**1. HTML теги. Блочные и строчные теги**

Поведение любого тега сегодня (в стандарте **HTML5**) легко изменить по своему усмотрению. Деление элементов на блочные и строчные сложилось исторически (именно поэтому они изначально относятся к одному из этих типов).

Сейчас таких типов больше, но их определение связано с каскадными таблицами стилей, о которых пойдет детальная речь в одном из последующих уроков.

**Блочные теги** занимают всю ширину страницы или родительского элемента. Если вы зададите два таких тега подряд, то каждый из них будет начинаться как бы с новой строки, абзаца.

**Строчные теги** не имеют строго размера. Он зависит от того, сколько символов в них содержится. Несколько подряд идущих строчных элементов не будут разделяться новыми строками, а будут выстраиваться друг за другом.

**Полезно знать**

В **CSS-свойствах** отображение тегов можно менять по следующим типам:

1. **Блочные** (**display: block**) – получает свойство блочности, занимает всю ширину страницы;
2. **Строчные** (**display: inline**) – тег становится строчным;
3. **Флекс** (**display: flex**) – элемент приобретает свойство особого типа блоков – флекс (ведет себя как блок, но внутреннее содержимое гибко настраивается);
4. **Грид** (**display: grid**) – еще один блочный тип с особенностями внутреннего поведения элементов (строятся на основании «сетки», состоящей из строк и колонок);
5. **Таблица** (**display: table**) – тег и его содержимое будут наследовать свойства таблиц;
6. **Строчный блок** (**display: inline-block**) – блочный элемент ведет себя как строка, но сохраняет часть свойств (можно задать размер, границы).
7. **Спрятанный** (**display: none**) – делает элемент невидимым и полностью удаляет его отображение со страницы.

Существуют и другие свойства отображения, многие из которых узконаправлены и зачастую экспериментальны.

На примере двух самых ярких представителей (1 блочный и 1 строчный соответственно) посмотрим на их поведение.

**1. Блочный тэг <div>**

Самый популярный блочный тег времен **HTML 4**. Основная его задача – отобразить содержимое как блок. Он и сегодня входит в список часто используемых. Достаточно открыть код любой страницы в сети и проанализировать его. Почти везде **<div>** применяется максимально широко. Посмотрим на простой пример.

Пример - HTML

|  |
| --- |
| <div>Этот текст занимает всю ширину страницы</div>  <div>Этот текст начнется с новой строки</div> |

Если сохранить приведенный код в виде **html-файла** и открыть в браузере, то увидим, что куски текста начинаются с разных строк. Причина – в сути блочных элементов (они стремятся занять всю доступную ширину, которая им предоставлена).

Другие блочные теги:

1. **<table></table>** Позволяет рисовать таблицы
2. **<p>Параграф</p>** Обозначает параграф текста
3. **<h1>Большой заголовок</h1>** Для оформления заголовка самого верхнего уровня (в роли такового обычно выбирается основная статья страницы).
4. **<form></form>** Отражает на сайте форму (авторизации, регистрации, опроса и др.)

**2. Строчный тэг <span>**

Часто используемый строчный элемент. Фактически, он **ничего не делает с текстом**, но при помощи атрибутов его содержимое можно определенным образом выделять. Важно и то, что информация не будет прерываться, переноситься на новую строку, выделяться абзацем.

Пример - HTML

|  |
| --- |
| <span>Обычный текст без особенностей. </span>  <span>Этот текст не будет начинаться с новой строки</span> |

Вторая часть текста идет вслед за первой без каких-либо разрывов. Строчные элементы занимают столько места, сколько необходимо для их отображения.

На практике нередко используются следующие строчные теги:

1. **<b>Выделение жирным шрифтом</b>** Выделяет участок содержимого жирным шрифтом без придания особой важности. Тег применяют в целях визуализации, чтобы информация воспринималась лучше
2. **<strong>Важный текст</strong>** Обозначение значимого участка текста. Чаще всего выглядит как выделенный жирным шрифтом, но воспринимается поисковыми системами как имеющий особую важность в приведенном контексте
3. **<a href="#">Ссылка в никуда</a>** Ссылка занимает тот объем на странице, который требуется для демонстрации ее текста
4. **<sub>, <sup>** Необходимы для представления степеней и индексов. Рассмотрим пример:

Пример - HTML

|  |
| --- |
| <span>sin<sup>2</sup>x + cos<sup>2</sup>x = 1</span>  <br>  <span>CO<sub>2</sub></span> |

В первом случае мы получаем тригонометрическое тождество, а во втором – формулу углекислого газа.

В **HTML5** прямого разделения тегов на блочные и строчные фактически нет. Сейчас говорят о **категориях контента**, который содержится внутри элемента.

