

Storybook - проблема или решение проблемы?

HolyJS

Нагайко Иван



О чем доклад?



Какие проблемы
помогает решать
storybook?

О чем доклад?



Какие проблемы
помогает решать
storybook?



Примеры **решения**
проблем из нашей
практики

О чем доклад?



Какие проблемы
помогает решать
storybook?



Примеры **решения**
проблем из нашей
практики



Проблемы при
использовании
storybook

Какие проблемы
помогает решать
storybook?



Какие проблемы помогает решать storybook?

1

Библиотека ui-компонентов

2

Написание документации на компоненты

3

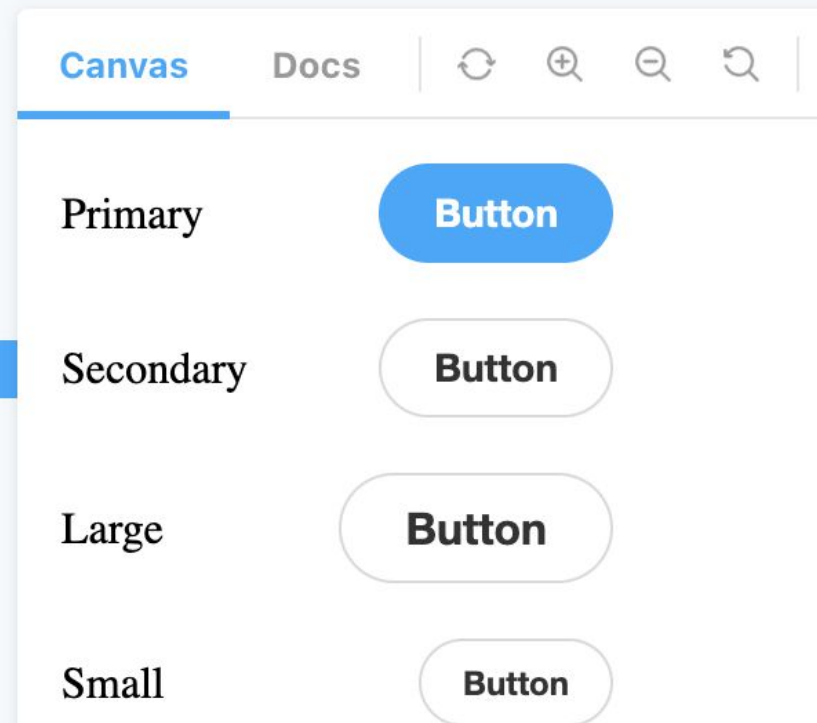
Тестирование с помощью storybook

4

Помощь при разработке в старом проекте

Библиотека UI-КОМПОНЕНТОВ

- Button
 - Primary
 - Secondary
 - Large
 - Small
 - Un Control
 - Button Default**
 - Product Simple
 - Product Editable
- Form
 - Empty Form
 - Inputed Form
- Header
 - Logged In
 - Logged Out



Какие проблемы помогает решать storybook?

1

Библиотека ui-компонентов

2

Написание документации на компоненты

3

Тестирование с помощью storybook

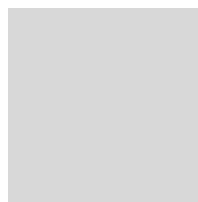
4

Помощь при разработке в старом проекте

Документация в Storybook

Компонент предназначен для того, чтобы привязывать какой-то контент в портале к позиции элемента на странице. С его помощью можно делать тултипы, дропдауны, селекты, подсказки и прочее

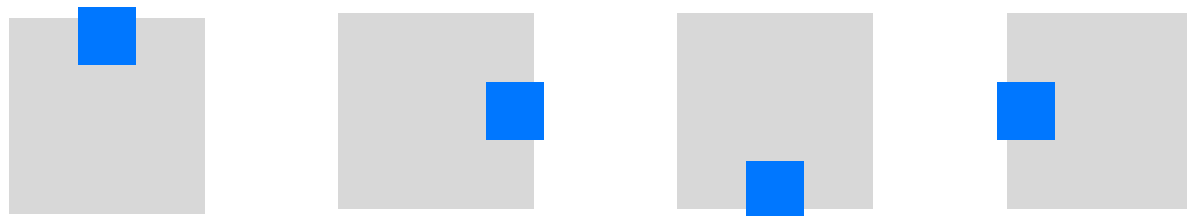
Пример с простым тултипом



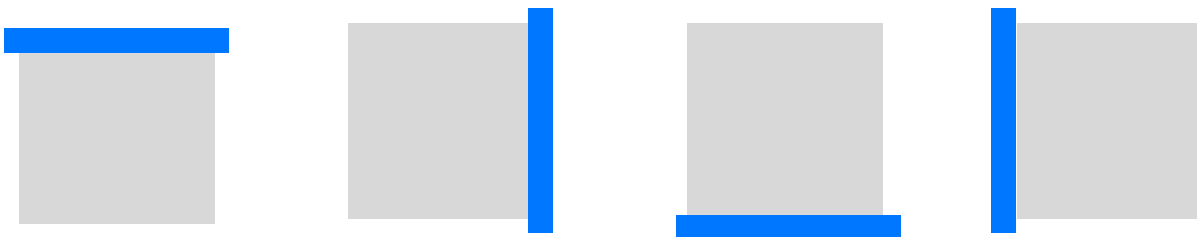
Режимы

Существуют 2 режима:

* **point** - обертка для элемента занимает точку 0x0 посередине верхней, правой или левой границы



* **side** - обертка для элемента будет занимать отрезок, совпадающий с верхней, правой, нижней или левой стороной




Документация в Storybook

- Control
- Control
- Copy Control
- Header
- Icon
- Item
- List More
- List Multi Select Control
- Members Grid
- Modal
- Multi Select
- Owner Name
- Page Block
- Pagination
- Photo Area Selector
- Popper**
- Popup Header
- Progress
- Pure Multi Select
- Radio
- Select
- Slider
- Spinner
- Sprite Animation
- Stats Table
- Stub
- Submit Area
- Switch
- Tabs
- Text Truncate
- Textarea
- Text Input
- Tip
- Tip New
- Toggle Button
- Marker

Canvas Docs | ↺ 🔍 🔍 🔍 🖼

Компонент предназначен для того, чтобы привязывать какой-то контент в портале к позиции элемента на странице. С его помощью можно делать тултипы, дропдауны, селекты, подсказки и прочее.


Пример с простым тутипом




Режимы

Существует 2 режима:

- * *point* — обертка для элемента будет занимать точку 0x0 посередине верхней, правой, нижней или левой границы.



- * *side* — обертка для элемента будет занимать отрезок, совпадающий с верхней, правой, нижней или левой стороной.



Какие проблемы помогает решать storybook?

