GeekBrains

HTML/CSS. Интерактивный курс

Псевдоклассы, табличная вёрстка

На этом уроке

1. Узнаем о блочных и строчных элементах. Поговорим об их сходствах и отличиях.
2. Узнаем, что такое псевдоклассы и псевдоэлементы, и где они применяются.
3. Разберём табличную вёрстку, где её нужно использовать, а где применять уже не стоит.

Оглавление

Основные теги вёрстки

В вёрстке сайта с помощью слоёв самый часто используемый HTML-тег — . Он и формирует слой на веб-странице. Это блочный тег.

Второй тег, который используется в вёрстке, — строчный тег .

Сами по себе эти теги ничего на экране не отображают и оформляются стилями CSS. Простыми словами, всё на странице состоит из прямоугольников. Это и будут элементы div.

Элементы span используются для стилизации какой-то произвольной части текста. Например, текст зелёного цвета. Чтобы эту часть реализовать в вёрстке, требуется обернуть этот текст тегом span.

```
<div>Блочный элемент</div>
<span>Строчный элемент</span>
```

Блочный элемент создаёт разрыв строки перед тегом и после него. Он образует прямоугольную область. Она занимает всю ширину веб-страницы (100%) или блока-родителя, если для него нет значения width.

Блочные элементы содержат элементы любого типа. Нельзя размещать блочные элементы внутри строчных, кроме элемента <img>. Для блочных элементов задаются отступы. Свойства width и height устанавливают ширину и высоту области содержимого элемента. Фактическая ширина элемента складывается из ширины полей (внутренних отступов), границ и внешних отступов.

Строчные элементы не создают блоки, а отображаются на одной строке с содержимым рядом стоящих тегов. Это потомки блочных элементов. Они игнорируют верхние и нижние отступы.

Ширина и высота строчного элемента зависит только от его содержания. Задать его размеры с помощью CSS нельзя. Можно увеличить расстояние между соседними элементами по горизонтали с помощью горизонтальных полей и отступов.

Рассмотрим основные блочные и строчные элементы HTML.

Блочные элементы

- <h1>, <h2>...<h6>;
- …;
- …
- <body>…</body>;
- … ;

- … ;
- …;
- ...

Строчные элементы

- ...;
- ...;
- ...;
- ...;
- ...;
- ...;
- <img>;
- <input>;
- ...

В этом списке тег картинки <img> — замещаемый строчный элемент. При помощи замещаемых элементов указывается, что в этом месте должен быть какой-то сторонний объект, в нашем случае, картинка. Замещающему элементу задаётся ширина и высота. Но это всё равно строчный элемент.

Формирование блочной модели

Атрибуты width и height — неокончательные размеры элемента. Чтобы вычислить размеры, требуется учитывать следующие моменты.

Согласно схеме, напрашивается вывод, что ширина блока складывается из таких свойств:

```
margin-left +
border-left +
padding-left +
width +
padding-right +
border-right +
margin-right
```

Соответственно, высота из следующих:

```
margin-top +
border-top +
padding-top +
height +
padding-bottom +
border-bottom +
margin-bottom
```

Внутренний отступ или поле элемента (padding) добавляет отступы внутри элемента, между его основным содержимым и границей. Если элементу задать фон, он распространится и на поля элемента. Такой отступ не принимает отрицательных значений, в отличие от внешнего отступа.

Внешний отступ (margin) добавляет отступы за границами элемента, создавая тем самым промежутки между элементами. Они всегда остаются прозрачными. Через них виден фон родительского элемента. Значения padding и margin задаются в следующем порядке: верхнее, правое, нижнее и левое.

Граница или рамка элемента задаётся с помощью свойства border. Если цвет рамки не указывается, она принимает цвет основного содержимого элемента, например, текста. Если рамка имеет разрывы, то сквозь них проступает фон элемента.

Внешние, внутренние отступы и рамка элемента — необязательные свойства. По умолчанию их значение равно нулю. Но некоторые браузеры добавляют к ним положительные значения автоматически на основе собственных таблиц стилей.

