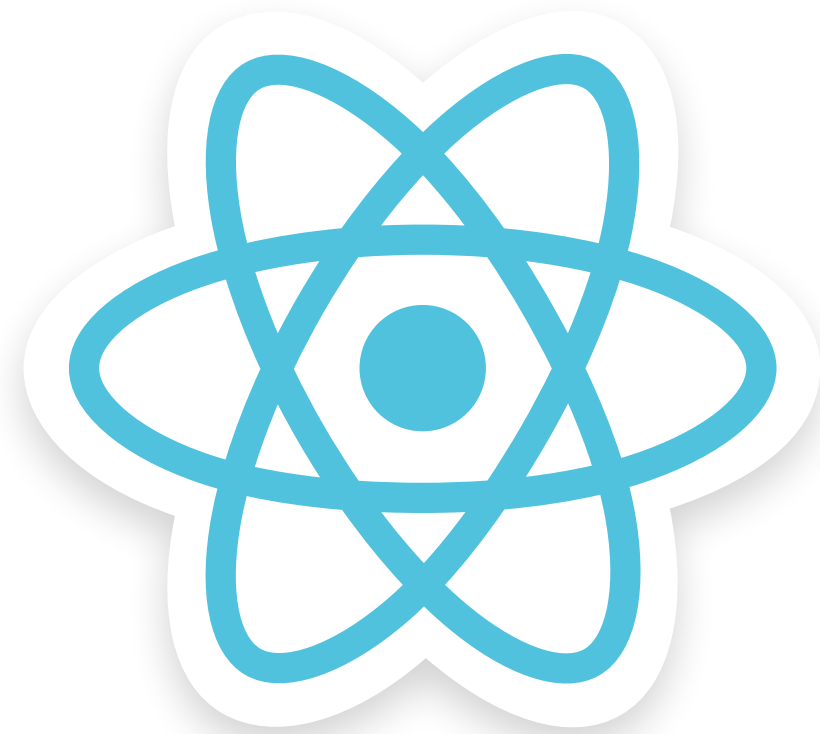
Embeddable in native or managed applications

 Java™

 node
JS

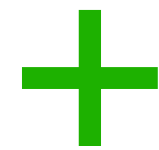
 Database

standalone

<https://reactjs.org/>

GraalVM + React



- Java + JS
- Сообщество
- Низкий порог входа
- Эксплуатация не усложнилась
- Найм сотрудников



- Нет поддержки типов в JS
- Риски эксплуатации GraalVM

Часть 3

Запуск React в GraalVM

Как GraalVM работает с JS

JS Context

JAVA

JS Context

global

```
Context context = Context.create("js");
```

```
//получаем global данного контекста  
Value js = context.getBindings("js");
```

JS Context

JAVA

JS Context

global

eval()

load()

app.js

```
Context context = Context.create("js");
```

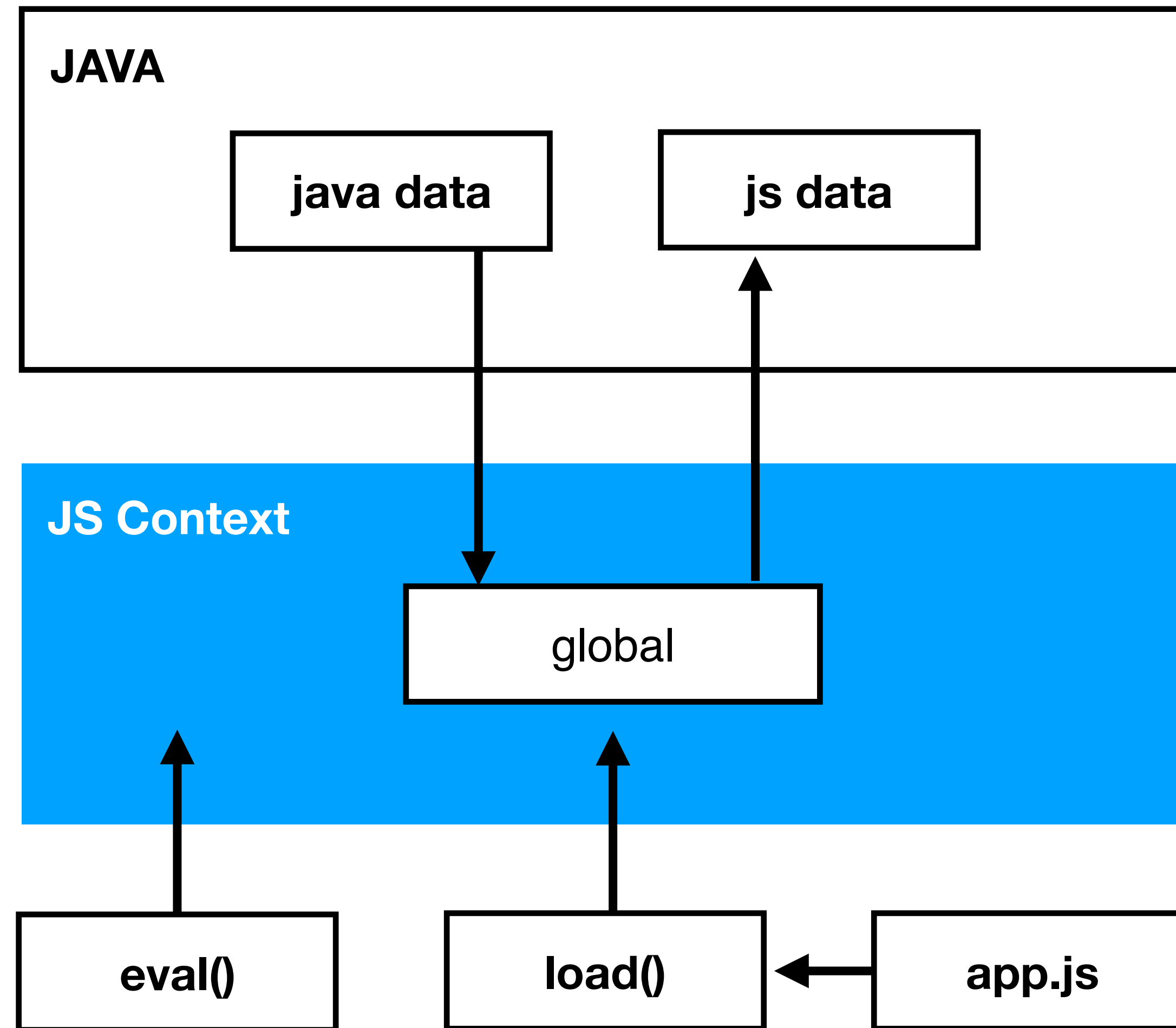
```
//получаем global данного контекста  
Value js = context.getBindings("js");
```

```
//можем читать из global  
Value load = js.getMember("load");
```

```
//можем вызвать функцию из контекста  
load.execute(pathToModule);
```

```
//можем выполнить код в данном контексте  
context.eval("js", someCode);
```

