



HTML и CSS. Адаптивная вёрстка и автоматизация

Уровень 2, с 21 ноября 2022 по 30 января 2023

Меню курса

[Главная](#) / [2. Методологии вёрстки](#) /

📖 2.18. Типовые ошибки

🕒 ~ 12 минут

Перечислим типовые ошибки, которые совершают разработчики, когда используют БЭМ.


Ошибка 1

Определяется неверный уровень абстракции. Что это значит? Мы находим общие свойства у блоков, которые не очень похожи. Добавляем условия, чтобы скрыть лишние и показать нужные свойства у блока. Много переопределений стилей. Когда мы используем принципы *KISS* (*Keep It Simple, Stupid*) и *DRY* (*Don't Repeat Yourself*), возникает когнитивная ошибка, и мы видим общее там, где его нет. Плохая абстракция приводит к мудрению в коде. Это попытка объединить в один блок разные сущности. Не факт, что такая композиция приживётся.


Лучше сделать несколько отдельных компонентов, чем объединить необъединимое и «подставить костыли». Пусть поначалу будет дублирование в коде, пусть мы создадим похожие блоки как разные компоненты. Потом, если появится необходимость, можно отрефакторить, увидев понятную абстракцию.

К примеру, на странице есть такие блоки:


Новинки




Палатка 2-местная
5500₽



Палатка 2-местная
4700₽



Палатка 1-местная
2600₽



Палатка 4-местная
9900₽

[Смотреть все](#)

Есть несколько блоков, которые выглядят для нас одинаково, хотя на самом деле это не так. Интуитивно хочется реализовать их как один блок, но если внимательно присмотреться, то схожими будут только шапки блоков, вот их можно выделить в один компонент.

```
<section class="catalog">
  <header class="catalog__headline headline">
    <h2 class="headline__title">Новинки</h2>
    <a class="headline__link link">Смотреть все</a>
  </header>
  <ul class="catalog__list">
    ...
  </ul>
</section>
<section class="blog">
  <header class="blog__headline headline">
    <h2 class="headline__title">Блог</h2>
    <a class="headline__link link">Все статьи</a>
  </header>
  <ul class="blog__list">
    ...
  </ul>
</section>
```

Ошибка 2

У блока присутствует внешняя геометрия (внешние отступы, позиционирование, ширина жёстко зафиксирована). К примеру, для того, чтобы попасть в *PerfectPixel*, добавляют странные модификаторы для блоков.

```
<section class="section section--catalog">
  ...
</section>
<section class="section section--novelty">
  ...
</section>
```

```
/* такое использование модификаторов — ошибка */
.section--catalog {
  margin-bottom: 960px;
}

.section--novelty {
  margin-bottom: 480px;
}
```

Геометрия, в том числе и внешние отступы, могут применяться только к элементам блока. За очень редким исключением блокам задают ширину.

В этом примере нужно подняться на уровень вверх и внимательно изучить родительский блок. Подумать, почему один элемент шире, другой уже, и всё-таки определить ширину для элемента. Например, так:

```
<main class="page-content">
  <section class="page-content__section section">
    ...
  </section>
  <section class="page-content__section page-content__section--half section">
    ...
  </section>
</main>
```

```
/* шириной разделов управляем через элементы родительского блока */
.page-content__section {
  margin-bottom: 960px;
}

.page-content__section--half {
  margin-bottom: 480px;
}
```

Ошибка 3

Неверная декомпозиция на блоки. Не видно БЭМ-дерева, почти везде просто блок, вложенный в другой блок, и всё живёт просто благодаря тому, что имена разные. Но это не декомпозиция, это не БЭМ.

Например:

```
<ul class="menu">
  <li class="menu-item">
    <a class="menu-link">
      <span class="menu-text">Кабинет ученика</span>
    </a>
  </li>
</ul>
```

menu-item > menu-link > menu-text — это отдельные блоки?

Ну, конечно же, нет, но выглядит, как будто да. Всё разбито на никак не связанные собой блоки. Там, где должны быть элементы блока — `блок__элемент`, написано

два отдельных несвязанных блока.

