

Яндекс



# State of ВЕМ 2019

## Эволюция Большого фронта

Хохулин Алексей  
@xoxulin

Яндекс

фантастические фильмы 2019

Найти

76

Алексей Хохулин

Поиск Картинки Видео Карты Маркет Новости Эфир Коллекции Знатоки Услуги Ещё

Каталог > Фильмы

фантастика по популярности 2019

163 оценки

Развернуть

Дитя робота

Люди в чёрном: Интернэшнл

Годзилла 2: Король монстров

Шазам!

Битва за Землю

Стекло

Мёртвые не умирают

6.4 89% 5.9 72% 6.0 68% 6.8 69% 5.4 85% 6.4 5 6.6 92%

2019, ужасы, фантастика

2019, фантастика, боевик

2019, фантастика, боевик

2019, фэнтези, боевик

2019, фантастика, триллер

2019, фантастика, триллер

Больше похожих фильмов

Фантастика 2019 года – списки лучших фильмов...

[kinopoisk.ru](#) > [lists/navigator/sci-fi/2019/](#) ▾

Все страны. 2019. Навигатор по фильмам. фантастика. ... Фантастика 2019 года. Смотреть онлайн. Фильмы Сериалы С высоким рейтингом Российские Зарубежные. Читать ещё >

Похожие подборки



# Большой поиск

Яндекс  Найти

Поиск Картинки Видео Карты Маркет Новости Эфир Коллекции Знатоки Услуги Ещё

Алексей Хохулин 76

Каталог > Фильмы

фантастика по популярности 2019

163 оценки Развернуть

Фильм	Рейтинг	Оценка	Год	Жанр
Дитя робота	6,4	89%	2019	фантастика
Люди в чёрном: Интернэшнл	5,9	72%	2019	фантастика, боевик
Годзилла 2: Король монстров	6,0	68%	2019	фантастика, боевик
Шазам!	6,8	69%	2019	фэнтези, боевик
Битва за Землю	5,4	85%	2019	фантастика, триллер
Стекло	6,4	5	2019	фантастика, триллер
Мёртвые не умирают	6,6	92%	2019	ужасы, фэнтези

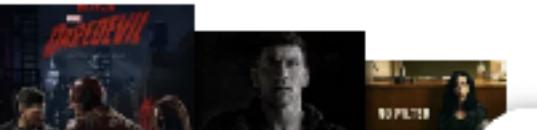
Колдунщик – специальный ответ  
микро приложение



Фант

кифор

Все страны. 2019. Навигатор по фильмам. фантастика. ... Фантастика 2019 года.  
Смотреть онлайн. Фильмы Сериалы С высоким рейтингом Российские Зарубежные.  
[Читать ещё >](#)



State of BEM 2019

БЭМ 2010



State of BEM 2019

# Методология

Методология



# Блок \_\_Элемент \_Модификатор

Методология

- Блок \_\_Элемент \_Модификатор
- КОМПОНЕНТНЫЙ ПОДХОД

- Блок \_\_Элемент \_Модификатор
- КОМПОНЕНТНЫЙ подход
- декларативность

- Блок \_\_Элемент \_Модификатор
- КОМПОНЕНТНЫЙ подход
- декларативность
- КОМПОЗИЦИЯ

- Блок \_\_Элемент \_Модификатор
- компонентный подход
- декларативность
- композиция
- уровни переопределения

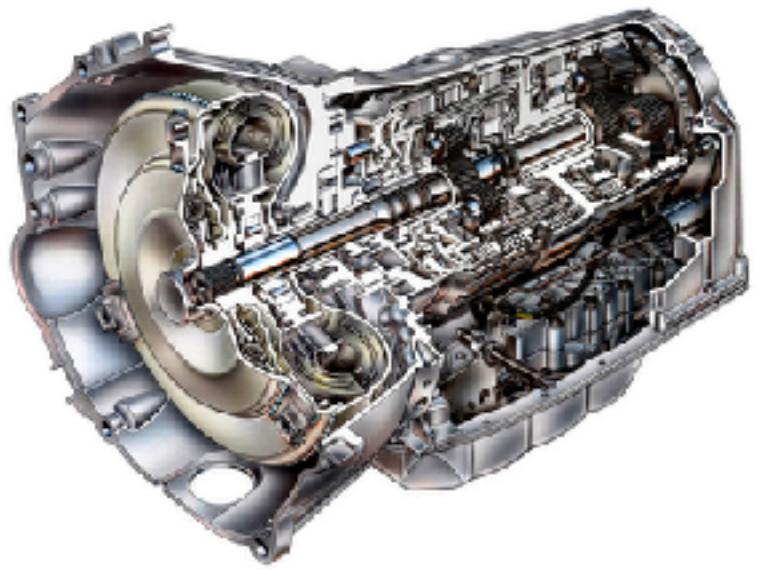
- Блок \_\_Элемент \_Модификатор
- компонентный подход
- декларативность
- композиция
- уровни переопределения
- эксперименты