1

Библиотека ui-компонентов

2

Написание документации на компоненты

3

Тестирование с помощью storybook

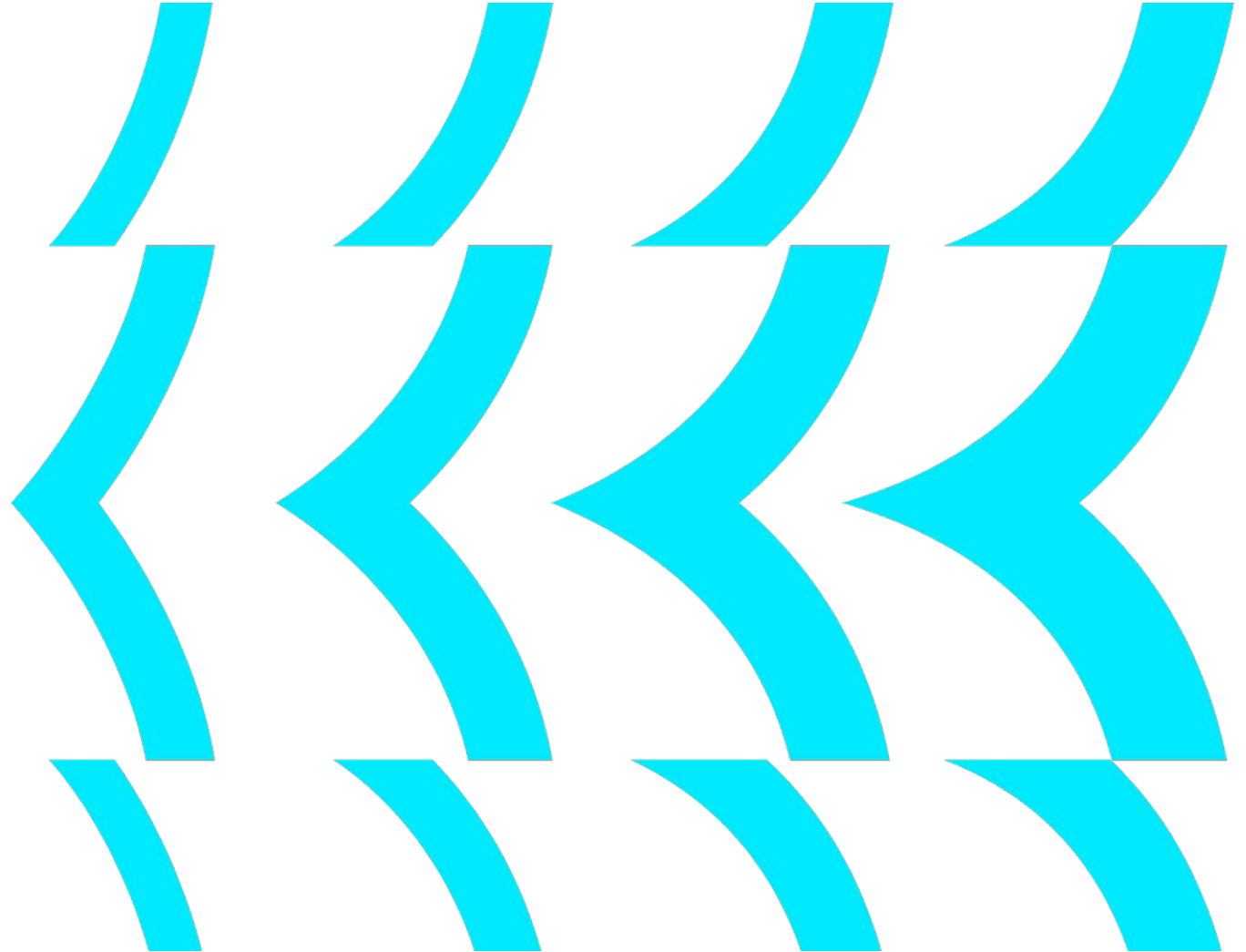
4

Помощь при разработке в старом проекте

Тестирование с помощью storybook



Скриншот-тестирование



Тестирование с помощью storybook



Скриншот-тестирование

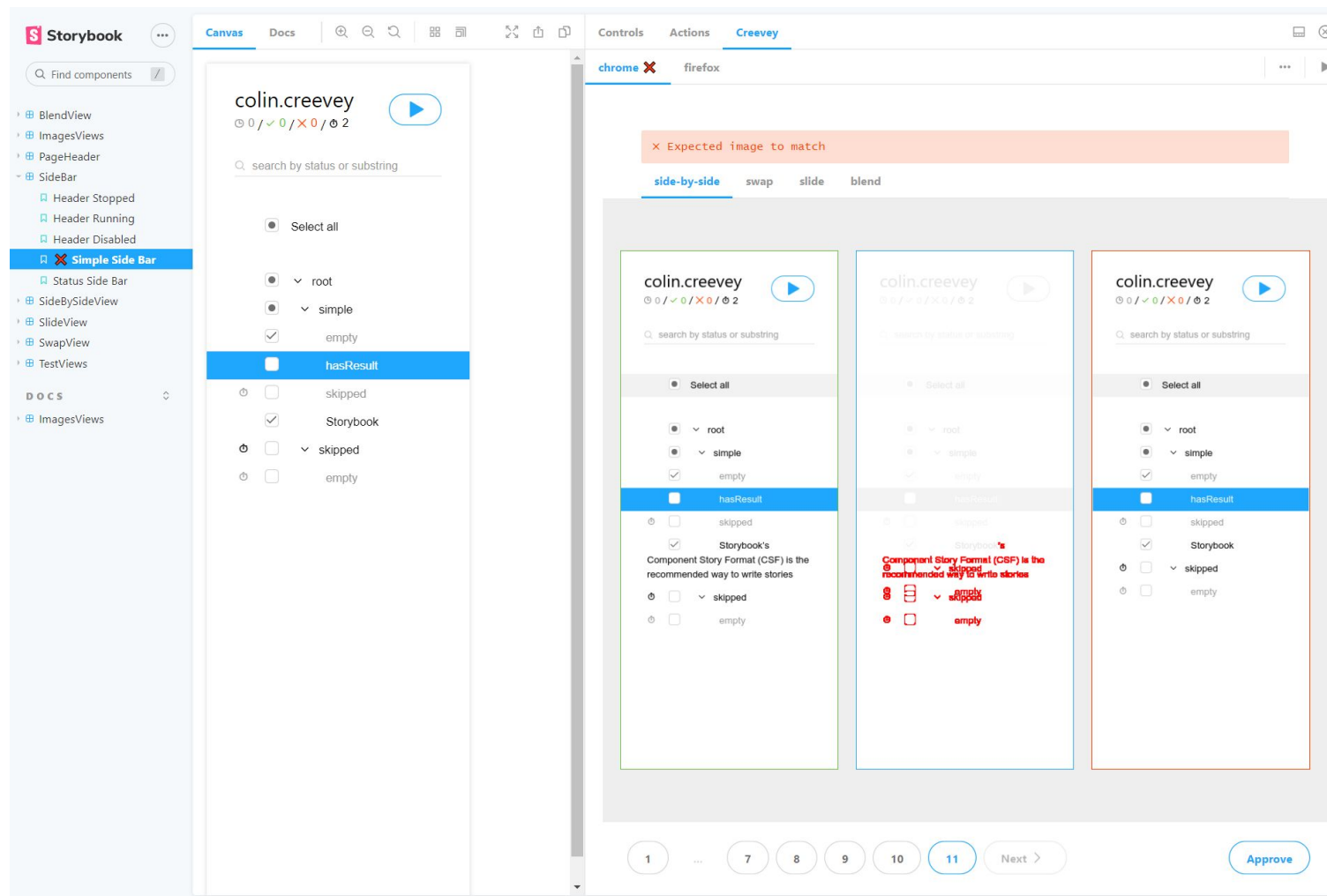


storybook-accessibility



Скриншот тестирование

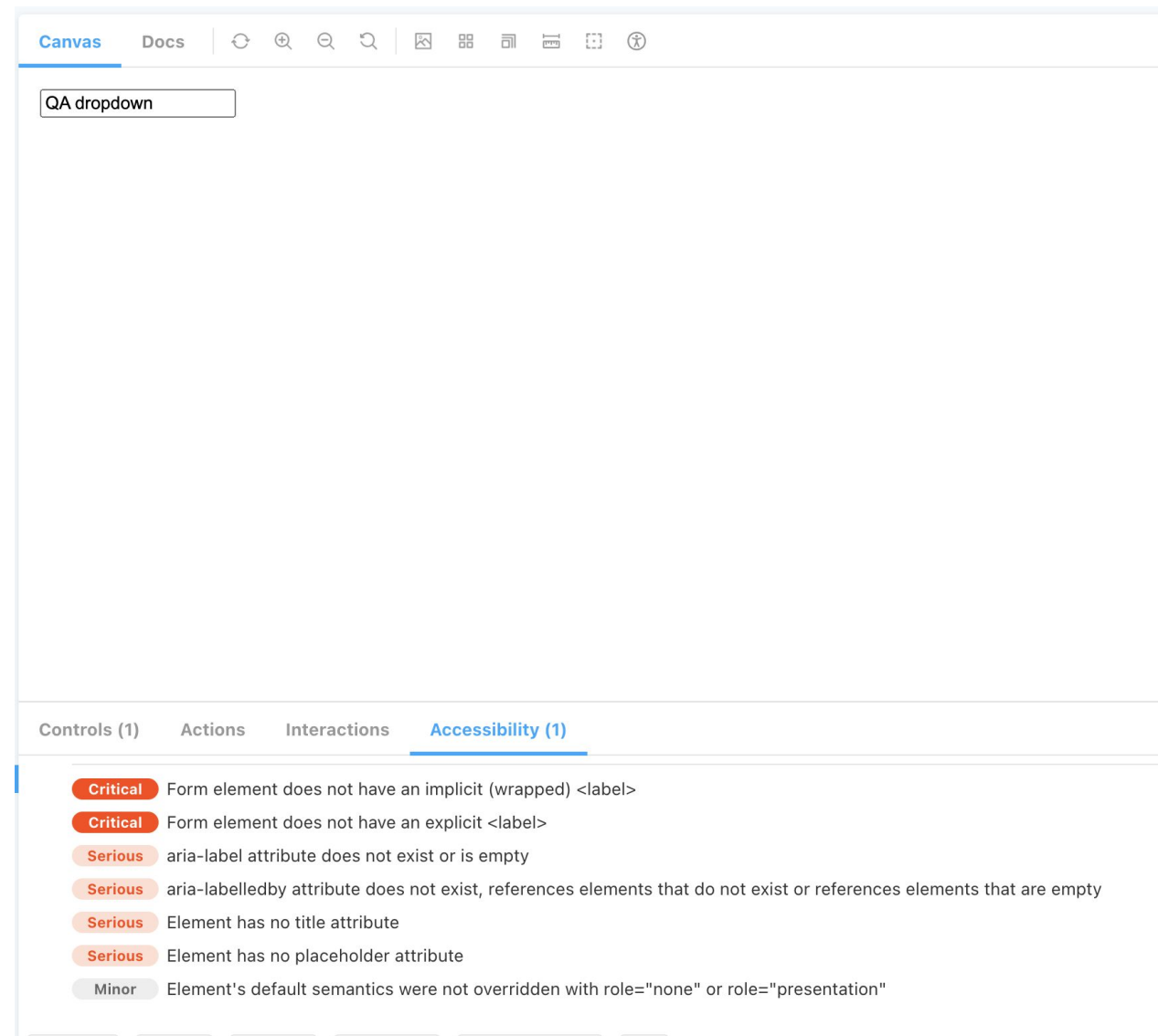
- creevey
- loki
- storyshots
- chromatic
- hermione
- happo



Сравнение библиотек для скриншот-тестирования

Features\Tools	Creevey	Loki	Storyshots	Hermione	BackstopJS	Percy/Happo	Chromatic
Easy-to-Setup	✓	✓	⚠	🚫	🚫	✓	✓
Storybook Support	✓	✓	✓	🚫	🚫	✓	✓
Run tests from Storybook UI	✓	🚫	🚫	🚫	🚫	🚫	🚫
Cross-browser	✓	⚠	🚫	✓	🚫	✓	✓
Test Interaction	✓	🚫	⚠	✓	✓	🚫	🚫
UI Test Runner	✓	🚫	🚫	✓	✓	✓	✓
Built-in Docker	✓	✓	🚫	🚫	✓	⚠	⚠
Tests hot-reload	✓	🚫	🚫	🚫	🚫	🚫	🚫
OSS/SaaS	OSS	OSS	OSS	OSS	OSS	SaaS	SaaS

Storybook accessibility



The screenshot shows the Storybook Accessibility panel for a component named "QA dropdown". The panel is divided into tabs: "Controls (1)", "Actions", "Interactions", and "Accessibility (1)". The "Accessibility (1)" tab is selected and displays a list of accessibility issues:

- Critical** Form element does not have an implicit (wrapped) <label>
- Critical** Form element does not have an explicit <label>
- Serious** aria-label attribute does not exist or is empty
- Serious** aria-labelledby attribute does not exist, references elements that do not exist or references elements that are empty
- Serious** Element has no title attribute
- Serious** Element has no placeholder attribute
- Minor** Element's default semantics were not overridden with role="none" or role="presentation"

Какие проблемы помогает решать storybook?

1

Библиотека ui-компонентов

2

Написание документации на компоненты

3

Тестирование с помощью storybook

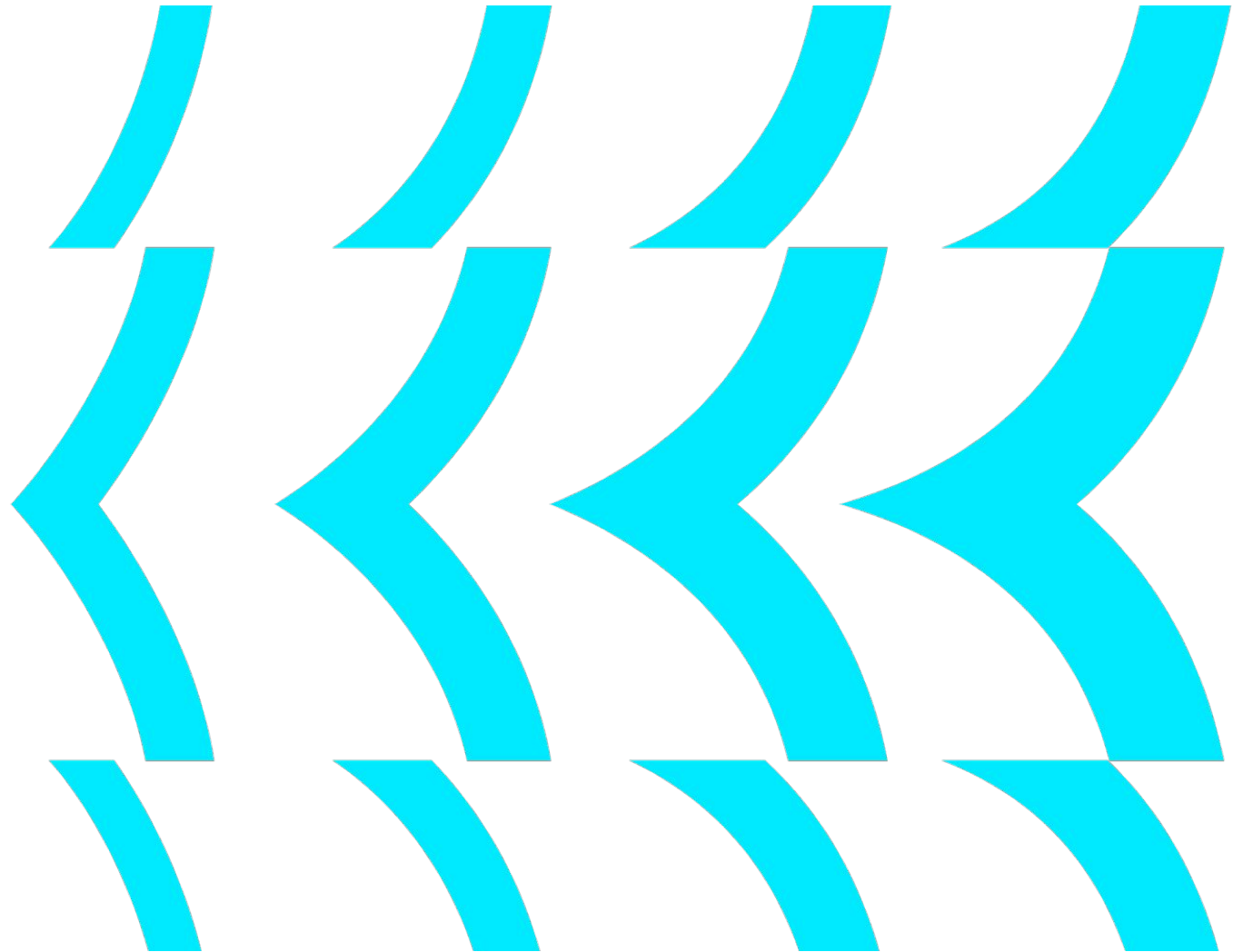
4

Помощь при разработке в старом проекте

Помощь в рефакторинге старого приложения



Изолированный запуск
компонентов



Помощь в рефакторинге старого приложения



Изолированный запуск
компонентов



Помощь в решении
проблем с зависимостями



Какие проблемы помогает решать storybook?

1

Библиотека ui-
компонентов

2

Написание
документации на
компоненты

3

Тестирование с
помощью storybook

4

Помощь при
разработке в старом
проекте

Примеры решения проблем из нашей практики



Какие проблемы помогает решать `storybook`?

1

Рефакторинг

2

Разработка новой функциональности

Рефакторинг

Какую проблему решали?

- Большой проект, в котором запущен кусок react-приложения, разработка проекта долгая и неудобная

Название поста

Описание

+ Создать пост

Помощь в размещении

Рефакторинг

Какую проблему решали?

- Большой проект, в котором запущен кусок react-приложения, разработка проекта долгая и неудобная
- Решили запустить react-приложение внутри storybook

Название поста

Описание

+ Создать пост

Помощь в размещении

Рефакторинг

Какую проблему решали?

