

## Разметка по БЭМ

Выполнен на 0%

Введение

Углублённая теория

Предыстория

Частые проблемы из-за неправильной CSS-архитектуры проекта

Основные понятия

Свод правил БЭМ-а

БЭМ для CMS

Названия классов по БЭМ

БЭМ — это про компоненты

Темизация

Плюсы и минусы БЭМ

Методика

Кейс 1, лёгкий уровень

Кейс 2, лёгкий уровень

Кейс 3, лёгкий уровень

Кейс 4, средний уровень

Кейс 5, средний уровень

Кейс 6, средний уровень

Кейс 7, сложный уровень

Кейс 8, сложный уровень

Кейс 9, сложный уровень

Главная / Моё обучение / Разметка по БЭМ / Углублённая теория /

# Предыстория

БЭМ — одно из таких российских изобретений, которые распространились по всему миру и используются массово и повсеместно. Как автомат Калашникова, только цель была мирная и пользу приносит всем, а вреда никакого.

БЭМ родился в Яндексе примерно в середине 2000-х. Изначально это была попытка использовать единый подход к вёрстке в разных проектах Яндекса, но постепенно БЭМ развился и стал всеобъемлющим, и при желании можно сделать «по БЭМ» всё, от вёрстки до скриптов, утилит и инструментов для сборки.

К нашему времени вся концепция БЭМ, его полная история и вариации описаны на сайте [bem.info](#), так что пересказывать всё вообще мы не станем.

БЭМ создавался как рабочий инструмент, а не как философия. Нужно было сделать вёрстку проекта более управляемой: при вёрстке крупного проекта оказалось, что прошлые подходы с разветвлённым каскадом делают код непрозрачным, осложняют изменения и командную работу. Вот пример сложноуправляемого и запутанного кода:

```
div#popup {
  z-index: 100500;
}

div#popup div span.child-of-child {
  color: #666666;
}

#some-id .label {
  color: #111111 !important;
}
```

Те правила, которые использовались для структурирования кода в маленьком проекте, сложно было перенести в большой проект. К примеру, для большого проекта тяжело подбирать названия для классов так, чтобы не повторить уже задействованные имена, да и код всего проекта сложно держать в голове. А чтобы найти что-либо в одном длинном стилевом файле, нужно хорошо покопаться, и нет гарантий, что изменение повлияет только на тот объект, который нужно изменить. В этих условиях сложно не сломать уже существующую функциональность. Вёрстка становится неуправляемой. Вот и Яндексу с их множеством проектов нужно было выработать новый подход к архитектуре CSS и проекта в целом.

Ещё один фактор, который повлиял на становление БЭМ — единство головной организации. Все проекты — это проекты одной компании, с близким дизайном и отдельными повторяющимися блоками. К примеру, во всех проектах есть схожие шапка, подвал и часть управляющих элементов. А всем хорошо известно, что копипáст (от англ. *copy & paste*, копирование и вставка) — это плохая практика (читайте про принцип *DRY*).

Так появилась идея создать общепортальный фреймфорк, основной фишкой которого были независимые блоки. Эти блоки можно использовать в любом месте портала. Фреймворку дали очевиднейшее имя — «Лего».

«Лего» решено использовать для создания брендбука общекорпоративного стиля. Все элементы дизайна разрабатываются как `html+css`, чтобы не хранить устаревающие тексты. Формулируются правила для полностью независимых блоков — *Блок как самостоятельная единица в дизайне*. Фреймворк подключается как внешняя библиотека к другим проектам. Используется версионность. Можно подключить нужную версию библиотеки.

«Наружу», вне Яндекса, про «Вёрстку независимыми блоками» начали рассказывать в 2007 году. В это время правила становятся более ясными: всё делается ради прозрачности, единообразия и реиспользования (к примеру, отказ от селекторов по `id` и тегам, длинного каскада и некоторых других практик). БЭМ приобретает законченный вид в 2009 году, это называется *Лего 2.0*, с этого времени блок превалирует над технологиями.

Сейчас БЭМ — это в большей степени концепция, чем жёстко зафиксированный свод правил. Его библия сформирована на сайте [bem.info](#), и составители рекомендуют использовать БЭМ так и в том объёме, чтобы это было полезно для вашего проекта.

## Ознакомились со статьёй?

Сохранить прогресс

Углублённая теория

Частые проблемы из-за неправильной CSS-архитектуры проекта



### Практикум

Тренажёры

Подписка

Для команд и компаний

Учебник по PHP

### Профессии

Фронтенд-разработчик

React-разработчик

Фулстек-разработчик

Бэкенд-разработчик

### Услуги

Работа наставником

Для учителей

Стать автором

### 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

PHP. Профессиональная веб-разработка

PHP и Yii. Архитектура сложных веб-сервисов

Node.js. Разработка серверов приложений и API

Анимация для фронтендеров

Вёрстка email-рассылок

Vue.js для опытных разработчиков

Регулярные выражения для фронтендеров

Шаблонизаторы HTML

Алгоритмы и структуры данных

Анатомия CSS-каскада

### Блог

С чего начать

Шпаргалки для разработчиков

Отчеты о курсах

### Информация

Об Академии

О центре карьеры

### Остальное

Написать нам

Мероприятия

Форум