Псевдоклассы и псевдоэлементы

С псевдоклассами и псевдоэлементами назначаются CSS-стили необязательным на веб-странице структурам. Но стили применяются к частям документа не на основании информации в его структуре. Часто по структуре невозможно понять, что для этих элементов используются стили.

Псевдоклассы

С псевдоклассами добавляются особые классы к элементам. Выбираются объекты, которые:

- отсутствуют в структуре веб-страницы;
- нельзя выбрать с помощью обычных селекторов. Например, первая буква или первая строка одного абзаца.

Вы наверняка замечали на сайтах: когда наводите мышкой на конкретный пункт меню, он меняет свой вид. У него изменяется цвет фона, цвет ссылки, даже шрифт или его размер. Это происходит благодаря псевдоклассам. Рассмотрим их синтаксис.

```
Селектор:псевдокласс {
      свойство1: значение1;
}

a:hover {
     color: #ccc;
}
```

После селектора ставится двоеточие. Сразу после него без пробела указывается название псевдокласса.

Псевдоклассы, определяющие состояние элементов

1. a:link — ссылается на непосещённую ссылку.
2. a:visited — ссылается на уже посещённую ссылку.
3. a:hover — ссылается на любой элемент, по которому проводят курсором мыши.
4. a:focus — ссылается на любой элемент, над которым находится курсор мыши.
5. a:active — ссылается на активизированный пользователем элемент.
6. :valid — выберет поля формы, содержимое которых прошло проверку в браузере на соответствие указанному типу.
7. :invalid — выберет поля формы, содержимое которых не соответствует указанному типу.
8. :enabled — выберет все доступные (активные) поля форм.
9. :disabled — выберет заблокированные поля форм, т. е. находящиеся в неактивном состоянии.
10. :in-range — выберет поля формы, значения которых находятся в заданном диапазоне.
11. :out-of-range — выберет поля формы, значения которых не входят в установленный диапазон.
12. :lang() — выбирает абзацы на указанном языке.
13. :not(селектор) — выберет элементы, которые не содержат указанный селектор. Например, класс, идентификатор или селектор элемента :not([type="submit"]).
14. :checked — выбирает выделенные (выбранные пользователем) элементы.

Структурные псевдоклассы

1. :nth-child(odd) — выбирает нечётные дочерние элементы.
2. :nth-child(even) — выбирает чётные дочерние элементы.
3. :nth-child(3n) — выбирает каждый третий элемент среди дочерних.
4. :nth-child(3n+2) — выбирает каждый третий элемент, начиная со второго дочернего элемента (+2).
5. :nth-child(n+2) — выбирает все элементы, начиная со второго.
6. :nth-child(3) — выбирает третий дочерний элемент.
7. :nth-last-child() — в списке дочерних элементов выбирает элемент с указанным местоположением, аналогично с :nth-child(), но начиная с последнего, в обратную сторону.
8. :first-child — позволяет оформить только самый первый дочерний элемент тега.
9. :last-child — позволяет форматировать последний дочерний элемент тега.
10. :only-child — выбирает элемент, являющийся единственным дочерним элементом.
11. :empty — выбирает элементы, у которых нет дочерних элементов.
12. :root — выбирает корневой элемент в документе (элемент html).

Псевдоклассы по типу дочернего элемента

1. :nth-of-type() — выбирает элементы по аналогии с :nth-child(), при этом берёт во внимание только тип элемента.
2. :first-of-type — позволяет выбрать первый дочерний элемент.
3. :last-of-type — выбирает последний тег конкретного типа.
4. :nth-last-of-type() — выбирает элемент заданного типа в списке элементов в соответствии с указанным местоположением начиная с конца.
5. :only-of-type — выбирает единственный элемент указанного типа среди дочерних элементов родительского элемента.