А надо так:

```
<ul class="menu">
  <li class="menu__item menu-item">
    <a class="menu-item__link">
      <span class="menu-item__text">Кабинет ученика</span>
    </a>
  </li>
</ul>
```

Ошибка 4

Попытка использовать элементы элементов, допустим, так:

```
<div class="company-logo">
  <div class="company-logo__cat"></div>
  <div class="company-logo__text">
    <!-- Ошибка в имени класса -->
    <span class="company-logo__text__first-letter">C</span>at
  </div>
</div>
```

В примере выше для элемента, который отвечает за особое оформление первой буквы в названии компании, используется некорректное имя класса `company-logo__text__first-letter` (читай как элемент `first-letter` элемента `text` блока `company-logo`).

А надо так:

```
<div class="company-logo">
  <div class="company-logo__cat"></div>
  <div class="company-logo__text">
    <span class="company-logo__first-letter">C</span>at
  </div>
</div>
```

Ошибка 5

Неверное подчинение. Очень линейное БЭМ-дерево, один блок и куча элементов, нет попыток разбить, сделать декомпозицию. Например, пишут `sources` > `sources__sources__logo`, а могли бы `sources__logos` сделать отдельным блоком, но не ст. К БЭМ не стоит подходить формально, когда синтаксис соблюден, но при этом

композиционно решено неверно, зависимости указаны неправильно, много связей и другое. План выражения выполнен, а план содержания — нет.

```
<!-- Очень линейное БЭМ-дерево -->
<section class="sources">
  <h2 class="sources__title">Наши партнёры/Источники финансирования</h2>
  <ul class="sources__logos">
    <li class="sources__logo">
      
    </li>
    ...
  </ul>
</section>
```

А надо так:

```
<!-- Видим и выделяем более мелкие блоки и составляем из них композицию -->
<section class="sources">
  <h2 class="sources__title">Наши партнёры/Источники финансирования</h2>
  <ul class="sources__logos logos">
    <li class="logos-item logo">
      
    </li>
    ...
  </ul>
</section>
```

План выражения и план содержания

Ошибка 6

Модификатор используется сам по себе, не относится ни к блоку ни к элементу. А у модификатора обязательно должен быть класс для блока или элемента!

```
<a class="button--primary" href="#">Смотреть каталог</a>
```

А надо так:

```
<a class="button button--primary" href="#">Смотреть каталог</a>
```

Ошибка 7

Элементы без блока, которые болтаются сами по себе. Элемент обязательно должен находиться внутри блока, но элементы могут находиться «друг в друге» в разметке в рамках одного блока.

```
<!-- Ошибка -->
<div class="logo__container">
  <div class="logo"></div>
</div>
```

А надо так:

```
<div class="page-header">
  ...
  <div class="page-header__logo">
    <div class="logo"></div>
  </div>
  ...
</div>
```

Ошибка 8

Названия классов включают в себя внешний вид: `button--red` (кнопка красная), `footer-dark` (подвал тёмный) и подобные. Лучше использовать `button--primary` (кнопка первичного действия).

Возможное исключение — `button--size-xl`.

Не должно быть блоков с именами `uppercase`, `text-center` — это стили вне блоков. Нежелательно использовать классы вида `vertical-wrap` или `push-md--right`. Стили должны всегда относиться к конкретному блоку. А что такое `push-md--right`? Это блок такой? Нет.

Ошибка 9

Используются классы из списка запрещённых имён: `for`, `list__item1`, `list__item2` и другие.

