## Занятие #8. Псевдоклассы :first-child, :last-child, :nth-child. Псевдоэлементы ::before и ::after. Подключение нестандартных шрифтов через @font-face

### Псевдоклассы - :first-child, :last-child, :nth-child

Псевдокласс **:first-child** применяет стилевое оформление к первому дочернему элементу своего родителя. Псевдокласс **:last-child** задаёт стилевое оформление последнего элемента своего родителя.

Например, на странице есть список:  
 <ul class="list">

<li>Lorem ipsum</li>

<li>Lorem ipsum</li>

<li>Lorem ipsum</li>

</ul>

Если в CSS записать: **.list li:first-child{color: red}**, то в результате только в первом пункте списка цвет текста станет красным, так как этот пункт является первым дочерним элементом тега <ul> с классом list

**Псевдокласс :nth-child** используется для добавления стиля к элементам на основе нумерации в дереве элементов.

**Псевдоклассу :nth-child**  можно задавать следующие значения

**odd** - все нечетные номера элементов.

**even** - все четные номера элементов.

**число** - порядковый номер дочернего элемента относительно своего родителя (нумерация начинается с 1)

**выражение** - задается в виде an+b, где a и b целые числа, а n — счетчик, который автоматически принимает значение 0, 1, 2...

a и b могут быть отрицательными числами, в этом случае знак плюс меняется на минус, например: 5n-1, при этом следует учитывать, что общее значение выражения должно быть положительным числом, т.к. нумерация элементов начинается

с 1.

|  |  |  |
| --- | --- | --- |
| Синтаксис | Номера элементов | Описание |
| :nth-child (1) | 1 | Первый элемент, является синонимом псевдокласса **:first-child.** |
| :nth-child (5) | 5 | Пятый элемент. |
| :nth-child (2n) | 2, 4, 6, 8, 10 | Все четные элементы, аналог значения **even.** |
| :nth-child (2n+1) | 1, 3, 5, 7, 9 | Все нечетные элементы, аналог значения **odd.** |
| :nth-child (even) | 2, 4, 6, 8, 10 | Все четные элементы. |
| :nth-child (odd) | 1, 3, 5, 7, 9 | Все нечетные элементы. |
| :nth-child (3n+2) | 2, 5, 8, 11, 14 |  |

### Псевдоэлементы

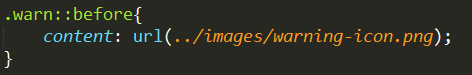
Мы с вами уже хорошо знакомы с элементами разметки html - тэгами, знаем, как они работают и как нам их применять. Сегодня мы узнаем про псевдоэлементы.

Псевдоэлементы :before и :after используются для того, чтобы добавить какое-то содержимое перед элементом или после него.

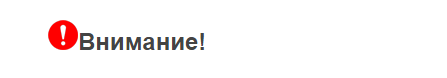
Например, если мы захотим вставить иконку warning-icon.png перед важным заголовком, нам нужно будет в разметке html создать только заголовок



И добавить к нему в стилях вот такой код: мы обращаемся к заголовку по классу .warn, и перед ним создаем элемент(:before), в который с помощью атрибута content помещаем картинку.



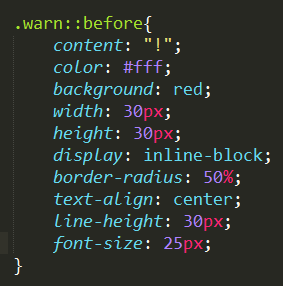
В браузере мы увидим, что картинка расположилась прямо перед текстом так, будто мы вставили ее тэгом img:



Раньше мы вставляли дополнительные тэги <i>, <b> или <span> в разметку html, только для того, чтобы добавить, например, иконочку для кнопки. Теперь мы можем воспользоваться для этого красивым способом, и не писать сложную разметку. Этот способ удобен тем, что он разгружает html, т.е. нам не нужны дополнительные тэги в разметке.

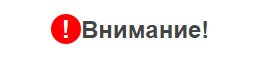
Обратите внимание на синтакис - в нашем css псевдоэлемент ::before написан с двумя двоеточиями. Два двоеточия были введены чтобы различать псевдоэлементы (например, before, after) от псевдоклассов (например, hover, first-child).

