## Структура HTML-кода

Если открыть любую веб-страницу, то она будет содержать в себе типичные элементы, которые остаются исходными, независимо от вида и направления сайта. Рассмотрим код, который первоначально создается в редакторе. Для удобства добавлены номера строк, но в коде их, конечно же, быть не должно, иначе это приведет к ошибке.

Пример 4.1. Исходный код веб-страницы

01. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

02. <html>

03. <head>

04. <meta content="text/html; charset=Windows-1251" http-equiv="content-type">

05. <title>Вторая страница</title>

06. </head>

07. <body>

08. <br>

09. </body>

10. </html>

Далее разберем отдельные строки нашего кода.

01. !DOCTYPE предназначен для указания типа текущего документа — DTD (document type definition, описание типа документа). Существует несколько видов <!DOCTYPE>, они различаются в зависимости от версии HTML, на которую ориентированы (пример 4.2).

Пример 4.2. Допустимые DTD

HTML 3.2 (предыдущая версия HTML)

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">

Строгий HTML 4.01

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">

Переходный HTML 4.01

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

HTML 4.01 при использовании фреймов

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

Разница между строгим и переходным описанием документа состоит в различном подходе к написанию кода документа. Строгий HTML требует жесткого соблюдения спецификации HTML и не прощает ошибок, что обычно приводит к иному отображению документа, чем планировал разработчик. Переходный HTML более «спокойно» относится к некоторым огрехам кода.

Замечание

Часто можно встретить код HTML вообще без использования <!DOCTYPE>, веб-страница в подобном случае все равно будет показана. Тем не менее, может получиться, что один и тот же документ отображается по-разному при использовании <!DOCTYPE> и без него. Чтобы не произошло подобных ситуаций, всегда добавляйте этот тег в начало документа.

02. Тег <html> определяет начало HTML-файла, внутри него хранится заголовок (<head>) и тело документа (<body>).

03. Заголовок документа, как еще называют блок <head>, может содержать текст и теги, но содержимое этого раздела не показывается напрямую на странице.

04. Тег <meta> является универсальным и добавляет целый класс возможностей, в частности, с помощью метатегов, как обобщенно называют этот тег с разными параметрами, можно изменять кодировку страницы, добавлять ключевые слова, описание документа и многое другое. Так, чтобы браузер понимал, что имеет дело с кодировкой «Кириллица (Windows)» и добавляется строка, как в примере 4.1.

05. Тег <title> определяет заголовок веб-страницы, это один из важных элементов предназначенный для решения множества задач. В операционной системе Windows текст заголовка отображается в левом верхнем углу окна браузера (рис. 4.1).

06. Обязательно следует добавлять закрывающий тег </head>, чтобы показать, что блок заголовка документа завершен.

07. Тело документа <body> предназначено для размещения тегов и содержательной части.

08. Внутри контейнера <body> могут размещаться практически любые элементы веб-страницы, в частности, тег <br>, как показано в примере 4.1, создает перенос строки.

09. Следует добавить закрывающий тег </body>, чтобы показать, что тело документа завершено.

10. Последним элементом в коде идет закрывающий тег </html>.

## Основная разметка страницы (layout). DIV vs. TABLE

Табличная или блочная верстка? Как же правильнее? Это неправильный вопрос!

Презентационная или семантическая верстка? Конечно семантическая!

### Недостатки табличной верстки

У табличной верстки есть много недостатков. Рассмотрим их по порядку:

1. **Web-стандарты.** Используя табличную верстку невозможно сверстать более или менее серьезный макет, который бы не содержал синтаксических ошибок в коде.
2. **Семантика.** Таблицы были введены в HTML исключительно для облегчения представления табличной информации (прайс лист и т. д.), но поскольку в спецификации HTML 3.2 не было каких-либо других способов позиционирования элементов на странице, тогдашние web-мастера использовали таблицы для разметки страницы на блоки (меню, шапка, подвал и т. д.).
3. **Нелогичность поведения таблиц.** Если вы верстали или верстаете таблицами, то наверняка сталкивались с нелогичностью поведения таблиц в сложных макетах и разных браузерах. Это связано с тем, что таблицы не предназначены для верстки и с тем, что большинство браузеров отрисовывают страницы с табличной версткой в кварк режиме, а этот режим у всех браузеров разный.
4. **Читабельность кода и редизайн.** Вы когда-нибудь пробовали разобраться в коде чужого сайта, сверстанного таблицами — наверное это было нелегко. А внести какие-либо серьезные изменения в дизайн иногда вообще невозможно, легче сверстать все заново.
5. **Размер кода.** В подавляющем большинстве случаев верстка таблицами значительно увеличивает код страницы по сравнению с блочной версткой.