Ошибка 10

Разнобой в синтаксисе или нотации.

```

<footer class="page-footer container">
  ...
  <ul class="page-footer__contacts contacts">
    <li class="contacts__item">
      <!-- для имени модификатора используется классический стиль -->
      <a class="contacts__link contacts__link_phone" href="tel:88009995522">8
(800) 999 55 22</a>
    </li>
    <li class="contacts__item">
      <!-- для имени модификатора используется стиль Two Dashes -->
      <a class="contacts__link contacts__link--mail"
href="mailto:info@edisonshoes.com">info@edisonshoes.com</a>
    </li>
    <li class="contacts__item">
      <!-- для имени модификатора используется стиль CamelCase -->
      <a class="contacts__link contacts__link-mapPin" href="#">Санкт-Петербург,
ул. Набережная, д. 40</a>
    </li>
  </ul>
  ...
</footer>

```

Выбираем что-то одно, к примеру, нотацию `блок__элемент - -модификатор`:

```

<footer class="page-footer container">
  ...
  <ul class="page-footer__contacts contacts">
    <li class="contacts__item">
      <a class="contacts__link contacts__link--phone" href="tel:88009995522">8
(800) 999 55 22</a>
    </li>
    <li class="contacts__item">
      <a class="contacts__link contacts__link--mail"
href="mailto:info@edisonshoes.com">info@edisonshoes.com</a>
    </li>
    <li class="contacts__item">
      <a class="contacts__link contacts__link--map-pin" href="#">Санкт-
Петербург, ул. Набережная, д. 40</a>
    </li>
  </ul>
  ...
</footer>

```

Ошибка 11

Ошибкой будет одновременно использовать два одинаковых модификатора с разными значениями.

```

<form class="search-form
          search-form--theme-islands

```

```

        search-form--theme-lite">

<input class="search-form__input">

<button class="search-form__button
            search-form__button--size-s
            search-form__button--size-m">

    Найти
</button>
</form>

```

В этом примере мы не можем использовать одновременно модификаторы `search-form--theme-islands` и `search-form--theme-lite`, `search-form__button--size-s` и `search-form__button--size-m`. Нужно выбрать что-то одно.

```

<form class="search-form search-form--theme-lite">

    <input class="search-form__input">

    <button class="search-form__button search-form__button--size-s">
        Найти
    </button>
</form>

```

Ошибка 12

БЭМ-дерево повторяет *DOM*.

Например:

```

<section class="feature">
  <header class="feature-head">
    <svg class="feature-head__logo" width="11" height="14"
xmlns="http://www.w3.org/2000/svg">...</svg>
    <h2 class="feature-head__title">Монтаж сантехнического оборудования</h2>
  </header>
  ...
</section>

```

Блоки `feature` и `feature-head` называются так, как-будто они не разбиты на части и каждый зависит от другого. Такая декомпозиция на компоненты похожа на попытку спрятать элементы элементов и повторить *DOM*.

Если шапка, действительно, это часть блока `feature`, пусть она будет его элементом, например, так:


```

<section class="feature">
  <header class="feature__head">
    <svg class="feature__logo" width="11" height="14"
xmlns="http://www.w3.org/2000/svg">...</svg>
    <h2 class="feature__title">Монтаж сантехнического оборудования</h2>
  </header>
  ...
</section>

```

Если этот блок может существовать сам по себе, в том числе и в других блоках, тогда надо так:

```

<section class="feature">
  <header class="headline">
    <svg class="headline__logo" width="11" height="14"
xmlns="http://www.w3.org/2000/svg">...</svg>
    <h2 class="headline__title">Монтаж сантехнического оборудования</h2>
  </header>
  ...
</section>

```

Ошибка 13

Добавление дополнительных имён классов-блоков, а не модификаторов.

Чтобы добавить визуальное отличие для базового блока делают его композицией из двух блоков, например: `class="section support"`, `class="section advantages"`.

А надо так: `class="section section--support"`, `class="section section--advantages"`.

Но классы также могут примиксоваться и как блоки, всё зависит от ситуации.

Ошибка 14

Использование модификаторов вместо элементов. Модификатор — это всегда только внешнее изменение для блока/элемента.