JS Context



```
Context context = Context.create("js");

//получаем global данного контекста
Value js = context.getBindings("js");

//можем читать из global
Value load = js.getMember("load");

//вызываем функцию из контекста
load.execute(pathToModule);

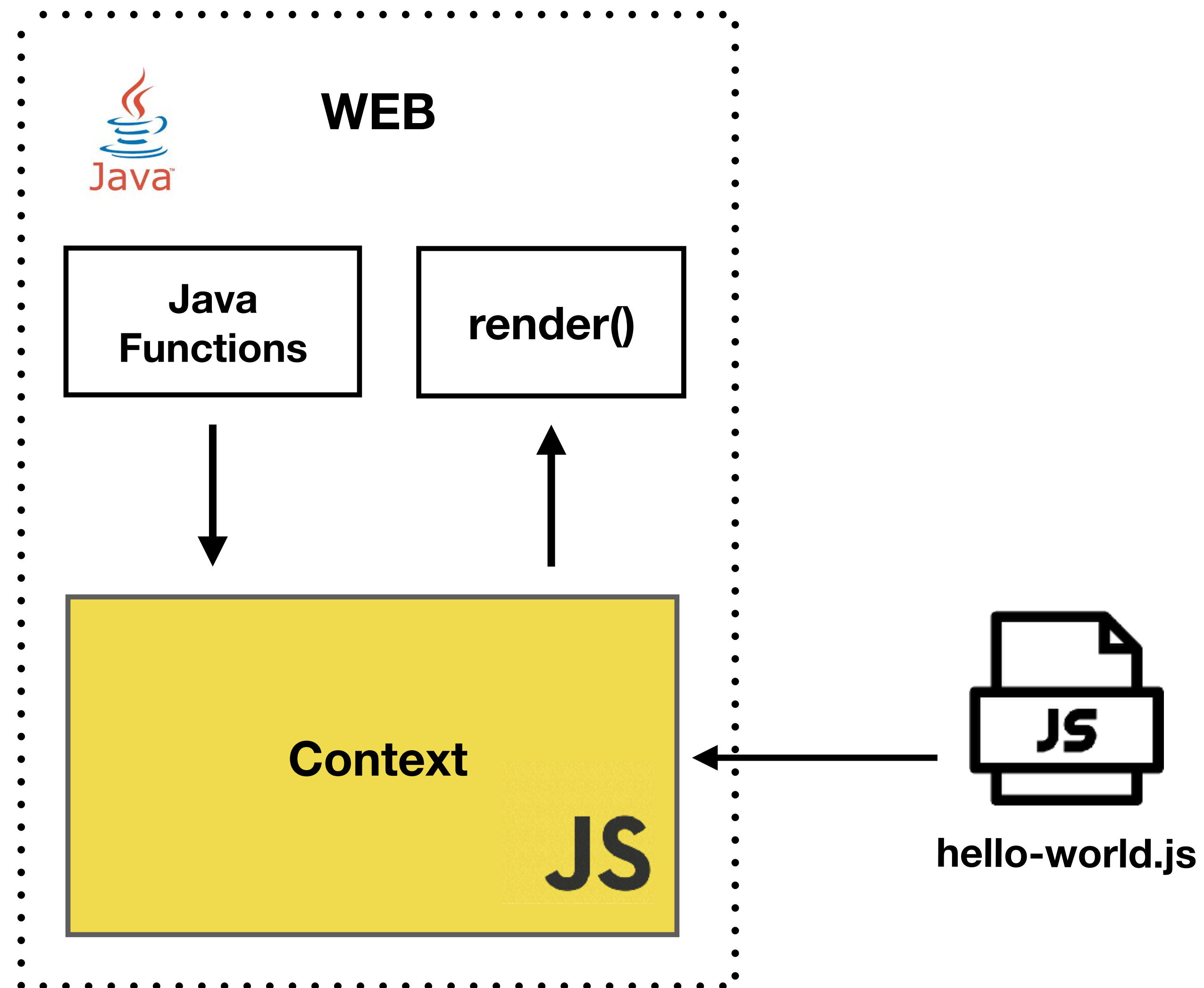
//можем выполнить код в данном контексте
context.eval("js", someCode);

//можем записать в global
js.putMember("serverProxy", serverProxy);

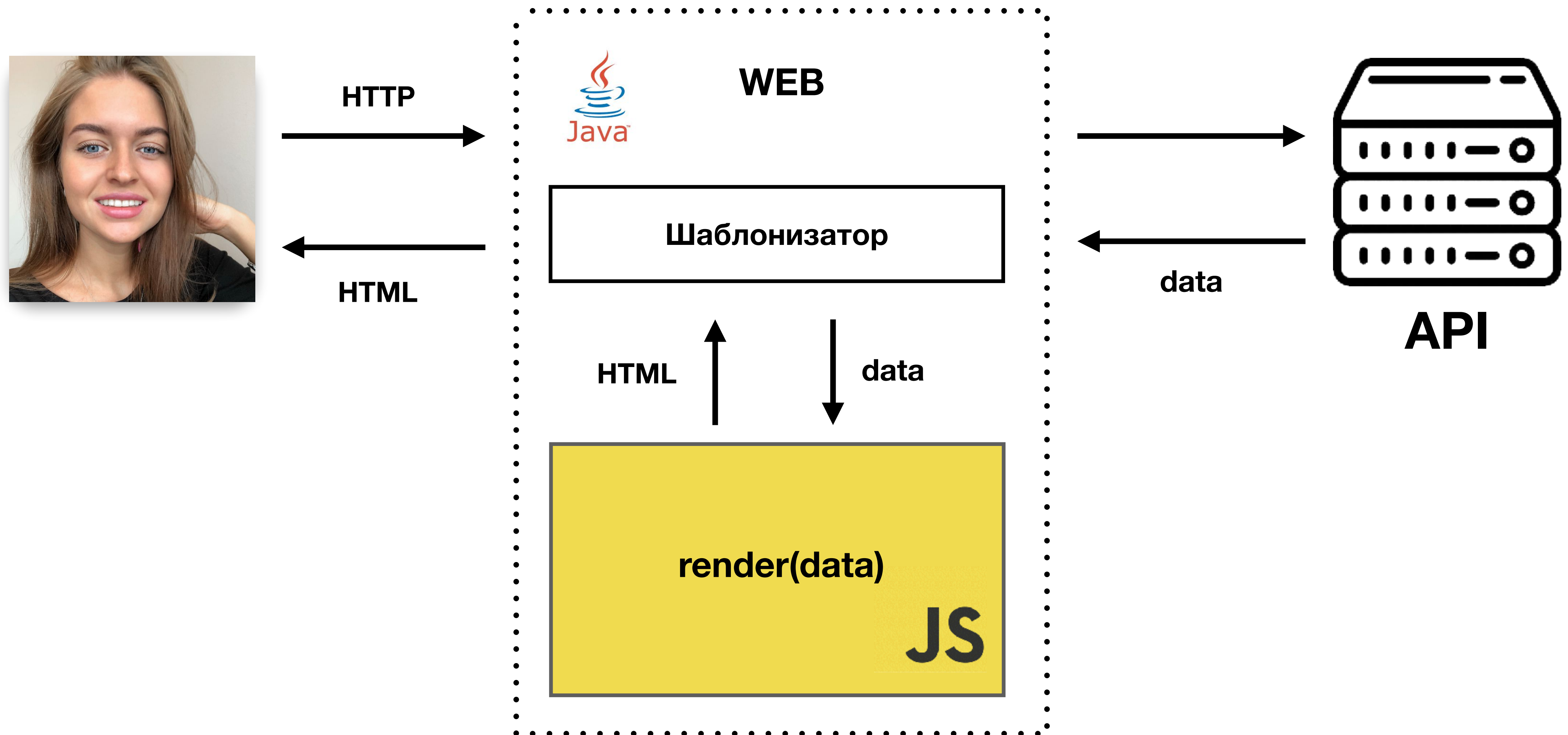
//можем читать из global
Value app = js.getMember("app");
```