### Преимущества блочной верстки

1. **Соответствие стандартам.** Правильная блочная верстка полностью соответствует стандартам HTML и CSS. Это обеспечивает легкое понимание вашего кода теми, кто знаком со стандартами и гарантируемую поддержку в новых версиях браузеров.
2. **Отделение содержания от представления.** Это обеспечивает легко читабельный код и смену дизайна всего сайта простым изменением файла стилей.
3. **Правильное отображение страниц разными устройствами.** Возможность создания разных файлов стилей для разных устройств (КПК, принтер и т. д.) без изменения HTML кода.
4. **Легкость и логичность верстки.** Вы просто берете нужный блок и, используя стили, определяете его размеры, вид и место на странице.

## Основные тэги и их CSS свойства

**<тег атрибут1="значение" атрибут2="значение">**

**<тег атрибут1="значение" атрибут2="значение">**...**</тег>**

### Блочные элементы

http://htmlbook.ru/samhtml/tipy-tegov/blochnye-elementy

#### <div>

Тег <div> относится к универсальным блочным контейнерам и применяется в тех случаях, где нужны блочные элементы без дополнительных свойств. Также с помощью тега <div> можно выравнивать текст внутри этого контейнера с помощью атрибута align.

#### <h1>,...,<h6>

Эта группа тегов определяет текстовые заголовки разного уровня, которые показывают относительную важность секции, расположенной после заголовка.

#### <hr>

Рисует горизонтальную линию, которая по своему виду зависит от используемых атрибутов. Линия всегда начинается с новой строки, а после нее все элементы отображаются на следующей строке.

#### <p>

Определяет параграф (абзац) текста.

#### <pre>

Задает блок предварительно форматированного текста. Такой текст отображается обычно моноширинным шрифтом и со всеми пробелами между словами. В HTML любое количество пробелов идущих в коде подряд на веб-странице показывается как один. Тег <pre> позволяет обойти эту особенность и отображать текст как требуется разработчику.

### Строчные элементы

http://htmlbook.ru/samhtml/tipy-tegov/strochnye-elementy

Строчными называются такие элементы веб-страницы, которые являются непосредственной частью другого элемента, например, текстового абзаца. В основном они используются для изменения вида текста или его логического выделения.

#### <a>

Тег <a> является одним из важных элементов HTML и предназначен для создания ссылок. В зависимости от присутствия атрибутов name или href тег <a> устанавливает ссылку или якорь.

Синтаксис

<a href="URL">...</a>

<a name="идентификатор">...</a>

Атрибуты

[accesskey](http://htmlbook.ru/html/a/accesskey)

Активация ссылки с помощью комбинации клавиш.

[charset](http://htmlbook.ru/html/a/charset)

Указывает кодировку текста, на который ведет ссылка.

[coords](http://htmlbook.ru/html/a/coords)

Устанавливает координаты активной области.

[href](http://htmlbook.ru/html/a/href)

Задает адрес документа, на который следует перейти.

[hreflang](http://htmlbook.ru/html/a/hreflang)

Идентифицирует язык текста по ссылке.

[media](http://htmlbook.ru/html/a/media)

Указывает тип носителя документа, на который ведёт ссылка.

[name](http://htmlbook.ru/html/a/name)

Устанавливает имя якоря внутри документа.

[rel](http://htmlbook.ru/html/a/rel)

Отношения между ссылаемым и текущим документами.

[rev](http://htmlbook.ru/html/a/rev)

Отношения между текущим и ссылаемым документами.

[shape](http://htmlbook.ru/html/a/shape)

Задает форму активной области ссылки для изображений.

[tabindex](http://htmlbook.ru/html/a/tabindex)

Определяет последовательность перехода между ссылками при нажатии на кнопку <Tab>.

[target](http://htmlbook.ru/html/a/target)

Имя окна или фрейма, куда браузер будет загружать документ.

[title](http://htmlbook.ru/html/a/title)

Добавляет всплывающую подсказку к тексту ссылки.