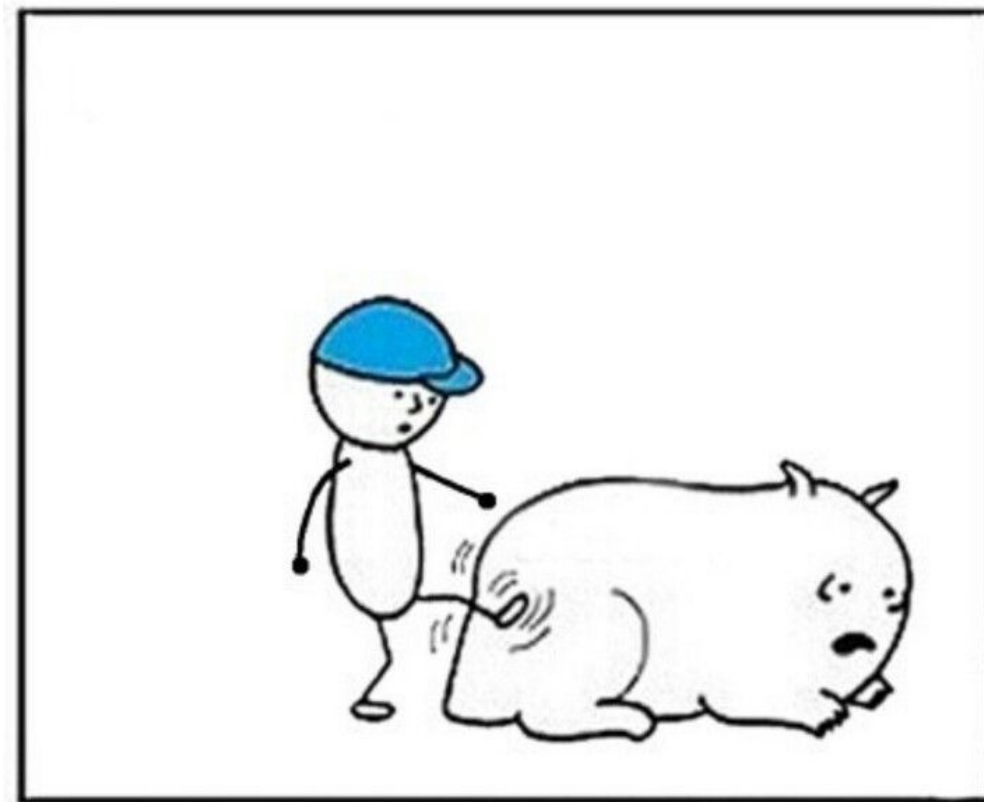
- Большой проект, в котором запущен кусок react-приложения, разработка проекта долгая и неудобная
- Решили запустить react-приложение внутри storybook

```
✖ ▶ TypeError: Cannot read 754.afb220b9.iframe.bundle.js:2
properties of undefined (reading 'apiDomain')
    at getExternalAppOptions (main.ea23cc04.iframe.bundl
e.js:2:7053806)
    at Widget_Widget (main.ea23cc04.iframe.bundle.js:2:73
30674)
    at Ch (754.afb220b9.iframe.bundle.js:2:4365102)
    at li (754.afb220b9.iframe.bundle.js:2:4374575)
    at ck (754.afb220b9.iframe.bundle.js:2:4417918)
    at bk (754.afb220b9.iframe.bundle.js:2:4404284)
    at ak (754.afb220b9.iframe.bundle.js:2:4404215)
    at Tj (754.afb220b9.iframe.bundle.js:2:4404084)
    at Lj (754.afb220b9.iframe.bundle.js:2:4401109)
    at 754.afb220b9.iframe.bundle.js:2:4351018
```

Рефакторинг

Какую проблему решали?

- Большой проект, в котором запущен кусок react-приложения, разработка проекта долгая и неудобная
- Решили запустить react-приложение внутри storybook



Рефакторинг

Какую проблему решали?

- Большой проект, в котором запущен кусок react-приложения, разработка проекта долгая и неудобная
- Решили запустить react-приложение внутри storybook

С какими проблемами столкнулись?

- Приложение ходит в арі

Рефакторинг

Какую проблему решали?

- Большой проект, в котором запущен кусок react-приложения, разработка проекта долгая и неудобная
- Решили запустить react-приложение внутри storybook

С какими проблемами столкнулись?

- Приложение ходит в арі
- Вызов глобальных методов крашит приложение

Рефакторинг

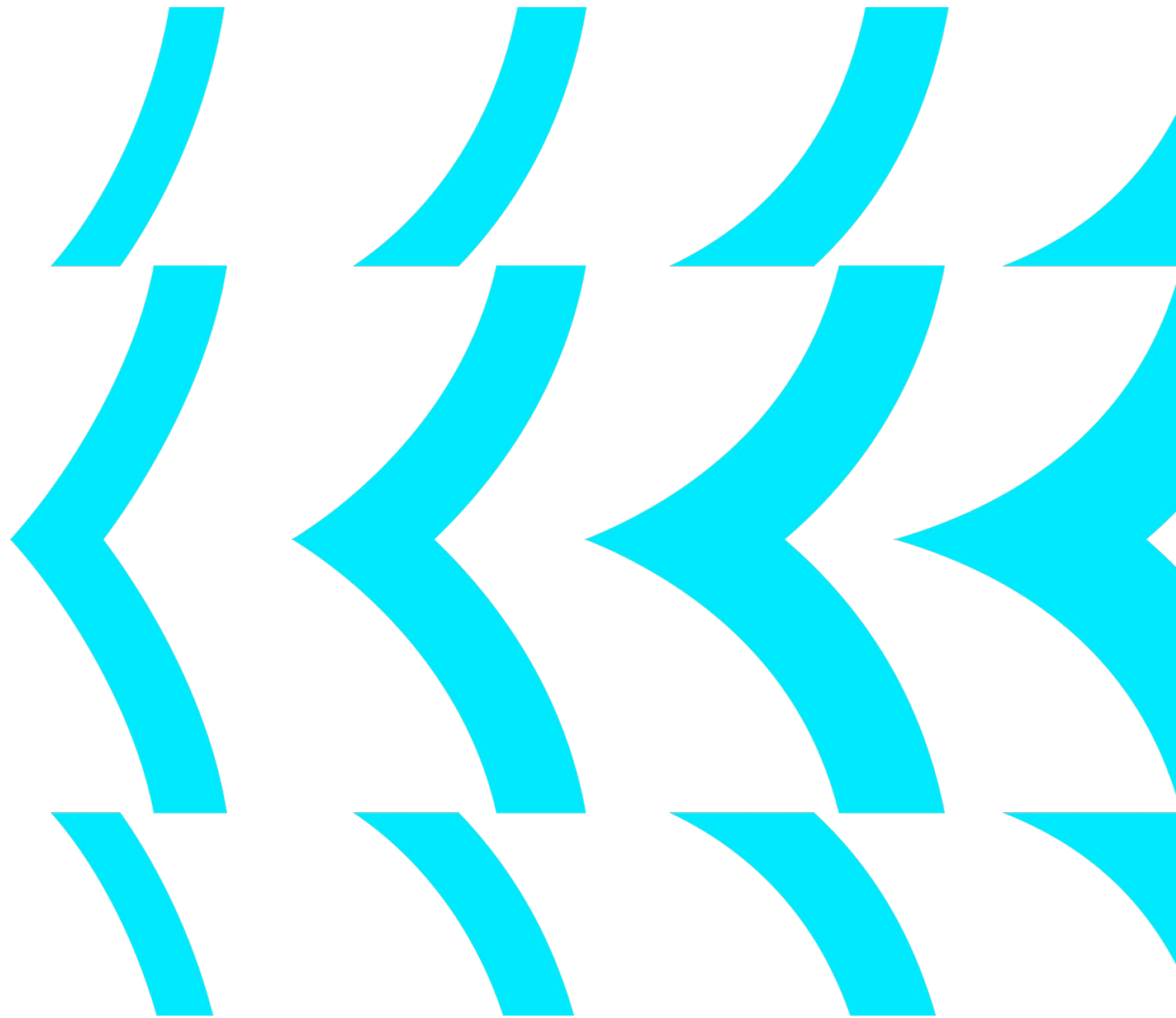
Какую проблему решали?

- Большой проект, в котором запущен кусок react-приложения, разработка проекта долгая и неудобная
- Решили запустить react-приложение внутри storybook

С какими проблемами столкнулись?

- Приложение ходит в арі
- Вызов глобальных методов крашит приложение
- Некорректно отображаются глобальные стили

**Как решали
проблемы?**



Проблема зависимостей.

Как выглядел компонент?

```
import { apiService } from "../services/apiService";
import { analyticsService } from "../services/analyticsService";
import { routingService } from "../services/routingService";

type Props = {
  analyticsData: Data;
};

export const PostCreate: React.FC<Props> = ({ analyticsData }) => {
  const handleCreatePost = async () => {
    analyticsService.sendPostCreate(analyticsData);
    const { postId } = await apiService("/create-post");

    routingService.push(`/post/${postId}`);
  };

  return (
    <div>
      <button onClick={handleCreatePost}>Создать пост</button>
    </div>
  );
};
```

Проблема зависимостей. Как выглядел компонент?

```
import { apiService } from "../services/apiService";
import { analyticsService } from "../services/analyticsService";
import { routingService } from "../services/routingService";

type Props = {
  analyticsData: Data;
};

export const PostCreate: React.FC<Props> = ({ analyticsData }) => {
  const handleCreatePost = async () => {
    analyticsService.sendPostCreate(analyticsData);
    const { postId } = await apiService("/create-post");

    routingService.push(`/post/${postId}`);
  };

  return (
    <div>
      <button onClick={handleCreatePost}>Создать пост</button>
    </div>
  );
};
```


Проблема зависимостей. Как выглядел компонент?

```
import { apiService } from "../services/apiService";
import { analyticsService } from "../services/analyticsService";
import { routingService } from "../services/routingService";

type Props = {
  analyticsData: Data;
};

export const PostCreate: React.FC<Props> = ({ analyticsData }) => {
  const handleCreatePost = async () => {
    analyticsService.sendPostCreate(analyticsData);
    const { postId } = await apiService("/create-post");

    routingService.push(`/post/${postId}`);
  };

  return (
    <div>
      <button onClick={handleCreatePost}>Создать пост</button>
    </div>
  );
};
```

Проблема зависимостей. Как выглядел компонент?

```
import { apiService } from "../services/apiService";
import { analyticsService } from "../services/analyticsService";
import { routingService } from "../services/routingService";

type Props = {
  analyticsData: Data;
};

export const PostCreate: React.FC<Props> = ({ analyticsData }) => {
  const handleCreatePost = async () => {
    analyticsService.sendPostCreate(analyticsData);
    const { postId } = await apiService("/create-post");

    routingService.push(`/post/${postId}`);
  };

  return (
    <div>
      <button onClick={handleCreatePost}>Создать пост</button>
    </div>
  );
};
```

Проблема зависимостей.

Делаем разную реализацию сервиса в зависимости от тестовой среды

```
type AnalyticsService = {  
  sendPostCreate: (data: Data) => void;  
};
```

Проблема зависимостей.

Делаем разную реализацию сервиса в зависимости от тестовой среды

```
type AnalyticsService = {  
  sendPostCreate: (data: Data) => void;  
};
```

```
export const AnalyticsServiceInstance: AnalyticsService = {  
  sendPostCreate: analyticsService.sendPostCreate,  
};
```

Проблема зависимостей.

Делаем разную реализацию сервиса в зависимости от тестовой среды

```
type AnalyticsService = {  
  sendPostCreate: (data: Data) => void;  
};
```

```
export const AnalyticsServiceMock: AnalyticsService = {  
  sendPostCreate: jest.fn,  
};
```

Проблема зависимостей.

Делаем разную реализацию сервиса в зависимости от тестовой среды

```
type AnalyticsService = {  
  sendPostCreate: (data: Data) => void;  
};
```

```
export const AnalyticsServiceStorybook: AnalyticsService = {  
  sendPostCreate: ({ params }) => {  
    action("Post Create")(params);  
  },  
};
```

Проблема зависимостей.

Как подключить сервис в storybook?

```
export const AnalyticsServiceContext = React.createContext<AnalyticsService>(
  {} as AnalyticsService
);

export const AnalyticsServiceProvider: React.FC = ({ children }) => {
  return (
    <AnalyticsServiceContext.Provider value={AnalyticsServiceInstance}>
      {children}
    </AnalyticsServiceContext.Provider>
  );
};
```

Проблема зависимостей.

Как подключить сервис в storybook?

```
export const withAnalyticsProvider = (Story: React.FC) => {  
  return (  
    <AnalyticsServiceContext.Provider value={AnalyticsServiceStorybook}>  
      <Story />  
    </AnalyticsServiceContext.Provider>  
  );  
};
```

```
export default {  
  title: "Post/PostCreate",  
  component: PostCreate,  
  decorators: [withAnalyticsProvider],  
};
```


Проблема зависимостей.

Внешние зависимости

```
import { apiService } from "./services/apiService";
import { analyticsService } from "./services/analyticsService";
import { routingService } from "./services/routingService";

type Props = {
  analyticsData: Data;
};

export const PostCreate: React.FC<Props> = ({ analyticsData }) => {
  const handleCreatePost = async () => {
    analyticsService.sendPostCreate(analyticsData);
    const { postId } = await apiService("/create-post");

    routingService.push(`/post/${postId}`);
  };

  return (
    <div>
      <button onClick={handleCreatePost}>Создать пост</button>
    </div>
  );
};
```

Проблема зависимостей.

Внешние зависимости

```
import { apiService } from "../services/apiService";
import { routingService } from "../services/routingService";

type Props = {
  analyticsData: Data;
};

export const PostCreate: React.FC<Props> = ({ analyticsData }) => {
  const AnalyticsService = React.useContext(AnalyticsServiceContext);

  const handleCreatePost = async () => {
    AnalyticsService.sendPostCreate(analyticsData);
    const { postId } = await apiService("/create-post");

    routingService.push(`/post/${postId}`);
  };

  return (
    <div>
      <button onClick={handleCreatePost}>Создать пост</button>
    </div>
  );
};
```

Проблема зависимостей.

