



# Зависимости в компонентном вебе, приготовленные правильно

Владимир Гриненко  
FrontTalks 2017

# О чём речь

- › Как обычно собирается фронтенд
- › Как сборку можно улучшить
- › Декларативное множественное наследование
- › Слои для компонентов
- › Как начать использовать новое знание

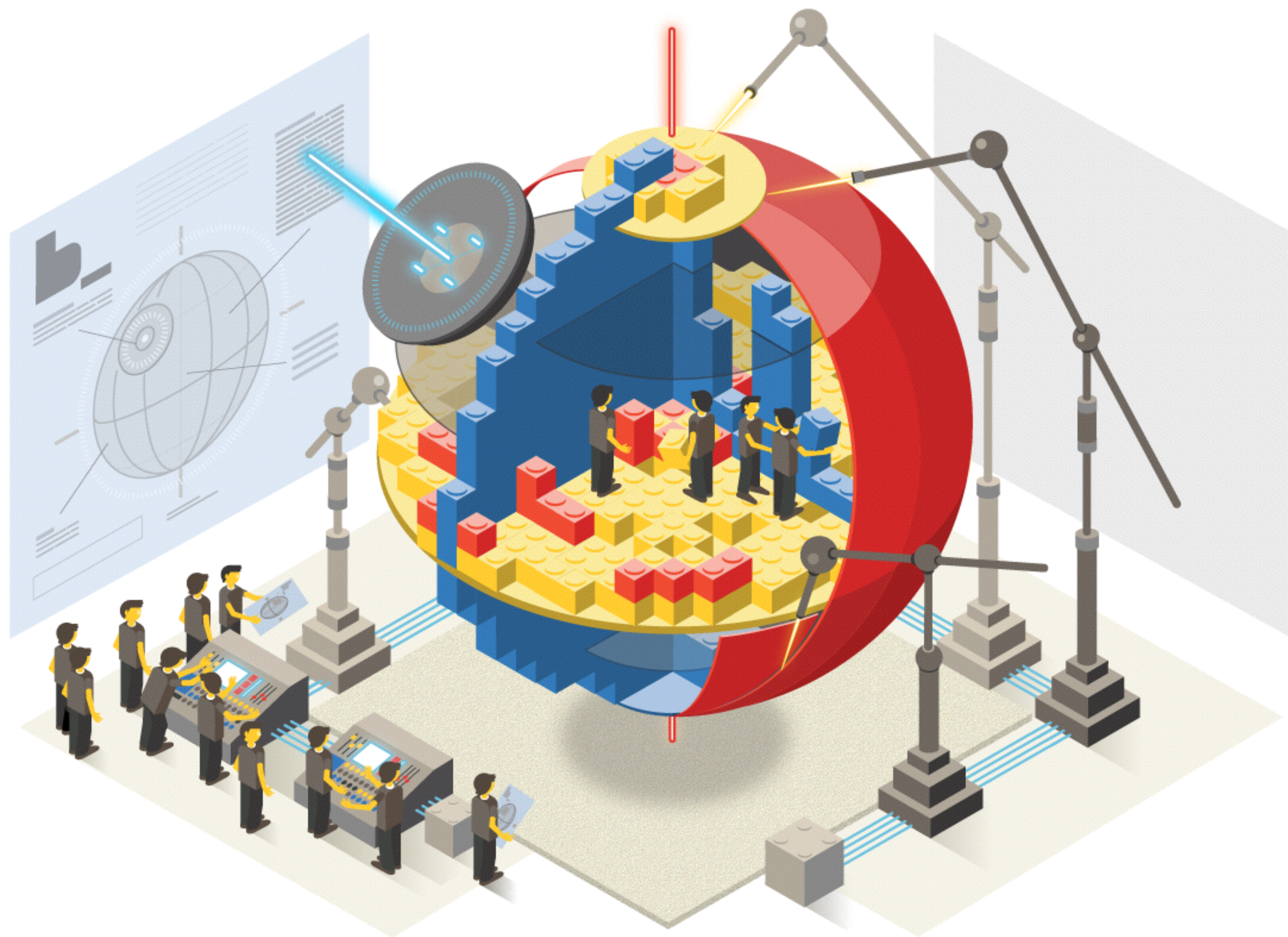
# Компонентный подход везде

- › Bootstrap
- › БЭМ
- › React
- › Web Components
- › ТЫСЯЧИ ИХ









# Сборка при компонентном подходе

- › Найти все компоненты на FS
- › Написать `require/import` в каждой технологии

# Найти все компоненты на FS

- › Собираем лишнее
- › Не можем гарантировать порядок

# Написать `require/import` в каждой технологии

- › Много ручной работы
- › Вербозно
- › Захардкожены пути
- › Медленно
- › Мухи и котлеты





















# Было

## **index.css**

```
@import "../..path/to/button/button.css";  
@import "../..path/to/link/link.css";  
/* project styles */
```

## **index.js**

```
require('../..path/to/button/button.js');  
require('../..path/to/link/link.js');  
/* project logic */
```



# Стало


**index.decl.js**

```
['button', 'link']
```

- › ~~Много ручной работы~~
- › ~~Вербозно~~
- › ~~Захардкожены пути~~
- › ~~Медленно~~
- › ~~Мухи и котлеты~~

# Декларируем бандл в терминах компонентов

- › Не говорим, где физически находятся файлы
- › Не говорим, какие технологии нужны



Декларация — что и  
в каком порядке  
подключать в сборку



# Получение декларации

- › Интроспекция с файловой системы
- › Написание руками
- › Генерация по описанию страницы

# Генерация по описанию страницы

```
<button class="button"></button>
```

```
<a class="link"></a>
```

# Алгебра деклараций





# Алгебра деклараций

- › Объединение
- › Пересечение
- › Вычитание

# Объединение

Декларация 1

```
[  
    'header',  
    'button',  
    'link',  