- 📁 bem in css // styles
- 📁 i-bem.js // client logic
- 📁 bemjson // blocks / pages
- 📁 bemhtml // templates
- 📁 bemdeps // dependencies

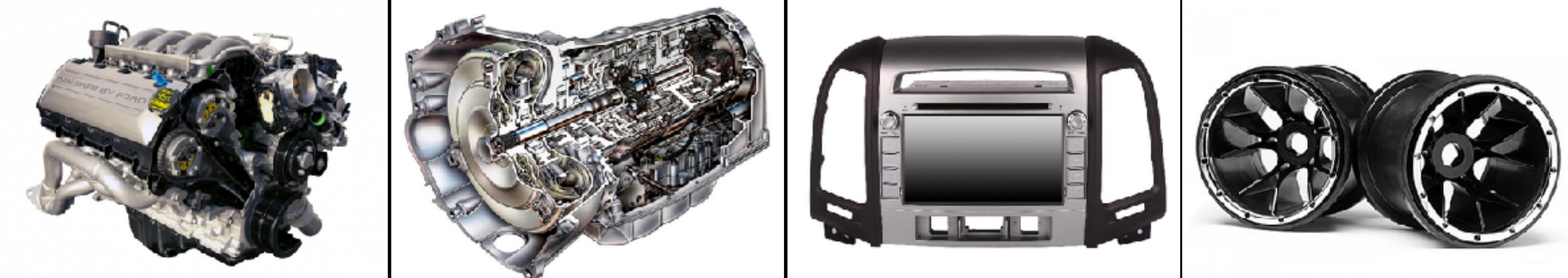
i-bem.js + jquery.js = ❤



## БЛОКИ



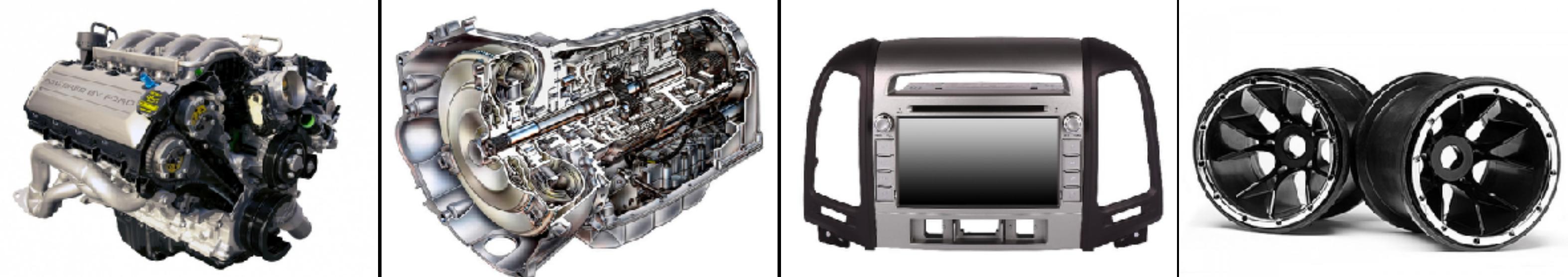
БЛОКИ



УРОВНИ



БЛОКИ



Уровни



Таргеты

Econom

Comfort

Business

Luxury

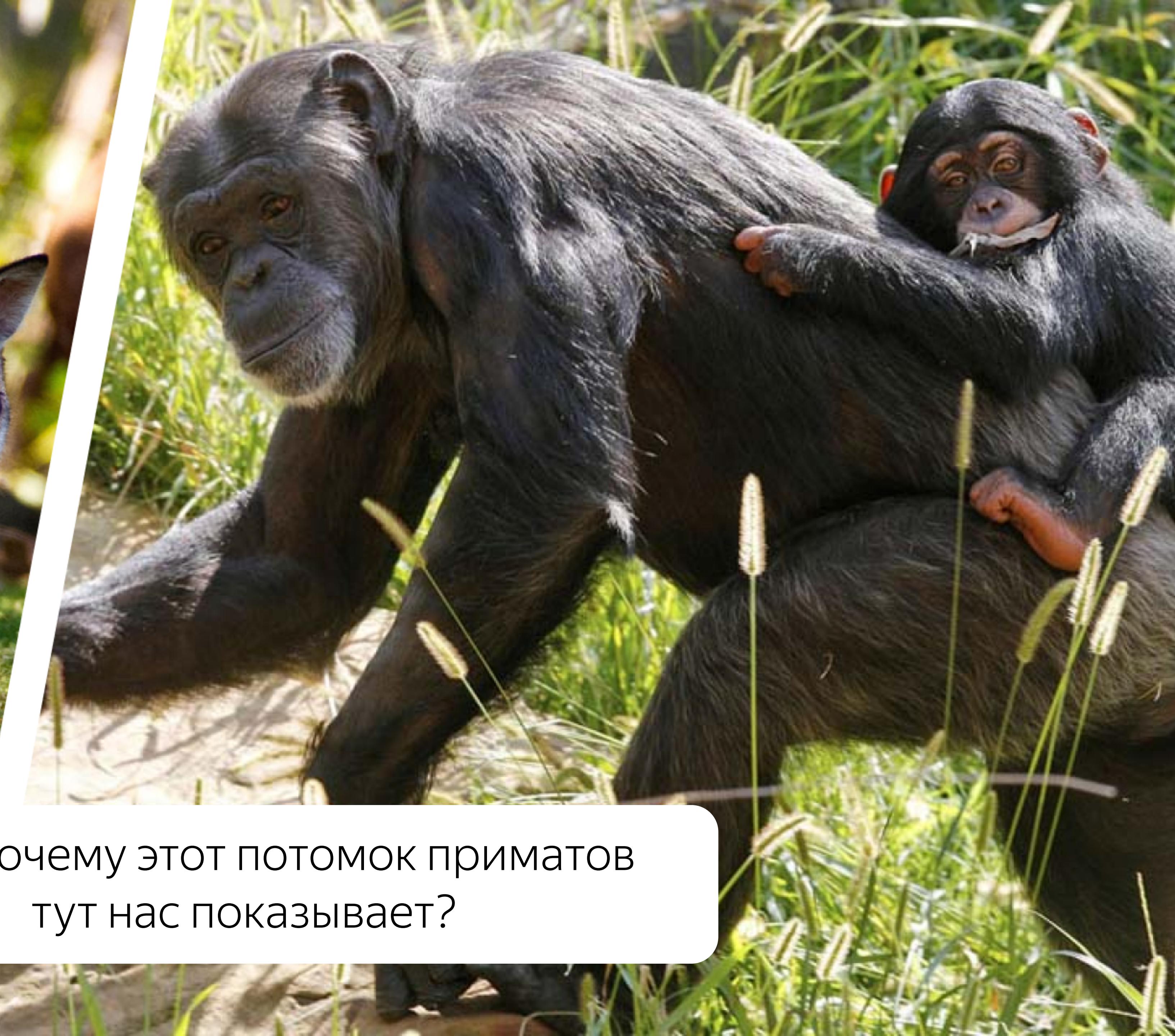
State of BEM 2019

# Эксперименты





Мам, почему этот потомок приматов  
тут нас показывает?









Мы с тобой эксперименты, прикинь!





Эксперименты и искусственный отбор!

Мутации и естественный отбор!