Внешние зависимости

```
import { routingService } from "../services/routingService";

type Props = {
  analyticsData: Data;
};

export const PostCreate: React.FC<Props> = ({ analyticsData }) => {
  const AnalyticsService = React.useContext(AnalyticsServiceContext);
  const ApiService = React.useContext(ApiServiceContext);

  const handleCreatePost = async () => {
    AnalyticsService.sendPostCreate(analyticsData);
    const { postId } = await ApiService("/create-post");

    routingService.push(`/post/${postId}`);
  };

  return (
    <div>
      <button onClick={handleCreatePost}>Создать пост</button>
    </div>
  );
};
```

Проблема зависимостей.

Внешние зависимости

```
type Props = {
  analyticsData: Data;
};

export const PostCreate: React.FC<Props> = ({ analyticsData }) => {
  const AnalyticsService = React.useContext(AnalyticsServiceContext);
  const ApiService = React.useContext(ApiServiceContext);
  const RoutingService = React.useContext(RoutingServiceContext);

  const handleCreatePost = async () => {
    AnalyticsService.sendPostCreate(analyticsData);
    const { postId } = await ApiService("/create-post");

    RoutingService.push(`/post/${postId}`);
  };

  return (
    <div>
      <button onClick={handleCreatePost}>Создать пост</button>
    </div>
  );
};
```

Проблема зависимостей. А если не через react-context?

```
import { AnalyticsService } from './services/Analytics'  
import { postCreate } from './postCreate'  
  
export function createPost() {  
  return async (dispatch) => {  
    AnalyticsService.sendPrePostCreate()  
    await dispatch(postCreate())  
    AnalyticsService.sendAfterPostCreate()  
  }  
}
```

Проблема зависимостей. А если не через react-context?

```
import { AnalyticsService } from './services/Analytics'  
import { postCreate } from './postCreate'  
  
export function createPost() {  
  return async (dispatch) => {  
    AnalyticsService.sendPrePostCreate()  
    await dispatch(postCreate())  
    AnalyticsService.sendAfterPostCreate()  
  }  
}
```

Проблема зависимостей. Подключение в redux-thunk

```
export const createStore = <E extends Record<string, unknown>>({
  initialState = {},
}: {
  initialState?: unknown;
} = {}) => {
  return createStore(
    defaultReducer,
    initialState,
    composeWithDevTools(applyMiddleware(thunk)),
  );
};
```

Проблема зависимостей. Подключение в redux-thunk

```
export const createStore = <E extends Record<string, unknown>>({
  extraArgument,
  initialState = {},
}): {
  extraArgument?: E;
  initialState?: unknown;
} = {} => {
  return createStore(
    defaultReducer,
    initialState,
    composeWithDevTools(applyMiddleware(thunk.withExtraArgument(extraArgument))),
  );
};
```


Проблема зависимостей. Подключение в redux-thunk

```
export const createStore = <E extends Record<string, unknown>>({
  extraArgument,
  initialState = {},
}): {
  extraArgument?: E;
  initialState?: unknown;
} = {} => {
  return createStore(
    defaultReducer,
    initialState,
    composeWithDevTools(applyMiddleware(thunk.withExtraArgument(extraArgument))),
  );
};
```

```
const store = createStore<Extra>({
  extraArgument: {
    AjaxService: AjaxServiceInstance,
    AnalyticsService: AnalyticsServiceInstance,
    RoutingService: RoutingServiceInstance,
  },
});
```

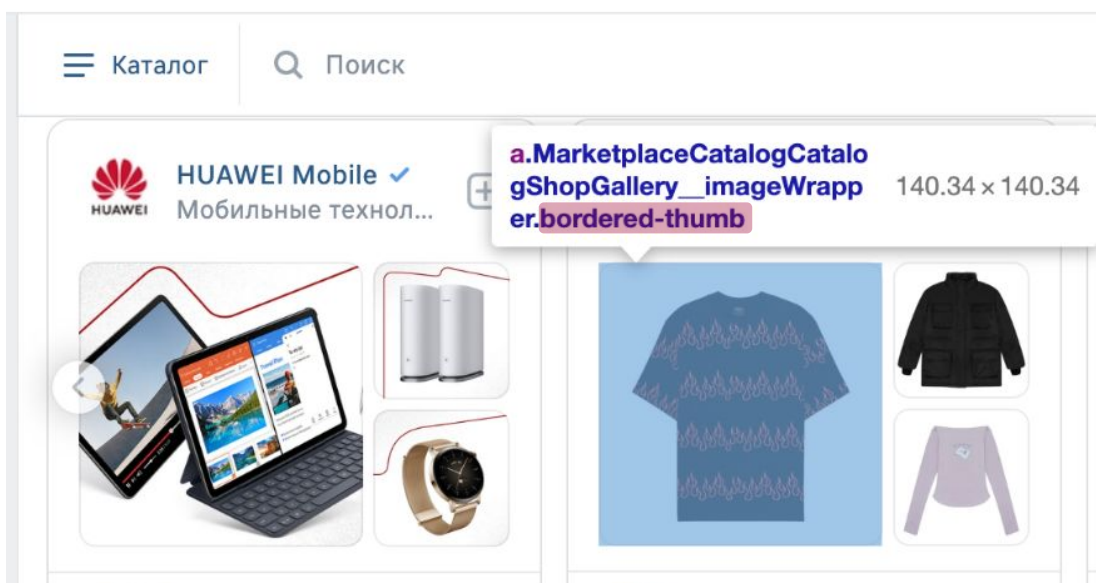
Проблема зависимостей. А если не через react-context?

```
import { AnalyticsService } from './services/Analytics'  
import { postCreate } from './postCreate'  
  
export function createPost() {  
  return async (dispatch) => {  
    AnalyticsService.sendPrePostCreate()  
    await dispatch(postCreate())  
    AnalyticsService.sendAfterPostCreate()  
  }  
}
```

Проблема зависимостей. А если не через react-context?

```
import { postCreate } from './postCreate'  
  
export function createPost() {  
  return async (dispatch, _, { AnalyticsService }) => {  
    AnalyticsService.sendPrePostCreate()  
    await dispatch(postCreate())  
    AnalyticsService.sendAfterPostCreate()  
  }  
}
```

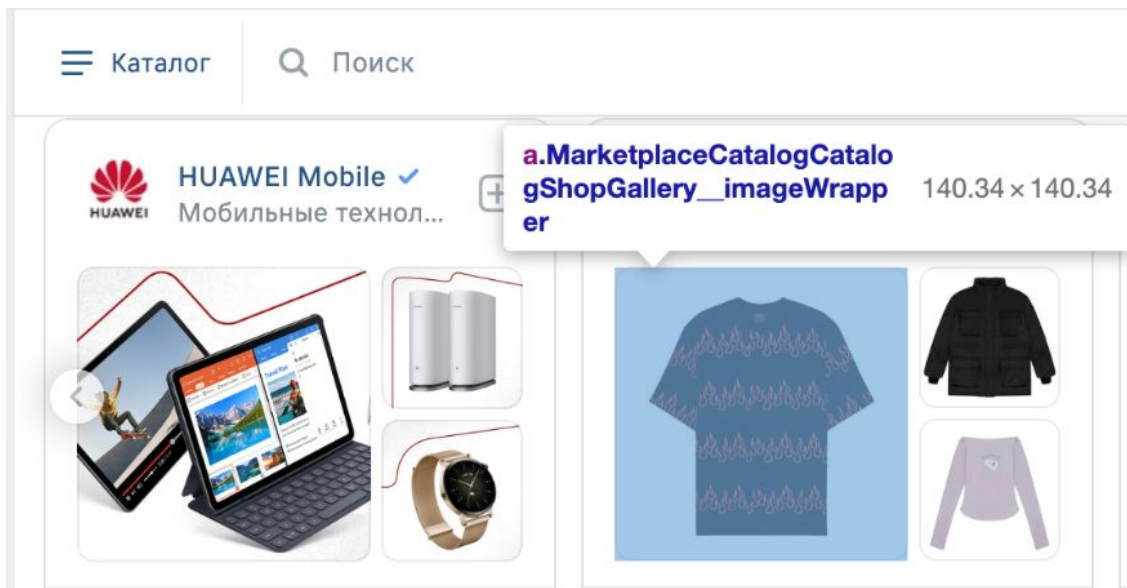
Проблема зависимостей. Глобальные стили



```
</div> flex
▼ <div class="MarketplaceCatalogCatalogShopGallery MarketplaceCatalogCatalogShopShop_gallery"> grid
  ▼ <a class="MarketplaceCatalogCatalogShopGallery_imageWrapper bordered-thumb" href="/market-6437840?w=product-6437840_5497217"> == $0
    
    ::after
  </a>
  ▼ <a class="MarketplaceCatalogCatalogShopGallery_imageWrapper bordered-thumb" href="/market-6437840?w
```

Проблема зависимостей.

Глобальные стили



```
...  
</div> flex  
▼ <div class="MarketplaceCatalogCatalogShopGallery MarketplaceCatalogCatalogShopShop__gallery"> grid  
...  
▼ <a class="MarketplaceCatalogCatalogShopGallery__imageWrapper" href="/market-6437840?w=product-6437840_5497217" style="> == $0  
  
::after  
</a>  
...
```

Решение: Перенести эти стили в компонент

Проблема зависимостей. Глобальные стили

```
import  
'../static/css/all/common.css'  
;
```

Решение: Подключить в `preview.js`

Проблема зависимостей.

Внешние зависимости

Что в итоге получилось?

- Компонент не знает, как реализован сервис

Проблема зависимостей.

Внешние зависимости

Что в итоге получилось?

- Компонент не знает, как реализован сервис
- Стало сильно проще писать story

Проблема зависимостей.

Внешние зависимости

Что в итоге получилось?

- Компонент не знает, как реализован сервис
- Стало сильно проще писать story
- Стало проще писать компоненты

Проблема зависимостей.

Внешние зависимости

Что в итоге получилось?

- Компонент не знает, как реализован сервис
- Стало сильно проще писать story
- Стало проще писать компоненты
- Заодно отрефакторили компонент

Внедрение `storybook` от большого к маленькому

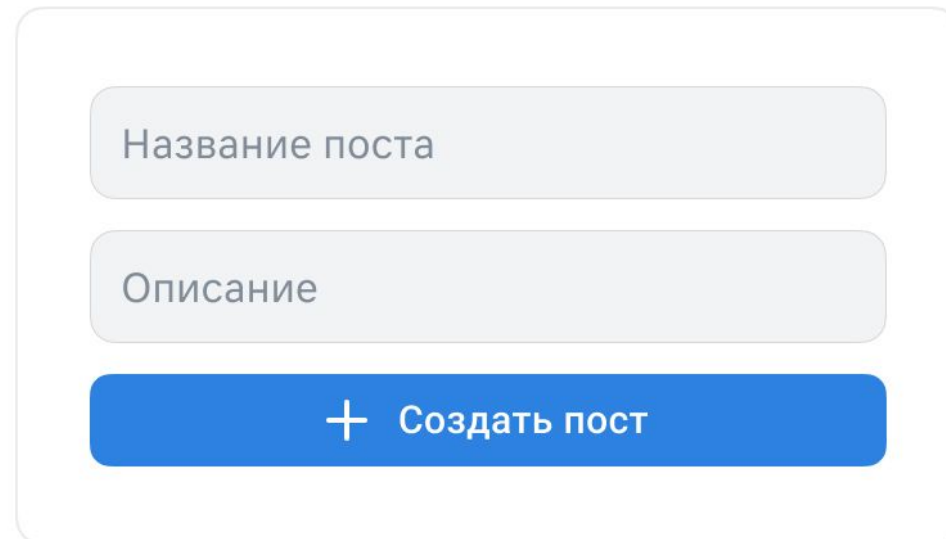
Когда стоит начинать
с больших компонентов?

Когда у нас компонент не обернут
в `storybook` и мы хотим его отрефачить

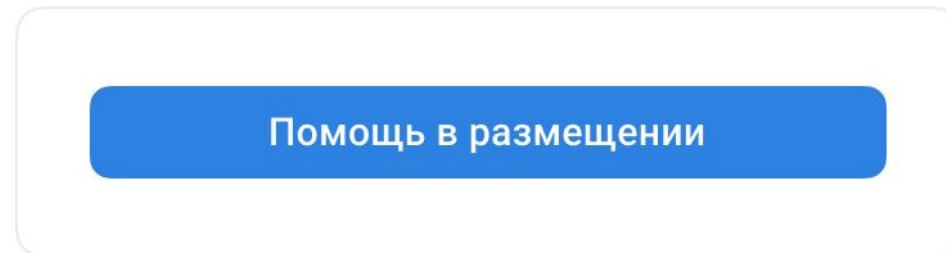
Когда стоит начинать с больших компонентов?

Как происходит разработка?

