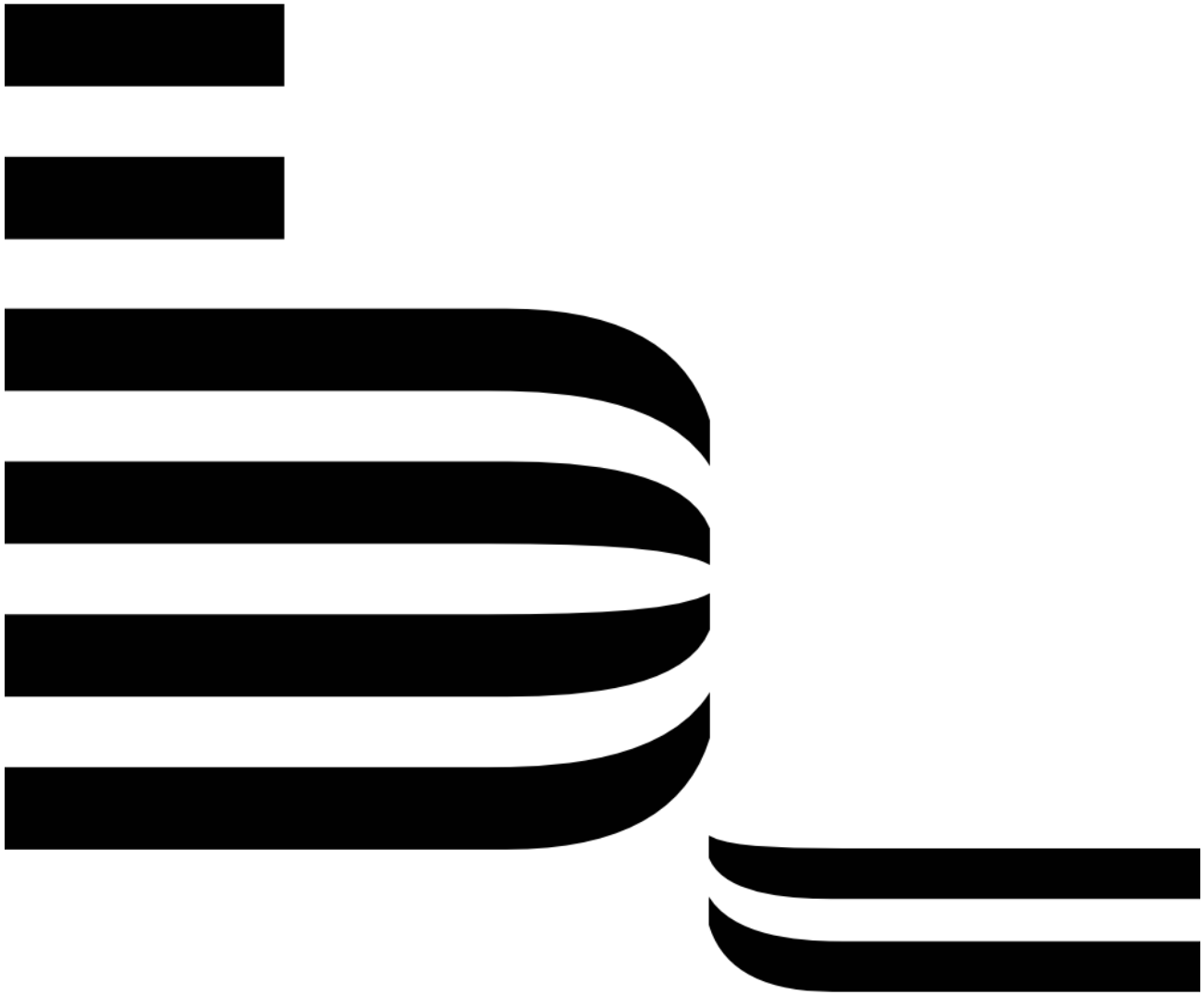

БЭМ

# БЭМ

БЭМ (Блок, Элемент, Модификатор) — компонентный подход к веб-разработке. В его основе лежит принцип разделения интерфейса на независимые блоки.

Он позволяет легко и быстро разрабатывать интерфейсы любой сложности и повторно использовать существующий код, избегая «Copy-Paste».