$$\text{feature}(x) = \sum_{i=1}^x \text{experiments}_i$$

$$\text{KPI}(\text{feature}(i+1)) > \text{KPI}(\text{feature}(i))$$

Experiments

Mutations

$$\text{feature}(x) = \sum_{i=1}^x \text{experiments}_i$$

$$\text{KPI}(\text{feature}(i+1)) > \text{KPI}(\text{feature}(i))$$

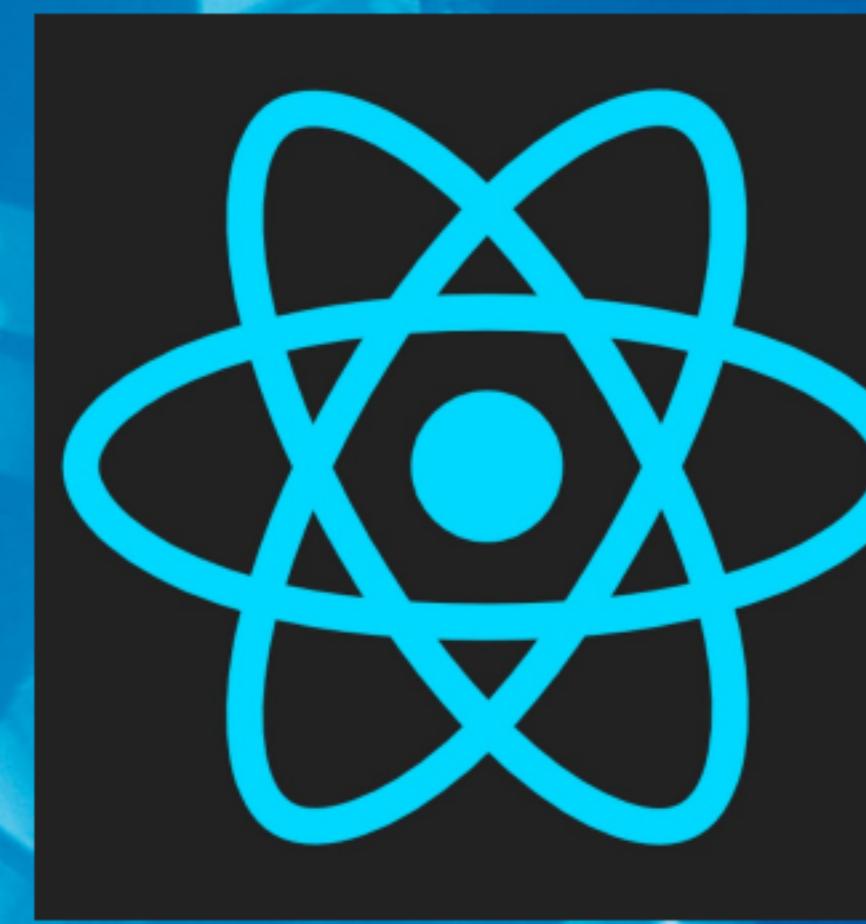