**Полезно знать**

Контентная модель **HTML5** позволяет более точно описать типы содержимого, с которым сталкиваются авторы при разработке сайтов. Она включает такие типы:

1. **Поток** (**flow**) – отображает основное содержимое страницы (большая часть тегов включается именно сюда. Например: **a**, **button,** **header**, **nav**, **ol**, **section**);
2. **Метаданные** (**metadata**) – применяются в заголовке документа и определяют поведение остального содержимого и связь с иными данными (Пример тегов: **title**, **meta**, **script**, **style**);
3. **Текст** (**phrasing**) – включает непосредственно текст страницы и используемые для его форматирования теги. Фактически, это строчные элементы: **a**, **button**, **i**, **img**, **span**, **textarea**;
4. **Встроенный контент** (**embedded**) – часть текстового, строчного содержимого, включающая импортированный контент (Пример тегов: **audio**, **video**, **img**, **svg**, **canvas**);
5. **Интерактивные элементы** (**interactive**) – позволяют посетителю сайта взаимодействовать с ресурсом (Включает теги: **button**, **select**, **video**, **a**, **input**, **textarea**);
6. **Заголовки** (**heading**) – для определения заголовков сайта, статей, подразделов (теги от **h1** до **h6**);
7. **Секции** (**sectioning**) – выделение изолированного контента в блоки. Включает теги: **article**, **aside**, **nav**, **section**.

Несмотря на это, все элементы с начальными свойствами являются или строчными или блочными по своей сути.

**2. Семантические теги**

Появление **HTML5** позволило подойти к вопросу размещения контента на сайтах по-новому. До этого решался лишь технический вопрос: как сделать сайт, который будет нормально отображаться у разных пользователей с разными устройствами и системами. Так как данная проблема сегодня не столь актуальна (разработчики браузеров более-менее договорились и пришли к единому стандарту), возникла другая: **огромное количество сайтов затрудняет демонстрацию пользователям максимально релевантных**.

Немаловажной стала задача обеспечения доступности ресурсов посетителям с ограниченными возможностями (например, слепым).

Обычного набора тегов стало не хватать. Потребовался **семантический набор**. Обычный пользователь особой разницы не увидит: страницы выглядят так же, имеют ту же скорость загрузки. Однако решена важная проблема: поисковые системы стали лучше понимать интернет-ресурсы, а люди с ограниченными возможностями теперь могут удобнее пользоваться сайтами (так, специализированный софт для таких граждан понимает семантику тегов: заголовки будут озвучены как заголовки, побочную информацию из того же тега **<aside>** можно будет пропустить и т.п.).

Семантические теги помогают браузерам, поисковикам, разработчикам. Теперь с проблемой однозначности трактования практически покончено.

Рассмотрим простой пример.

|  |
| --- |
| <div>Чтобы создавать сайты необходимо знать HTML, CSS, JavaScript и серверную технологию (например, Node.js, Python или php)</div>  <div>Первый сайт, к слову, создал Тим Бернес-Ли</div> |

Анализ содержимого такой страницы не сложен. Многие, скорее всего, поймут, что первый блок текста затрагивает основное содержимое раздела, тогда как второй – дополнение, интересный факт, побочная, но не обязательная информация. А если речь идет о более комплексной ситуации? **Как выяснить, какую смысловую нагрузку имеет тот или иной** **<div>?** Фактически, никак.

Если человек еще хоть как-то способен логически сопоставить данные, то машине это не под силу. Да и вместо того, чтобы оценивать код, его качество и эффективность, программисту потребуется тратить время на понимание смысловой нагрузки используемых элементов. Это очень неудобно.

**Семантические элементы позволяют упростить такую ситуацию.** Разработчик помогает не только своим коллегам, но и браузерам и поисковикам. Да и сам сайт существенно быстрее попадет к целевой аудитории (ведь какой-нибудь Яндекс будет показывать вашу страничку не тем, кому интересна биография основателя **W3-Консорциума**, а тем, кому хочется научиться создавать сайты. Именно в этом заключается суть обозначенного выше **HTML-кода**).

К основным тегам такого типа относят:

1. **<article>Будущая статья</article>**

Для выделения независимого, самоопределяемого контента. Тег подразумевает содержимое, которое несет ценность само по себе, обосабливается в общем контексте сайта. Хорошим примером такого вида служит пост на форуме или в блоге, отдельная статья сайта, новость;

2. **<aside>Побочная информация</aside>**

Некие дополнительные данные, косвенно связанные с основным содержанием страницы. Например, вы пишете про списки в Python. В блоке aside можно рассказать про списки и их свойства в других языках программирования;

3. **<details>Дополнительные сведения</details>**

Справочная, дополняющая информация, которую пользователь по своему желанию может не смотреть. Это раскрывающийся блок с надписью **Подробнее** (по умолчанию). Сюда могут включать сведения об авторе, некие уточнения;

4. **<summary></summary>**

Дает краткую характеристику содержимого блока **details**. Посмотрим на примере совместную работу элементов.