# БЛОК

Блок - функционально независимый компонент страницы, который может быть повторно использован. В HTML блоки представлены атрибутом class.

Особенности:

❖ Название блока характеризует смысл («что это?» — «меню»: menu, «кнопка»: button), а не состояние («какой, как выглядит?» — «красный»: red, «большой»: big).

```
<!-- Верно. Семантически осмысленный блок `error` -->
<div class="error"></div>

<!-- Неверно. Описывается внешний вид -->
<div class="red-text"></div>
```

# БЛОК

Особенности:

- Блок не должен влиять на свое окружение, т. е. блоку не следует задавать внешнюю геометрию (в виде отступов, границ, влияющих на размеры) и позиционирование.

- В CSS по БЭМ также не рекомендуется использовать селекторы по тегам или id.

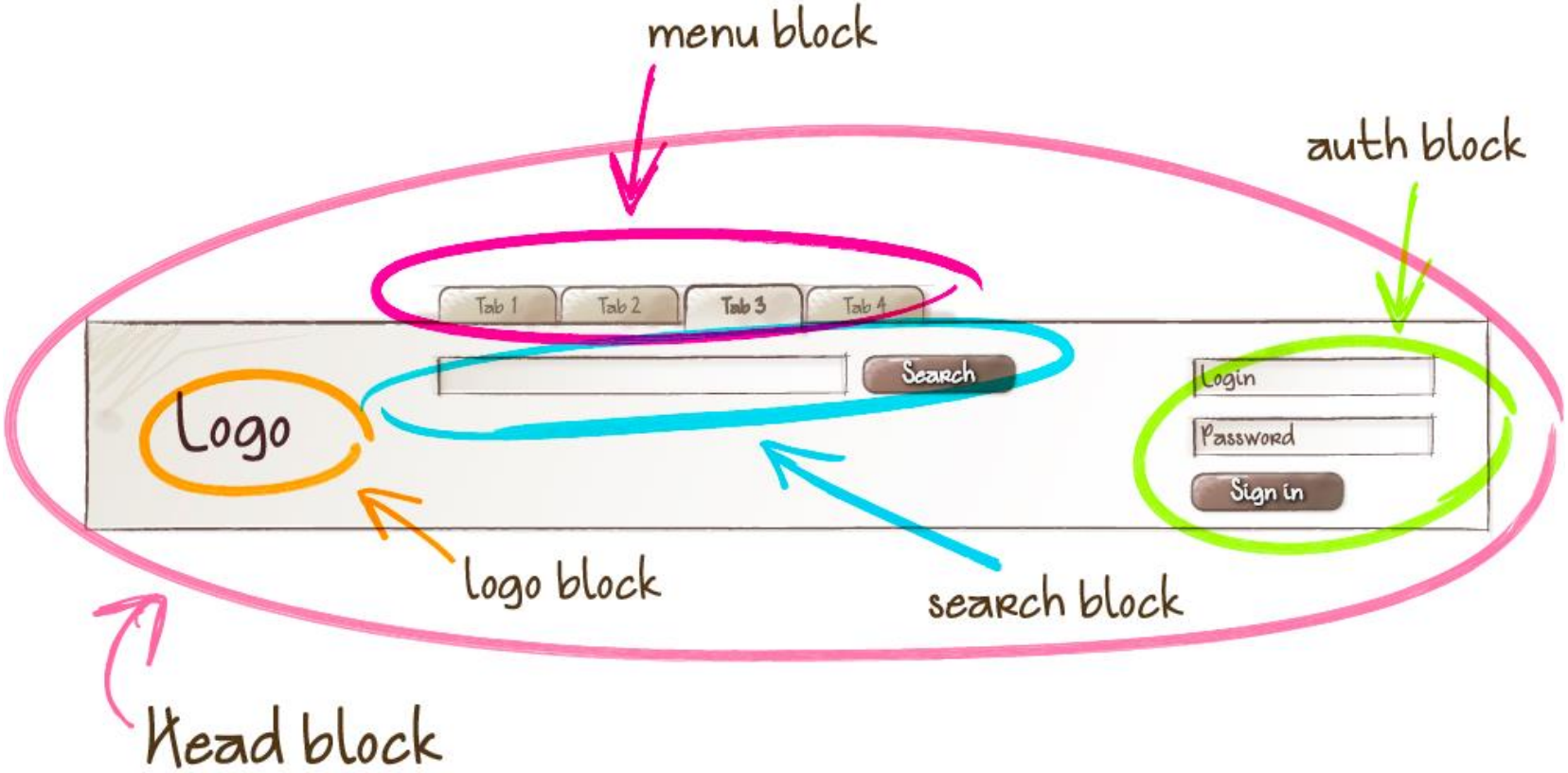
Таким образом обеспечивается независимость, при которой возможно повторное использование или перенос блоков с места на место.