Selection

Experiments

State of BEM 2019

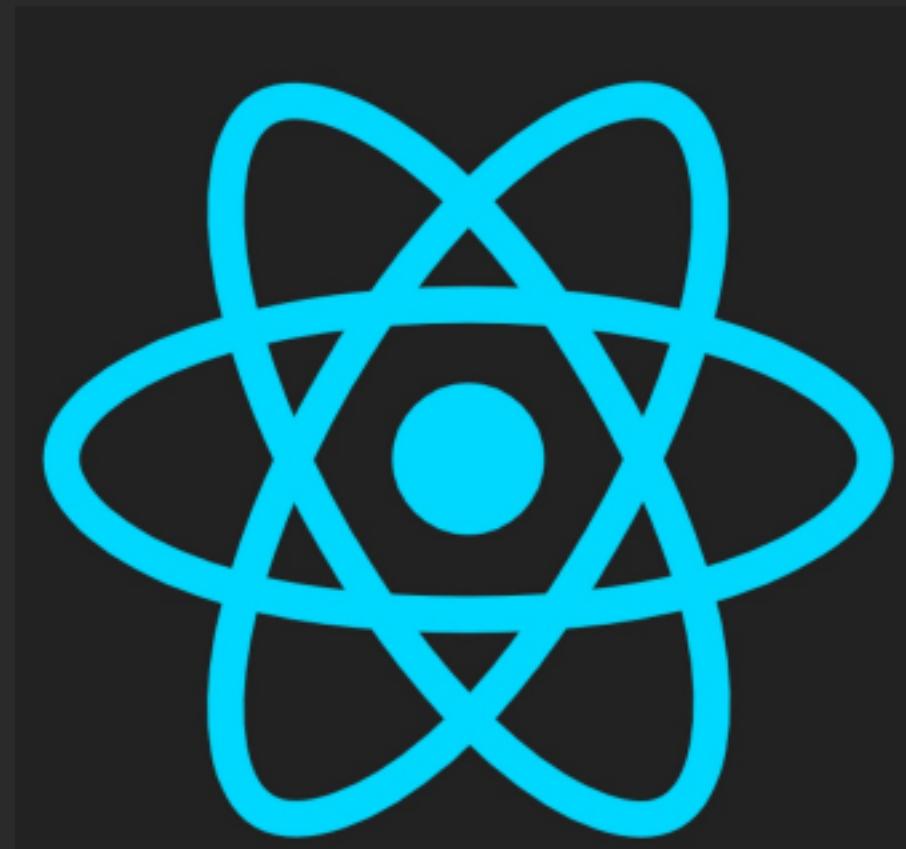
БЭМ 2019





TS

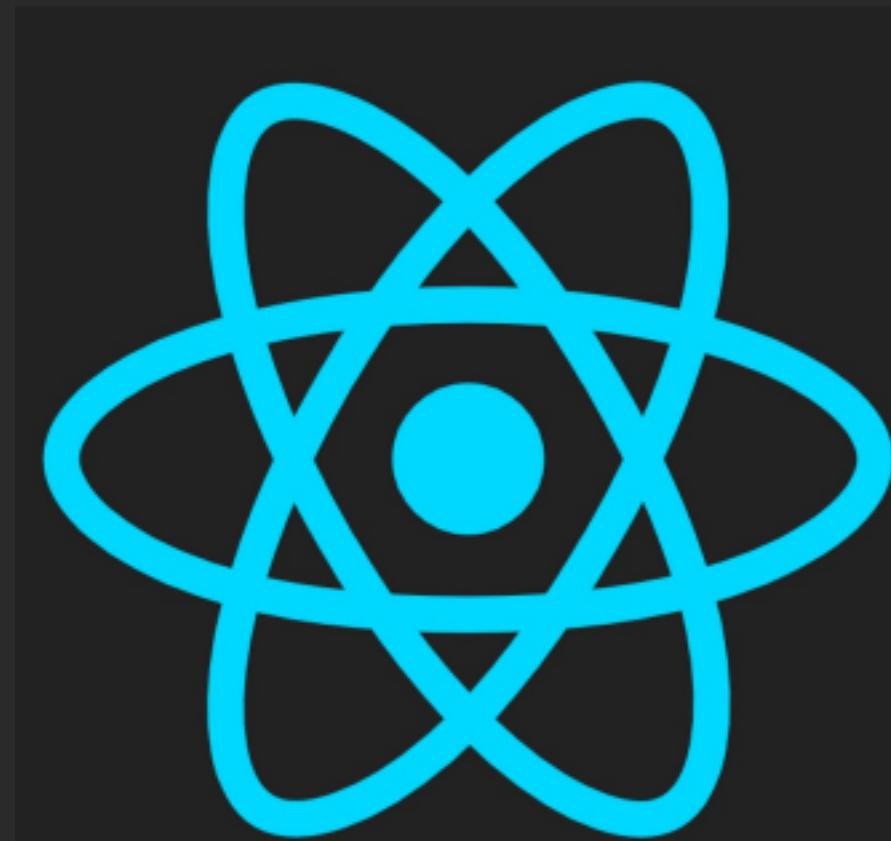
react



📄 всё явно

📄 всё декларативно

react



typescript



- 📁 интерфейсы
- 📁 обобщенное программирование
- 📁 функциональность
- 📁 поддержка IDE