Пример использования псевдоклассов

HTML	CSS

 - Первый элемент
 - Второй элемент
 - Третий элемент
 	
	li:first-child{
	font-size: 24px;
	color: #F23401;
	}

Чтобы понять, как работает этот псевдокласс, рассмотрим простой пример. Зададим элементу списка <li> псевдокласс first-child, и прописываем у него определённый стиль. Как видно, маркированный список состоит из трёх элементов. Указанный стиль применится ТОЛЬКО к первому элементу списка. Это происходит потому, что первый элемент списка <li> будет первым дочерним у тега <ul>.

HTML	CSS

  Первый элемент
  Второй элемент
  Третий элемент
 	
	li:first-child a {
	color: red;
	}

Одно из самых простых правил — просто проговорить, что требуется сделать.

Если нужно найти первый элемент списка (ссылку), запись будет следующей li:first-child a { color: red; }

Комбинирование псевдоклассов

Псевдоклассы комбинируются в одном селекторе и перечисляются через двоеточие:

```
/* При навсылку цвет текста будет зеленымедении на не посещенную с */
a:link:hover {
    color: #0F0;
```

```
}

/* При наведении на посещенную ссылку цвет текста будет красным*/
a:visited:hover {
    color: #F00;
}
```

Псевдоэлементы

Псевдоэлементы позволяют ввести несуществующие элементы в веб-документ и придать им определённые стили. Псевдоэлементы появились ещё в CSS1, но пошли в релиз только в CSS2.1. В самом начале в синтаксисе использовалось одно двоеточие, но в CSS3 используется двойное двоеточие для отличия от псевдоклассов. Современные браузеры умеют понимать оба типа синтаксиса псевдоэлементов, кроме Internet Explorer 8. Он воспринимает только одно двоеточие. Поэтому надёжнее использовать одно. С помощью свойства content можно изменить внешний вид части элемента.

Рассмотрим часто используемые псевдоэлементы:

1. :first-letter — выбирает первую букву каждого абзаца, применяется только к блочным элементам.
2. :first-line — выбирает первую строку текста элемента, применяется только к блочным элементам.
3. :before — вставляет генерируемое содержимое перед элементом.
4. :after — добавляет генерируемое содержимое после элемента.

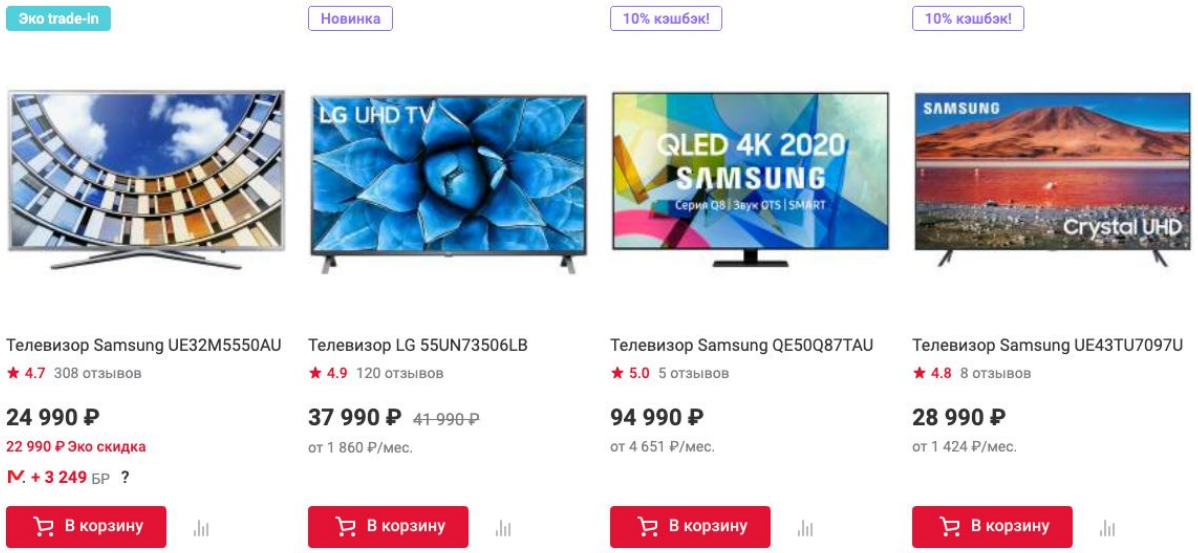
Пример использования псевдоэлементов

HTML	CSS

`<ul>`
`  <li>Первый элемент</li>`
`  <li>Второй элемент</li>`
`  <li>Третий элемент</li>`
`</ul>`
	`li:after{`
	`  content: "new";`
	`  color: #F00;`
	`}`

После всех элементов списка li появится текст new красного цвета.