# ВОЗМОЖНОСТИ БЛОКОВ

❖ Вложенная структура.

Блоки можно вкладывать в любые другие блоки.

Например, блок head может содержать логотип (logo), форму поиска (search) и блок авторизации (auth).

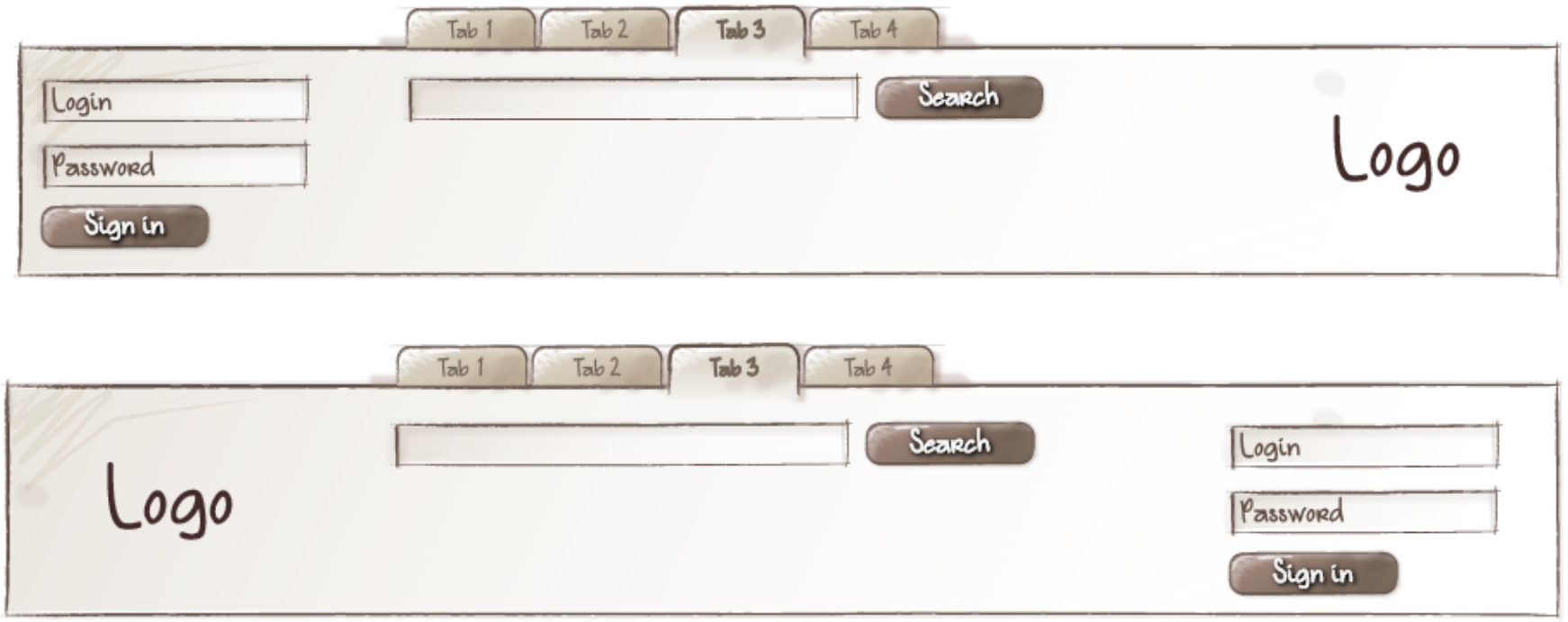


# ВОЗМОЖНОСТИ БЛОКОВ

- Свободное перемещение.

Блоки можно перемещать в пределах одной страницы и разных проектов. Независимость блока позволяет изменять его положение на странице и обеспечивает корректную работу и внешний вид.