typescript





## Methodology

```
import { Methodology } from 'BEM';
```

Methodology

```
import { Methodology } from 'BEM';
```

Methodology

```
import { Methodology } from 'BEM';

class IBemJS extends Methodology<JQuery>;
```

Methodology

```
import { Methodology } from 'BEM';

class IBemJS extends Methodology<JQuery>;
class BemReact extends Methodology<React>;
```

- 📁 bem in css // styles
- 📁 react.js // client logic
- 📁 react.js // blocks / pages
- 📁 jsx // templates
- 📁 imports // dependencies

bem-react



- 📁 компонентный подход
- 📁 изоморфный код
- 📁 уровни переопределения
- 📁 явные зависимости



bem-react

📁 bem-react

📁 packages

  📁 classname ⚒

  📁 core ⚒

  📁 di 🚀



# classname

```
import { IClassNameProps } from '@bem-react/core'
import { cn } from '@bem-react/classname';

export interface IButtonProps extends IClassNameProps {
  tag?: string;
}

export const cnButton = cn('Button');
cnButton('Text');           // Button Button-Text
cnButton({ size: 'l' });    // Button Button_size_l
```

## core baseBlock

```
import React, { FC } from 'react';

import { IButtonProps, cnButton } from './index';

export const Button: FC<IButtonProps> = ({  
    children,  
    className,  
    tag: TagName = 'button',  
}) => (  
    <TagName className={cnButton({}, [className])}>  
        {children}  
    </TagName>  
);
```

## core modifier

```
import React from 'react';
import { withBemMod } from '@bem-react/core';

import { IButtonProps, cnButton } from '../index';

interface IButtonTypeLinkProps {
  type?: 'link';
}

const withButtonTypeLink = withBemMod<IBUTTONTypeLinkProps,
IButtonProps>(cnButton(), { type: 'link' }, (Button) => (
  (props) => (
    <Button {...props} tag="a" />
  )
));

```

## core compose

```
import { Button as ButtonPresenter } from 'Button';
import { withButtonThemeAction, ... } from '...';
import { compose, composeU } from '@bem-react/core';

const Button = compose(
  composeU(withButtonThemeAction, withButtonThemeDefault),
  withButtonTypeLink,
)(ButtonPresenter);

export const App: FC = () => (
  <div className="App">
    <Button>Basic</Button>
    <Button theme="action" type="link">All</Button>
  </div>
);
```



## di primitive App.tsx

```
import { cn } from '@bem-react/classname';
import { Footer } from './Components/Footer';
import { Header } from './Components/Header';

const cnApp = cn('App');

export const App = () => {
  return (
    <div className={cnApp()}>
      <Header />
      <Footer />
      <div/>
    );
};
```

## di by hooks App.tsx

```
import { cn } from '@bem-react/classname';
import { useComponentRegistry } from '@bem-react/di';

const cnApp = cn('App');

export const App = () => {
  const { Header, Footer } = useComponentRegistry(cnApp());
  return (
    <>
      <Header />
      <Footer />
    </>
  );
};
```

## di injection App@desktop.tsx

```
import { cn } from '@bem-react/classname';
import { Registry, withRegistry } from '@bem-react/di';
import { App as AppCommon, registryId } from './App';
import { Header } from './Components/Header/Header@desktop';
import { Footer } from './Components/Footer/Footer@desktop';

export const cnApp = cn('App');
export const registryId = cnApp();
export const registry = new Registry({ id: registryId });

registry.set('Header', Header); registry.set('Footer', Footer);

export const AppDesktop = withRegistry(registry)(AppCommon);
```

```
import { AppDesktop, registryId } from './App@desktop';
import { HeaderExp } from './exp/Components/Header/Header';

const expRegistry = new Registry({ id: registryId });

expRegistry.set('Header', HeaderExp);

export const AppDesktopExp = withRegistry(
  expRegistry
)(AppDesktop);
```