Серверный рендеринг JS - концепт

Серверный рендеринг JS - концепт

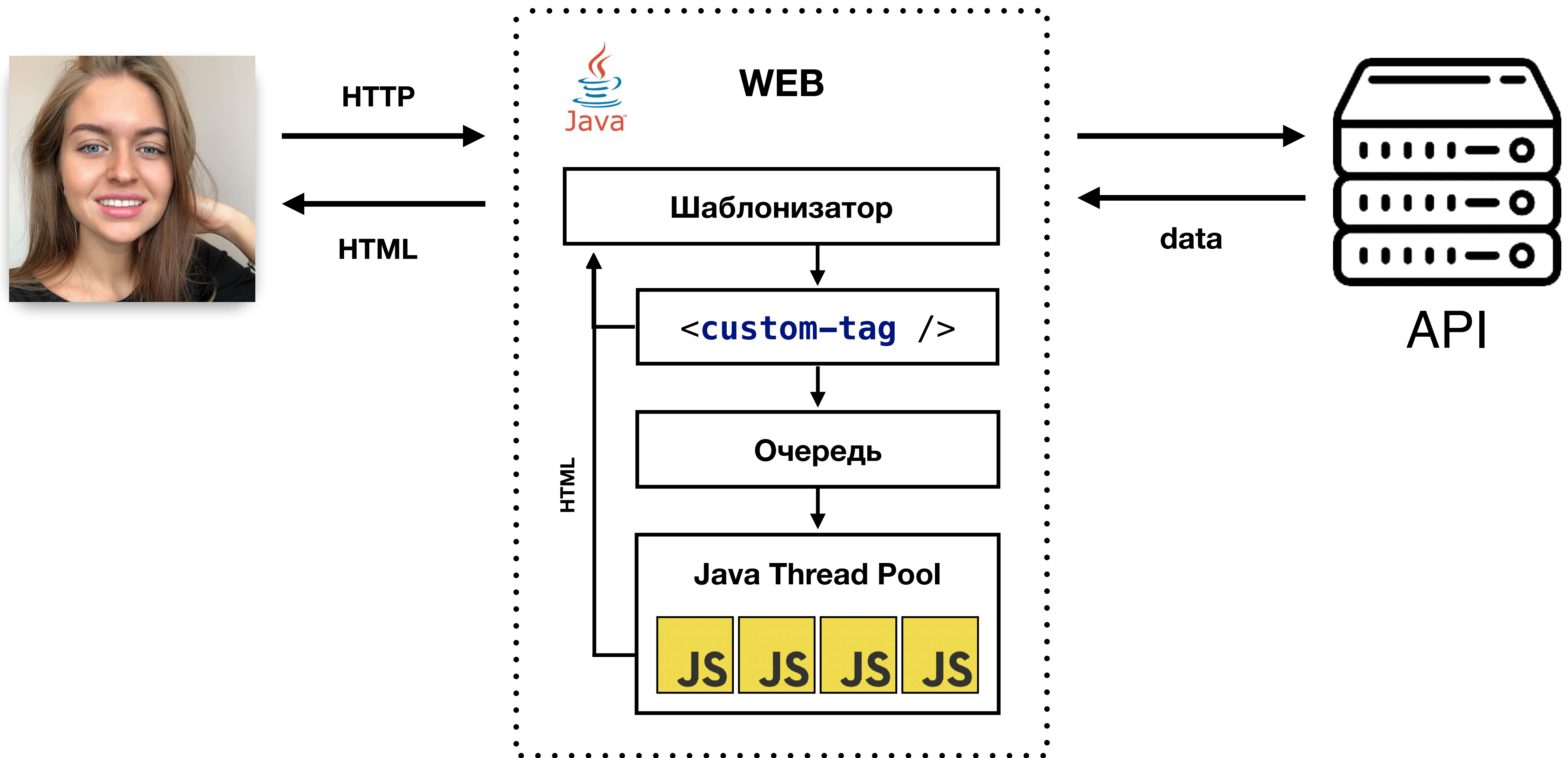


Серверный рендеринг JS - концепт

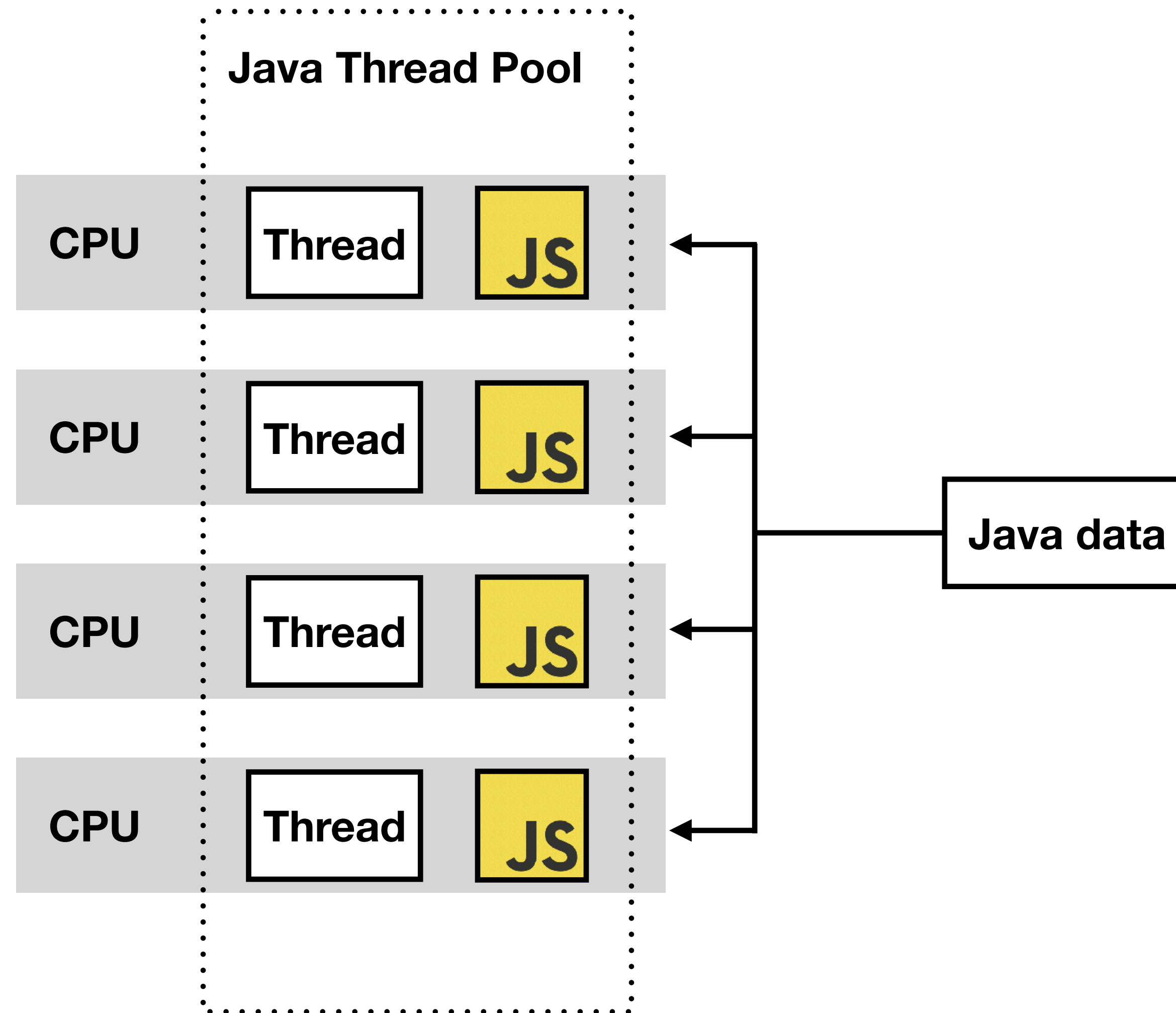