    'attach',  
  
    'checkbox',  
  
    'textarea'  
]
```

Декларация 2

```
[  
    'header',  
    'button',  
  
    'menu',  
    'image',  
  
    'popup'  
]
```

Декларация 3

```
[  
    'header',  
    'button',  
    'link',  
    'attach',  
    'menu',  
    'image',  
    'checkbox',  
    'popup',  
    'textarea'  
]
```

# Пересечение

Декларация 1

```
[  
    'header',  
    'input',  
    'link',  
    'attach',  
    'checkbox',  
    'textarea',  
    'footer'  
]
```

∩

Декларация 2

```
[  
    'header',  
    'menu',  
    'button',  
    'input',  
    'image',  
    'popup',  
    'footer'  
]
```

=

Декларация 3

```
[  
    'header',  
  
  
  
  
  
  
    'footer'  
]
```

# Вычитание

Декларация 1

```
[  
    'button',  
    'checkbox',  
    'textarea',  
    'suggest'  
]
```

Декларация 2

```
[  
    'button',  
  
    'header',  
    'input',  
    'button',  
    'menu'  
]
```

-

=

Декларация 3

```
[  
  
    'checkbox',  
    'textarea',  
    'suggest'  
]
```



Композиция.  
Зависимости между  
компонентами



# Было

## **my-component.css**

```
@import "../..path/to/button/button.css";  
@import "../..path/to/link/link.css";  
/* my-component styles */
```

## **my-component.js**

```
require('../..path/to/button/button.js');  
require('../..path/to/link/link.js');  
/* my-component logic */
```

## Пример

# Стало

**my-component.deps.js**

**`['button', 'link']`**

# Стало

**my-component.deps.js**

```
{  
  mustDeps: ['button'],  
  shouldDeps: ['link']  
}
```



Зависимость как  
технология компонента







Декларативное  
множественное  
наследование



# Модификаторы

```
<button class="button">
```

```
<button class="button button_type_submit">
```

```
<button class="button button_type_submit button_disabled">
```

А шаблоны?