Для новых, введённых в CSS3, псевдоэлементов в спецификации вообще запрещают использовать одно двоеточие, допустимо только два. Для псевдоэлементов из CSS2 (before, after) можно по прежнему использовать и одно двоеточие, но чтобы отличать их от псевдоклассов, их часто пишут с двумя двоеточиями, по самым последним стандартам. Имейте ввиду, что IE8 поддерживает только старый синтаксис для старых псевдоэлементов, т.е. с одним двоеточием :before и :after



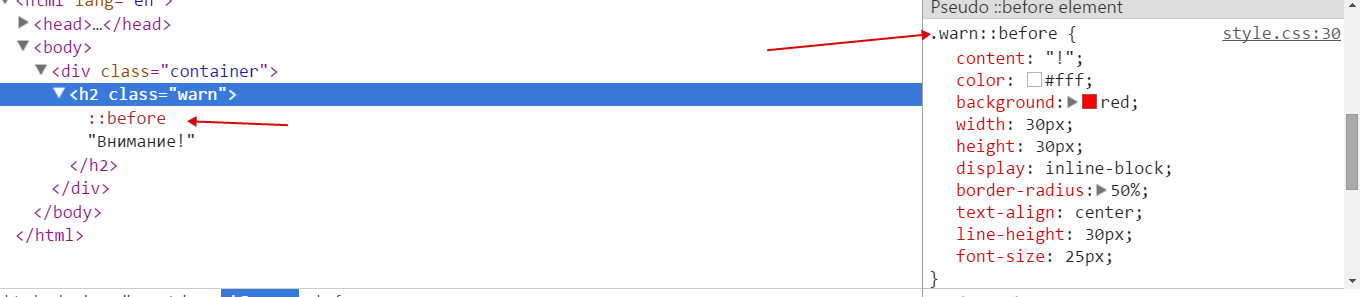
Обратите внимание, что содержимое псевдоэлементу добавляется особым образом - из css атрибутом content. В него можно вложить не только картинку, но и любой текст или символ. Давайте попробуем вложить в псевдоэлемент не готовую картинку, а символ, и стилизовать его:

В браузере получим почти то же самое:



Еще стоит помнить, что

* псевдоэлементы создаются отдельно от разметки, они на особом счету
* им обязательно нужно добавить содержимое (content: ;)
* к ним нельзя обратиться средствами javascript (запомните на будущее)
* в них нельзя вложить никакие другие элементы (они не могут быть родителями)
* их нельзя создать внутри элемента, который не принимает дочерних тэгов. Т.е. если элемент сам не может быть родителем, то у него не может быть ::before или ::after (таким образом псевдоэлементы неприменимы к input и img)
* изначально ::before и ::after - строчные элементы
* в инспекторе псевдоэлементы тоже выглядят по особому, их можно просмотреть как отдельный элемент, или как псевдоэлемент родителя.



### @font-face

Правило @font-face позволяет импортировать внешний файл шрифта в документ. Это значит, что если у пользователя не установлен нужный шрифт, он подгрузится с сервера без установки в операционную систему.

Внутри конструкции @font-face может находиться набор свойств для изменения параметров шрифта (font-family,font-size, font-style и др.), а также ссылка на шрифтовой файл. Ссылка записывается в виде src: url(URI), где URI — относительный или абсолютный путь к файлу.

Для обеспечения кроссбраузерности необходимо подключать различные форматы одного и того же шрифта если необходима очень широкая поддержака браузеров в том числе и старых то это следующие форматы (ttf, eot,woff, woff2, svg). Для современных браузеров достаточно 2-х форматов woff и woff2

Для конвертации шрифтов можно пользоваться онлайн сервисами, например этим: <https://www.fontsquirrel.com/tools/webfont-generator>

**Полный синтаксис внедрения шрифта, в котором учтены некоторые возможные баги IE и поддержка разных браузеров, выглядит следующим образом:**

@font-face {

font-family: 'MyWebFont';  
 src: url('webfont.eot'); /\* IE9 Compat Modes \*/  
 src: url('webfont.eot?#iefix') format('embedded-opentype'), /\* IE6-IE8 \*/  
 url('webfont.woff2') format('woff2'), /\* Super Modern Browsers \*/  
 url('webfont.woff') format('woff'), /\* Pretty Modern Browsers \*/  
 url('webfont.ttf') format('truetype'), /\* Safari, Android, iOS \*/  
 url('webfont.svg#svgFontName') format('svg'); /\* Legacy iOS \*/

font-weight: normal;

font-style: normal;

}

Если поддерживать только современные версии браузеров, то достаточно использовать форматы шрифтов: woff и woff2

@font-face {

font-family: 'MyWebFont';  
 src: url('webfont.woff2') format('woff2'),  
 url(webfont.woff') format('woff');

font-weight: normal;

font-style: normal;

}

Для более глубокой поддержки, можно также добавить формат ttf

@font-face {

font-family: 'MyWebFont';  
 src: url(‘webfont.woff2') format('woff2'),  
 url('webfont.woff') format('woff'),  
 url(webfont.ttf') format('truetype');

font-weight: normal;

font-style: normal;

}