**Структура HTML-документа**

Любой HTML-документ состоит из трех основных секций:

* строки, содержащей *декларацию типа документа*;
* *заголовка* документа (заключенного в теги <**HEAD**>…<**/HEAD**>);
* *тела* документа (заключенного в теги <**BODY**>…<**/BODY**> или <**FRAMESET**>…<**/FRAMESET**>);

Перед каждой секцией и после нее могут находиться символы пробела, табуляции, новой строки и комментарии. Заголовок и тело документа должны быть заключены в теги <**HTML**>…<**/HTML**>.

Пример простого HTML-документа:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">

<HTML>

<HEAD>

<TITLE>Мой первый HTML-документ</TITLE>

</HEAD>

<BODY>

<P>Это HTML-документ.</P>

</BODY>

</HTML>

Поясним подробнее содержимое каждой секции.

**Декларация типа документа**

Рекомендуется начинать любой HTML-документ со строки, содержащей декларацию типа документа (DTD, document type declaration). HTML 4.0 поддерживает три типа таких деклараций:

* Документ *строго соответствует* стандарту HTML 4.0, т. е. не содержит ни морально устаревших элементов, ни фреймов. Декларация имеет вид:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"

"http://www.w3.org/TR/html4/strict.dtd">

* Документ является *переходным* к стандарту HTML 4.0, т. е. может содержать морально устаревшие элементы. Декларация имеет вид:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

* Документ *содержит фреймы*. Декларация имеет вид:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN"

"http://www.w3.org/TR/html4/frameset.dtd">

Поясним структуру приведенных деклараций. Каждая из них говорит о том, что последующий документ является HTML-документом, созданным в соответствии со стандартом HTML 4.0, разработанном [W3C](http://arininav.ru/js/organs.htm#w3c). Последние две буквы первой строки декларации обозначают язык DTD, который всегда является английским ( "EN"). Вторая строка декларации содержит [URI](http://arininav.ru/js/html01.htm#uri), откуда обозреватель может загрузить данную DTD.

**Примечания**.

1. Многие авторы оформляют эту декларацию в укороченном виде, опуская URI, т. е. так, как указано в приведенном выше примере.
2. Декларация типа документа является рекомендуемой, но не обязательной частью HTML-документа, поскольку старые обозреватели ее игнорируют. Первым обозревателем, учитывающим ее, стал Netscape 6, который при отсутствии DTD отображает документы в режиме совместимости с Netscape 4, а при ее наличии — в соответствии с Веб-стандартами. Internet Explorer 6 также поддерживает два режима отображения в зависимости от заданной DTD.

**Элемент HTML**

**Синтаксис**: <HTML>…</HTML> (оба тега не обязательны)

**Атрибуты**: [lang](http://arininav.ru/js/html01.htm#alang), [dir](http://arininav.ru/js/html01.htm#adir)

**Поддержка**: Атрибуты **lang** и **dir** игнорируются

Атрибуты **lang** и **dir** игнорируются

Этот элемент указывает, что последующий документ является HTML-документом. Он следует за декларацией типа документа и включает в себя все остальное содержимое документа. Он часто содержит атрибут [**lang**](http://arininav.ru/js/html01.htm#alang), задающий базовый язык документа. Иными словами, типичный HTML-документ имеет следующее строение:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"

"http://www.w3.org/TR/html4/strict.dtd">

**<HTML lang="en">**

*…Заголовок и тело документа…*

**</HTML>**

**Элемент HEAD**

**Синтаксис**: <HEAD>…</HEAD> (оба тега не обязательны)

**Атрибуты**: [lang](http://arininav.ru/js/html01.htm#alang), [dir](http://arininav.ru/js/html01.htm#adir)

profile = [URI](http://arininav.ru/js/html01.htm#uri) (URI словаря метаданных)

**Поддержка**: Атрибуты **lang**, **dir** и **profile** игнорируются

Атрибуты **lang**, **dir** и **profile** игнорируются

Заголовок документа, заключенный в элемент **HEAD**, содержит информацию об общих свойствах документа и не отображается обозревателями. Заголовок может включать в себя следующую информацию:

* единственный титул документа ([**TITLE**](http://arininav.ru/js/html02.htm#etitle));
* метаописатели документа ([**META**](http://arininav.ru/js/html02.htm#emeta));
* базовый URI внешних ссылок ([**BASE**](http://arininav.ru/js/html02.htm#ebase));
* ссылки на другие документы ([**LINK**](http://arininav.ru/js/html02.htm#elink));
* внутренние таблицы стилей ([**STYLE**](http://arininav.ru/js/html02.htm#estyle));
* сценарии клиента ([**SCRIPT**](http://arininav.ru/js/html09.htm#escript)).

Элемент **HEAD** может иметь атрибут **profile**, указывающий местонахождения словаря метаданных. Предполагается, что такой словарь должен содержать имена метапеременных, значения которых определяются элементами [**META**](http://arininav.ru/js/html02.htm#emeta) и [**LINK**](http://arininav.ru/js/html02.htm#elink) в заголовке документа, но пока работа по спецификации формата словарей метаданных не завершена.

**Титул документа: элемент TITLE**

**Синтаксис**: <TITLE>…</TITLE>

**Атрибуты**: [lang](http://arininav.ru/js/html01.htm#alang), [dir](http://arininav.ru/js/html01.htm#adir)

**Поддержка**: Атрибуты игнорируются, титул отображается в заголовке обозревателя

Атрибуты игнорируются, титул отображается в заголовке обозревателя

Каждый HTML-документ должен иметь единственный *титул*, который отображается обозревателями в строке заголовка. Текст титула может содержать специальные символы, но не должен содержать других элементов. Пример:

<HEAD>

**<TITLE>**Справочник Веб-разработчика**</TITLE>**

</HEAD>

Титул документа должен кратко отражать суть его содержимого; рекомендуемый размер титула — не более 60 символов.

**Метаописатели документа: элемент META**

**Синтаксис**: <META> (содержимого и конечного тега нет)

**Атрибуты**: [lang](http://arininav.ru/js/html01.htm#alang), [dir](http://arininav.ru/js/html01.htm#adir) (для атрибута **content**)

http-equiv = [NAME](http://arininav.ru/js/html01.htm#name) ([заголовок сообщения HTTP](http://arininav.ru/js/http.htm#http-headers))

name = [NAME](http://arininav.ru/js/html01.htm#name) (название свойства)

**content** = [CDATA](http://arininav.ru/js/html01.htm#cdata) (значение свойства)

scheme = [CDATA](http://arininav.ru/js/html01.htm#cdata) (имя схемы, интерпретирующей значение свойства)

**Поддержка**: Атрибуты **lang**, **dir** и **scheme** игнорируются (2.0+)

Атрибуты **lang**, **dir** и **scheme** игнорируются (4.0+)

Элемент **META** содержит *метаописатели* таких свойств документа, как имя автора документа, его описание, ключевые слова и т. д. Спецификация HTML 4.0 не содержит стандартного списка этих свойств, поэтому авторы пока свободны в их определении.

Каждый элемент **META** содержит пару атрибутов: *название свойства* (**name**) и *значение свойства* (**content**), например, следующий метаописатель задает имя автора документа:

<META name="Author" content="Ю. С. Лукач">

Дополнительно он может содержать атрибут **lang**, указывающий язык, на котором написано значение свойства:

<META name="Author" lang="en" content="Yury S. Lukach">

Многие поисковые системы используют свойства **description** (описание документа) и **keywords** (ключевые слова) для извлечения дополнительной информации о документе. При этом описание должно быть кратким (не длиннее 200 символов), а список ключевых слов разделяться запятыми:

<META name="Description" content="Web Developer's Handbook. HTML Reference.">

<META name="Keywords" content="Web development, HTML reference">

Некоторые поисковые системы поддерживают также свойство **robots**, содержащее указания для роботов, собирающих информацию о документах в Сети. Значение свойства **robots** — это список следующих директив, разделенных запятыми:

|  |  |
| --- | --- |
| index | эта страница должна быть индексирована |
| noindex | эта страница не должна быть индексирована |
| follow | прослеживать гиперссылки на странице |
| nofollow | не прослеживать гиперссылки на странице |
| all | = index, follow (принято по умолчанию) |
| none | = noindex, nofollow |

Например, следующий метаописатель указывает поисковым роботам, что данный документ индексировать не нужно, но гиперссылки, которые в нем содержаться, следует просмотреть.