- Пытаемся завести компонент



A form for creating a post, consisting of two text input fields and a submit button. The first field is labeled 'Название поста' (Post title) and the second is labeled 'Описание' (Description). The submit button is blue with a white plus sign and the text 'Создать пост' (Create post).

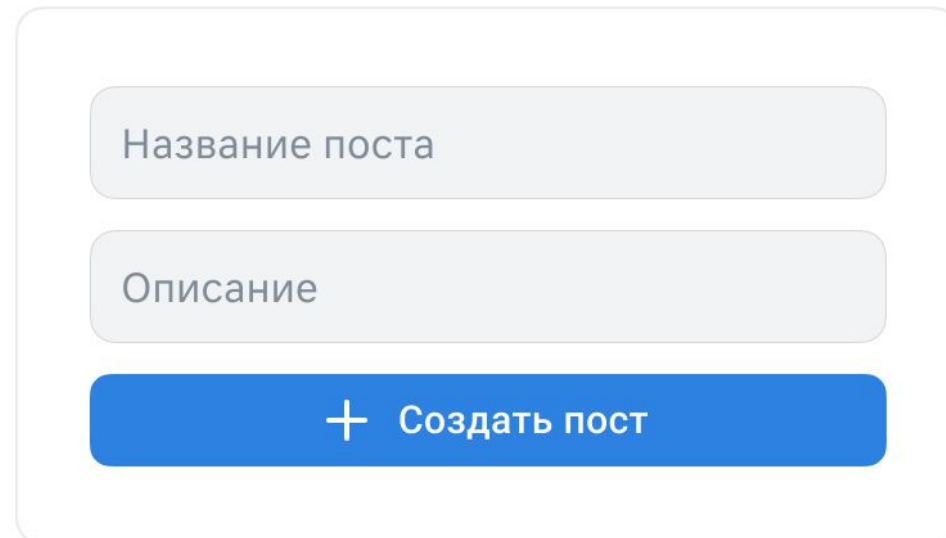


A blue button with white text that reads 'Помощь в размещении' (Help with posting).

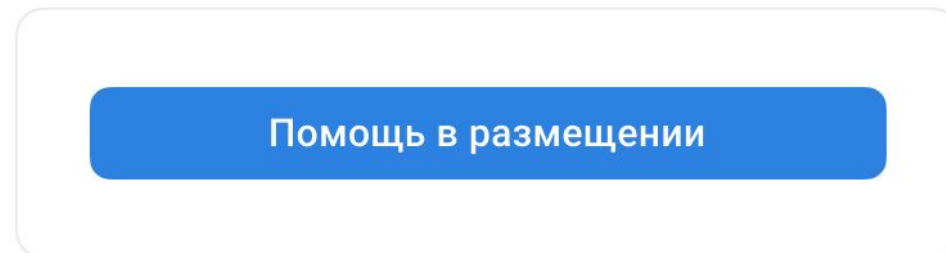
Когда стоит начинать с больших компонентов?

Как происходит разработка?

- Пытаемся завести компонент
- Обеспечение работоспособности компонента



A form for creating a post, consisting of two text input fields and a submit button. The first input field is labeled "Название поста" (Post title) and the second is labeled "Описание" (Description). Below the inputs is a blue button with a white plus sign and the text "Создать пост" (Create post).

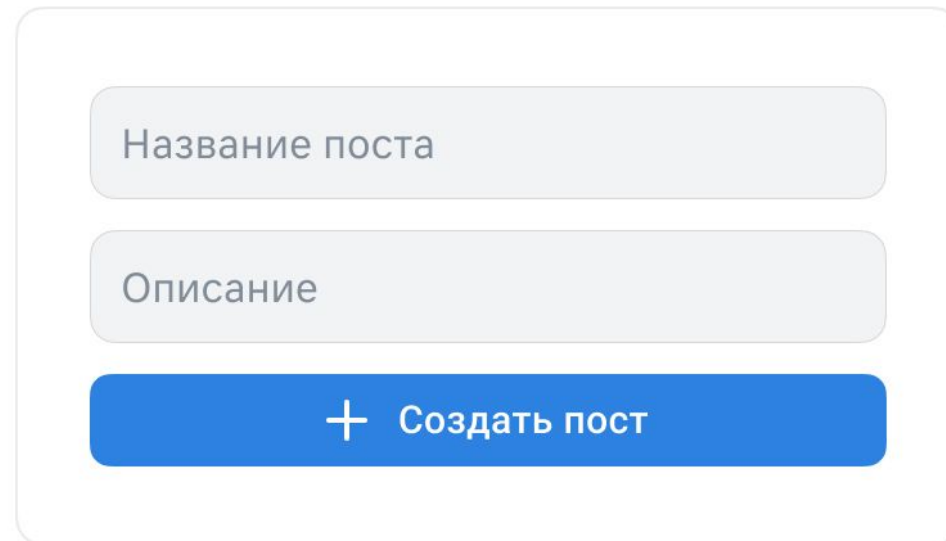


A blue button with white text that reads "Помощь в размещении" (Help with posting).

Когда стоит начинать с больших компонентов?

Как происходит разработка?

- Пытаемся завести компонент
- Обеспечение работоспособности компонента
- Story для формы создания поста



Название поста

Описание

+ Создать пост

Когда стоит начинать с больших компонентов?

Как происходит разработка?

- Пытаемся завести компонент
- Обеспечение работоспособности компонента
- Story для формы создания поста
- Story для кнопки создания поста

+ Создать пост

Какие проблемы помогает решать `storybook`?

1

Рефакторинг

2

Разработка новой функциональности

Внедрение *storybook* от маленького к большому

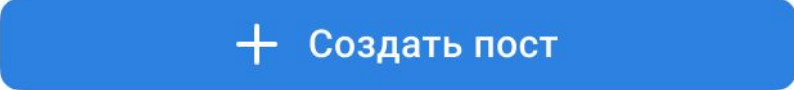
Когда стоит начинать с маленьких
компонентов?

Когда начинаем разработку нового

Внедрение storybook от маленького к большому

Как происходит разработка?

- Кнопка создания поста

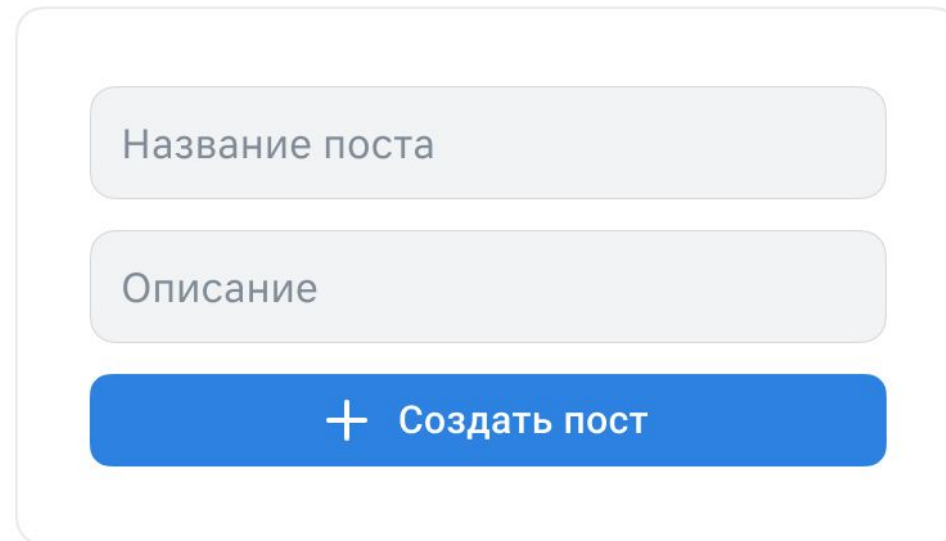


+ Создать пост

Когда стоит начинать с маленьких компонентов?

Как происходит разработка?

- Кнопка создания поста
- Форма создания поста



Название поста

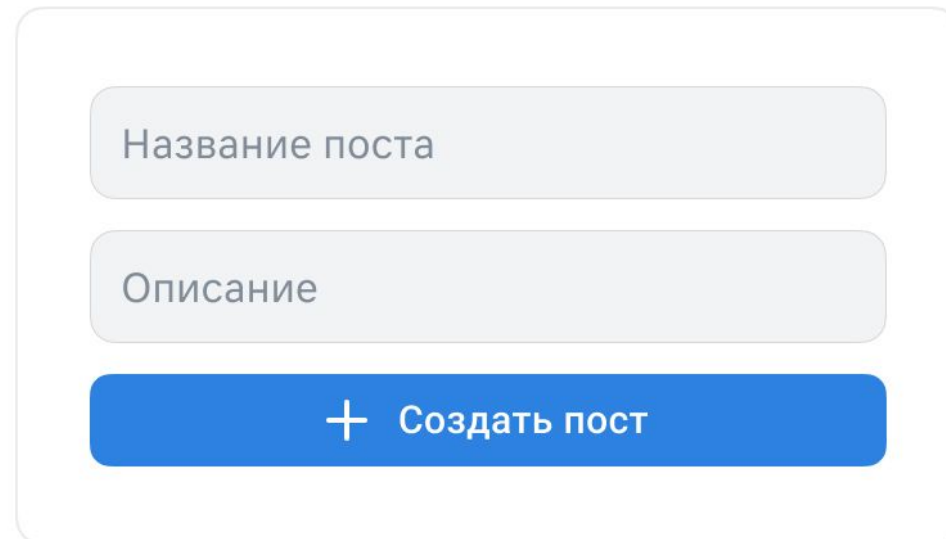
Описание

+ Создать пост

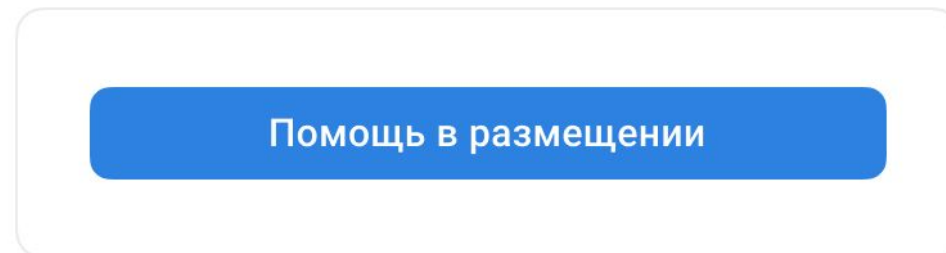
Когда стоит начинать с маленьких компонентов?

Как происходит разработка?

- Кнопка создания поста
- Форма создания поста
- Создание поста



A form for creating a post, consisting of three stacked rounded rectangular fields. The top field is labeled 'Название поста' (Post title). The middle field is labeled 'Описание' (Description). The bottom field is a blue button with a white plus sign and the text 'Создать пост' (Create post).



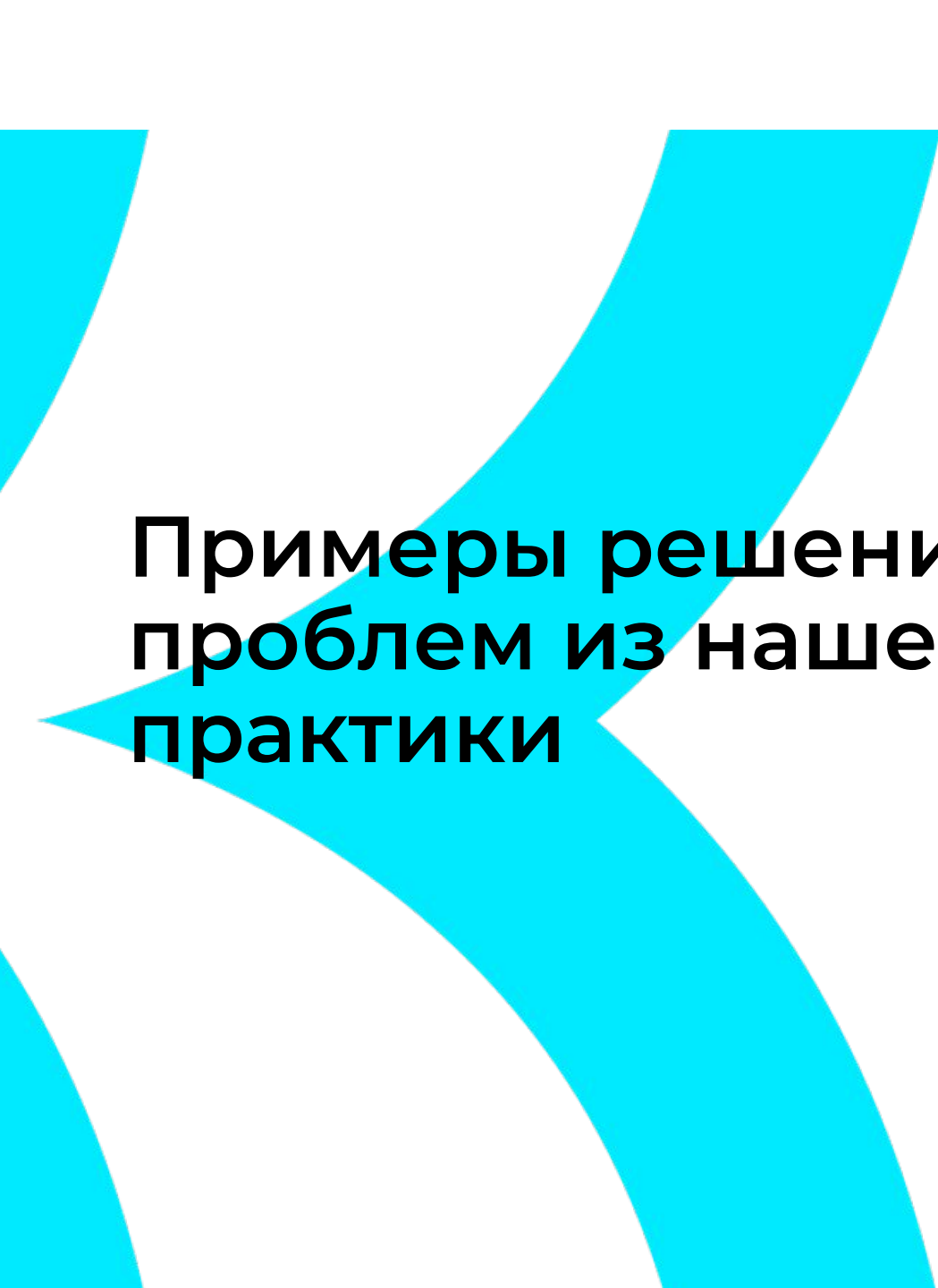
A single blue rounded rectangular button with the text 'Помощь в размещении' (Help with posting).