Серверный рендеринг JS - реализация

Серверный рендеринг JS - реализация



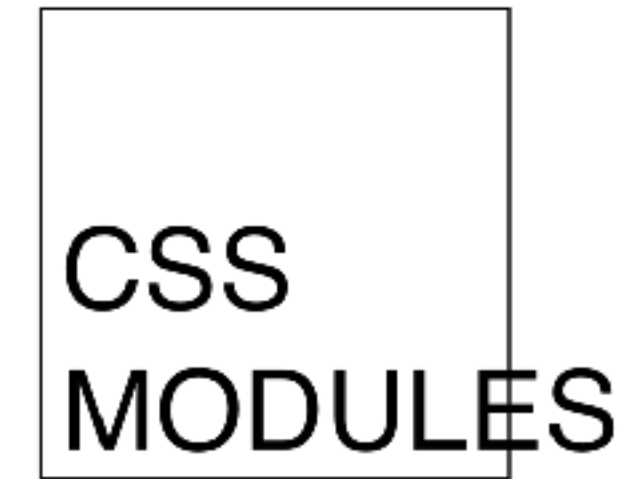
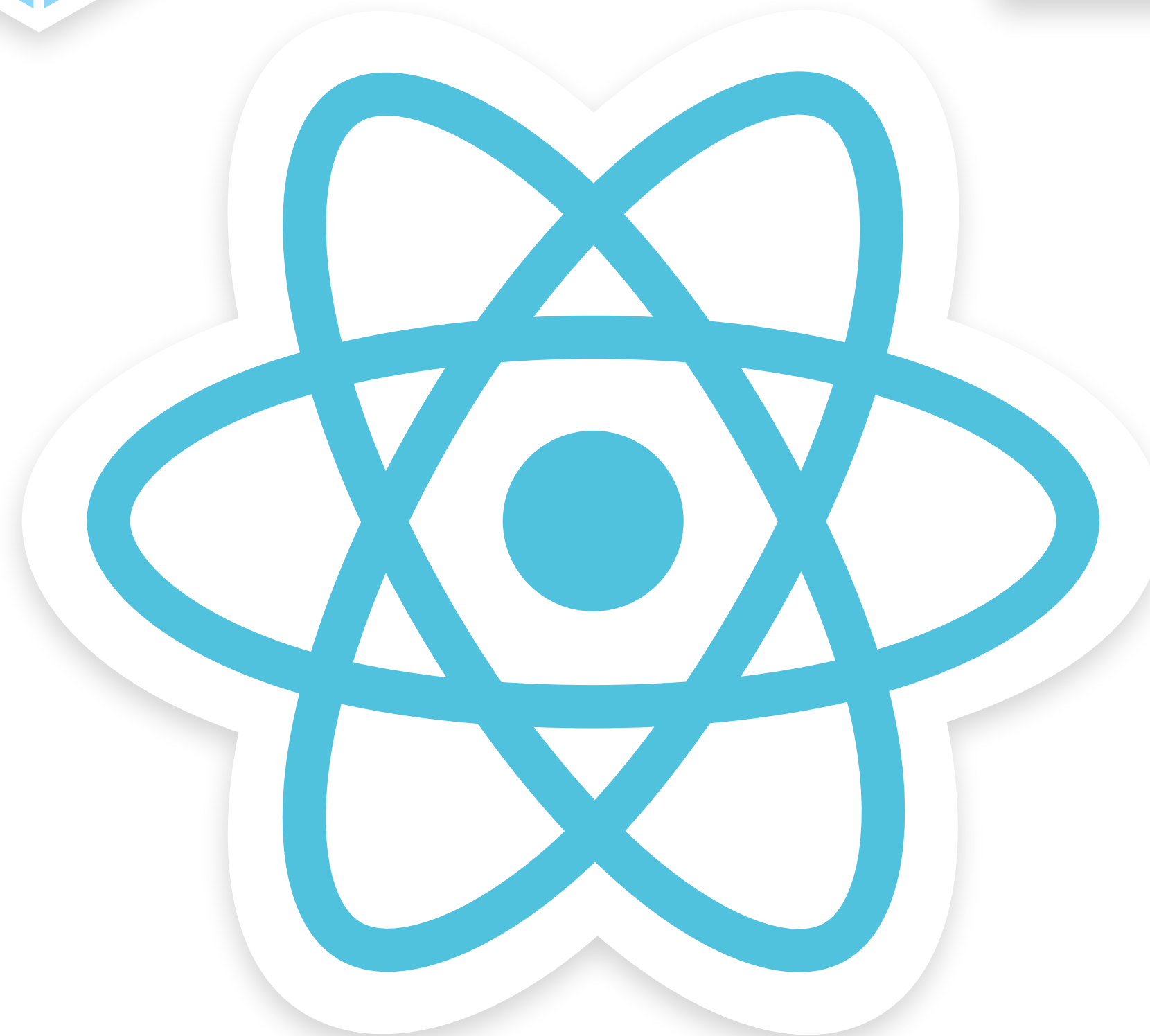
Почему пул контекстов - это хорошо?