<META name="robots" content="noindex,follow">

Вместо атрибута **name** метаописатель может содержать атрибут **http-equiv** для указания того, что данное свойство является [заголовком сообщения HTTP](http://arininav.ru/js/http.htm#http-headers). Такие метаописатели указывают обозревателю, как следует отображать данный документ. Существует три базовых свойства документа, которые задаются таким образом:

* кодировка символов документа:

<META http-equiv="Content-Type" content="text/html; charset=windows-1251">

* язык таблиц стилей, принятый по умолчанию:

<META http-equiv="Content-Style-Type" content="text/css">

* язык программирования сценариев, принятый по умолчанию:

<META http-equiv="Content-Script-Type" content="text/javascript">

Кроме того, атрибут **http-equiv** используется в следующих конструкциях:

<META http-equiv="Expires" content="Sun, 7 May 2000 12:04:32 GMT">

* задает дату и время истечения срока действия документа: полезен для того, чтобы заставить обозреватель загрузить документ не из кэш-памяти, а с сервера;

<META http-equiv="Refresh" content="10; URL=http://www.newserver.com/newpage.htm">

* указывает серверу, что через 10 секунд после завершения загрузки текущего документа нужно загрузить вместо него **http://www.newserver.com/newpage.htm**.

Метаописатели могут также содержать атрибут **scheme**, задающий формат значения свойства и предусмотренный для использования совместно со [словарями метаданных](http://arininav.ru/js/html02.htm#ehead). В настоящее время этот атрибут не обозревателями не поддерживается.

**Базовый URI документа: элемент BASE**

**Синтаксис**: <BASE> (содержимого и конечного тега нет)

**Атрибуты**: **href** = [URI](http://arininav.ru/js/html01.htm#uri) (базовый URI документа)

target = [фрейм](http://arininav.ru/js/html01.htm#frametarget) (имя фрейма для отображения ссылок)

**Поддержка**: Соответствует стандарту

Соответствует стандарту

Элемент **BASE** задает в обязательном атрибуте **href** базовый URI для данного документа, который используется обозревателем для приведения относительных URI к полным. Подробности см. в [Приложении 2](http://arininav.ru/js/uri.htm#reluri).

Если документ не содержит элемента **BASE**, то его собственный URI считается базовым по умолчанию. В большинстве случаев этого вполне достаточно, поэтому на практике элемент **BASE** употребляется только в двух ситуациях:

* когда документ хранится в нескольких узлах Сети, и мы хотим указать "эталонное" хранилище;
* когда URI документа неизвестен (например, при его получении по электронной почте).

Рассмотрим следующий пример:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN"

"http://www.w3.org/TR/html4/strict.dtd">

<HTML>

<HEAD>

<TITLE>Наша продукция</TITLE>

**<BASE href="http://www.bestseller.com/products/intro.html">**

</HEAD>

<BODY>

<P>Лучшие в мире <A href="../images/hornhoof.gif">рога и копыта</A>!

</BODY>

</HTML>

С учетом базового URI ссылка **"../images/hornhoof.gif"** в этом примере соответствует полному URI **"http://www.bestseller.com/images/hornhoof.gif"**.

Необязательный атрибут **target** используется только во [фреймовых документах](http://arininav.ru/js/html07.htm#ref17) и задает [имя фрейма](http://arininav.ru/js/html01.htm#frametarget), в котором должны отображаться по умолчанию все документы, на которые в данном документе имеются ссылки.

**Взаимосвязь документов: элемент LINK**

**Синтаксис**: <LINK> (содержимого и конечного тега нет)

**Атрибуты**: [id](http://arininav.ru/js/html01.htm#aid), [class](http://arininav.ru/js/html01.htm#aclass), [style](http://arininav.ru/js/html01.htm#astyle), [title](http://arininav.ru/js/html01.htm#atitle), [lang](http://arininav.ru/js/html01.htm#alang), [dir](http://arininav.ru/js/html01.htm#adir), [события](http://arininav.ru/js/html01.htm#aevent)

rel = [типы ссылок](http://arininav.ru/js/html01.htm#linktypes) (прямая ссылка)

rev = [типы ссылок](http://arininav.ru/js/html01.htm#linktypes) (обратная ссылка)

href = [URI](http://arininav.ru/js/html01.htm#uri) (URI ссылки)

target = [фрейм](http://arininav.ru/js/html01.htm#frametarget) (имя фрейма для отображения ссылки)

type = [тип файла](http://arininav.ru/js/html01.htm#contenttype) (тип файла, на который указывает ссылка)

media = [устройства](http://arininav.ru/js/html01.htm#mediadesc) (устройства для отображения ссылки)

hreflang = [код языка](http://arininav.ru/js/html01.htm#languagecode) (язык ссылки)

charset = [кодировка](http://arininav.ru/js/html01.htm#charset) (кодировка ссылки)

**Поддержка**: Атрибуты **type**, **hreflang** и **charset** игнорируются;

допустимые значения **media**: screen, print и all (5.0+)

Атрибуты **type**, **media**, **hreflang** и **charset** игнорируются (4.0+)

Элемент **LINK** определяет *взаимосвязь* между документами. Заголовок документа может содержать любое количество этих элементов. Многие обозреватели не поддерживают элементы **LINK**, поэтому авторы не должны полагаться на то, что обозреватель сделает перечисленные в них ссылки доступными пользователю.

Каждый элемент **LINK** должен содержать атрибут **rel** или **rev** и атрибут **href**. При этом атрибут **rel** определяет прямую ссылку, а атрибут **rev** — обратную ссылку. Например,

<LINK rel="Glossary" href="glossary.html">

означает, что документ **glossary.html** является глоссарием терминов для текущего документа (прямая ссылка), а

<LINK rev="Subsection" href="section2.html">

означает, что текущий документ является подразделом документа **section2.html** (обратная ссылка). Значением атрибутов **rel** и **rev** является список [типов ссылок](http://arininav.ru/js/html01.htm#linktypes), разделенных пробелами.

Важным применением элемента **LINK** является подключение к документу [внешней таблицы стилей](http://arininav.ru/js/css01.htm#ref2121). В этом случае элемент **LINK** имеет вид:

<LINK rel="StyleSheet" href="style.css" type="text/css">

Такой элемент может дополнительно содержать атрибут **media** для указания того, к отображению на каких [устройствах](http://arininav.ru/js/html01.htm#mediadesc) применяется данная таблица стилей:

<LINK rel="StyleSheet" href="aural.css" type="text/css" media="aural">

Необязательный атрибут **target** используется только во [фреймовых документах](http://arininav.ru/js/html07.htm#ref17) и задает [имя фрейма](http://arininav.ru/js/html01.htm#frametarget), в котором должна отображаться по умолчанию цель ссылки.