Пример использования псевдоэлементов на реальном сайте



На картинке шильдики «новинка», «10% кэшбэк!» и т. д. Эту часть проще всего сделать с помощью псевдоэлемента. Он не будет находиться в структуре html. Добавление подобного элемента состоит только из применения класса к любому блоку. Их можно легко поменять один на другой, не затрагивая html-файл.

Для чего нужны таблицы в HTML

Основное назначение таблиц в HTML — представление табличных данных:

- просмотр информации о пользователях;
- просмотр заказанных товаров в интернет-магазине;
- просмотр отчётов о продажах и др.

Второе назначение таблиц в HTML — верстать веб-страницы. А верстать – это разбивать страницу на элементы, т. е. формировать различные блоки сайта:шапка, меню, контент, подвал и другие. Табличный способ вёрстки в настоящее время считается неправильным и устаревшим, потому что таблицы используются не по назначению.

Структура таблицы в HTML

```
<table>
      <tr>
            <td>Столбец 1</td>
            <td>Столбец 2</td>
            <td>Столбец 3</td>
```

```
        </tr>
</table>
```

Любая таблица в HTML помещается в контейнер <table>. После чего в контейнер строки <tr> вкладываются уже столбцы таблицы. Они включаются в тег <td>. В этом примере таблица состоит из трёх столбцов и одной строки. То есть в таблице будет 3 ячейки. Чтобы добавить строку в таблицу, нужно вложить в контейнер <table> ещё один контейнер строки <tr>, затем поместить в него то же количество столбцов <td> (3 столбца).

```
<table>
      <tr>
            <td>Столбец 1</td>
            <td>Столбец 2</td>
            <td>Столбец 3</td>
      </tr>
      <tr>
            <td>Столбец 1</td>
            <td>Столбец 2</td>
            <td>Столбец 3</td>
      </tr>
</table>
```

Создание ячеек таблицы

Элемент <td> создаёт ячейки таблицы с данными, добавляя их в строку таблицы. Парные теги <td>…</td>, расположенные между тегами соответствующей строки, определяют количество ячеек в пределах одной строки. Количество пар ячеек таблицы равно количеству пар ячеек заголовка.

Элемент <th> создаёт заголовок — специальную ячейку, текст в которой выделяется полужирным шрифтом. Количество ячеек заголовка определяется количеством пар тегов <th>…</th>.

Объединение ячеек таблицы

Чтобы в полной мере начать использовать таблицы в HTML, нужно научиться объединять ячейки. Для этого у тега <td> или <th> есть атрибуты colspan и rowspan.

- colspan объединяет ячейки по горизонтали;
- rowspan объединяет ячейки по вертикали.

rowspan — объединение по вертикали (строк)	**colspan** — объединение по горизонтали (столбцов)

При rowspan в первом столбце объединяются 2 строки. А при атрибуте colspan в первой строке объединяются уже два столбца.

Посмотрим, как это выглядит в HTML.

```
<table>
    <tr>
        <td rowspan="2">Столбец 1</td>
        <td colspan="2">Столбец 2</td>
    </tr>
    <tr>
        <td>Столбец 5</td>
        <td>Столбец 6</td>
    </tr>
    <tr>
        <td>Столбец 7</td>
        <td>Столбец 8</td>
        <td>Столбец 9</td>
    </tr>
</table>
```

В первой строке таблицы планируются два столбца. Первая строка первого столбца объединяется со второй строкой, поэтому у первой ячейки указываем атрибут rowspan со значением 2. Это означает, что нужно объединить две ячейки.