# Слои для компонентов





# Слои позволяют

- › Дешево обновлять библиотечный код
- › Разделять общие части реализации блоков от частных
- › Разделять проект на платформы
- › Разный код в разработке и продакшене
- › Держать темы отдельно от общей логики

# Пример

**library-components/**  
  input/  
  button/  
  popup/

# Уровень библиотеки

# Базовая реализация блока button

**project/**  
  input/  
  button/  
  header/

# Уровень проекта

# Измененная реализация блока button

# Пример

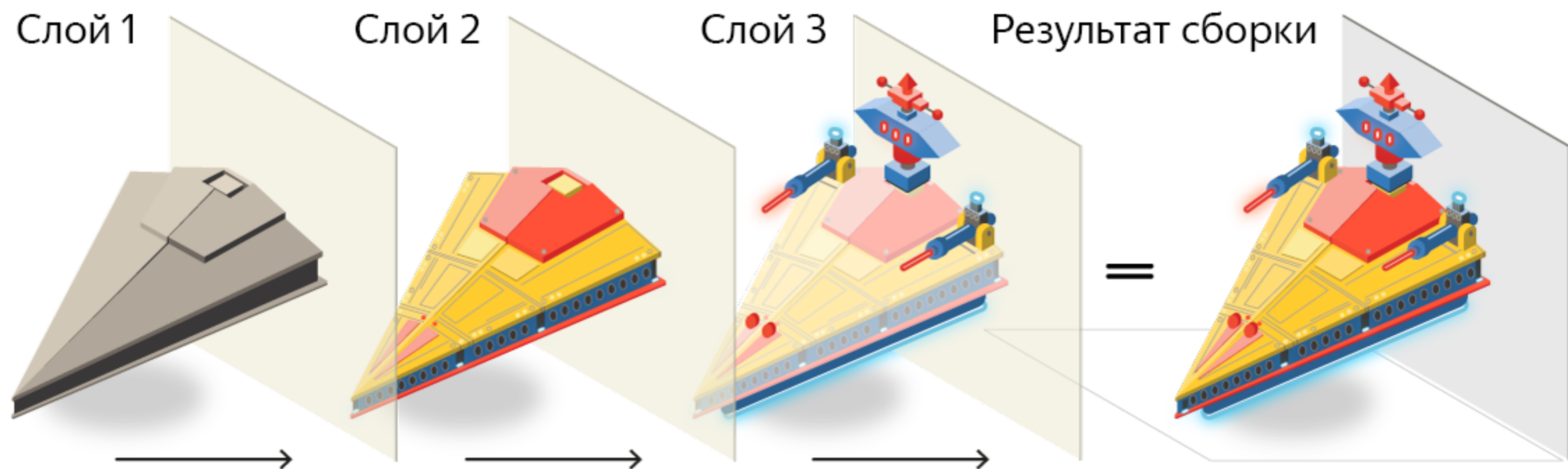
**common.blocks/button/button.css**

```
.button {  
    height: 25px;  
}
```

**touch.block/button/button.css**

```
.button {  
    height: 50px;  
    tap-highlight-color: #ccc;  
}
```





# Было

```
@import "library/common.blocks/button/button.css";
```

```
@import "library/touch.block/button/button.css";
```

```
@import "library/design/common.block/button/button.css";
```

```
@import "library/design/touch.block/button/button.css";
```

```
@import "project/common.blocks/button/button.css";
```

```
@import "project/touch.blocks/button/button.css";
```

**И аналогично для JS.**

# Стало

```
[ 'button' ]
```







# Как еще может быть

```
import Button from 'b:button';
```

Как начать использовать  
НОВОЕ знание





# Как начать использовать

- › Для React + Пример
- › Для всех остальных + Пример

# Что дальше

- › [bem.info/methodology/build](https://bem.info/methodology/build)
- › [bem.info/forum](https://bem.info/forum)
- › [t.me/bem\\_ru](https://t.me/bem_ru)
- › слайды







Ваши вопросы!





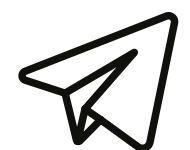


Владимир Гриненко

Руководитель группы общих  
компонентов интерфейсов



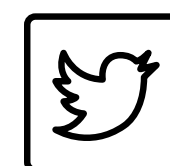
[tadatuta@yandex-team.ru](mailto:tadatuta@yandex-team.ru)



tadatuta



tadatuta



tadatuta