Когда стоит начинать с маленьких компонентов?

Как происходит разработка?

- Кнопка создания поста
- Форма создания поста
- Создания поста
- Страница создания поста

Скриншот интерфейса VK «Создание поста». В левом меню: Главная, Мои посты, Поиск, Помощь. Основная форма включает поля «Название поста» и «Описание», кнопку «+ Создать пост» и кнопку «Помощь в размещении».




Примеры решения проблем из нашей практики

1

Рефакторинг

2

Разработка новой
функциональности



Проблемы при использовании storybook

Проблемы при использовании **storybook**

1

Инфраструктура
storybook

2

Свой “storybook”

3

Культура написания
stories

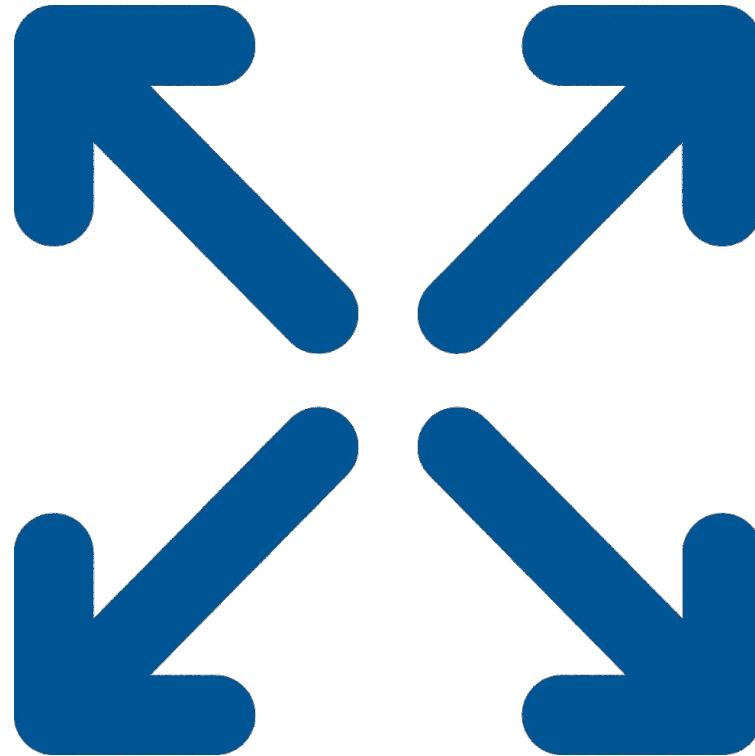
Инфраструктура `storybook`.

- Сборки



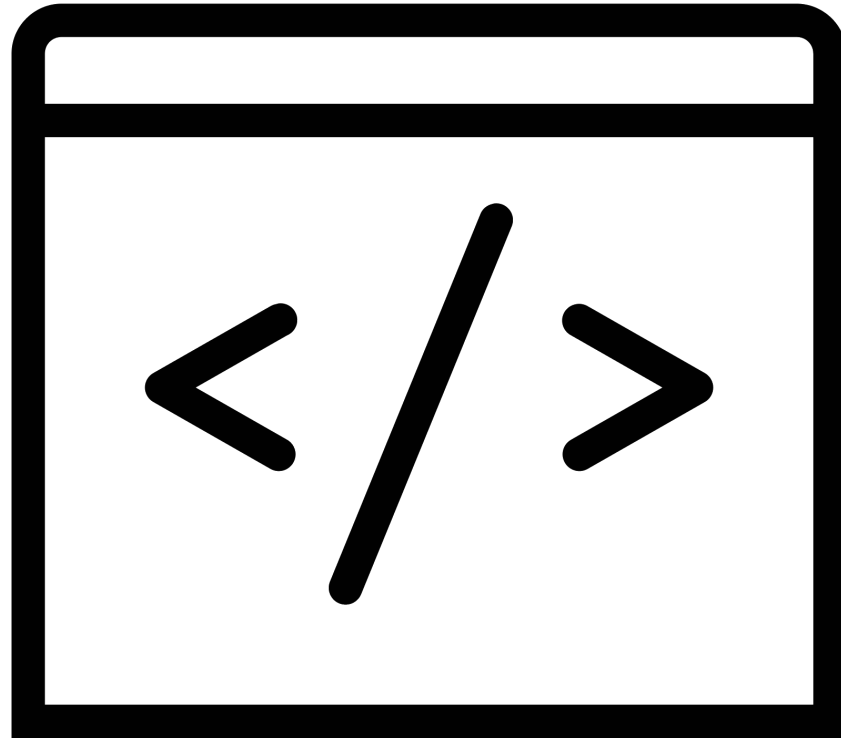
Инфраструктура *storybook*.

- Сборки
- Масштабируемость



Инфраструктура storybook.

- Сборки
- Масштабируемость
- Iframe



Инфраструктура `storybook`. Сборки

- Чем сложнее сборка у приложения, тем сложнее поддерживать `storybook`

Инфраструктура `storybook`. Сборки

- Чем сложнее сборка у приложения, тем сложнее поддерживать `storybook`
- Используем в `storybook` и приложении один конфиг

Инфраструктура storybook. Сборки

- Чем сложнее сборка у приложения, тем сложнее поддерживать storybook
- Используем в storybook и приложении один конфиг
- Используем разные конфиги для storybook и приложения

Используем в `storybook` и приложения один конфиг

- Зависимости `storybook` и приложения могут отличаться

Используем в `storybook` и приложении один конфиг

- Зависимости `storybook` и приложения могут отличаться
- При обновлении зависимостей приложения приходится обновлять библиотеки для `storybook`

Используем в `storybook` и приложении один конфиг

- Зависимости `storybook` и приложения могут отличаться
- При обновлении зависимостей приложения приходится обновлять библиотеки для `storybook`

```
{  
  "name": "@storybook/react",  
  ...  
  "dependencies": {  
    ...  
    "@babel/preset-flow": "^7.12.1",  
    "@babel/preset-react": "^7.12.10",  
    ...  
  },  
  ...  
}
```

Используем в `storybook` и приложении разный конфиг

- Тяжелая настройка

Используем в `storybook` и приложении разный конфиг

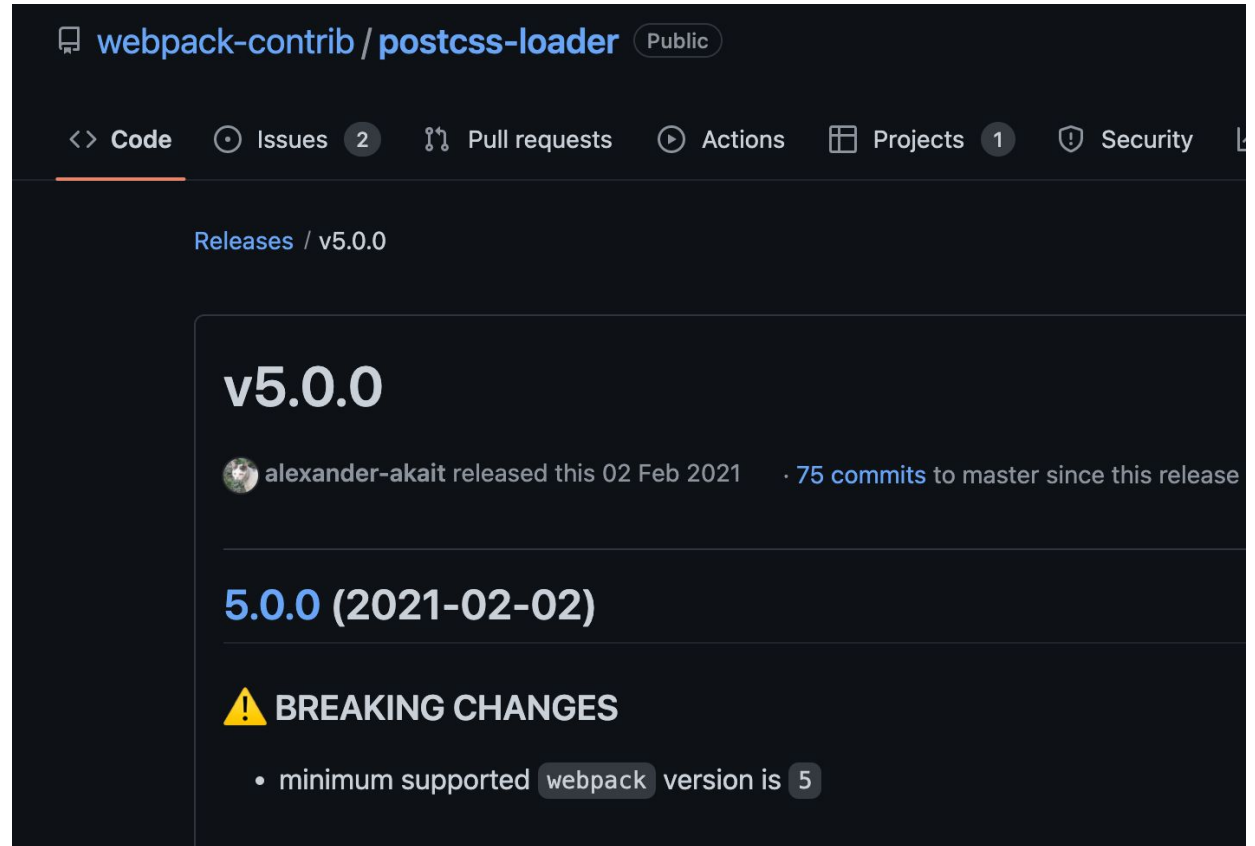
- Тяжелая настройка
- Проще обновлять

Используем в `storybook` и приложении разный конфиг

- Тяжелая настройка
- Проще обновлять
- Все же иногда возникают проблемы

Используем в storybook и приложении разный конфиг

- Тяжелая настройка
- Проще обновлять
- Все же иногда возникают проблемы



The screenshot shows the GitHub release page for the repository `webpack-contrib/postcss-loader`. The page is in dark mode. At the top, the repository name is displayed with a "Public" badge. Below the repository name, there are navigation links for "Code", "Issues" (with a count of 2), "Pull requests", "Actions", "Projects" (with a count of 1), and "Security". The main content area shows the release information for version `v5.0.0`. The release was made by `alexander-akait` on `02 Feb 2021`, and it includes `75 commits` to master since this release. Below the release information, there is a section for `5.0.0 (2021-02-02)` with a yellow warning icon and the text `BREAKING CHANGES`. A list of changes includes `minimum supported webpack version is 5`.

Инфраструктура `storybook`. Масштабируемость

Проблемы

- При росте числа `stories` увеличивается время запуска и переключения между `stories`

Инфраструктура `storybook`.

Масштабируемость

Проблемы

- При росте числа `stories` увеличивается время запуска и переключения между `stories`

Что может помочь?

- Сделать несколько `storybook`

Инфраструктура storybook.

Масштабируемость

Проблемы

- При росте числа stories увеличивается время запуска и переключения между stories

Что может помочь?