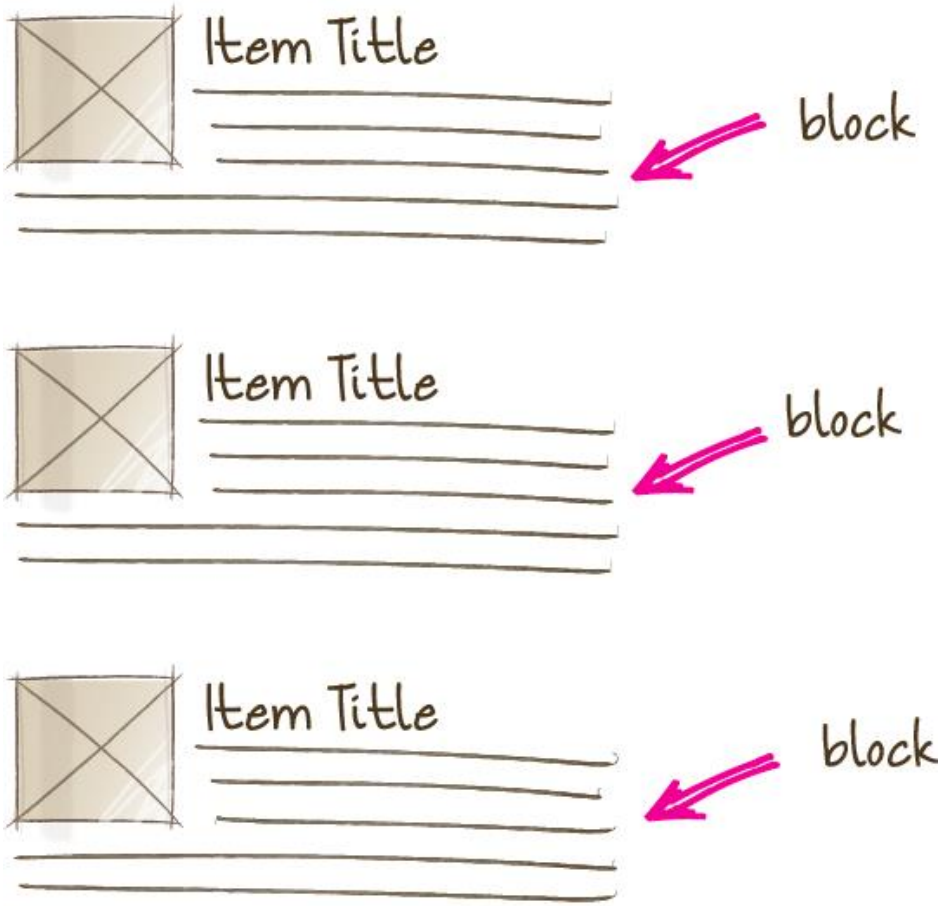
Так, например, логотип и форму авторизации можно поменять местами. При этом вносить изменения в CSS или JavaScript-код блоков не нужно.



# ВОЗМОЖНОСТИ БЛОКОВ

❖ Повторное использование.

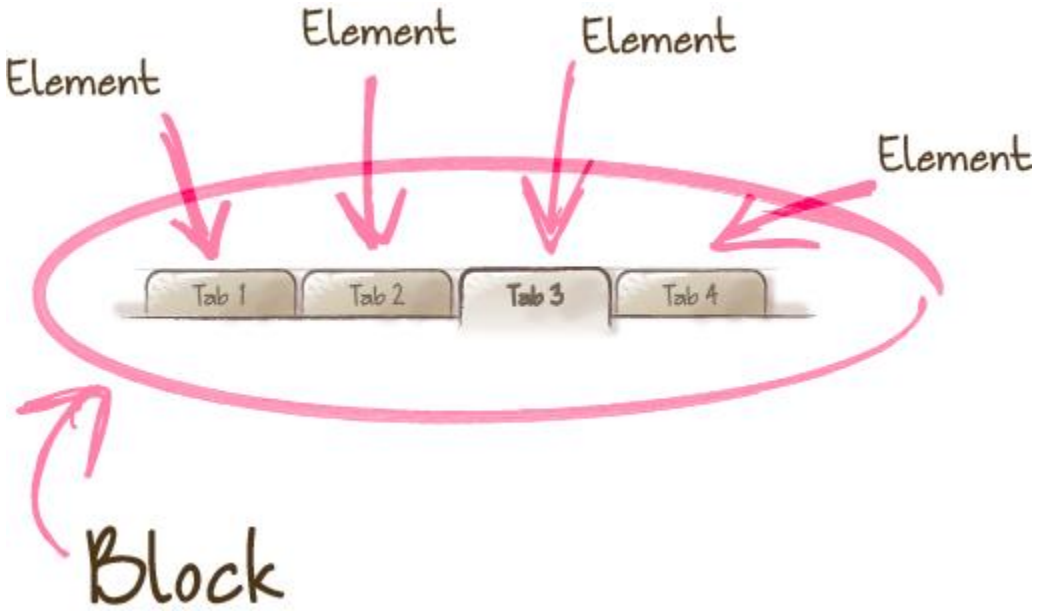
В интерфейсе может одновременно присутствовать несколько экземпляров одного и того же блока.