Конфигурация UI фреймворка



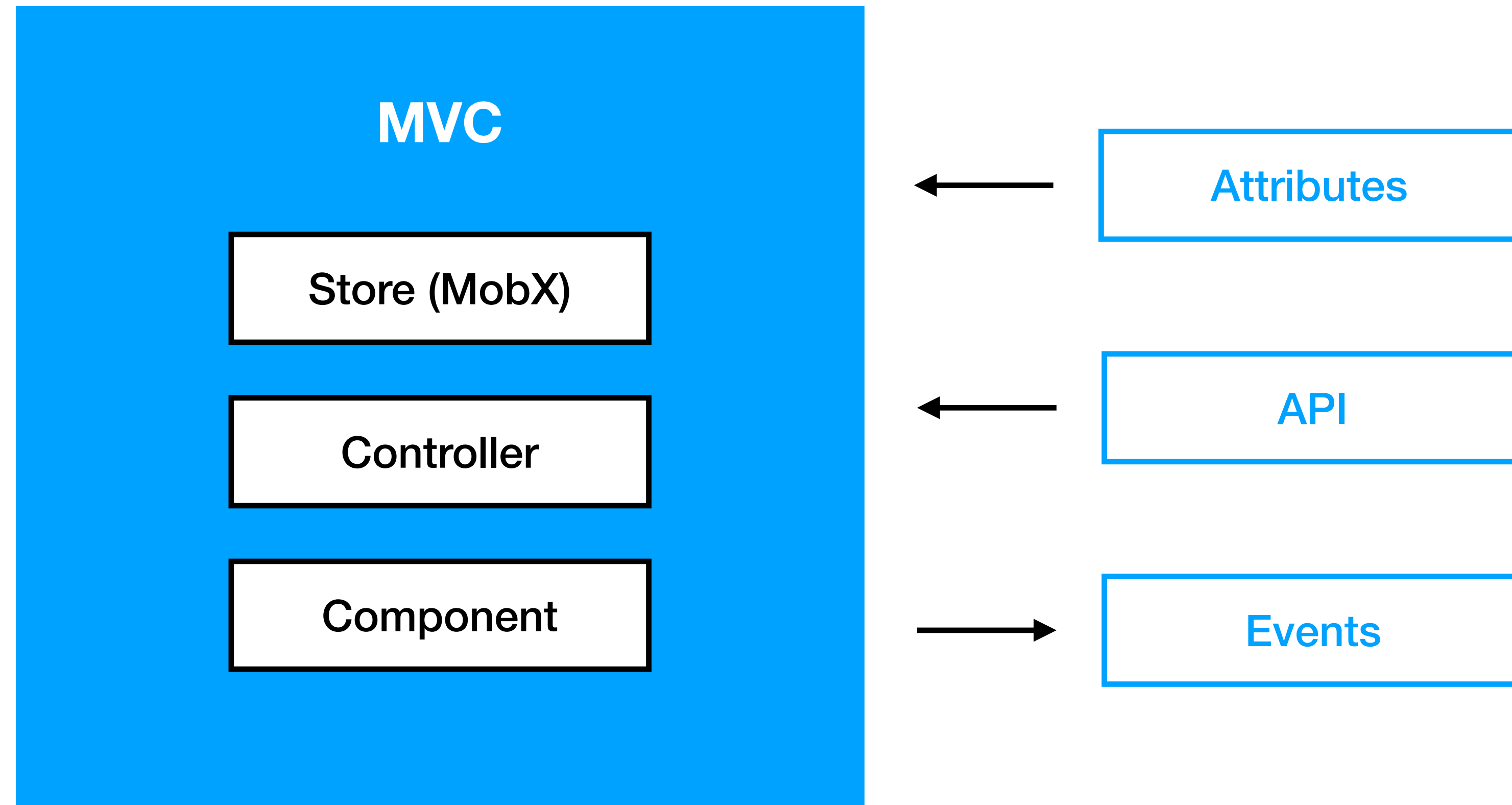
Lerna



Архитектура приложений

Архитектура приложений

Приложение в DOM узле

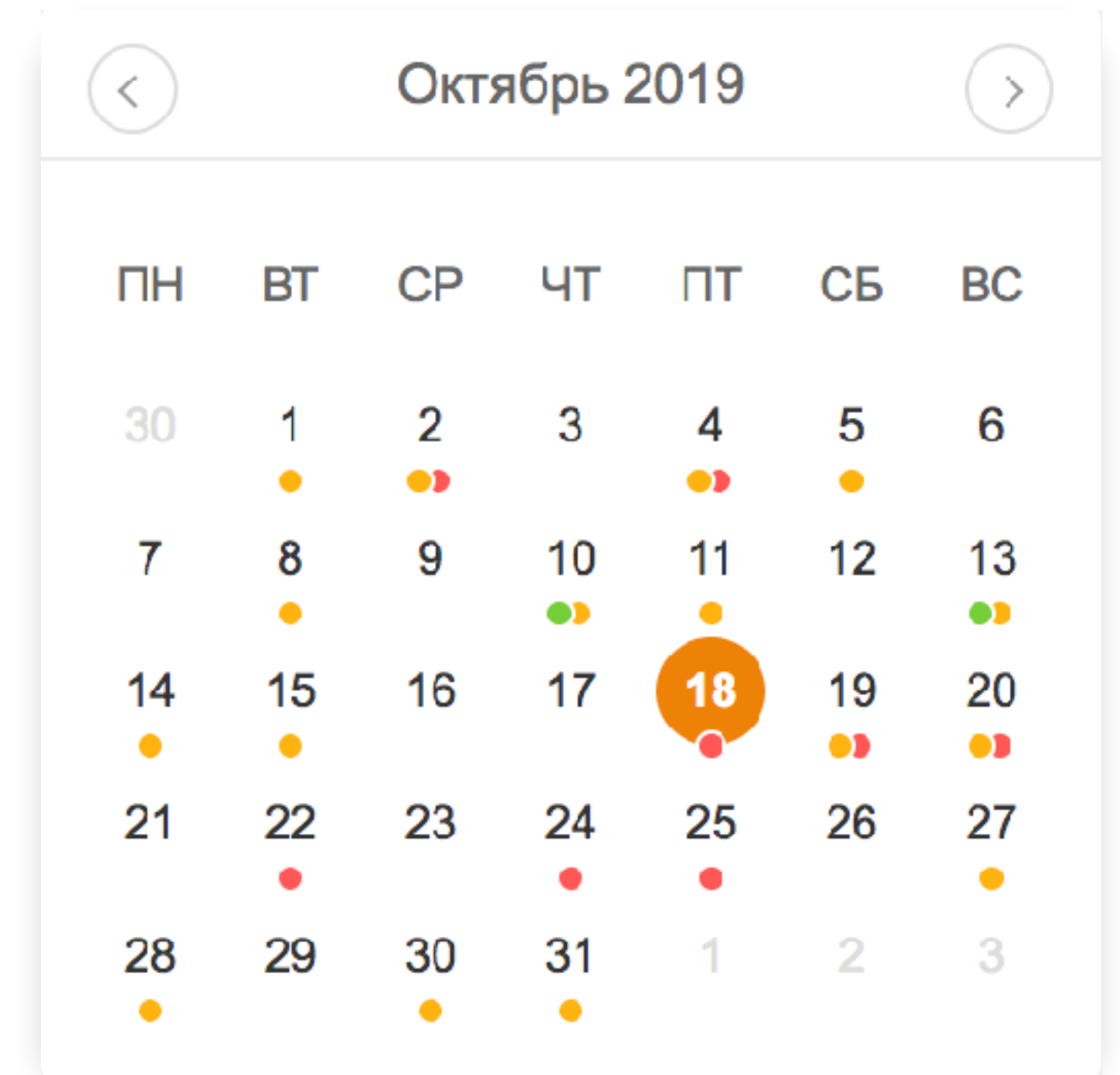


React Engine



Движок приложения

- Инициализирует приложение
- Связывает приложение с дом узлом
- Предоставляет
 - единое апи для рендеринга и уничтожения приложения
 - сервисы локализации, настроек, пр..



Информация о приложении

```
{  
  "events-calendar": {  
    "bundleName": "events-calendar",  
    "js": "events-calendar_h4h5m.js",  
    "css": "events-calendar_h4h5m.css"  
  }  
}
```

Запуск приложения на клиенте

```
<events-calendar data-module="react-loader"  
                  data-bundle="events-calendar.js"  
                  date=".."  
                  marks="[{}]"  
                  ...  
>
```

Как запускать код?

Как запускать реальный код?

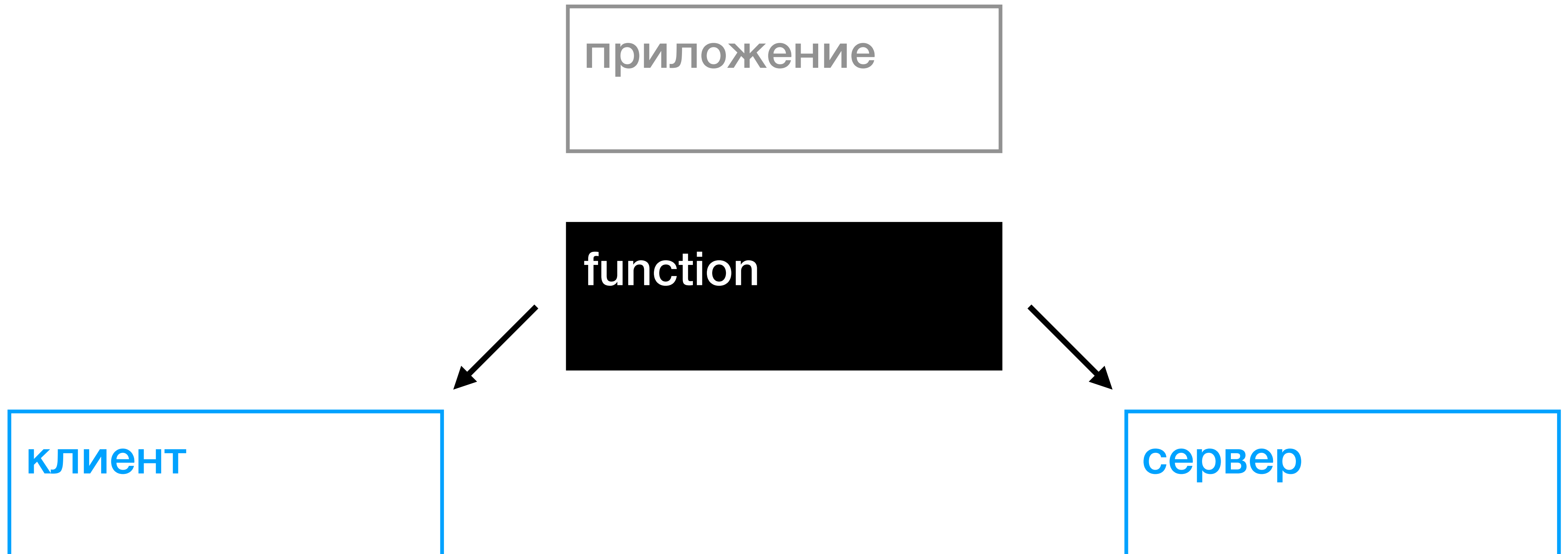


Как запускать код в GraalVM?