|  |
| --- |
| <details>  <summary>W3-Консорциум</summary>  Возглавляется Тимом Бернесом-Ли, разработчиком языка разметки HTML.  </details> |

Вместо кнопки **Подробнее** пользователь увидит надпись **W3-Консорциум**, нажав на которую получит дополнительные сведения;

5. **<figcaption>Описание</figcaption>**

Комментарий к картинке, пояснения, название;

6. **<figure></figure>**

Выделяет рисунок и его описание в единый блок. Посмотрим пример:

|  |
| --- |
| <figure>  <img src="cat.jpg">  <figcaption>Мой кот сегодня доволен</figcaption>  </figure> |

В тег **<figure>** мы поместили картинку кота и дополнили пояснениями;

7. **<footer>Заключение</footer>**

Позволяет выделить нижний колонтитул статьи или раздела сайта. Здесь часто подводят итоги, пишут информацию об авторе, резюмируют написанное;

8. **<header>Введение</header>**

Служит для вступительного содержания страницы сайта или статьи. Тут указывают заголовок, подзаголовок, некие вводные данные;

9. **<main>Основное содержимое</main>**

Основной контент документа. Если речь о статье, то именно тут находится ее тело за исключением вводной и заключительной частей;

10. **<nav>Разделы сайта</nav>**

Основное меню сайта заключают в тег nav. Содержит ссылки на разделы сервиса;

11. **<section>Некий блок сайта</section>**

Отдельный раздел документа, не обязательно независимый. Представляет некую тему с содержимым, которые требуется выделить в изолированный блок;

12. **<time></time>**

Позволяет браузеру и поисковым системам понять, что заключенные в тег данные представляют собой дату. Например:

Мы работаем до <time>22:00</time> ежедневно

Визуально этот элемент себя никак не проявляет, тем не менее подразумевает семантический контекст;

11. **<mark>Выделенные данные</mark>**

Содержит информацию, требующую выделения или подчеркивания.

Структура web-страницы, построенной на семантических тегах

**Подытожим**Представленные семантические теги являются новыми, введенными в **HTML5**. Важно понимать, что и многие старые элементы относятся по сути к семантическим (**<h1>**, **<strong>**, **<table>**, **<form>** и др.).

Никто не запрещает вам создавать сайт без использования семантических элементов, но его эффективность и «рейтинг» будут существенно ниже тех, которые применяют новые возможности.

**3. Теги логического форматирования**

Теги логического форматирования предназначены в основном для отметки логических частей текстовой информации и явно не несут стилевого оформления. Но стилевое оформление для них (как и для других тегов) можно закладывать с помощью CSS-каскадных таблиц стилей.

|  |  |
| --- | --- |
| **html-тег** | **Пояснение** |
| <CITE>. . .</CITE> | от Citation – цитата. Служит для обозначения названия  источника информации или автора |
| <Q> … </Q> | обозначение цитаты из источника, берет текст в кавычки. У тега q есть атрибут с названием cite, который дает возможность обозначать ссылку на источник цитаты |
| **<p>**  Как сказал **<cite>**Антуан де Сент-Экзюпери**</cite>** в своей книге  **<cite>**Маленький принц**</cite>,**  **<q>**Таким был прежде мой Лис. Он ничем не отличался от ста тысяч других лисиц. Но я с ним подружился, и теперь он — единственный в целом свете**.</q>**  **</p>** | |
| <CODE>. . .</CODE> | вывод кода программы шрифтом фиксированной ширины |
| <EM>. . .</EM> | акцентирование текста от Emphasis – особое внимание, браузер выделяет такой текст курсивом |
| <KBD>. . .</KBD> | от Keyboard – клавиатура; вывод текста шрифтом фиксированной ширины, введенного с клавиатуры |
| <SAMP>. . .</SAMP> | от Sample – пример, образец; вывод нескольких символов шрифтом фиксированной ширины |
| <SMALL>…</SMALL> | мелкий текст, обозначающий отступления, предупреждения, ограничения или авторские права |
| <VAR>. . .</VAR> | от Variable – переменная |
| <TT>. . .</TT> | от Teletype – телетайп; текст, выведенный устройством вывода компьютера |
| <DFN>. . .</DFN> | от Definition – формулировка; термин, встречающийся в тексте первый раз |
| **<P>** **<DFN>** **<ABBR** TITLE="HyperText Markup Language">HTML**</ABBR>**  **</DFN>** это основной язык разметки веб-контента. **</P>** | |
| <ACRONYM TITLE= "Высшее учебное заведение"> ВУЗ </ACRONYM> | акроним; вид аббревиатуры, образованной из начальных букв слов или словосочетаний (например, НИИ, АЭС), произносимой как единое слово, а не по буквам |
| <ABBR> …</ABBR> | аббревиатура |

.