# ЭЛЕМЕНТ

Элемент - составная часть блока, которая не может использоваться в отрыве от него. Элементы не существуют вне блока. Каждый элемент может принадлежать только одному блоку.

Например, пункт меню вне блока меню не используется, значит является его элементом.



# ЭЛЕМЕНТ

Особенности:

❖ Название элемента характеризует смысл («что это?» — «пункт»: item, «текст»: text), а не состояние («какой, как выглядит?» — «красный»: red, «большой»: big).

❖ Структура полного имени элемента соответствует схеме: имя-блока__имя-элемента. Имя элемента отделяется от имени блока двумя подчеркиваниями (__).

```
<!-- Блок `search-form` -->
<form class="search-form">
    <!-- Элемент `input` блока `search-form` -->
    <input class="search-form__input">
    <!-- Элемент `button` блока `search-form` -->
    <button class="search-form__button">Найти</button>
</form>
```

# ПРИНЦИПЫ РАБОТЫ С ЭЛЕМЕНТАМИ

❖ Вложенность

Элементы можно вкладывать друг в друга. Допустима любая вложенность элементов. Элемент — всегда часть блока, а не другого элемента. Это означает, что в названии элементов нельзя прописывать иерархию вида block__elem1__elem2.

```
<form class="search-form">
   <div class="search-form__content">
      <input class="search-form__input">
      <button class="search-form__button">Найти</button>
   </div>
</form>
```

Это позволяет изменять DOM-структуру блока без внесения правок в названия классов.

# ПРИНЦИПЫ РАБОТЫ С ЭЛЕМЕНТАМИ

❖ Принадлежность

Элемент — всегда часть блока и не должен использоваться отдельно от него.

```
<!-- Блок `search-form` -->
<form class="search-form">
    <!-- Элемент `input` блока `search-form` -->
    <input class="search-form__input">
</form>

<!-- Неверно. Элементы лежат вне контекста блока `search-form` -->
<!-- Блок `search-form` -->
<form class="search-form"></form>
<!-- Элемент `input` блока `search-form` -->
<input class="search-form__input">
```

# ПРИНЦИПЫ РАБОТЫ С ЭЛЕМЕНТАМИ

❖ Необязательность

Элемент — необязательный компонент блока. Не у всех блоков должны быть элементы.

```
<!-- Блок `search-form` -->
<div class="search-form">
    <!-- Блок `input` -->
    <input class="input">
    <!-- Блок `button` -->
    <button class="button">Найти</button>
</div>
```

# КОГДА СОЗДАВАТЬ БЛОК, КОГДА — ЭЛЕМЕНТ?

- Создавайте блок

Если фрагмент кода может использоваться повторно и не зависит от реализации других компонентов страницы.