Наконец, атрибуты **hreflang** и **charset** указывают на [язык](http://arininav.ru/js/html01.htm#languagecode) и [кодировку](http://arininav.ru/js/html01.htm#charset) ссылки и предназначены для указания поисковым системам, где искать альтернативные версии данного документа. Для этого используется тип ссылки **Alternate**:

<LINK

rel="Alternate"

type="text/html"

href="indexru.htm"

hreflang="ru"

charset="windows-1251"

lang="ru" title="Русская версия">

<LINK

rel="Alternate"

type="text/html"

href="indexfr.htm"

hreflang="fr"

lang="fr" title="Version francaise">

Тип ссылки **Alternate** используется также для указания на версии текущего документа, предназначенного для отображения другими типами устройств. Например, указатель на версию документа для печати может иметь вид:

<LINK

rel="Alternate"

media="print"

lang="ru" title="Справочник в формате PostScript"

type="application/postscript"

href="manual.ps">

В частности, альтернативная внешняя таблица стилей может быть задана элементом:

<LINK rel="Alternate StyleSheet" href="style.css" type="text/css" media="...">

**Внутренние таблицы стилей: элемент STYLE**

**Синтаксис**: <STYLE>…</STYLE>

**Атрибуты**: [lang](http://arininav.ru/js/html01.htm#alang), [dir](http://arininav.ru/js/html01.htm#adir)

**type** = [тип файла](http://arininav.ru/js/html01.htm#contenttype) (тип таблицы стилей)

media = [устройства](http://arininav.ru/js/html01.htm#mediadesc) (устройства для отображения документа)

title = [текст](http://arininav.ru/js/html01.htm#text) (титул таблицы стилей)

**Поддержка**: Атрибуты **lang** и **dir** игнорируются;

допустимые значения **media**: screen, print и all (5.0+)

Атрибуты **lang**, **dir** и **media** игнорируются (4.0+)

Элемент **STYLE** позволяет включать в документ *внутренние таблицы стилей*. Заголовок документа может содержать любое количество этих элементов. Старые обозреватели не поддерживают элементы **STYLE**, поэтому рекомендуется заключать содержимое элемента **STYLE** в комментарий, как показано ниже.

Обязательный атрибут **type** указывает на тип таблицы стилей, т. е. на язык, на котором описываются стили. Для каскадных таблиц стилей этот атрибут всегда должен иметь значение **"text/css"**. Пример:

<HEAD>

**<STYLE type="text/css">**

<!--

H1 {border-width: 1; border: solid; text-align: center}

-->

**</STYLE>**

</HEAD>

<BODY>

<H1>Этот заголовок имеет указанный выше стиль</H1>

</BODY>

HTML позволяет авторам создавать документы, учитывающие особенности устройств, на которых документ будет отображаться. Для того, чтобы использовать разные таблицы стилей для различных устройств, включите в элемент **STYLE** атрибут **media**:

**<STYLE type="text/css" media="screen">**

<!--

H1 {color: blue}

-->

**</STYLE>**

**<STYLE type="text/css" media="print">**

<!--

H1 {text-align: center}

-->

**</STYLE>**

**Тело документа: элемент BODY**

**Синтаксис**: <BODY>…</BODY> (оба тега не обязательны)

**Атрибуты**: [id](http://arininav.ru/js/html01.htm#aid), [class](http://arininav.ru/js/html01.htm#aclass), [style](http://arininav.ru/js/html01.htm#astyle), [title](http://arininav.ru/js/html01.htm#atitle), [lang](http://arininav.ru/js/html01.htm#alang), [dir](http://arininav.ru/js/html01.htm#adir), [события](http://arininav.ru/js/html01.htm#aevent)

onload = [сценарий](http://arininav.ru/js/html01.htm#script) (документ загружен)

onunload = [сценарий](http://arininav.ru/js/html01.htm#script) (документ выгружен)

**Поддержка**: Атрибуты **lang** и **dir** игнорируются

Атрибуты **lang** и **dir** игнорируются

*Тело документа* содержит HTML-элементы, предназначенные для отображения обозревателем. Тело заключается в теги **<BODY>**…**</BODY>** или **<**[**FRAMESET**](http://arininav.ru/js/html07.htm#eframeset)**>**…**</FRAMESET>**, если документ содержит фреймы (о них см. [Главу 1.7](http://arininav.ru/js/html07.htm#ref17)). Напомним, что наше описание не включает [морально устаревшие элементы и атрибуты](http://arininav.ru/js/html01.htm#deprecated). Поэтому элемент **BODY** должен использовать только общие атрибуты и два нестандартных обработчика событий:

* **onload** — происходит, когда обозреватель завершил загрузку документа в окно;
* **onunload** — происходит, когда обозреватель завершил удаление документа из окна.

Все остальные атрибуты тела документа являются морально устаревшими и должны заменяться на [стили CSS](http://arininav.ru/js/css01.htm), например:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">

<HTML>

<HEAD>

<STYLE type="text/css">

BODY { background: white; color: black}

A:link { color: red }

A:visited { color: maroon }

A:active { color: purple }

</STYLE>

</HEAD>

**<BODY>**

*…тело документа…*

**</BODY>**

</HTML>

**Блочные и текстовые элементы**

Большинство элементов HTML, используемых в теле документа, подразделяются на *блочные* (block-level) и *текстовые* (inline) элементы. *Блочные элементы* могут содержать как текстовые элементы, так и другие блочные элементы. При отображении они всегда выводятся как отдельный абзац. *Текстовые элементы* могут содержать только текст и другие текстовые элементы, но не могут содержать блочных элементов. При отображении они выводятся в текущей строке.

Ниже при описании каждого из элементов HTML мы указываем, к какому из этих двух типов он относится.

# HTML-текст

**HTML-текст** представлен в спецификации элементами для форматирования и группировки текста. Данные элементы являются контейнерами для текста и не имеют визуального отображения.

Элементы для форматирования текста несут смысловую нагрузку и обычно задают для текста, заключенного внутрь, стилевое оформление, например, выделяют текст жирным начертанием или отображают его шрифтом другого семейства (свойство font-family).

Грамотно отформатированный текст дает понять поисковым системам, какие слова несут важную смысловую нагрузку, по каким из них предпочтительно ранжировать веб-страницу в поисковой выдаче. Вся текстовая информация, отображаемая на сайте, размещается внутри элемента <body>.

## HTML-элементы для текста

* **СОДЕРЖАНИЕ:**
* [1. Заголовки: <h1...h6>](https://html5book.ru/html-text/#part1)
* [2. Форматирование текста: <b>, <em>, <i>, <small>, <strong>, <sub>, <sup>, <ins>, <del>, <mark>](https://html5book.ru/html-text/#part2)
* [3. Ввод «компьютерного» текста: <code>, <kbd>, <samp>, <var>, <pre>](https://html5book.ru/html-text/#part3)
* [4. Оформление цитат и определений: <abbr>, <bdo>, <blockquote>, <q>, <cite>, <dfn>](https://html5book.ru/html-text/#part4)
* [5. Абзацы, средства переноса текста: <p>, <br>, <hr>](https://html5book.ru/html-text/#part5)

### 1. HTML-элементы для заголовков

Заголовки являются важными элементами веб-страницы, они упорядочивают текст, формируя его визуальную структуру. Элементы <h1>...<h6> должны использоваться только для выделения заголовков нового раздела или подраздела.

При использовании заголовков необходимо учитывать их иерархию, т.е. за <h1> должен следовать <h2> и т.д. Также не рекомендуется вкладывать в заголовки другие элементы.

# Заголовок 1-го уровня

## Заголовок 2-го уровня

### Заголовок 3-го уровня

#### Заголовок 4-го уровня

##### Заголовок 5-го уровня

###### Заголовок 6-го уровня

ФИГУРА. 1. ЭЛЕМЕНТЫ ДЛЯ ЗАГОЛОВКОВ HTML-ДОКУМЕНТА

#### 1.1. Элемент <h1>

Заголовок самого верхнего уровня, на странице рекомендуется использовать только один раз, по возможности частично дублируя заглавие страницы. Элемент <h1> должен быть уникальным для каждой страницы сайта.