- Storybook-composition

```
// .storybook/main.js

module.exports = {
  refs: {
    react: {
      title: 'React',
      url: 'http://localhost:7007',
    },
    angular: {
      title: 'Angular',
      url: 'http://localhost:7008',
    },
  },
};
```

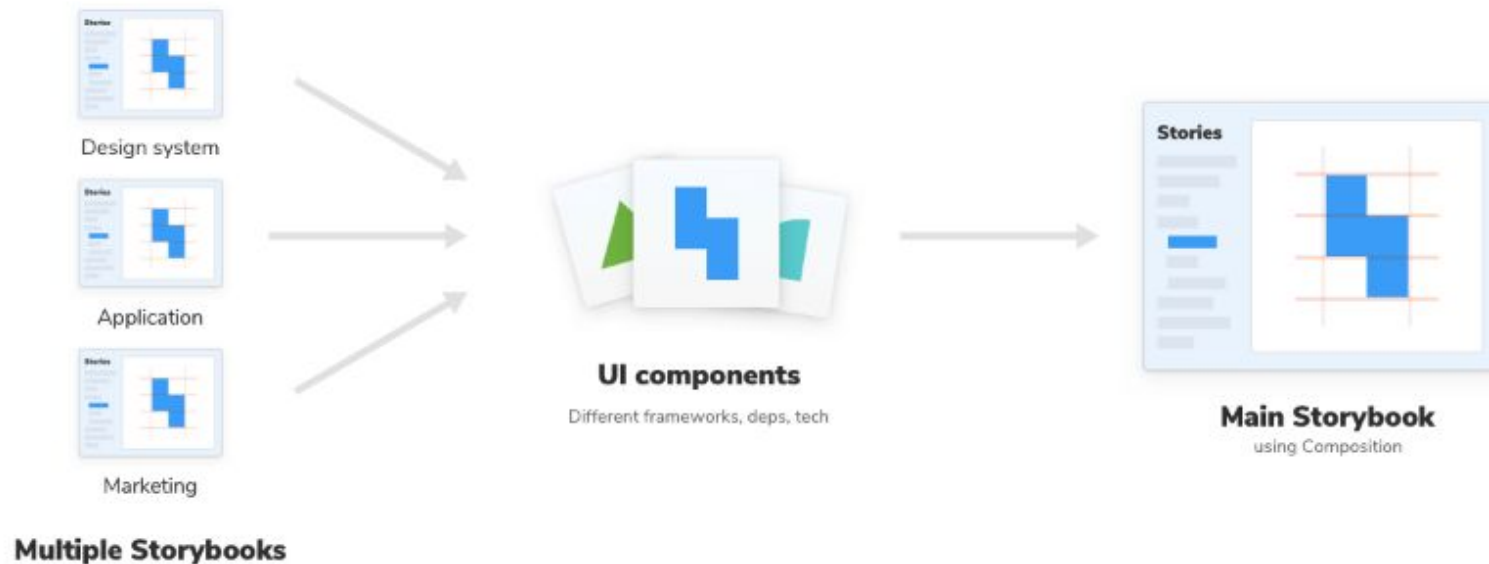

Инфраструктура storybook. Масштабируемость

Проблемы

- При росте числа stories увеличивается время запуска и переключения между stories

Что может помочь?

- Storybook-composition



Инфраструктура `storybook`.

Масштабируемость

Проблемы

- При росте числа `stories` ухудшается их работа

Что может помочь?

```
typescript {  
  reactDocgen: false  
}
```

Инфраструктура `storybook`.

Масштабируемость

Проблемы

- При росте числа `stories` ухудшается их работа

Что может помочь?

- `lazyCompilation`

```
module.exports = {
  features: {
    storyStoreV7: true
  },
  core: {
    builder: {
      name: 'webpack5',
      options: {
        lazyCompilation: true,
      },
    },
  },
};
```

Инфраструктура storybook.

Масштабируемость

Проблемы

- При росте числа stories ухудшается их работа

Что может помочь?

- lazyCompilation
- fsCache

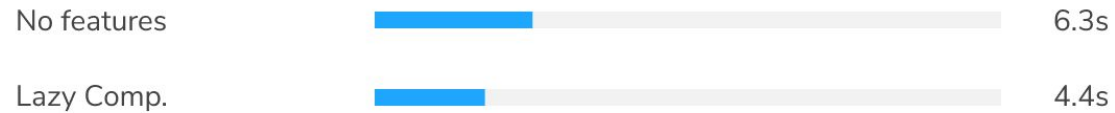
```
module.exports = {
  features: {
    storyStoreV7: true
  },
  core: {
    builder: {
      name: 'webpack5',
      options: {
        lazyCompilation: true,
        fsCache: true,
      },
    },
  },
};
```

Инфраструктура storybook. Масштабируемость

Mealdrop

Benchmark with 92 stories. Smaller is better.

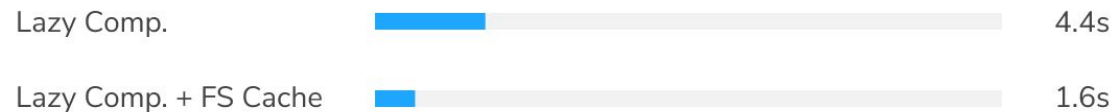
Time to first story Build time + Dev boot time



Mealdrop

Benchmark with 92 stories. Smaller is better.

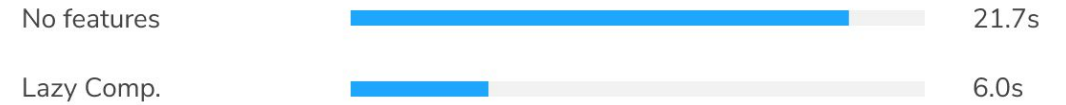
Time to first story Build time + Dev boot time



chromatic

Benchmark with 2000+ stories. Smaller is better.

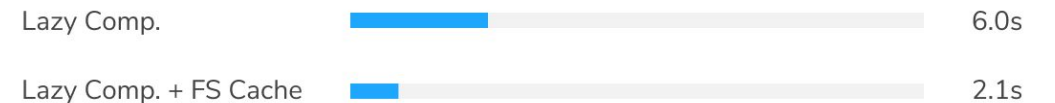
Time to first story Build time + Dev boot time



chromatic

Benchmark with 2000+ stories. Smaller is better.

Time to first story Build time + Dev boot time



Инфраструктура `storybook`.

Масштабируемость

Проблемы

- При росте числа `stories` ухудшается их работа
- `hot-reloading` падает при переполнении памяти

Что может помочь?

- Не использовать глобальные поиск `story` по всему проекту

Инфраструктура storybook. Масштабируемость

Проблемы

- При росте числа stories ухудшается их работа
- hot-reloading падает при переполнении памяти

Что может помочь?

- Не использовать глобальные поиск story по всему проекту

```
module.exports = {  
  stories: ['../../**/*.stories.@(js|tsx)'],  
}
```

Инфраструктура storybook.

Масштабируемость

Проблемы

- При росте числа stories ухудшается их работа
- hot-reloading падает при переполнении памяти

Что может помочь?

- Не использовать глобальные поиск story по всему проекту
- <https://github.com/storybookjs/storybook/issues/14342>

```
module.exports = {
  stories: [
    '../..../advertising/**/*stories.@(js|tsx)',
    '../..../app-b2b/**/*stories.@(js|tsx)',
    '../..../app-classified/**/*stories.@(js|tsx)',
    '../..../app-store/**/*stories.@(js|tsx)',
    '../..../app-dashboard/**/*stories.@(js|tsx)',
    '../..../app-dash-v2/**/*stories.@(js|tsx)',
    '../..../app-promo/**/*stories.@(js|tsx)',
    '../..../platform/**/*stories.@(js|tsx)',
    '../..../ui/**/*stories.@(js|tsx)',
  ],
}
```


Инфраструктура storybook. Iframe

- Storybook запущен в iframe
- Запуск storybook вне привычной среды исполнения

```
▼ <div>
  ▶ <div class="os-host os-host-foreign os-theme-dark os-host-overflow os-host-overflow-x os-host-resize-disabled os-host-scrollbar-vertical-hidden sto-6v9ktq os-host-transition">...</div>
  ▼ <div offset="40" class="sto-10ro1m">
    ▼ <div id="storybook-preview-wrapper" class="sto-sqc8g1">
      <a tabindex="0" href="#example-button--primary" class="sto-1ltb70r">Skip to sidebar</a>
      ▼ <iframe data-is-storybook="true" id="storybook-preview-iframe" title="storybook-preview-iframe" src="iframe.html?args=&viewMode=story&id=example-button--primary" allowfullscreen class="sto-crh05v" data-is-loaded="true">
        == $0
      </iframe>
    </div>
  </div>
  ▼ #document
    <!DOCTYPE html>
    ▼ <html lang="en">
      ▶ <head>...</head>
      ▼ <body class="sb-main-padded sb-show-main">
        ▶ <div class="sb-preparing-story sb-wrapper">...</div>
        ▶ <div class="sb-preparing-docs sb-wrapper">...</div>
        ▶ <div class="sb-nopreview sb-wrapper">...</div>
        ▶ <div class="sb-errordisplay sb-wrapper">...</div>
        ▶ <div id="root">...</div>
        <div id="docs-root" hidden="true"></div>
        ▶ <script>...</script>
        <script src="runtime~main.iframe.bundle.js"></script>
        <script src="vendors~node_modules_pmmwh_react-refresh-webpack-plugin_in_lib_runtime_RefreshUtils_js~node_mod-46d30e.iframe.bundle.js"></script>
        <script src="main.iframe.bundle.js"></script>
      </body>
    </html>
  </div>
</div>
```

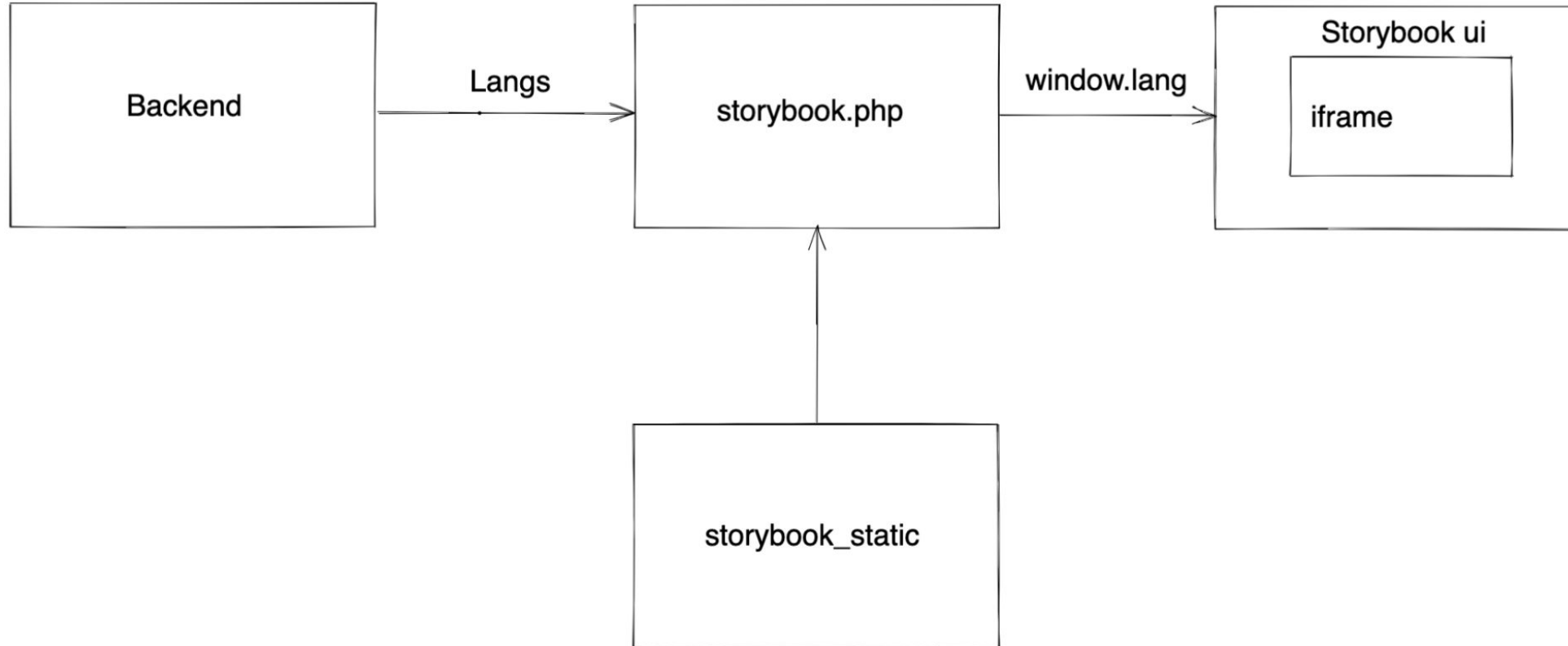
Iframe в storybook. Пример: Локализация

- Все ключи языков хранятся на бекенде

```
window.lang = {  
  edit: 'Редактировать',  
  cancel: 'Отменить'  
}
```

Iframe в storybook. Пример: Локализация

- Все ключи языков хранятся на бекенде
- Как передавать ключи в storybook?