Например:

```

<footer class="footer">
  <span class="footer--brand">Elenea &copy;</span>
  <span class="footer--description">Copyright 2021</span>
</footer>

```

А надо так:

```
<footer class="footer">
  <span class="footer__brand">Elenea &copy;</span>
  <span class="footer__description">Copyright 2021</span>
</footer>
```

Ошибка 15

Формально это не ошибка, но будут проблемы, когда компонуют блоки так: `list-item > service-menu__item > footer__link`. Так-то можно «взболтать, но не смешивать», однако при этом не получится поменять стили только в одном месте, в конкретном блоке. Стили сменятся во всех блоках, где их используют.

Например, вот такой микс:

```
<footer class="footer">
  <section class="service-menu">
    <ul class="list-item">
      <li class="service-menu__item">
        <a class="footer__link"
href="mailto:company@mail.com">company@mail.com</a>
      </li>
    </ul>
  </section>
</footer>
```

Блоки не должны «вплетаться» друг в друга. В такой разметке не учитывается один момент: как их использовать по отдельности и где тут внутри чья зона ответственности?

А надо так:

```
<section class="section">
  <h2 class="section__title">Контакты</h2>
  <ul class="contacts">
    <li class="contact">
      <a class="contact__link"
href="mailto:company@mail.com">company@mail.com</a>
    </li>
    ...
  </ul>
</section>
```

Ошибка 16

Блок хочет управлять CSS-свойствами вложенных блоков. К примеру, при использовании светлой темы цвет текста на панели с закладками должен быть `#` а фоновый цвет вкладок — `#eeeeee`.

```
<body class="page
  theme
  theme--light">
  ...
  <div class="tabs">
    <ul class="tabs__navigation">
      <li class="tabs__navigation-item">Характеристики</li>
      <li class="tabs__navigation-item">Отзывы</li>
      ...
    </ul>
  </div>
</body>
```

Устанавливаем цвет шрифта для закладок:

```
.theme--light .tabs {
  color: #2e2c9c;
}
```

Задаём фоновый цвет самим вкладкам:

```
.theme--light .tabs__navigation-item {
  background-color: #eeeeee;
}
```

Используется межблочный каскад, что категорически не рекомендуется делать. Это делает код разных блоков связанным.

Прочитали главу?

Нажмите кнопку «Готово», чтобы сохранить прогресс.

Готово

⚠ Если вы обнаружили ошибку или неработающую ссылку, выделите ее и нажмите Ctrl + Enter

Поиск по материалам

Git

[Все материалы](#)

В самом начале



- ☐ [Пройдите опрос](#)
- ☐ [Укажите персональные данные](#)
- ☐ [Изучите регламент](#)
- ☐ [Прочитайте FAQ](#)
- ☐ [Добавьте свой Гитхаб](#)
- ☐ [Выберите наставника](#)
- ☐ [Создайте проект](#)

Мой наставник



[Выбрать наставника](#)

Работа с наставником

У вас осталось **10** из 10 консультаций.

[История](#)



Практикум

[Тренажёры](#)

[Подписка](#)

[Для команд и компаний](#)

[Учебник по PHP](#)

Профессии

[Фронтенд-разработчик](#)

[JavaScript-разработчик](#)

[Фулстек-разработчик](#)

Курсы

HTML и CSS. Профессиональная вёрстка сайтов

HTML и CSS. Адаптивная вёрстка и автоматизация

JavaScript. Профессиональная разработка веб-интерфейсов

JavaScript. Архитектура клиентских приложений

React. Разработка сложных клиентских приложений

Node.js. Профессиональная разработка REST API

Node.js и Nest.js. Микросервисная архитектура

TypeScript. Теория типов

Алгоритмы и структуры данных

Паттерны проектирования

Webpack

Vue.js 3. Разработка клиентских приложений

Git и GitHub

Анимация для фронтендеров

Блог

С чего начать

Шпаргалки для разработчиков

Отчеты о курсах

Информация

Об Академии

О центре карьеры

Услуги

Работа наставником

Для учителей

Стать автором

Остальное

Написать нам

Мероприятия

Форум

Соглашение

Конфиденциальность

Сведения об образовательной организации

Лицензия № 4696