Рекомендуется прописывать в начале статьи, используя ключевое слово в тексте заголовка. Размер шрифта в браузере равен **2em**, верхний и нижний отступ по умолчанию **0.67em**.

#### 1.2. Элемент <h2>

Представляет подзаголовки элемента <h1>. Размер шрифта в браузере равен **1.5em**, верхний и нижний отступ по умолчанию **0.83em**.

#### 1.3. Элемент <h3>

Показывает подзаголовки элемента <h2>. Размер шрифта в браузере равен **1.17em**, верхний и нижний отступ по умолчанию **1em**.

#### 1.4. Элементы <h4>, <h5>, <h6>

Обозначают подзаголовки четвёртого, пятого и шестого уровня. Размер шрифта в браузере равен **1em** / **0.83em** / **0.67em**, верхний и нижний отступ по умолчанию **1.33em** / **1.67em** / **2.33em** соответственно.

### 2. Элементы для форматирования текста

#### 2.1. Элемент <b>

Задаёт полужирное начертания шрифта. Выделяет текст без акцента на его важность.

#### 2.2. Элемент <em>

Отображает шрифт курсивом, придавая тексту значимость.

#### 2.3. Элемент <i>

Отображает шрифт курсивом.

#### 2.4. Элемент <small>

Уменьшает размер шрифта на единицу по отношению к обычному тексту.

#### 2.5. Элемент <strong>

Задаёт полужирное начертание шрифта, относится к элементам логической разметки, указывая браузеру на важность текста.

#### 2.6. Элемент <sub>

Используется для создания нижних индексов. Сдвигает текст ниже уровня строки, уменьшая его размер.

#### 2.7. Элемент <sup>

Используется для создания степеней. Сдвигает текст выше уровня строки, уменьшая его размер.

#### 2.8. Элемент <ins>

Выделяет текст в новой версии документа, подчёркивая его.

Для элемента доступны атрибуты cite и datetime.

#### 2.9. Элемент <del>

Перечёркивает текст. Используется для выделения текста, удаленного из документа.

Для элемента доступны атрибуты cite и datetime.

#### 2.10. Элемент <mark>

Применяется для выделения фрагментов текста в справочных целях, окрашивая блок символов желтым цветом.

### 3. Элементы для ввода «компьютерного» текста

#### 3.1. Элемент <code>

Служит для выделения фрагментов программного кода. Отображает текст моноширинным шрифтом.

#### 3.2. Элемент <kbd>

Отмечает фрагмент как вводимый пользователем с клавиатуры. Отображает текст моноширинным шрифтом.

#### 3.3. Элемент <samp>

Применяется для выделения результата, полученного в ходе выполнения программы. Отображает текст моноширинным шрифтом.

#### 3.4. Элемент <var>

Выделяет имена переменных, отображая текст курсивом.

#### 3.5. Элемент <pre>

Позволяет вывести текст на экран, сохранив изначальное форматирование. Пробелы и переносы строк при этом не удаляются.

### 4. Элементы для оформления цитат и определений

#### 4.1. Элемент <abbr>

Применяется для форматирования аббревиатур. Браузером обычно подчеркивается пунктирной линией. Расшифровка сокращения осуществляется с помощью атрибута title, она появляется при наведении курсора мыши на текст.

#### 4.2. Элемент <bdo>

Используется для изменения текущего направления текста.

Для элемента доступен атрибут dir.

#### 4.3. Элемент <blockquote>

Выделяет цитаты внутри документа, выделяя его отступами и переносами строк.

Для элемента доступен атрибут cite.

#### 4.4. Элемент <q>

Используется для выделения коротких цитат. Браузерами заключается в кавычки.

Для элемента доступен атрибут cite.

#### 4.5. Элемент <cite>

Применяется для выделения цитат, названий произведений, сносок на другие документы.

#### 4.6. Элемент <dfn>

Позволяет выделить текст как определение. Несмотря на наличие данного элемента, рекомендуется выделять текст силами CSS.

Для элемента доступен атрибут title.

### 5. Абзацы, средства переноса текста

#### 5.1. Элемент <p>

Разбивает текст на отдельные абзацы, отделяя друг от друга пустой строкой. Браузер автоматически добавляет верхнее и нижнее внешнее поле margin, равное 1em, при этом поля соседних абзацев «схлопываются».

#### 5.2. Элемент <br>

Переносит текст на следующую строку, создавая разрыв строки.

#### 5.3. Элемент <hr>

Используется для разделения контента на веб-странице. Отображается в виде горизонтальной линии.

# HTML-списки

**HTML-списки** используются для группировки связанных между собой фрагментов информации. Существует три вида списков:

* **маркированный список** — <ul> — каждый элемент списка <li> отмечается маркером,
* **нумерованный список** — <ol> — каждый элемент списка <li> отмечается цифрой,
* **список определений** — <dl> — состоит из пар термин <dt> — <dd> определение.

Каждый список представляет собой контейнер, внутри которого располагаются элементы списка или пары термин-определение.

Элементы списка ведут себя как блочные элементы, располагаясь друг под другом и занимая всю ширину блока-контейнера. Каждый элемент списка имеет дополнительный блок, расположенный сбоку, который не участвует в компоновке.

## Создание HTML-списков

* **СОДЕРЖАНИЕ:**
* [1. Маркированный список <ul>](https://html5book.ru/html-lists/#part1)
* [2. Нумерованный список <ol>](https://html5book.ru/html-lists/#part2)
* [3. Список определений <dl>](https://html5book.ru/html-lists/#part3)
* [4. Вложенный список](https://html5book.ru/html-lists/#part4)
* [5. Многоуровневый нумерованный список](https://html5book.ru/html-lists/#part5)

### 1. Маркированный список

**Маркированный список** представляет собой неупорядоченный список (от англ. Unordered List). Создаётся с помощью элемента <ul>. В качестве маркера элемента списка выступает метка, например, закрашенный кружок.

Браузеры по умолчанию добавляют следующее форматирование блоку списка:

ul {

padding-left: 40px;

margin-top: 1em;

margin-bottom: 1em;

}

CSS

Каждый элемент списка создаётся с помощью элемента <li> (от англ. List Item).

<ul>

<li>Microsoft</li>

<li>Google</li>

<li>Apple</li>

<li>IBM</li>

</ul>

HTML

* Microsoft
* Google
* Apple
* IBM

ФИГУРА. 1. МАРКИРОВАННЫЙ СПИСОК

### 2. Нумерованный список

**Нумерованный список** создаётся с помощью элемента <ol>. Каждый пункт списка также создаётся с помощью элемента <li>. Браузер нумерует элементы по порядку автоматически и если удалить один или несколько элементов такого списка, то остальные номера будут автоматически пересчитаны.

Блок списка также имеет стили браузера по умолчанию:

ol {

padding-left: 40px;

margin-top: 1em;

margin-bottom: 1em;

}

CSS

Для элемента <li> доступен атрибут value, который позволяет изменить номер по умолчанию для выбранного элемента списка. Например, если для первого пункта списка задать <li value="10">, то остальная нумерация будет пересчитана относительно нового значения.