- Создавайте элемент

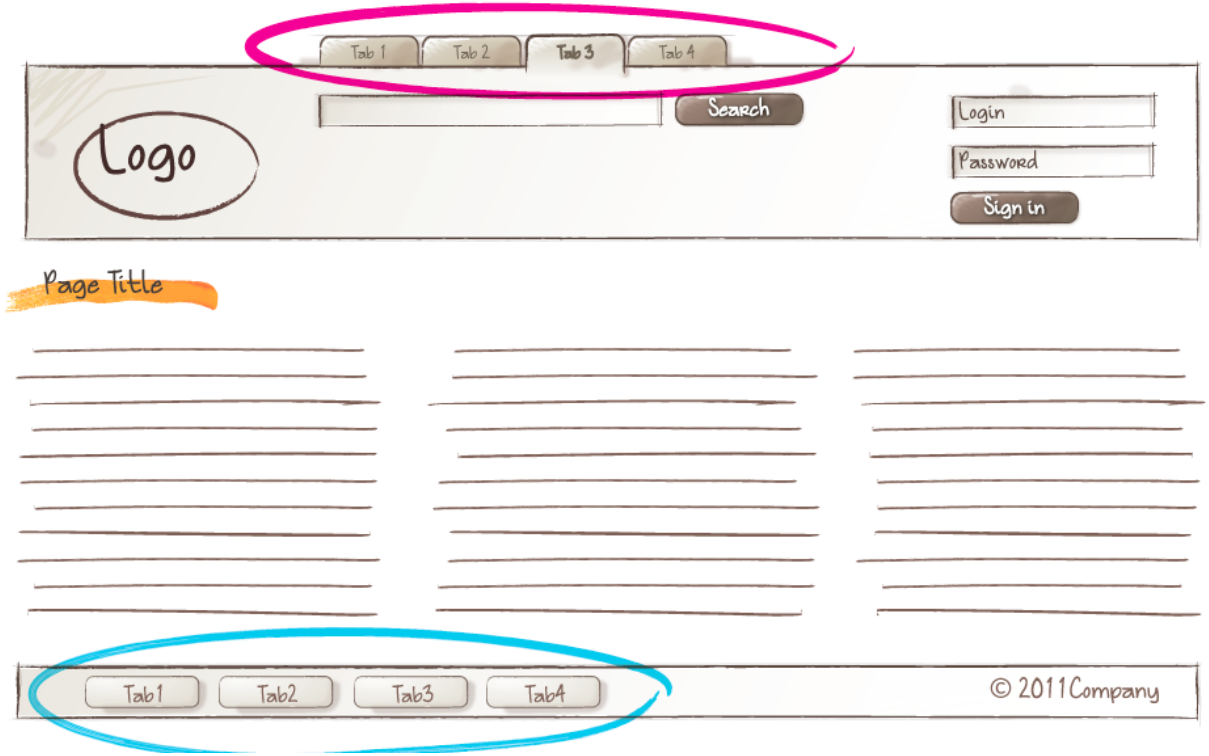
Если фрагмент кода не может использоваться самостоятельно, без родительской сущности (блока).

Исключение составляют элементы, реализация которых для упрощения разработки требует разделения на более мелкие части — подэлементы. В БЭМ-методологии нельзя создавать элементы элементов. В подобном случае вместо элемента необходимо создавать служебный блок.

# МОДИФИКАТОР

Модификатор - сущность, определяющая внешний вид, состояние или поведение блока либо элемента.

Например, внешний вид блока меню (menu) может меняться в зависимости от примененного модификатора.



# МОДИФИКАТОР

Особенности:

- Название модификатора характеризует внешний вид («какой размер?», «какая тема?» и т. п. — «размер»: size_s, «тема»: theme_islands), состояние («чем отличается от прочих?» — «отключен»: disabled, «фокусированный»: focused) и поведение («как ведет себя?», «как взаимодействует с пользователем?» — «направление»: directions_left-top).
- Имя модификатора отделяется от имени блока или элемента одним подчеркиванием (_).

# ТИПЫ МОДИФИКАТОРОВ

- Булевый

Используют, когда важно только наличие или отсутствие модификатора, а его значение несущественно. Например, «отключен»: disabled. Считается, что при наличии булевого модификатора у сущности его значение равно true.

Структура полного имени модификатора соответствует схеме:

имя-блока_имя-модификатора;

имя-блока__имя-элемента_имя-модификатора.

```
<!-- Блок `search-form` имеет булевый модификатор `focused` -->
<form class="search-form search-form_focused">
   <input class="search-form__input">
   <!-- Элемент `button` имеет булевый модификатор `disabled` -->
   <button class="search-form__button
         search-form__button_disabled">Найти</button>
</form>
```

# ТИПЫ МОДИФИКАТОРОВ

❖ Ключ-значение

Используют, когда важно значение модификатора. Например, «меню с темой оформления islands»: menu_theme_islands.