- Транспайлить TS в JS, как для NodeJs
- Собирать вебпаком, как для клиента

Методы с разной имплементацией на сервере и клиенте

Методы с разной имплементацией



Alias



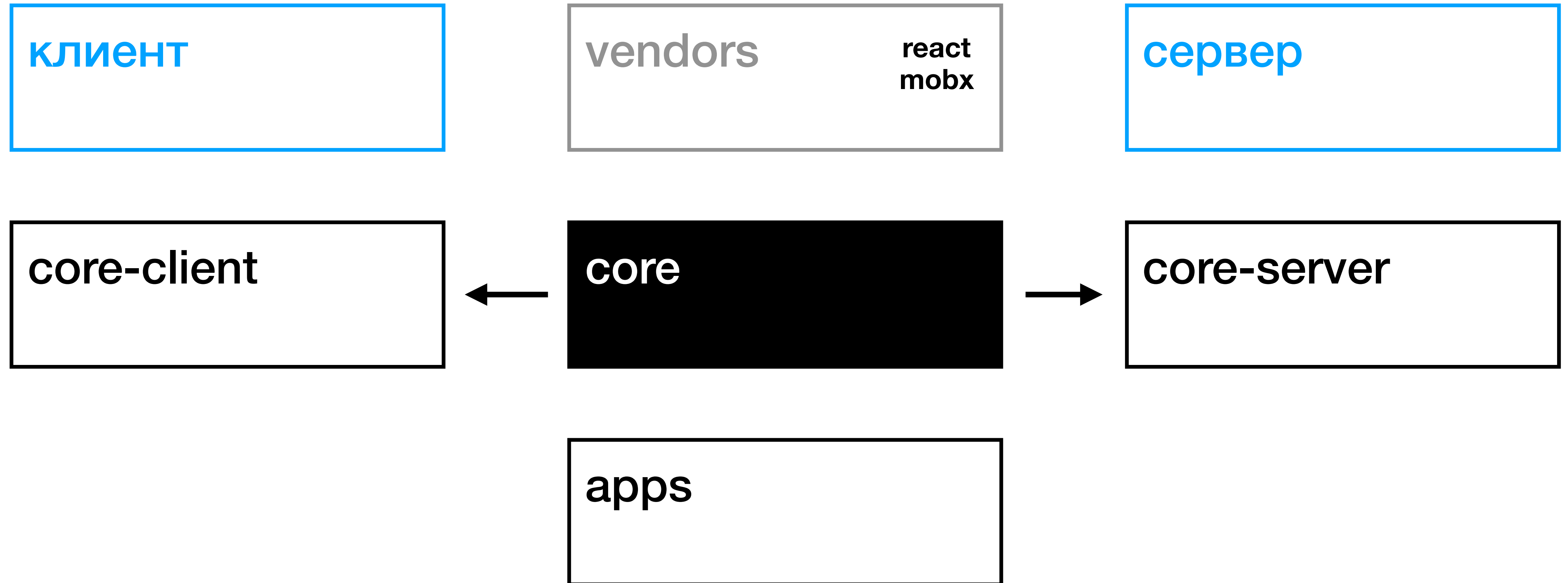
```
resolve: {  
  alias: {  
    './CfgImpl': folders.projects("core", "cfg", "CfgImplServer.ts")  
  }  
}
```

2 конфига - 2 сборки?



Конфигурация Webpack для запуска js на сервере и клиенте

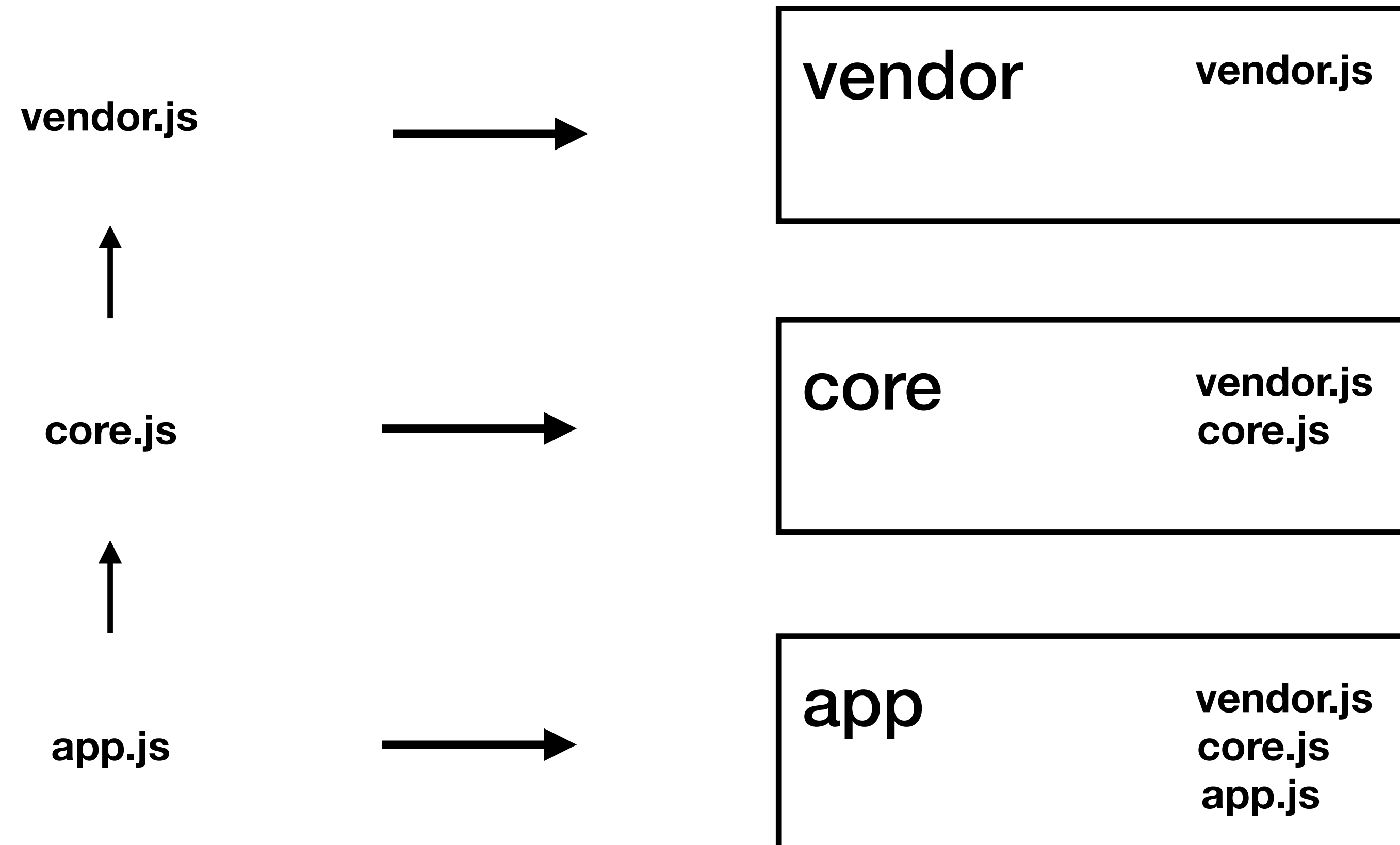
Что есть в исходниках?



Как надо собирать

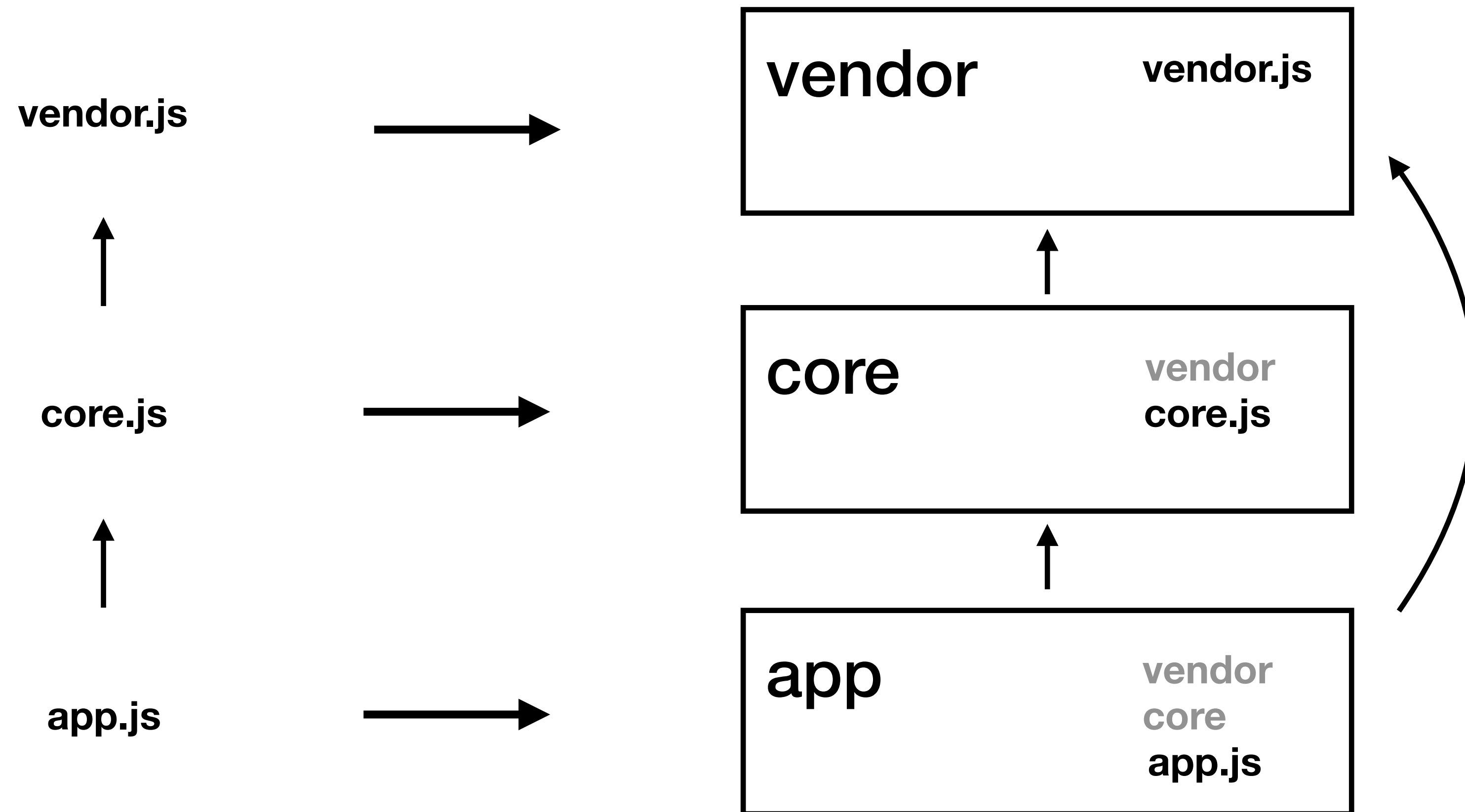
- Возможность сконфигурировать отдельно сборку каждой из частей
- Проставить зависимости между ними
- Собирать все за 1 раз