Для элемента <ol> доступны следующие атрибуты:

|  |  |
| --- | --- |
| ТАБЛИЦА 1. АТРИБУТЫ ЭЛЕМЕНТА <OL> | |
| **Атрибут** | **Описание, принимаемое значение** |
| reversed | Атрибут reversed задает отображение списка в обратном порядке (например, 9, 8, 7…). |
| start | Атрибут start задает начальное значение, от которого пойдет отсчет нумерации, например, конструкция <ol start="10"> первому пункту присвоит порядковый номер «10». Также можно одновременно задавать тип нумерации, например, <ol type="I" start="10">. |
| type | Атрибут type задает вид маркера для использования в списке (в виде букв или цифр). Принимаемые значения: 1 — значение по умолчанию, десятичная нумерация. A — нумерация списка в алфавитном порядке, заглавные буквы (A, B, C, D). a — нумерация списка в алфавитном порядке, строчные буквы (a, b, c, d). I — нумерация римскими заглавными цифрами (I, II, III, IV). i — нумерация римскими строчными цифрами (i, ii, iii, iv). |

<ol>

<li>Microsoft</li>

<li>Google</li>

<li>Apple</li>

<li>IBM</li>

</ol>

HTML

1. Microsoft
2. Google
3. Apple
4. IBM

ФИГУРА. 2. НУМЕРОВАННЫЙ СПИСОК

### 3. Список определений

**Списки определений** создаются с помощью элемента <dl>. Для добавления термина применяется элемент <dt>, а для вставки определения — элемент <dd>.

Блок списка определений имеет следующие стили браузера по умолчанию:

dl {

margin-top: 1em;

margin-bottom: 1em;

}

CSS

Для элементов <dl>, <dt> и <dd> доступны [‎глобальные атрибуты](https://html5book.ru/html-attributes/).

<dl>

<dt>Режиссер:</dt>

<dd>Петр Точилин</dd>

<dt>В ролях:</dt>

<dd>Андрей Гайдулян</dd>

<dd>Алексей Гаврилов</dd>

<dd>Виталий Гогунский</dd>

<dd>Мария Кожевникова</dd>

</dl>

HTML

Режиссер:

Петр Точилин

В ролях:

Андрей Гайдулян

Алексей Гаврилов

Виталий Гогунский

Мария Кожевникова

ФИГУРА. 3. СПИСОК ОПРЕДЕЛЕНИЙ

### 4. Как создать вложенный список

Зачастую возможностей простых списков не хватает, например, при создании оглавления никак не обойтись без **вложенных пунктов**. Разметка для вложенного списка будет следующей:

<ul>

<li>Пункт 1.</li>

<li>Пункт 2.

<ul>

<li>Подпункт 2.1.</li>

<li>Подпункт 2.2.

<ul>

<li>Подпункт 2.2.1.</li>

<li>Подпункт 2.2.2.</li>

</ul>

</li>

<li>Подпункт 2.3.</li>

</ul>

</li>

<li>Пункт 3.</li>

</ul>

HTML

* Пункт 1.
* Пункт 2.
  + Подпункт 2.1.
  + Подпункт 2.2.
    - Подпункт 2.2.1.
    - Подпункт 2.2.2.
  + Подпункт 2.3.
* Пункт 3.

ФИГУРА. 4. ВЛОЖЕННЫЙ СПИСОК

### 5. Многоуровневый нумерованный список

Многоуровневый список используется для отображения элементов списка на разных уровнях с различными отступами. Разметка для многоуровневого нумерованного списка будет следующей:

<ol>

<li>пункт</li>

<li>пункт

<ol>

<li>пункт</li>

<li>пункт</li>

<li>пункт

<ol>

<li>пункт</li>

<li>пункт</li>

<li>пункт</li>

</ol>

</li>

<li>пункт</li>

</ol>

</li>

<li>пункт</li>

<li>пункт</li>

</ol>

HTML

Такая разметка по умолчанию создаст для каждого вложенного списка новую нумерацию, начинающуюся с единицы. Чтобы сделать вложенную нумерацию, нужно использовать следующие свойства:

* counter-reset сбрасывает один или несколько счётчиков, задавая значение для сброса;
* counter-increment задаёт значение приращения счётчика, т.е. с каким шагом будет нумероваться каждый последующий пункт;
* content — генерируемое содержимое, в данном случае отвечает за вывод номера перед каждым пунктом списка.

ol {

/\* убираем стандартную нумерацию \*/

list-style: none;

/\* Идентифицируем счетчик и даем ему имя li. Значение счетчика не указано - по умолчанию оно равно 0 \*/

counter-reset: li;

}

li:before {

/\* Определяем элемент, который будет нумероваться — li. Псевдоэлемент before указывает, что содержимое, вставляемое при помощи свойства content, будет располагаться перед пунктами списка. Здесь же устанавливается значение приращения счетчика (по умолчанию равно 1). \*/

counter-increment: li;

/\* С помощью свойства content выводится номер пункта списка. counters() означает, что генерируемый текст представляет собой значения всех счетчиков с таким именем. Точка в кавычках добавляет разделяющую точку между цифрами, а точка с пробелом добавляется перед содержимым каждого пункта списка \*/

content: counters(li, ".") ". ";

}

CSS

**HTML-ссылки**

**HTML-ссылки** создаются с помощью элементов <a>, <area> и <link>. Ссылки представляют собой связь между двумя ресурсами, одним из которых является текущий документ.

Ссылки можно поделить на две категории:

* **ссылки на внешние ресурсы** — создаются с помощью элемента <link> и используются для расширения возможностей текущего документа при обработке браузером;
* **гиперссылки** — ссылки на другие ресурсы, которые пользователь может посетить или загрузить.

## Как сделать гиперссылки на сайте

* **СОДЕРЖАНИЕ:**
* [1. Структура ссылки](https://html5book.ru/hyperlinks-in-html/#part1)
* [2. Абсолютный и относительный путь](https://html5book.ru/hyperlinks-in-html/#part2)
* [3. Якорь](https://html5book.ru/hyperlinks-in-html/#part3)
* [4. Как сделать изображение-ссылку](https://html5book.ru/hyperlinks-in-html/#part4)
* [5. Как сделать ссылку на телефонный номер, скайп или адрес электронной почты](https://html5book.ru/hyperlinks-in-html/#part5)
* [6. Атрибуты ссылок](https://html5book.ru/hyperlinks-in-html/#part6)

### 1. Структура ссылки

Гиперссылки создаются с помощью элемента <a></a>. Внутрь помещается текст, который будет отображаться на веб-странице. Текст ссылки отображается в браузере с подчёркиванием, цвет шрифта — синий, при наведении на ссылку курсор мыши меняет вид.

Обязательным параметром элемента <a> является атрибут href, который задает URl-адрес веб-страницы.

<a href="http://site.ru">указатель ссылки</a>

HTML

Ссылка состоит из двух частей — **указателя** и **адресной части**. **Указатель ссылки** представляет собой фрагмент текста или изображение, видимые для пользователя. **Адресная часть** ссылки пользователю не видна, она представляет собой адрес ресурса, к которому необходимо перейти.

Адресная часть ссылки состоит из URl. **URl** (Uniform Resource Locator) — унифицированный адрес ресурса. При создании адресов для разделения слов между собой рекомендуется использовать дефис, а не символ подчёркивания. В общем виде URl имеющий следующий формат:

метод доступа://имя сервера:порт/путь

HTML

**Метод доступа**, или протокол, осуществляет обмен данными между рабочими станциями в разных сетях. Наиболее распространенные протоколы передачи данных:

file обеспечивает чтение файла с локального диска:

file:/gallery/pictures/summer.html

HTML

http предоставляет доступ к веб-странице по протоколу HTTP:

http://site.ru/

HTML

https — специальная реализация протокола HTTP, использующая шифрование (как правило, SSL или TLS):

https://site.ru/

HTML

ftp осуществляет запрос к FTP-серверу на получение файла:

ftp://pgu/directory/library

HTML

mailto запускает сеанс почтовой связи с указанным адресатом и хостом:

mailto: nika@gmail.com

HTML

**Имя сервера** описывает полное имя машины в сети, например, site.ru. Если имя сервера не указано, то ссылка считается локальной, т.е. она относится к той же машине, на которой находится HTML-документ, содержащий ссылку.