<https://github.com/bem/bem-react/>



Package	Version	Size
@bem-react/classname	npm v1.5.3	minzipped size 532 B
@bem-react/classnames	npm v1.3.4	minzipped size 316 B
@bem-react/core	npm v2.0.2	minzipped size 638 B
@bem-react/di	npm v2.0.2	minzipped size 743 B



bem-react

Возможности

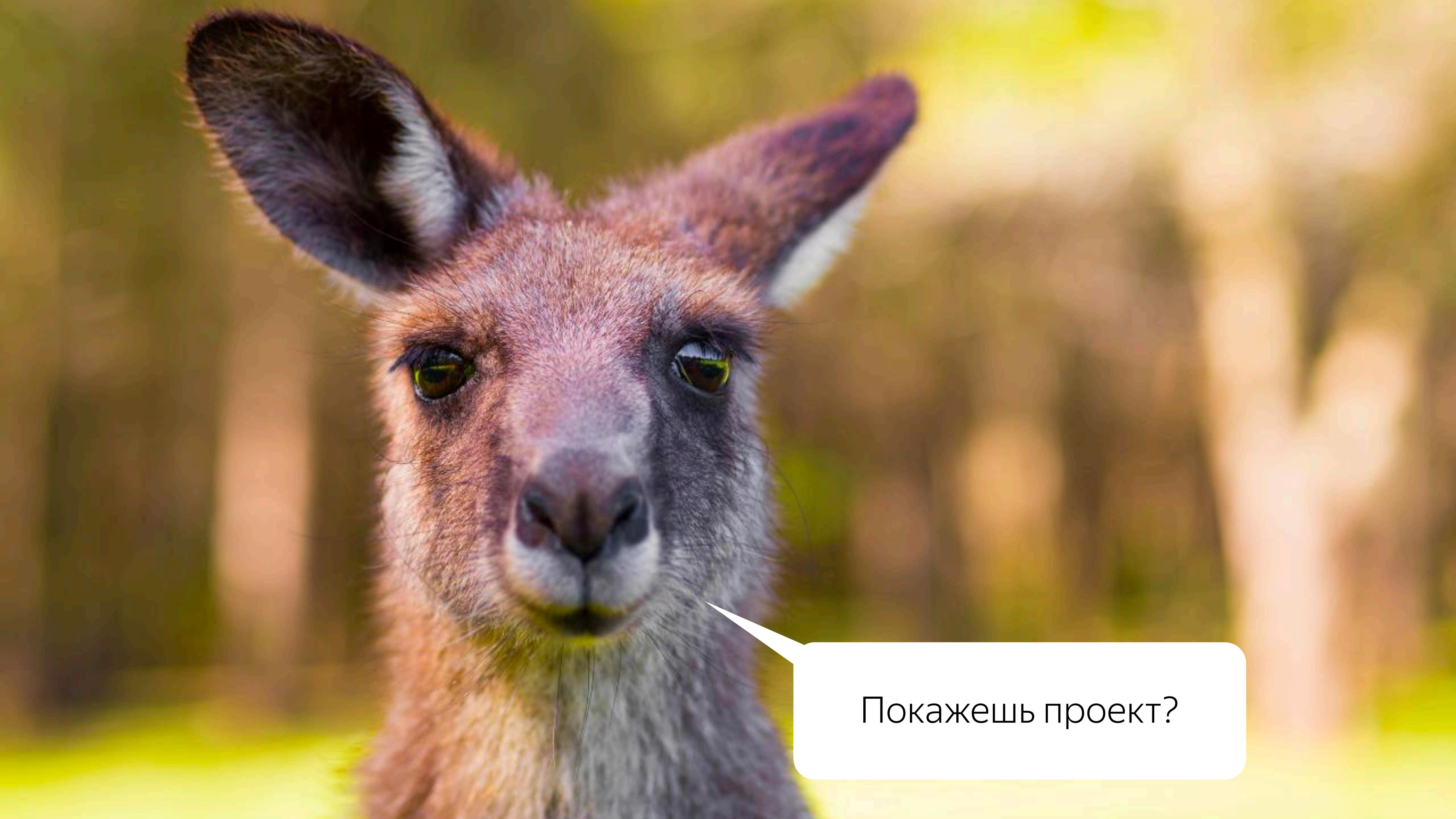


- █ малая инвазивность
- █ автономность модулей
- █ удобная сборка

Возможности







Покажешь проект?

Проект

- 📁 project
- 📁 src
  - 📁 components 🔧
  - 📁 features 🔧
  - 📁 experiments 🚀

Проект

Общие компоненты

```
■ components 🔧
  ■ Block
    ■ Element
    ■ _modifier
    ■ Block.test // enzyme + jest
      - Block.tsx
      - Block@desktop.tsx
      - Block@touch-phone.tsx
      - Block.examples.tsx
```

