**Псевдоелемет**

**Псевдоелемет::after**

Псевдоелемент, який використовується для виведення бажаного контенту після вмісту елемента, до якого він додається.

**Псевдоелемет::before**

Застосовується для відображення контенту до вмісту елемента, до якого він додається.

[**Псевдоелемет::first-letter**](https://webref.ru/css/first-letter)

Определяет стиль первого символа в тексте элемента, к которому добавляется.

[**Псевдоелемет::first-line**](https://webref.ru/css/first-line)

Задаёт стиль первой строки форматированного текста.

[**Псевдоелемет::placeholder**](https://webref.ru/css/placeholder)

Псевдоэлемент, с помощью которого задаётся стилевое оформление подсказывающего текста, созданного атрибутом placeholder.

[**Псевдоелемет::selection**](https://webref.ru/css/selection)

Применяет стиль к выделенному пользователем фрагменту текста.

**Псевдоклас:**

* визначає конкретний стан елемента;
* це ключове слово, яке починається з двокрапки.

**Синтаксис**

Псевдоклас не може існувати сам по собі. Він повинен бути прикріплений до селектора. Псевдоклас визначатиме тільки певний стан цього селектора.

Синтаксис виглядає наступним чином:

*селектор: псевдоклас {}*

Прогалини між селектором і псевдоклас немає, щоб показати, що вони пов'язані один з одним.

**:hover**

Наприклад, типовим використовуваним псевдоклас є: hover, який буде застосовувати стиль, коли на цільової елемент наводиться покажчик миші. Давайте перевіримо це на засланнях.

**HTML**

<p>Наведите курсор <a href="#">на эту ссылку</a> и увидите, как она становится красной.</p>

**CSS**

a { color: blue; }

a:hover { color: red; }

Перший рядок у CSS визначає, що всі елементи <a> повинні виглядати синіми. Другий рядок визначає, що при наведенні покажчика миші <a> повинні виглядати червоними.

Другий рядок орієнтована на ті ж елементи HTML, але тільки, коли відбувається щось конкретне (в даному випадку, наведення).

**:visited**

Цей псевдоклас націлюється на посилання, які вже були відвідані. За замовчуванням посилання відображаються синіми і при відвідуванні стають фіолетовими. Результати Google працюють так само.

**HTML**

<a href="https://www.google.com">Google</a>

<a href="https://twitter.com">Twitter</a>

<a href="https://www.facebook.com">Facebook</a>

<a href="https://www.mozilla.org">Mozilla</a>

<a href="http://marksheet.io/visited.html">MarkSheet</a>

**CSS**

a { color: dodgerblue; }

a:visited { color: rebeccapurple; }

Міняти колір відвіданих посилань часто забувають, але користувачам так зручніше переглядати список результатів. Це допомагає їм легко визначати, де вони вже були.

**:focus**

Даний псевдоклас відбувається, коли елемент HTML отримує фокус. Це особливо корисно для полів форм.

**HTML**

<p>

<input class="form-input" placeholder="Введіть ім’я">

</p>

**CSS**

.form-input { border: 2px solid grey; padding: 5px; }

.form-input:focus { background: lightyellow; border-color: blue; outline: none; }

**:First-child і: last-child**

Ці псевдокласи пов'язані з ієрархією в HTML. Вони націлюються на елементи HTML в залежності від порядку, в якому вони з'являються в коді.

**HTML**

<ul>

<li>Один</li>

<li>Два</li>

<li>Три</li>

<li>Чотири</li>

</ul>

**CSS**

li:first-child { background: greenyellow; }

li:last-child { background: lightsalmon; }

**:nth-child**

Даний псевдоклас є глобальною версією: first-child і: last-child. За допомогою: nth-child ви можете вирахувати конкретний дочірній цільової елемент.

Наприклад, якщо ви хочете націлитися на другий дочірній елемент, то використовуйте: nth-child (2).

**HTML**

<ul>

<li>Один</li>

<li>Два</li>

<li>Три</li>

<li>Чотири</li>

</ul>

**CSS**

li:nth-child(2) { background: violet; }

**odd і even**

Крім використання чисел, :nth-child поставляється з двома ключовими словами:

: Nth-child (odd) націлена на кожен непарний елемент;

: Nth-child (even) націлена на кожен парний елемент.

li:nth-child(odd) { background: gold; }

**Лічильник n**

Потужним аспектом: nth-child є те, що він може націлюватися на елементи, грунтуючись на розрахунках за допомогою ключового слова n.

Значення n підвищується від 0 до кількості представлених дочірніх елементів.

Що робити, якщо ви хочете націлитися на кожен третій елемент?

li:nth-child(3n) { background: hotpink; }

У нашому випадку n починається з нуля і закінчується шістьма.

Комп'ютери починають відлік з нуля. І оскільки в нашому списку сім елементів, ми будемо вважати до шести, тому що 0-1-2-3-4-5-6 представляють сім пунктів.

Ви можете прочитати: nth-child (3n) як «націлитися на кожен елемент, положення якого ділиться на 3». У нашому випадку, це третій і шостий пункт списку:

* 3 помножити на 0 дорівнює нулю;
* 3 помножити на 1 - третій пункт;
* 3 помножити на 2 - шостий пункт.

n + 1

Що робити, якщо ви хочете націлитися на перший пункт і кожен третій пункт після нього?

<https://webref.ru/css/type/pseudoelement>

<https://webref.ru/css/type/pseudoclass>

**Оптимізація картинок**

Картинки потрібно оптимізовувати, тобто зжимати їхній розмір. Ось 2 сайти за допомогою яких можна зжати картинки.

<http://optimizilla.com/ru/>

<https://compressor.io/>

**БЕМ -** абревіатура від слів Блок, Елемент, Модифікатор - методологія іменування, винайдена хлопцями з Яндекса.  
Ця методологія організаціїї CSS коду стала дуже швидко популярна в багатьох ІТ компаніях, навіть компанія Google використовує її на своїх проектах. В деталі історії та винекнення дуже вдаватися не будемо, так чим же корисна ця річ?

Діло в тому, що методологія **БЕМ** дозволяє нам ясніше організовувати свій код. Ми вже не бачимо CSS як набір селекторів ніяким чином не пов’язаних один з одним, а як окремі незалежні об’єкти які мають свої складові та модифікації.  
Це дозволяє легко розробляти великі так і не великі проекти. Також приходить зручність у підтримці проектів які написані за даною методологією.  
  
В чому суть?  
  
Є певна система найменувань класів, яка використовує подібні шаблони:  
  
.block{} – батьківський блок, найвищий рівень вкладеності елементів.  
.block\_\_element{} – дочірній елемент, як складова основного блоку, яка доповнює його цілісність.  
.block--modifier{} – інша форма основного блоку, іншими словами модифікація.

Мета БЕМ - розповісти іншим розробникам якомога більше про те, що робить частина коду, тільки за назвами класів в розмітці. Читаючи HTML з невеликою кількістю класів всередині, ви можете побачити взаємодії у всьому коді і між його частинами; щось може бути компонентом, щось дочірнім елементом - елементом цього компонента, а щось може бути зміненою копією - модифікатором компонента.

Для більш детального пізнання дуже рекомендовано перейти за посиланнями та уважно почитати статті про дану методологію:  
  
<https://noteskeeper.ru/527/>

<https://habrahabr.ru/post/256109/>

<https://ru.bem.info/methodology/key-concepts/>

<https://habr.com/ru/post/203440/>

https://ru.bem.info/methodology/quick-start/