**Номер порта ТСР**, на котором функционирует веб-сервер. Представляет собой число, которое необходимо указывать, если метод требует номер порта (отдельные сервера могут иметь свой отличительный номер порта). Если порт не указан, по умолчанию используется порт 80. Стандартными портами являются:

* 21 — FTP
* 23 — Telnet
* 70 — Gopher
* 80 — HTTP

**Путь** содержит имя папки, в которой находится файл.

### 2. Абсолютный и относительный путь

Когда в ссылке указывается только имя файла, браузер предполагает, что файл находится в той же папке, что и документ, содержащий гиперссылку. На практике веб-сайты содержат сотни документов, которые размещают в отдельные папки, чтобы ими было легче управлять. Чтобы создать ссылку на файл, находящийся вне папки, содержащей текущий документ, необходимо указать расположение файла или путь. HTML поддерживает два вида пути: абсолютный и относительный.

РИС. 1. ПРИМЕР СТРУКТУРЫ ПАПОК

#### 2.1. Абсолютный путь

**Абсолютный путь** указывает точное местоположение файла в пределах всей структуры папок на компьютере (сервере). Абсолютный путь к файлу даёт доступ к файлу со сторонних ресурсов и содержит следующие компоненты:

* протокол, например, http (опционально);
* домен (доменное имя или IP-адрес компьютера);
* папка (имя папки, указывающей путь к файлу);
* файл (имя файла).

Существует два вида записи абсолютного пути — с указанием протокола (полный) и без него:

http://site.ru/pages/tips/tips1.html

//site.ru/pages/tips/tips1.html

HTML

Когда вы ссылаетесь на страницу на другом сайте, вы можете использовать только полный абсолютный путь.

Если файл находится в корневой папке, то путь к файлу будет следующим:

http://site.ru/index.html

HTML

При отсутствии имени файла будет загружаться веб-страница, которая задана по умолчанию в настройках веб-сервера (так называемый индексный файл).

http://site.ru/

HTML

Обычно в качестве индексного файла выступает документ с именем index.html. Наличие завершающего слеша / означает, что обращение идет к папке, если его нет — напрямую к файлу.

#### 2.2. Относительный путь

**Относительный путь** описывает путь к указанному документу относительно текущего. Путь определяется с учётом местоположения веб-страницы, на которой находится ссылка. Относительные ссылки используются при создании ссылок на другие документы на одном и том же сайте. Когда браузер не находит в ссылке протокол http://, он выполняет поиск указанного документа на том же сервере.

Относительный путь содержит следующие компоненты:

* папка (имя папки, указывающей путь к файлу);
* файл (имя файла).

Путь для относительных ссылок имеет три специальных обозначения:

* / указывает на корневую директорию и говорит о том, что нужно начать путь от корневого каталога документов и идти вниз до следующей папки
* ./ указывает на текущую папку
* ../ подняться на одну папку (директорию) выше

Главное отличие относительного пути от абсолютного в том, что относительный путь не содержит имени корневой папки и родительских папок, что делает адрес короче, и в случае переезда с одного домена на другой не нужно прописывать новый абсолютный адрес. Но если сторонний ресурс будет ссылаться например, на ваши изображения с относительными адресами, то они не будут отображаться на другом сайте.

### 3. Якоря

**Якоря**, или внутренние ссылки, создают переходы на различные разделы текущей веб-страницы, позволяя быстро перемещаться между разделами. Это оказывается очень удобным в случае, когда на странице слишком много текста. Внутренние ссылки также создаются при помощи элемента <a> с разницей в том, что атрибут href содержит имя указателя — так называемый **якорь**, а не URl-адрес. Перед именем указателя всегда ставится знак #.

Следующая разметка создаст оглавление с быстрыми переходами на соответствующие разделы:

<h1>Времена года</h1>

<h2>Оглавление</h2>

<a href="#p1">Лето</a> <!--создаём якорь, указав #id элемента-->

<a href="#p2">Осень</a>

<a href="#p3">Зима</a>

<a href="#p4">Весна</a>

<p id="p1">...</p> <!--добавляем соответствующий id элементу-->

<p id="p2">...</p>

<p id="p3">...</p>

<p id="p4">...</p>

HTML

Если нужно сделать ссылку **с одной страницы** сайта на определенный раздел **другой страницы**, то необходимо задать id для этого раздела страницы, а затем добавить его к абсолютному адресу ссылки:

<th id="about-color">color</th>

<a href="https://html5book.ru/css-shrifty/#about-color" class="site" target="\_blank">color</a>

HTML

### 4. Как сделать изображение-ссылку

Чтобы сделать кликабельное изображение, необходимо поместить элемент <img> внутрь элемента <a>. Чтобы ссылка открывалась в другом окне, нужно добавить атрибут target="\_blank" для ссылки.

<a href="http://www.fast-torrent.ru/film/gran-za-granyu-tv.html" target="\_blank"><img src="//html5book.ru/wp-content/uploads/2014/07/Fringe.jpg" alt="Fringe"></a>

HTML

РИС. 2. ИЗОБРАЖЕНИЕ-ССЫЛКА

### 5. Как сделать ссылку на телефонный номер, скайп или адрес электронной почты

У ссылок появились новые возможности — по клику можно не только переходить на другие страницы и скачивать файлы, но и совершать звонки на телефоны, отправлять сообщения или звонить по скайпу.

ссылка на телефонный номер

<a href="tel:+74951234567">+7 (495) 123-45-67</a>

ссылка на адрес электронной почты

<a href="mailto:example@mail.ru">example@mail.ru</a>

ссылка на скайп (позвонить)

<a href="skype:имя-пользователя?call">Skype</a>

ссылка на скайп (открыть чат)

<a href="skype:имя-пользователя?chat">Skype</a>

ссылка на скайп (добавить в список контактов)

<a href="skype:имя-пользователя?add">Skype</a>

ссылка на скайп (отправить файл)

<a href="skype:имя-пользователя?sendfile">Skype</a>

HTML

### 6. Атрибуты ссылок

|  |  |
| --- | --- |
| ТАБЛИЦА 1. АТРИБУТЫ ЭЛЕМЕНТА <A> | |
| **Атрибут** | **Описание, принимаемое значение** |
| download | Дополняет атрибут href и сообщает браузеру, что ресурс должен быть загружен в момент, когда пользователь щелкает по ссылке, вместо того, чтобы, например, предварительно открыть его (как PDF-файл). Задавая имя для атрибута, мы таким образом задаем имя загружаемому объекту. Разрешается использовать атрибут без указания его значения:  <a href="/images/logo.png" download><img src="/images/logo.png" alt="logo"></a>  <a href="/images/logo.png" download="logo"><img src="/images/logo.png" alt="logo"></a>  <a href="files/20022014.pdf" download="Отчет Февраль 2014.pdf">Загрузить отчет за Февраль 2014</a> |
| href | Значением атрибута является URL-адрес документа, на который указывается ссылка. |
| hreflang | Определяет язык связанного веб-документа. Используется только вместе с атрибутом href. Принимаемые значения — аббревиатура, состоящая из пары букв, обозначающих [код языка](https://www.w3schools.com/tags/ref_language_codes.asp), например:  <a href="http://www.anysite.ru" hreflang="en">Anysite</a> |
| media | Определяет, для каких типов устройств файл будет оптимизирован. Значением может быть любой медиа-запрос. |
| ping | Отслеживает URL-адреса ресурсов, по которым переходил пользователь. |
| rel | Дополняет атрибут href информацией об отношении между текущим и связанным документом. Принимаемые значения: alternate — ссылка на альтернативную версию документа (например, печатную форму страницы, перевод или зеркало). author — ссылка на автора документа. bookmark — постоянный URL-адрес, используемый для закладок. help — ссылка на справку. license — ссылка на информацию об авторских правах на данный веб-документ. next/prev — указывает связь между отдельными URL. Благодаря этой разметке Google может определить, что содержание данных страниц связано в логической последовательности. nofollow — запрещает поисковой системе переходить по ссылкам на данной странице или по конкретной ссылке. noreferrer — указывает, что переходе по ссылке браузер не должен посылать заголовок HTTP-запроса (Referrer), в который записывается информация о том, с какой страницы пришел посетитель сайта. prefetch — указывает, что целевой документ должен быть кэширован, т.е. браузер в фоновом режиме загружает содержимое страницы к себе в кэш. search — указывает, что целевой документ содержит инструмент для поиска. tag — указывает ключевое слово для текущего документа. |
| target | Указывает на то, в каком окне должен открываться документ, к которому ведет ссылка. Принимает следующие значения: \_self — страница загружается в текущее окно; \_blank — страница открывается в новом окне браузера; \_parent — страница загружается во фрейм-родитель; \_top — страница загружается в полное окно браузера. |
| type | Указывает [MIME-тип](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_MIME-%D1%82%D0%B8%D0%BF%D0%BE%D0%B2) файлов ссылки, т.е. расширение файла, носит больше справочную информацию. |