Iframe в storybook. Пример: Локализация

- Все ключи языков хранятся на бекенде
- Канал для взаимодействия между storybook и бекендом

```
$html = str_replace('</head>', '<script>
  window.addEventListener("message", receiveMessage, false);
  function receiveMessage(event) {
    if (event.data === "load") {
      window.frames["storybook-preview-iframe"].contentWindow.lang = ' . getLangMap() . ';
    }
  }
</script></head>', $html);

echo $html;
```

Проблемы при использовании **storybook**

1

Инфраструктура
storybook

2

Свой “storybook”

3

Культура написания
stories

Свой “storybook”





**СВОЙ
“storybook”**

Свой “storybook”.

Как выглядит story?

```
export const BadgeStylekit: React.FC = () => (  
  <Group>  
    <Cell>  
      <Badge.Success>Продано</Badge.Success>  
    </Cell>  
    <Cell>  
      <Badge.Info>В резерве</Badge.Info>  
    </Cell>  
    <Cell>  
      <Badge.Warning>Неактивно</Badge.Warning>  
    </Cell>  
    <Cell>  
      <Badge.Error>Заблокировано</Badge.Error>  
    </Cell>  
  </Group>  
)  
  
export default {  
  component: BadgeStylekit,  
  title: 'Бейдж',  
}
```


Свой “storybook”.

Как подключать story?

```
import React from 'react'

const context = require.context(' ../../..', true, /\.stylekit.tsx$/)

type Config = {
  component: () => React.ReactElement
  title: string
}

export const config: Config[] = []

context.keys().forEach((fileName) => {
  config.push(context(fileName).default)
})
```

Свой “storybook”.

Рисуем список story

```
<Group>
  {config.map((item, index) => (
    <Link key={index} to={item.title}>
      {item.title}
    </Link>
  ))}
</Group>
```

Свой “storybook”.

Рисуем story

```
{config.map(({ title, component: Component }) => (  
  <Route to={title} key={title}>  
    <Header>  
      {title}  
    </Header>  
    <StylekitForm>  
      <Component />  
    </StylekitForm>  
  </Route>  
))}
```

Свой “storybook”.

А что если нужны контроллы?

```
export const CrosspostingGroupStylekit: React.FC = () => {
  const title = useString({ name: 'title', label: 'Название' })
  const description = useString({ name: 'description', label: 'Описание' })
  const mode = useSelect({ name: 'mode', label: 'Тип отображения', values: ['plain', 'card'] })

  return (
    <CrosspostingGroup title={title} description={description} mode={mode}>
      Группа
    </CrosspostingGroup>
  )
}

export default {
  component: CrosspostingGroupStylekit,
  title: 'CrosspostingGroup',
}
```

Свой “storybook”.

А как они устроены?

```
export function useSelect<T extends SelectTypeKnobValue>(params: Params<T>) {
  const dispatch = useDispatch()

  React.useEffect(() => {
    dispatch(
      initializeFormItem({
        type: 'select',
        ...(params as Params<SelectTypeKnobValue>),
      })
    )
  }, [params.name, params.label, params.defaultValue])

  return useSelector(selectFormValues)[params.name] as T
}
```

Проблемы при использовании storybook

1

Инфраструктура
storybook

2

Свой “storybook”

3

Культура написания
stories

Культура написания stories

- Есть большой story, в котором есть несколько props. Как группировать эти props, чтобы они позволяли получать только актуальные состояния компонента?

```
export type Product = {
  photos: ItemImage[];
  itemRaw: string;
  moderationStatus?: MarketModerationStatus;
  rejectInfo: MarketMarketItemFull['reject_info'];
  deliveryInfo?: MarketDeliveryInfo;
  canComment?: MarketMarketItemFull['can_comment'];
  oldPrice: MarketPrice['old_amount_text'];
  price: MarketPrice['text'];
  discount: MarketPrice['discount_rate'];
  name: MarketMarketItemFull['title'];
  description: MarketMarketItemFull['description'];
  id: MarketMarketItemFull['id'];
  availability: MarketMarketItemFull['availability'];
  sku: MarketMarketItemFull['sku'];
  viewsCount: MarketMarketItemFull['views_count'];
  date: MarketMarketItemFull['date'];
  isFavorite: MarketMarketItemFull['is_favorite'];
  canRepost: MarketMarketItemFull['can_repost'];
  reposts: MarketMarketItemFull['reposts'];
  serviceDuration: MarketMarketItemFull['service_duration'];
  likes: {
    count: BaseLikes['count'];
    userLikes: BaseLikes['user_likes'];
  };
  isPriceListService: boolean;
  actionButtons?: BaseLinkButton[];
  variantsGrid?: MarketVariantsGridProperty[];
  cartQuantity?: number;
  banner?: MarketItemBanner;
  ownerInfo: MarketMarketItemFull['owner_info'];
  wishListItemId?: MarketMarketItemFull['wishlist_item_id'];
  canEdit: MarketMarketItemFull['can_edit'];
  serviceEditEnabled: boolean;
  stockAmount: MarketMarketItemFull['stock_amount'] | null;
  canDelete: MarketMarketItemFull['can_delete'];
  promotion: MarketMarketItemFull['promotion'];
  adId: MarketMarketItemFull['ad_id'];
  category: MarketMarketItemFull['category'];
  canShowConvertToService: MarketMarketItemFull['can_show_convert_to_service'];
  ownerId: MarketMarketItemFull['owner_id'];
  similarItems: MarketMarketItemFull['similar_items'];
};
```

Как группировать props в компоненте

- Плохо написанный компонент

Как группировать props в компоненте

- Плохо написанный компонент
- Разделяем компонент на несколько

```
export const Product: React.FC = (product) => {  
  if (product.isEdit) {  
    return <ProductEditable {...editableProps} />;  
  }  
  
  return <ProductSimple {...simpleProps} />;  
};
```

Как группировать props в компоненте

- Создаем story под каждое состояние

```
export const ProductSimple = {  
  args: {  
    canEdit: false,  
    canDelete: false,  
  }  
};
```

```
export const ProductEditable = {  
  args: {  
    canEdit: true,  
    canDelete: true,  
  }  
};
```

Культура написания stories

- Стоит ли все состояния компонента описывать в одной story или разделять на несколько?

Базовый компонент стоит описывать в одной story

- Можем сразу видеть все состояния компонента

Primary



Button

Secondary



Button

Large



Button

Small



Button

Базовый компонент стоит описывать в одной story

- Можем сразу видеть все состояния компонента
- Поведение компонента при смене контролла

Primary



Secondary



Large

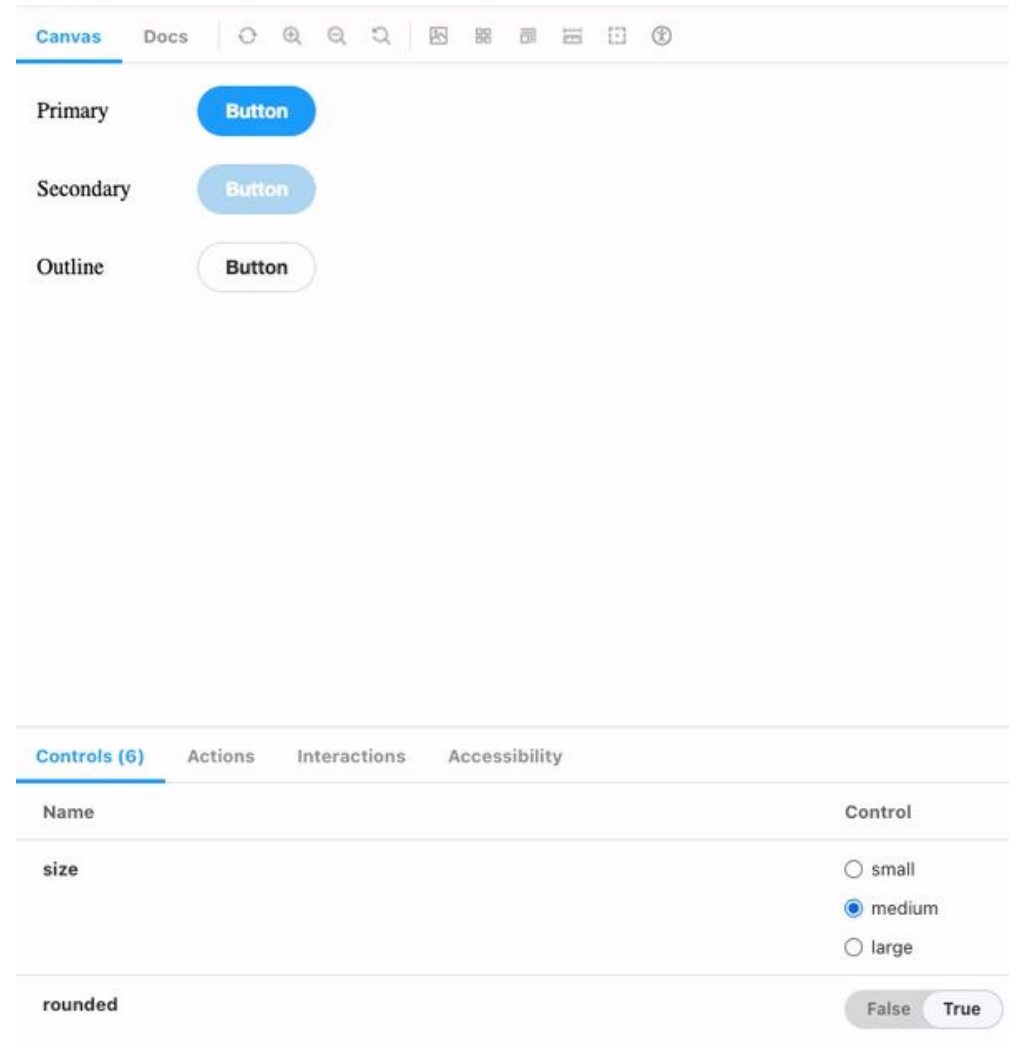


Small



Базовый компонент стоит описывать в одной story

- Можем сразу видеть все состояния компонента
- Поведение компонента при смене контролла



Базовый компонент стоит описывать в одной story

- Можем сразу видеть все состояния компонента
- Поведение компонента при смене контролла
- Компонент не зависит от сценария использования

Primary



Secondary



Large

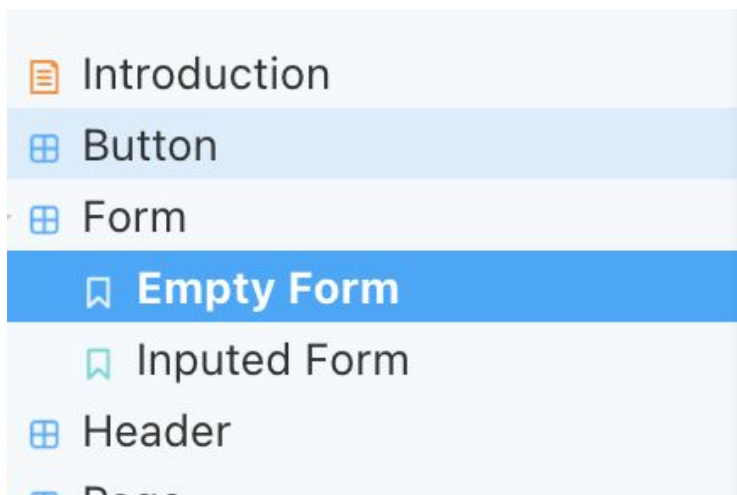


Small



Составной компонент

- Каждая story - конкретный сценарий использования компонентов
- Для каждого сценария подбираем определенные props компонента



Культура написания stories

- Когда проект состоит из нескольких команд, каждая команда пишет stories по-разному

Культура написания stories

- Когда проект состоит из нескольких команд, каждая команда пишет stories по-разному
- Подключить linter для storybook (eslint-plugin-storybook)

```
{
  "overrides": [
    {
      "files": ['*.stories.@(ts|tsx|js|jsx|mjs|cjs)'],
      "rules": {
        'storybook/hierarchy-separator': 'error',
        'storybook/default-exports': 'off',
      }
    }
  ]
}
```

Культура написания stories

- Когда проект состоит из нескольких команд, каждая команда пишет stories по-разному
- Подключить linter для storybook (eslint-plugin-storybook)
- Документация

Как писать story?

- Когда создаете react-компонент, к нему обязательно должна быть написана story
- Максимальный размер story - энтри поинт
- Если создаете переиспользуемый react-компонент, то каждое его состояние должно быть в отдельной story
- Если в компоненте есть дочерние компоненты, которые не существуют без родителя, то все состояния дочернего компонента следует описывать в одной story
- Скелетон для компонента должен лежать вместе с компонентом в одной story
- Все сервисы (апи, wiki, аналитика) должны быть вынесены во внешний сервис (context, thunk-extra) для удобства использования в storybook
- Если в компонент передается какой-то callback, который переопределен в storybook, то он должен быть обернут в action

Проблемы при использовании storybook

1

Инфраструктура
storybook

2

Свой “storybook”

3

Культура
написания stories

Итог




Какие проблемы
помогает решать
storybook?



Примеры **решения**
проблем из нашей
практики



Проблемы при
использовании
storybook



**И все-таки, *storybook* -
проблема или решение
проблемы?**



Спасибо
за внимание!

Нагайко Иван