АдAPTERЫ И ПРИЛОЖЕНИЯ



features 🔧



- 📁 App.components
- 📁 App.i18n
  - App.server.ts
  - App@desktop.server.ts
  - App@touch-phone.server.ts
- App.tsx
- App@desktop.tsx
- App@touch-phone.tsx
- App.test.ts
- App.hermione.ts

features 🔧

App

App.components

App.i18n

- App.server.ts

- App@desktop.server.ts

- App@touch-phone.server.ts

- App.tsx

- App@desktop.tsx

- App@touch-phone.tsx

- App.test.ts

- App.hermione.ts

Adapters<RawData>: AppProps

hermione.js



# hermione.js



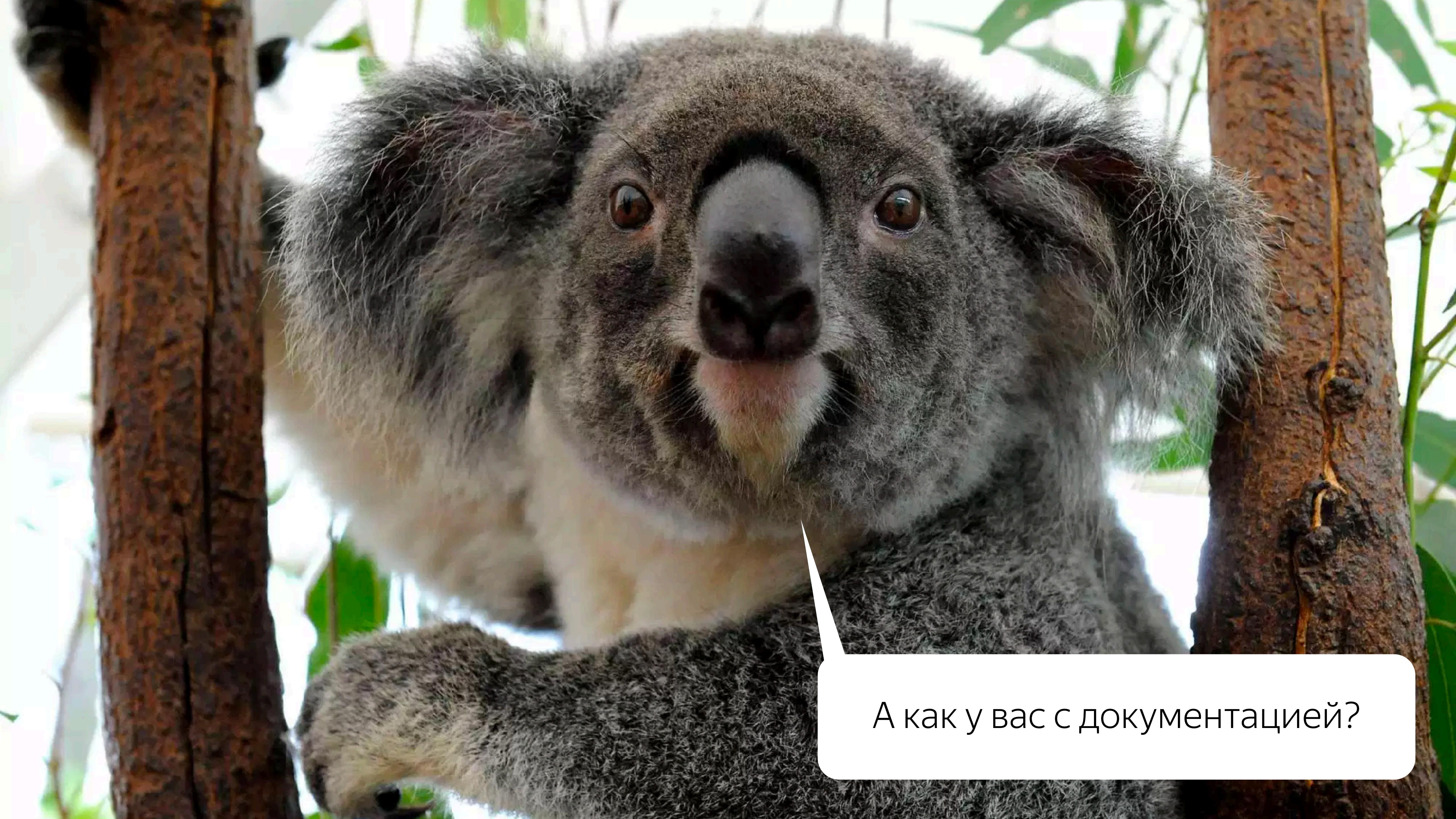
**Skip Ad ➤**

Protego! Интеграционные тесты!

Эксперименты

- 📁 experiments 🚀
  - 📁 .registry
    - desktop.ts
    - touch-phone.ts
  - 📁 experimental\_flag
    - 📁 features
      - 📁 App /\* modified app \*/
    - 📁 beauty\_comments
    - 📁 ... more 150





А как у вас с документацией?



# storybook



Press "/" to search...

## LEGO / COMPONENTS

- Button
  - default
  - @desktop
  - **\_action**
  - \_baseline
  - \_hasControl
  - \_pale
  - \_pin
  - \_size
  - \_theme
  - \_tone
  - \_type
  - \_view
  - \_width
- Checkbox
- Image
- Link
- Menu
- Modal
- Popup
- Progress
- Radiobox
- Select
- Spin

Песочница

Документация

## Button

Компонент используется для создания кнопок.