Структура полного имени модификатора соответствует схеме:
имя-блока_имя-модификатора_значение-модификатора;
имя-блока__имя-элемента_имя-модификатора_значение-модификатора.

```
<!-- Блок `search-form` имеет модификатор `theme` со значением `islands` -->
<form class="search-form search-form_theme_islands">
   <input class="search-form__input">

   <!-- Элемент `button` имеет модификатор `size` со значением `m` -->
   <button class="search-form__button
           search-form__button_size_m">Найти</button>
</form>
```

# ПРИНЦИПЫ РАБОТЫ С МОДИФИКАТОРАМИ

❖ Модификатор нельзя использовать самостоятельно

С точки зрения БЭМ-методологии модификатор не может использоваться в отрыве от модифицируемого блока или элемента. Модификатор должен изменять вид, поведение или состояние сущности, а не заменять ее.

```
<!-- Верно. Блок `search-form` имеет модификатор `theme` со значением `islands`-->
<form class="search-form search-form_theme_islands">
    <input class="search-form__input">
    <button class="search-form__button">Найти</button>
</form>
<!-- Неверно. Отсутствует модифицируемый класс `search-form` -->
<form class="search-form_theme_islands">
    <input class="search-form__input">
    <button class="search-form__button">Найти</button>
</form>
```

# МИКС

Прием, позволяющий использовать разные БЭМ-сущности на одном DOM-узле.

Миксы позволяют совмещать поведение и стили нескольких сущностей без дублирования кода; создавать семантически новые компоненты интерфейса на основе имеющихся.

```
<!-- Блок `header` -->
<div class="header">
    <!-- К блоку `search-form` примиксован элемент `search-form` блока `header`-->
    <div class="search-form header__search-form"></div>
</div>
```

В данном примере мы совместили поведение и стили блока search-form и элемента search-form блока header. Такой подход позволяет нам задать внешнюю геометрию и позиционирование в элементе header__search-form, а сам блок search-form оставить универсальным.

# Соглашение по именованию

Имя БЭМ-сущности уникально.

Правила формирования имен:
block-name__elem-name_mod-name_mod-val

- Имена записываются латиницей в нижнем регистре.
- Для разделения слов в именах используется дефис (-).
- Имя блока задает пространство имен для его элементов и модификаторов.
- Имя элемента отделяется от имени блока двумя подчеркиваниями (__).
- Имя модификатора отделяется от имени блока или элемента одним подчеркиванием (_).
- Значение модификатора отделяется от имени модификатора одним подчеркиванием (_).
- Значение булевых модификаторов в имени не указывается.

# Альтернативные схемы именования

Стиль Two Dashes

Правила формирования имен:
block-name__elem-name--mod-name--mod-val

- Имена записываются латиницей в нижнем регистре.
- Для разделения слов в именах БЭМ-сущностей используется дефис (-).
- Имя элемента отделяется от имени блока двумя подчеркиваниями (__).
- Булевые модификаторы отделяются от имения блока или элемента двумя дефисами (--).
- Значение модификатора отделяется от его имени двумя дефисами (--).

# Альтернативные схемы именования

Стиль CamelCase

Правила формирования имен:
blockName__elemName_modName_modVal

- Имена записываются латиницей.
- Каждое слово внутри имени пишется с заглавной буквы.
- Разделители элементов и модификаторов совпадают с классической схемой.

# Альтернативные схемы именования

Стиль React

Правила формирования имен:
BlockName-ElemName_modName_modVal

- Имена записываются латиницей.
- Имена блоков и элементов пишутся с заглавной буквы. Имена модификаторов — со строчной.
- Каждое слово внутри имени пишется с заглавной буквы.
- Имя элемента отделяется от имени блока одним дефисом (-).
- Разделители имени и значения модификаторов совпадают с классической схемой.

# Альтернативные схемы именования

Стиль No-namespace

Правила формирования имен:

_available

- Имена записываются латиницей.
- Имя блока или элемента перед модификатором не указывается.

Такая схема именования ограничивает использование миксов, так как не дает возможности определить, к какому блоку или элементу относится модификатор.

# ИСТОЧНИКИ

https://ru.bem.info/methodology/quick-start/ - Документация по БЭМ