[type](http://htmlbook.ru/html/a/type)

Указывает MIME-тип документа, на который ведёт ссылка.

#### <b>

Определяет жирное начертание шрифта.

#### <big>

Тег <big> увеличивает размер шрифта на единицу по сравнению с обычным текстом. В HTML размер шрифта измеряется в условных единицах от 1 до 7, средний размер текста, используемый по умолчанию, принят 3. Таким образом, добавление тега <big> увеличивает текст на одну условную единицу.

#### <br>

Тег <br> устанавливает перевод строки в том месте, где этот тег находится. В отличие от тега параграфа <p>, использование тега <br> не добавляет пустой отступ перед строкой.

#### <em>

Тег <em> предназначен для акцентирования текста. Браузеры отображают такой текст курсивным начертанием.

#### <i>

Устанавливает курсивное начертание шрифта.

#### <img>

Тег <img> предназначен для отображения на веб-странице изображений в графическом формате GIF, JPEG или PNG. Если необходимо, то рисунок можно сделать ссылкой на другой файл, поместив тег <img> в контейнер <a>. При этом вокруг изображения отображается рамка, которую можно убрать, добавив атрибут border="0" в тег <img>.

**Синтаксис**

|  |  |
| --- | --- |
| HTML | <img src="URL" alt="альтернативный текст"> |
| XHTML | <img src="URL" alt="альтернативный текст" /> |

**Атрибуты**

[align](http://htmlbook.ru/html/img/align)

Определяет как рисунок будет выравниваться по краю и способ обтекания текстом.

[alt](http://htmlbook.ru/html/img/alt)

Альтернативный текст для изображения.

[border](http://htmlbook.ru/html/img/border)

Толщина рамки вокруг изображения.

[height](http://htmlbook.ru/html/img/height)

Высота изображения.

[hspace](http://htmlbook.ru/html/img/hspace)

Горизонтальный отступ от изображения до окружающего контента.

[ismap](http://htmlbook.ru/html/img/ismap)

Говорит браузеру, что картинка является серверной картой-изображением.

[longdesc](http://htmlbook.ru/html/img/longdesc)

Указывает адрес документа, где содержится аннотация к картинке.

[src](http://htmlbook.ru/html/img/src)

Путь к графическому файлу.

[vspace](http://htmlbook.ru/html/img/vspace)

Вертикальный отступ от изображения до окружающего контента.

[width](http://htmlbook.ru/html/img/width)

Ширина изображения.

[usemap](http://htmlbook.ru/html/img/usemap)

Ссылка на тег <map>, содержащий координаты для клиентской карты-изображения.

#### <small>

Тег <small> уменьшает размер шрифта на единицу по сравнению с обычным текстом. По своему действию похож на тег <big>, но действует с точностью до наоборот.

#### <span>

Универсальный тег, предназначенный для определения строчного элемента внутри документа.

#### <strong>

Тег <strong> предназначен для акцентирования текста. Браузеры отображают такой текст жирным начертанием.

#### <sub>

Отображает шрифт в виде нижнего индекса. Текст при этом располагается ниже базовой линии остальных символов строки и уменьшенного размера — H2O.

#### <sup>

Отображает шрифт в виде верхнего индекса. По своему действию похож на <sub>, но текст отображается выше базовой линии текста — м2.

Разница между блочными и строчными элементами следующая.

* Строчные элементы могут содержать только данные или другие строчные элементы, а в блочные допустимо вкладывать другие блочные элементы, строчные элементы, а также данные. Иными словами, строчные элементы никак не могут хранить блочные элементы.
* Блочные элементы всегда начинаются с новой строки, а строчные таким способом не акцентируются.
* Блочные элементы занимают всю доступную ширину, например, окна браузера, а ширина строчных элементов равна их содержимому плюс значения отступов, полей и границ

### Тэги для списков

Списком называется взаимосвязанный набор отдельных фраз или предложений, которые начинаются с маркера или цифры. Списки предоставляют возможность упорядочить и систематизировать разные данные и представить их в наглядном и удобном для пользователя виде.

#### <ol>

Тег <ol> устанавливает нумерованный список, т.е. каждый элемент списка начинается с числа или буквы и увеличивается по нарастающей.