В этой же строке во втором и третьем столбце объединяем две ячейки по горизонтали при помощи атрибута colspan, также со значением 2. Далее во второй строке учитываем, что указывать для неё первый столбец не надо. Он уже объединился с первой строкой. Поэтому во второй строке указываются два столбца, начиная со второго.

В третьей строке присутствуют все три столбца по порядку.

Стилевое оформление таблиц

Для стилизации таблиц используются те же стили, что и ранее.

По умолчанию таблица и её ячейки не имеют видимых границ и фона. При этом ячейки внутри таблицы не прилегают вплотную друг к другу. Ширина ячеек таблицы определяется шириной их содержимого, поэтому может быть разной. Высота всех ячеек сетки строки по умолчанию одинаковая.

Граница таблице задаётся аналогично любому другому элементу. Это рассматривали на прошлом занятии. Единственное, если указать свойство border только тегу <table>, граница будет только у внешних границ таблицы. Это происходит потому, что свойство border не наследуется. Чтобы указать границы для ячеек, нужно задать их тегу <td>. Здесь уже можно задавать границы ячейкам любого стиля.

```
/*Внешняя граница таблицы*/
table {
      border: 1px solid #000;
}
/*Границы для ячеек таблицы*/
td {
      border: 1px solid #000;
}
```

Ширину и высоту таблицы задаём при помощи свойств CSS width и height.

```
table {
      width: 400px;
      height: 200px;
}
```

По умолчанию ширина и высота таблицы определяются содержимым её ячеек. Если не задать ширину, то она будет равна ширине самого широкого ряда.

Ширина таблицы и её столбцов указывается свойством width. Если таблице задана ширина table {width: 100%;}, то она будет равна ширине блока-контейнера. При этом ширина столбцов установится в соответствии с шириной содержимого ячеек. Чаще всего ширину таблицы и столбцов задают в px или %.

Есть ещё одно полезное свойство для оформления таблиц. Если для всех ячеек задать атрибут cellspacing=”0” или CSS свойство border-spacing=”0”, то границы соседних ячеек будут двойными. Чтобы убрать двойные границы, применяется свойство border-collapse.

```
table {
       border-collapse: collapse;
}
```

Свойство border-collapse принимает два значения. По умолчанию устанавливается separate — рамка двойная. При значении collapse рамка становится одинарной.

Дополнительные материалы

1. Популярно о псевдоэлементах :Before и :After.
2. Псевдоклассы и псевдоэлементы.
3. Дополнительный материал о таблицах.
4. div & span.

Используемые источники

1. Псевдоэлементы.
2. Псевдоклассы.
3. Основы css.
4. Псевдоклассы.
5. Таблицы в html.
6. Стилизация таблиц.

Практическое задание

1. Домашнее задание посвящается разбиению сайта на блоки.
2. кПредоставляется изображение сайта (просто картинка). На нём разными цветами выделены блоки, на которые требуется разбить страницу. Верстать по макету не нужно, просто добавить элементы div с class, которые выделены на изображении.
3. Всё содержимое 3 дз вам необходимо добавить в блоки, отмеченные на изображении сайта.
4. Никаких новых элементов из макета добавлять не нужно. Вы начнёте работать с ним на 6 уроке.
5. Всем блокам задать логичное название класса. Запрещается использовать перечисление (block1, block2, block3...block378).
6. * Для страницы «Контакты» также представлено изображение страницы, но без разбиения на блоки. Здесь требуется самостоятельно разделить сайт на блоки.

Задания со * — дополнительные, его выполнять необязательно.

Рассмотрим пример верхней части сайта

```html
<div class="top"> <!-- Бордовый прямоугольник -->
  <div class="top__content"> <!-- Фиолетовый блок -->
    <div class="menu"> <!-- Зеленый блок -->
      <a href="#">Главная</a>
      <a href="#">Контакты</a>
    </div>
    <div class="top__info"> <!-- Голубой блок -->
      <h1>Lorem ipsum dolor.</h1>
      <p>Lorem ipsum dolor sit amet</p>
    </div>
  </div>
</div>
```