Сборка по частям - мультиконфигурации

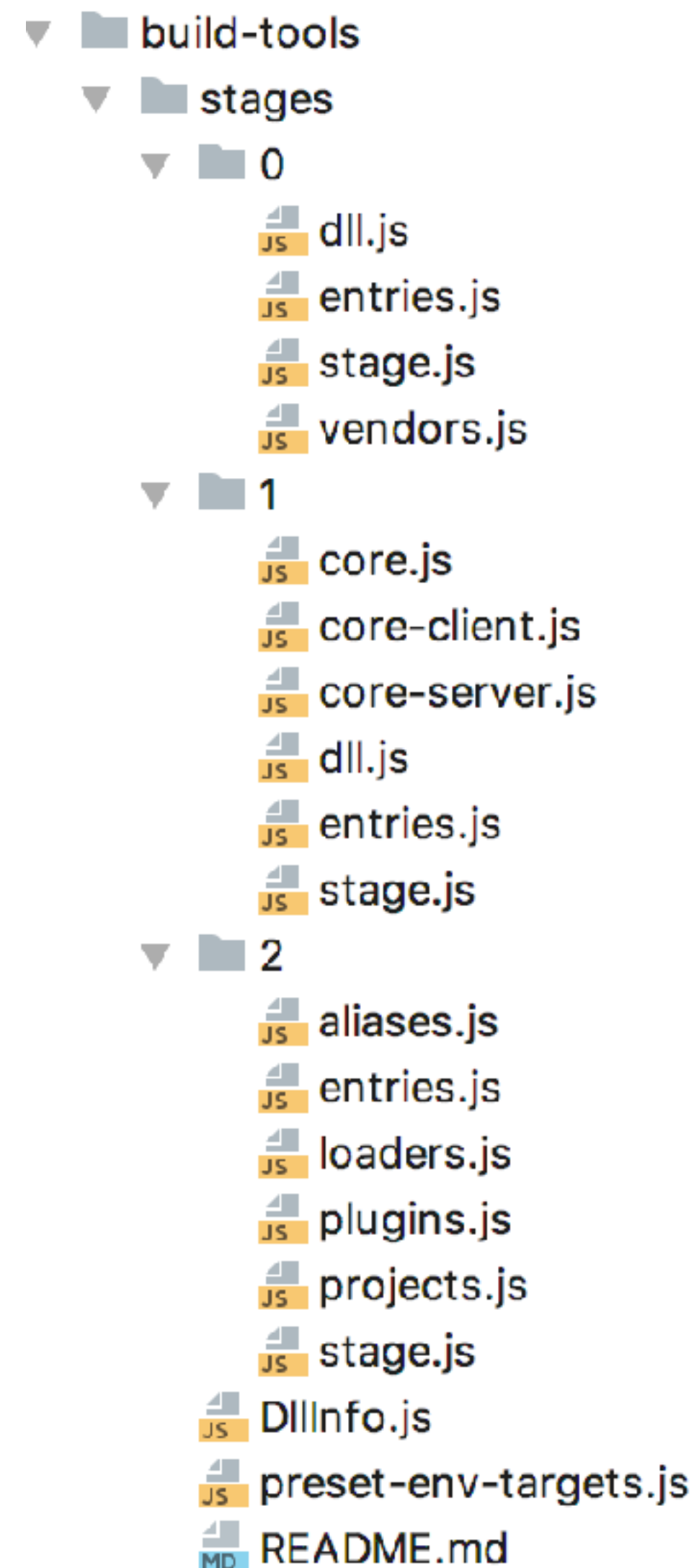


DllPlugin и DllReferencePlugin

<https://webpack.js.org/plugins/dll-plugin/>



Стадии сборки Webpack



Активно применяются

1. [HashedModuleIdsPlugin](#): При сборке бандла в качестве идентификатора модуля используется хэш, вместо инкрементирующегося счётчика.
2. [DllPlugin](#): В процессе сборки генерируется файл-манифест, содержащий список экспортируемых классов данной стадии
3. [DllReferencePlugin](#): Используя файл-манифест предыдущей стадии(-й) позволяет импортировать общий класс
4. [webpack-merge](#): Позволяет склеивать конфигурацию из множества различных кусочков

0я стадия сборки. Сторонние библиотеки.

- 0/dll.js — Параметры DllPlugin, используемые на этой стадии
- 0/stage.js — Базовая конфигурация
- 0/vendors.js — Список сторонних библиотек, включаемых в бандл
- 0/entries.js — Комбинирует кусочки для webpack-merge

1я стадия сборки. Базовые классы (зависимые от среды выполнения).

- 1/dll.js — Параметры DllPlugin, используемые на этой стадии
- 1/stage.js — Расширение конфигурации
- 1/core.js — Список базовых классов
- 1/core-server.js — Бандл классов для работы на сервере (GraalJS)
- 1/core-client.js — Бандл классов для работы на клиенте (Браузер)
- 1/entries.js — Комбинирует кусочки для webpack-merge

2я стадия сборки. Проектные классы.