**Синтаксис**

<ol>

<li>элемент нумерованного списка</li>

<li>элемент нумерованного списка</li>

</ol>

**Атрибуты**

[type](http://htmlbook.ru/html/ol/type)

Устанавливает вид маркера списка.

[reversed](http://htmlbook.ru/html/ol/reversed)

Нумерация в списке становится по убыванию (3,2,1).

[start](http://htmlbook.ru/html/ol/start)

Задаёт число, с которого будет начинаться нумерованный список.

#### <ul>

Устанавливает маркированный список, каждый элемент которого начинается с небольшого символа — маркера.

**Синтаксис**

<ul>

<li>элемент маркированного списка</li>

</ul>

**Атрибуты**

[type](http://htmlbook.ru/html/ul/type)

Устанавливает вид маркера списка.

#### <li>

Тег <li> определяет отдельный элемент списка. Внешний тег <ul> или <ol> устанавливает тип списка — маркированный или нумерованный.

**Синтаксис**

<ul>

<li>элемент маркированного списка</li>

</ul>

<ol>

<li>элемент нумерованного списка</li>

</ol>

**Атрибуты**

[type](http://htmlbook.ru/html/li/type)

Устанавливает вид маркера нумерованного или маркированного списка.

[value](http://htmlbook.ru/html/li/value)

Число, с которого будет начинаться нумерованный список.

#### <dd>, <dt>, <dl>

Тройка элементов предназначена для создания списка определений. Каждый такой список начинается с контейнера <dl>, куда входит тег <dt> создающий термин и тег <dd> задающий определение этого термина. Закрывающий тег </dd> не обязателен, поскольку следующий тег сообщает о завершении предыдущего элемента. Тем не менее, хорошим стилем является закрывать все теги.

**Синтаксис**

<dl>

<dt>Термин 1</dt>

<dd>Определение термина 1</dd>

<dt>Термин 2</dt>

<dd>Определение термина 2</dd>

</dl>

## Основные стили

* font-family
* font-size
* color
* text-align
* width
* height
* display
* position
* margin
* padding

### Стиль position

**Синтаксис**

position: absolute | fixed | relative | static | inherit

**Значения**

absolute

Указывает, что элемент абсолютно позиционирован, при этом другие элементы отображаются на веб-странице словно абсолютно позиционированного элемента и нет. Положение элемента задается свойствами left, top, right и bottom, также на положение влияет значение свойства position родительского элемента. Так, если у родителя значение position установлено как static или родителя нет, то отсчет координат ведется от края окна браузера. Если у родителя значение position задано как fixed, relative или absolute, то отсчет координат ведется от края родительского элемента.

fixed

По своему действию это значение близко к absolute, но в отличие от него привязывается к указанной свойствами left, top, right и bottom точке на экране и не меняет своего положения при прокрутке веб-страницы. Браузер Firefox вообще не отображает полосы прокрутки, если положение элемента задано фиксированным, и оно не помещается целиком в окно браузера. В браузере Opera хотя и показываются полосы прокрутки, но они никак не влияют на позицию элемента.

relative

Положение элемента устанавливается относительно его исходного места. Добавление свойств left, top, right и bottom изменяет позицию элемента и сдвигает его в ту или иную сторону от первоначального расположения.

static

Элементы отображаются как обычно. Использование свойств left, top, right и bottom не приводит к каким-либо результатам.

inherit

Наследует значение родителя.

### Стиль margin

**Описание**

Устанавливает величину отступа от каждого края элемента. Отступом является пространство от границы текущего элемента до внутренней границы его родительского элемента (рис. 1).

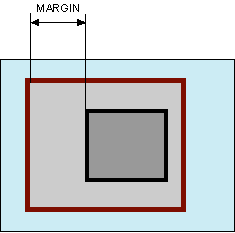


Рис. 1. Отступ от левого края элемента

Если у элемента нет родителя, отступом будет расстояние от края элемента до края окна браузера с учетом того, что у самого окна по умолчанию тоже установлены отступы. Чтобы от них избавиться, следует устанавливать значение margin для селектора <body> равное нулю.

Свойство margin позволяет задать величину отступа сразу для всех сторон элемента или определить ее только для указанных сторон.