# HTML-изображения

**HTML-изображения** добавляются на веб-страницы с помощью элемента <img>. Использование графики делает веб-страницы визуально привлекательнее. Изображения помогают лучше передать суть и содержание веб-документа.

Элементы <map> и <area> позволяют создавать **карты-изображения** с активными областями.

## Вставка изображений в HTML-документ

* **СОДЕРЖАНИЕ:**
* [1. Элемент <img>](https://html5book.ru/images-in-html/#part1)
* [1.1. Адрес изображения](https://html5book.ru/images-in-html/#part2)
* [1.2. Размеры изображения](https://html5book.ru/images-in-html/#part3)
* [1.3. Форматы графических файлов](https://html5book.ru/images-in-html/#part4)
* [2. Элемент <map>](https://html5book.ru/images-in-html/#part5)
* [3. Элемент <area>](https://html5book.ru/images-in-html/#part6)
* [4. Пример создания карты-изображения](https://html5book.ru/images-in-html/#part7)

### 1. Элемент <img>

Элемент <img> представляет изображение и его резервный контент, который добавляется с помощью атрибута alt. Так как элемент <img> является строчным, то рекомендуется располагать его внутри блочного элемента, например, <p> или <div>.

Элемент <img> имеет обязательный атрибут src, значением которого является абсолютный или относительный путь к изображению:

<img src="image.png" alt="Пример кода">

HTML

Для элемента <img> доступны следующие атрибуты:

|  |  |
| --- | --- |
| ТАБЛИЦА 1. АТРИБУТЫ ЭЛЕМЕНТА <IMG> | |
| **Атрибут** | **Описание, принимаемое значение** |
| alt | Атрибут alt добавляет альтернативный текст для изображения. Выводится на месте появления изображения до его загрузки или при отключенной графике. Синтаксис: alt="описание изображения". |
| crossorigin | Атрибут crossorigin позволяет загружать изображения с ресурсов другого домена с помощью CORS-запросов. Изображения, загруженные в холст с помощью CORS-запросов, могут быть использованы повторно. Допускаемые значения: anonymous — Cross-origin запрос выполняется с помощью HTTP-заголовка, при этом учетные данные не передаются. Если сервер не даёт учетные данные серверу, с которого запрашивается контент, то изображение будет испорчено и его использование будет ограничено. use-credentials — Cross-origin запрос выполняется с передачей учетных данных. Синтаксис: crossorigin="anonymous". |
| height | Атрибут height задает высоту изображения в px. Синтаксис: height="300". |
| ismap | Атрибутismap указывает на то, что картинка является частью изображения-карты, расположенного на сервере (изображение-карта — изображение с интерактивными областями). При нажатии на изображение-карту координаты передаются на сервер в виде строки запроса URL-адреса. Атрибут ismap допускается только в случае, если элемент <img> является потомком элемента <a> с действительным атрибутом href. Синтаксис: ismap. |
| longdesc | URL расширенного описания изображения, дополняющее атрибут alt. Синтаксис: longdesc="http://www.example.com/description.txt". |
| src | Атрибут src задает путь к изображению. Синтаксис: src="flower.jpg". |
| sizes | Задаёт размер изображения в зависимости от параметров отображения. Работает только при заданном атрибуте srcset. Значением атрибута является одна или несколько строк, указанных через запятую. |
| srcset | Создаёт список источников для изображения, которые будут выбраны, исходя из разрешения экрана. Может использоваться вместе или вместо атрибута src. Значением атрибута является одна или несколько строк, разделенных запятой.  <img src="flower.jpg"  srcset="  img/flower-mobile.jpg 320w,  img/flower-wide-mobile.jpg 480w,  img/flower-tablet.jpg 768w,  img/flower-desktop.jpg 1024w,  img/flower-hires.jpg 1280w"  sizes="  (max-width: 20em) 30vw,  (max-width: 30em) 60vw,  (max-width: 40em) 90vw"  alt="Роза в моём саду">  HTML |
| usemap | Атрибут usemap определяет изображение в качестве карты-изображения. Значение обязательно должно начинаться с символа #. Значение ассоциируется со значением атрибута name или id элемента <map> и создает связь между элементами <img> и <map>. Атрибут нельзя использовать, если элемент <map> является потомком элемента <a> или <button>. Синтаксис: usemap="#mymap". |
| width | Атрибут width задает ширину изображения в px. Синтаксис: width="500". |

#### 1.1. Адрес изображения

Адрес изображения может быть указан полностью (абсолютный URL), например: url(http://anysite.ru/images/anyphoto.png)

Или же через относительный путь от **документа** или **корневого каталога** сайта:

* url(../images/anyphoto.png) — относительный путь от документа,
* url(/images/anyphoto.png) — относительный путь от корневого каталога.

Это интерпретируется следующим образом:

* ../ — означает подняться вверх на один уровень, к корневому каталогу,
* images/ — перейти к папке с изображениями,
* anyphoto.png — указывает на файл изображения.

#### 1.2. Размеры изображения

Без задания размеров изображения рисунок отображается на странице в реальном размере. Отредактировать размеры изображения можно с помощью атрибутов width и height. Если будет задан только один из атрибутов, то второй будет вычисляться автоматически для сохранения пропорций рисунка.