### Пример использования

```
import React from 'react'
import { compose } from '@bem-react/core'
import {
  Button as ButtonDesktop,
  withThemeNormal,
  withViewDefault,
  withSizeS,
  withPinCircleCircle,
  withToneRed,
} from '@yandex-lego/components/Button/desktop'

// Композиция из различных модификаторов
const Button = compose(
  withViewDefault,
  withThemeNormal,
  withPinCircleCircle,
  withSizeS,
  withToneRed,
)(ButtonDesktop)

// Использование компонента в вашем приложении
```

Пример использования

Примеры

Стилевое оформление  
кнопки

Размер кнопки

Тип кнопки

Границы кнопки

Тон кнопки

Вид кнопки

Ширина кнопки

Свойства

Модификаторы

Ссылки

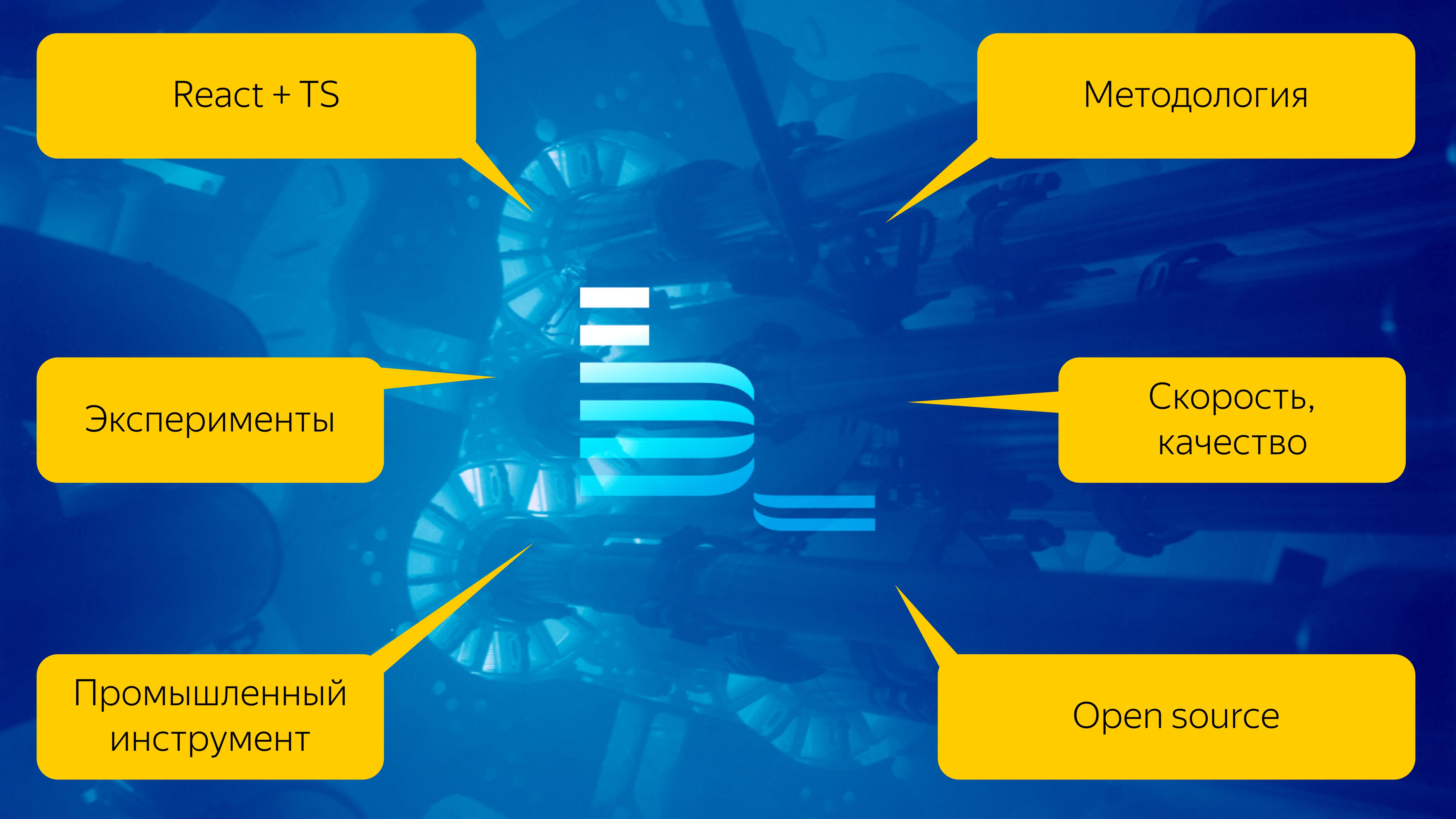
Документация

State of BEM 2019

# Summary







React + TS

Методология

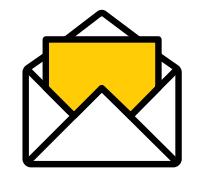
Эксперименты

Промышленный  
инструмент

Скорость,  
качество

Open source

Хохулин Алексей  
руководитель группы  
разработки интерфейсов



@xoxulin

Спасибо! Вопросы?

Хохулин Алексей  
руководитель группы  
разработки интерфейсов



@xoxulin