**Синтаксис**

margin: [значение | проценты | auto] {1,4} | inherit

**Значения**

Разрешается использовать одно, два, три или четыре значения, разделяя их между собой пробелом. Эффект зависит от количества значений и приведен в табл. 1.

|  |  |
| --- | --- |
| Табл. 1. Зависимость от числа значений | |
| **Число значений** | **Результат** |
| 1 | Отступы будут установлены одновременно от каждого края элемента. |
| 2 | Первое значение устанавливает отступ от верхнего и нижнего края, второе — от левого и правого. |
| 3 | Первое значение задает отступ от верхнего края, второе — одновременно от левого и правого края, а третье — от нижнего края. |
| 4 | Поочередно устанавливается отступ от верхнего, правого, нижнего и левого края. |

Величину отступов можно указывать в пикселах (px), процентах (%) или других допустимых для CSS единицах. Значение может быть как положительным, так и отрицательным числом.

auto

Указывает, что размер отступов будет автоматически рассчитан браузером.

inherit

Наследует значение родителя.

### Сталь padding

**Описание**

Устанавливает значение полей вокруг содержимого элемента. Полем называется расстояние от внутреннего края рамки элемента до воображаемого прямоугольника, ограничивающего его содержимое (рис. 1).

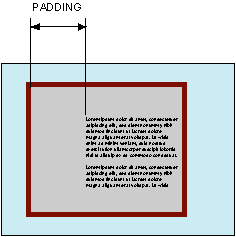


Рис. 1. Поле слева от текста

Свойство padding позволяет задать величину поля сразу для всех сторон элемента или определить ее только для указанных сторон.

**Синтаксис**

padding: [значение | проценты] {1, 4} | inherit

**Значения**

Разрешается использовать одно, два, три или четыре значения, разделяя их между собой пробелом. Эффект зависит от количества значений и приведен в табл. 1.

|  |  |
| --- | --- |
| Табл. 1. Зависимость от числа значений | |
| **Число значений** | **Результат** |
| 1 | Поля будут установлены одновременно от каждого края элемента. |
| 2 | Первое значение устанавливает поля от верхнего и нижнего края, второе — от левого и правого. |
| 3 | Первое значение задает поле от верхнего края, второе — одновременно от левого и правого края, а третье — от нижнего края. |
| 4 | Поочередно устанавливается поля от верхнего, правого, нижнего и левого края. |

Величину полей можно указывать в пикселах (px), процентах (%) или других допустимых для CSS единицах. Значение inherit указывает, что оно наследуется у родителя. При указании поля в процентах, значение считается от ширины контента блока.

## Подключение внешних CSS стилей при помощи тэга <link>

**Описание**

Устанавливает связь с внешним документом вроде файла со стилями или со шрифтами. В отличие от тега <a>, тег <link> размещается всегда внутри контейнера <head> и не создает ссылку.

**Синтаксис**

|  |  |
| --- | --- |
| HTML | <head>  <link атрибуты>  </head> |
| XHTML | <head>  <link атрибуты />  </head> |

**Атрибуты**

[charset](http://htmlbook.ru/html/link/charset)

Кодировка связываемого документа.

[href](http://htmlbook.ru/html/link/href)

Путь к связываемому файлу.

[media](http://htmlbook.ru/html/link/media)

Определяет устройство, для которого следует применять стилевое оформление.

[rel](http://htmlbook.ru/html/link/rel)

Определяет отношения между текущим документом и файлом, на который делается ссылка.

[sizes](http://htmlbook.ru/html/link/sizes)

Указывает размер иконок для визуального отображения.

[type](http://htmlbook.ru/html/link/type)

MIME-тип данных подключаемого файла.

**Закрывающий тег**

Не требуется.

Пример. Использование тега <link>

HTML 4.01IE 7IE 8IE 9Cr 13Op 11Sa 5Fx 6

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<title>Тег LINK</title>

<link rel="stylesheet" type="text/css" href="ie.css">

<link rel="alternate" type="application/rss+xml"

title="Статьи с сайта htmlbook.ru" href="http://htmlbook.ru/rss.xml">

<link rel="shortcut icon" href="http://htmlbook.ru/favicon.ico">

</head>

<body>

<p>...</p>

</body>

</html>

## Используемые источники

http://stepbystep.htmlbook.ru/?id=4 - Структура HTML-кода

http://www.w3.org/TR/html4/struct/global.html

http://www.freeartists.ru/articles/make-up/div-vs-table/ - DIV vs. TABLE

http://htmlbook.ru/html - Справочник по HTML