- 2/stage.js — Расширение конфигурации
- 2/projects.js — Список генерируемых бандлов
- 2/plugins.js — Список используемых плагинов
- 2/loaders.js — Список используемых загрузчиков
- 2/entries.js — Комбинирует кусочки для webpack-merge

Доставка ресурсов

Использование в коде

```
const pkg = l10n('smiles');
```

```
<div>  
  Текст: { pkg.getText('title') }  
</div>
```

Сервер



Методы Java для JS

```
public class ServerMethods {  
    ...  
  
    /**  
     * Получаем текст в виде строки  
     */  
    public String getText(String pkg, String key) {  
        ...  
    }  
  
    ...  
}
```

Методы Java для JS

//добавляем объект с методами Java в поле контекст
js.putMember("serverMethods", serverMethods);

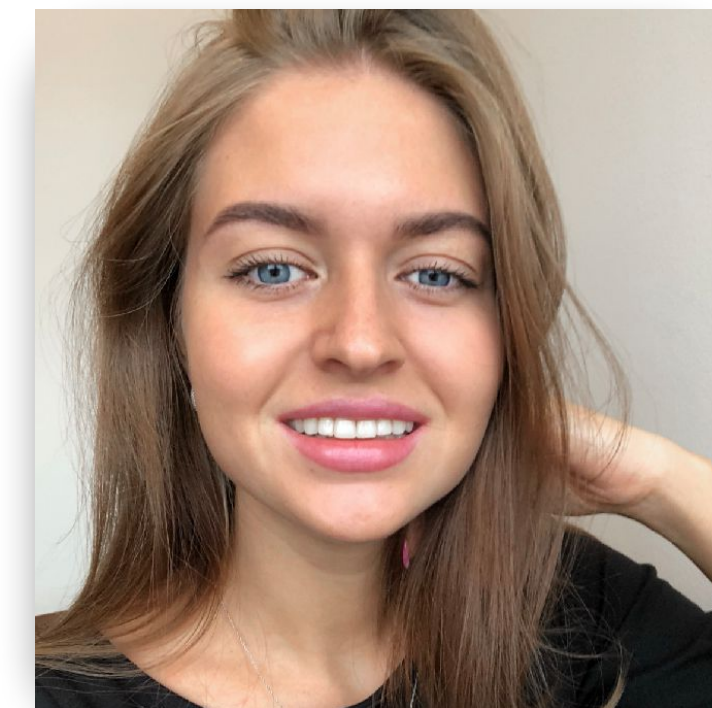
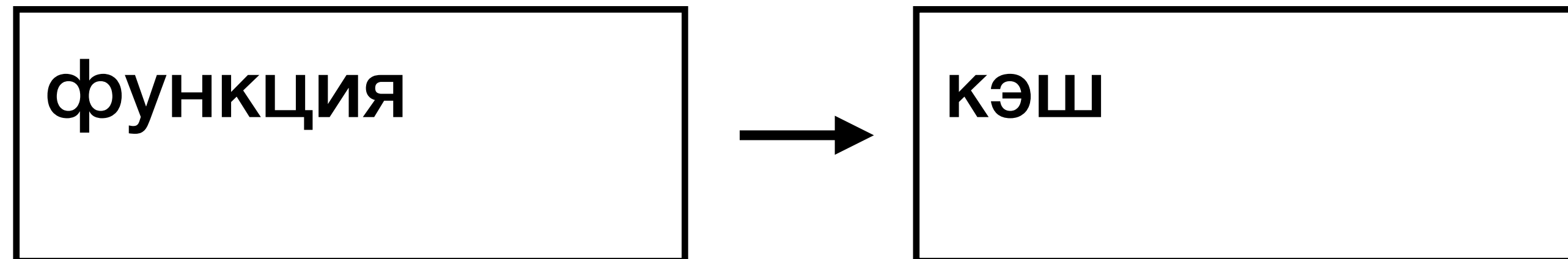
Методы Java в JS

```
function getText(key: string): string {  
    return global.serverMethods.getText(pkg, key);  
}
```

Клиент



Клиент



Клиент



Поиск текстов и настроек в коде

```
const pkg = l10n('smiles');
```

```
<div>  
  { pkg.getLMsg('title') }  
</div>
```

Результат работы webраск плагина

```
{  
  "some-tag": {  
    "pkg": [  
      "smiles",  
      "sadness"  
    ],  
    "cfg": [  
      "config1",  
      "config2"  
    ],  
    "bundleName": "some-tag",  
    "js": "some-tag.js",  
    "css": "some-tag.css"  
  }  
}
```

- Watcher

Итоги

Время рендеринга реакт-приложений

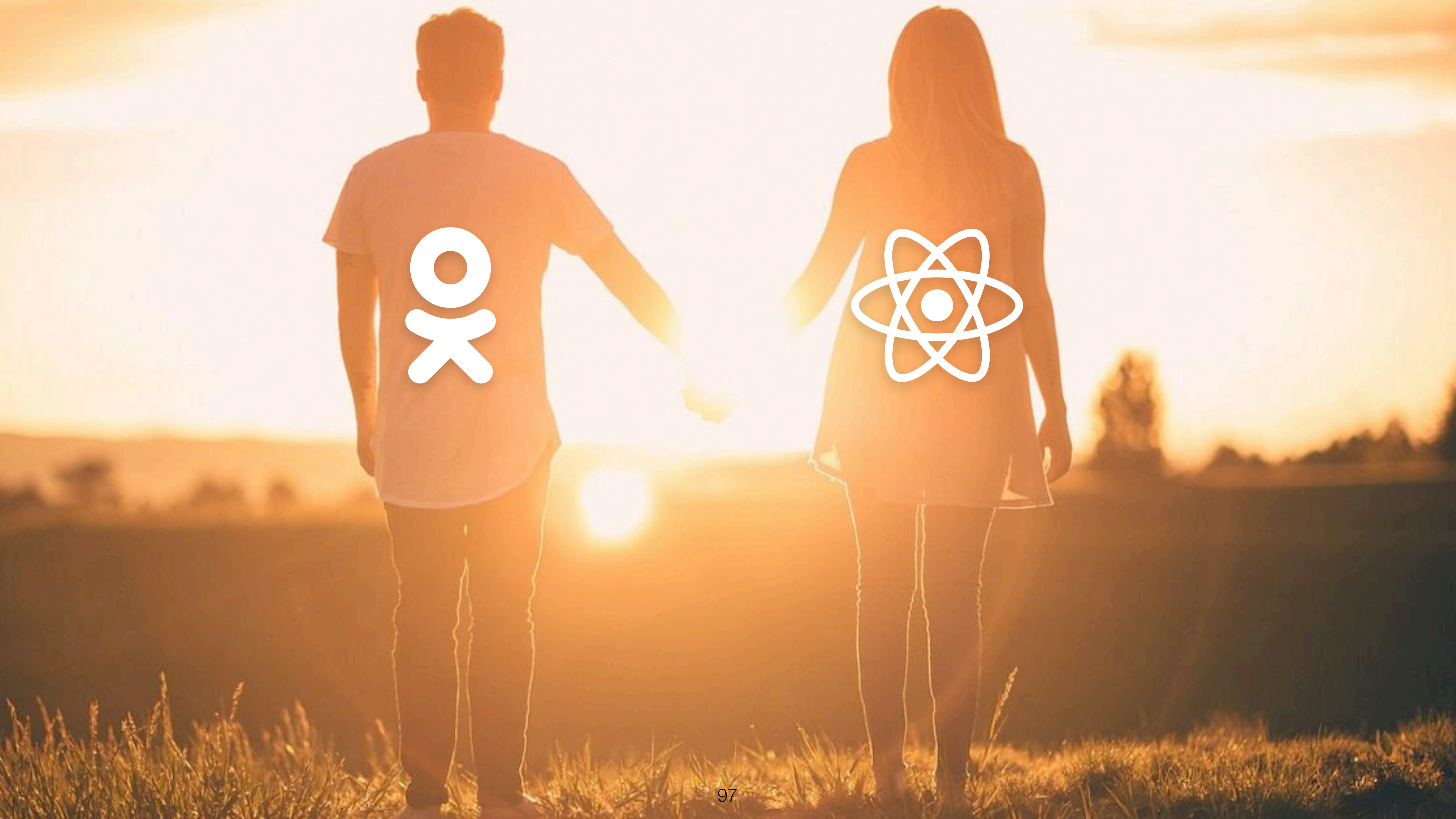
- Простая функция ~ 1 мкс
- Компоненты $\sim 0.5-6$ мс

Что удалось сделать

- Запустить JS на сервере, там где это казалось невозможным)
- Можем писать изоморфный код для UI
- Начали использовать современный стек для фронтенда, со всеми его плюсами
- Появилась единая современная платформа для создания UI
- Начали плавный переход (миграцию)

При этом

- Не произошло усложнения эксплуатации
- Не ухудшили серверный рендеринг



GraalVM — это веб
стандарт?

PHP в GraalVM

<https://github.com/oracle/graal/issues/361>

Да!
GraalVM — будущий веб
стандарт!



Спасибо за внимание!

Новый фронтенд в Одноклассниках

Священный грааль: React в Java

Олег Коровин

ok.ru/madcow

vk.com/madcow