#### 1.3. Форматы графических файлов

* Формат **JPEG** (Joint Photographic Experts Group). Изображения JPEG идеальны для фотографий, они могут содержать миллионы различных цветов. Сжимают изображения лучше GIF, но текст и большие площади со сплошным цветом могут покрыться пятнами.
* Формат **GIF** (Graphics Interchange Format). Идеален для сжатия изображений, в которых есть области со сплошным цветом, например, логотипов. GIF-файлы позволяют установить один из цветов прозрачным, благодаря чему фон веб-страницы может проявляться через часть изображения. Также GIF-файлы могут включать в себя простую анимацию. GIF-изображения содержат всего лишь 256 оттенков, из-за чего изображения выглядят пятнистыми и нереалистичного цвета, как плакаты.
* Формат **PNG** (Portable Network Graphics). Включает в себя лучшие черты GIF- и JPEG-форматов. Содержит 256 цветов и дает возможность сделать один из цветов прозрачным, при этом сжимает изображения в меньший размер, чем GIF-файл.
* Формат **APNG** (Animated Portable Network Graphics). Формат изображения, основанный на формате PNG. Позволяет хранить анимацию, а также поддерживает прозрачность.
* **SVG** (Scalable Vector Graphics). SVG-рисунок состоит из набора геометрических фигур, описанных в формате XML: линия, эллипс, многоугольник и т.п. Поддерживается как статичная, так и анимированная графика. Набор функций включает в себя различные преобразования, альфа-маски, эффекты фильтров, возможность использования шаблонов. Изображения в формате SVG могут изменяться в размере без снижения качества.
* Формат **BMP** (Bitmap Picture). Представляет собой несжатое (оригинальное) растровое изображение, шаблоном которого является прямоугольная сетка пикселей. Bitmap-файл состоит из заголовка, палитры и графических данных. В заголовке хранится информация о графическом изображении и файле (глубина пикселей, высота, ширина и количество цветов). Палитра указывается только в тех Bitmap-файлах, которые содержат палитровые изображения (8 и менее бит) и состоят не более чем из 256 элементов. Графические данные представляют саму картинку. Глубина цвета в данном формате может быть 1, 2, 4, 8, 16, 24, 32, 48 бит на пиксель.
* Формат **ICO** (Windows icon). Формат хранения значков файлов в Microsoft Windows. Также, Windows icon, используется как иконка на сайтах в интернете. Именно картинка такого формата отображается рядом с адресом сайта или закладкой в браузере. Один ICO-файл содержит один или несколько значков, размер и цветность каждого из которых задаётся отдельно. Размер значка может быть любым, но наиболее употребимы квадратные значки со сторонами 16, 32 и 48 пикселей.

### 2. Элемент <map>

Элемент <map> служит для представления графического изображения в виде карты с активными областями. Активные области определяются по изменению вида курсора мыши при наведении. Щелкая мышью на активных областях, пользователь может переходить к связанным документам.

Для элемента доступен атрибут name, который задает имя карты. Значение атрибут name для элемента <map> должно соответствовать имени в атрибуте usemap элемента <img>:

<img src="url" usemap="#имя\_карты">

<map name="имя\_карты">

...

</map>

HTML

Элемент <map> содержит ряд элементов <area>, определяющих интерактивные области в изображении карты.

### 3. Элемент <area>

Элемент <area> описывает только одну активную область в составе карты изображений на стороне клиента. Если одна активная область перекрывает другую, то будет реализована первая ссылка из списка областей.

<map name="имя\_карты">

<area атрибуты>

</map>

HTML

|  |  |
| --- | --- |
| ТАБЛИЦА 2. АТРИБУТЫ ЭЛЕМЕНТА <AREA> | |
| **Атрибут** | **Краткое описание** |
| alt | Задает альтернативный текст для активной области карты. |
| coords | Определяет позицию области на экране. Координаты задаются в единицах длины и разделяются запятыми: для **круга** — координаты центра и радиус круга; для **прямоугольника** — координаты верхнего левого и правого нижнего углов; для **многоугольника** — координаты вершин многоугольника в нужном порядке, также рекомендуется указывать последние координаты, равные первым, для логического завершения фигуры. |
| download | Дополняет атрибут href и сообщает браузеру, что ресурс должен быть загружен в момент, когда пользователь щелкает по ссылке, вместо того, чтобы, например, предварительно открыть его (как PDF-файл). Задавая имя для атрибута, мы таким образом задаем имя загружаемому объекту. Разрешается использовать атрибут без указания его значения. |
| href | Указывает URL-адрес для ссылки. Может быть указан абсолютный или относительный адрес ссылки. |
| hreflang | Определяет язык связанного веб-документа. Используется только вместе с атрибутом href. Принимаемые значения — аббревиатура, состоящая из пары букв, обозначающих код языка. |
| media | Определяет, для каких типов устройств файл будет оптимизирован. Значением может быть любой медиа-запрос. |
| rel | Дополняет атрибут href информацией об отношении между текущим и связанным документом. Принимаемые значения: alternate — ссылка на альтернативную версию документа (например, печатную форму страницы, перевод или зеркало). author — ссылка на автора документа. bookmark — постоянный URL-адрес, используемый для закладок. help — ссылка на справку. license — ссылка на информацию об авторских правах на данный веб-документ. next/prev — указывает связь между отдельными URL. Благодаря этой разметке Google может определить, что содержание данных страниц связано в логической последовательности. nofollow — запрещает поисковой системе переходить по ссылкам на данной странице или по конкретной ссылке. noreferrer — указывает, что переходе по ссылке браузер не должен посылать заголовок HTTP-запроса (Referrer), в который записывается информация о том, с какой страницы пришел посетитель сайта. prefetch — указывает, что целевой документ должен быть кэширован, т.е. браузер в фоновом режиме загружает содержимое страницы к себе в кэш. search — указывает, что целевой документ содержит инструмент для поиска. tag — указывает ключевое слово для текущего документа. |
| shape | Задает форму активной области на карте и ее координаты. Может принимать следующие значения: rect — активная область прямоугольной формы; circle — активная область в форме круга; poly — активная область в форме многоугольника; default — активная область занимает всю площадь изображения. |
| target | Указывает, куда будет загружен документ при переходе по ссылке. Принимает следующие значения: \_self — страница загружается в текущее окно; \_blank — страница открывается в новом окне браузера; \_parent — страница загружается во фрейм-родитель; \_top — страница загружается в полное окно браузера. |
| type | Указывает MIME-тип файлов ссылки, т.е. расширение файла. |

# HTML-таблицы

**HTML-таблицы** упорядочивают и выводят на экран данные с помощью строк или столбцов. Таблицы состоят из ячеек, образующихся при пересечении строк и столбцов.

**Ячейки таблиц** могут содержать любые HTML-элементы, такие как заголовки, списки, текст, изображения, элементы форм, а также другие таблицы. Каждой таблице можно добавить связанный с ней заголовок, расположив его перед таблицей или после неё.

Таблицы больше не используются для вёрстки веб-страниц и компоновки отдельных элементов, потому что такой приём не обеспечивает гибкость структуры и адаптивность сайта, существенно увеличивая HTML-разметку.

Для всех элементов таблицы доступны [‎глобальные атрибуты](https://html5book.ru/html-attributes/), а также собственные атрибуты.

## Создание таблиц в HTML

* **СОДЕРЖАНИЕ:**
* [1. Как создать таблицу](https://html5book.ru/html-table/#table)
* [2. Как создать строки (ряды) таблицы](https://html5book.ru/html-table/#tr)
* [3. Как сделать ячейку заголовка столбца таблицы](https://html5book.ru/html-table/#th)
* [4. Как сделать ячейку тела таблицы](https://html5book.ru/html-table/#td)
* [5. Как добавить подпись (заголовок) к таблице](https://html5book.ru/html-table/#caption)
* [6. Группирование строк и столбцов таблицы <colgroup> и <col>](https://html5book.ru/html-table/#colgroup-col)
* [7. Группировка разделов таблицы <thead>, <tbody> и <tfoot>](https://html5book.ru/html-table/#grouping)
* [8. Как объединить ячейки таблицы](https://html5book.ru/html-table/#grouping-cell)
* [9. Атрибуты элементов таблицы](https://html5book.ru/html-table/#table-attributes)
* [10. Пример создания таблицы](https://html5book.ru/html-table/#table-example)

### 1. Как создать таблицу

Таблица создаётся при помощи элемента <table></table>, который является контейнером для элементов таблицы и все элементы должны находиться внутри него.

Например, с помощью данной разметки можно создать таблицу, состоящую из **двух столбцов и двух строк**:

<table>

<tr><th>текст заголовка</th><th>текст заголовка</th></tr> <!--ряд с ячейками заголовков-->

<tr><td>данные</td><td>данные</td></tr> <!--ряд с ячейками тела таблицы-->

</table>

HTML

|  |  |
| --- | --- |
| **текст заголовка** | **текст заголовка** |
| данные | данные |

ФИГУРА 1. ВНЕШНИЙ ВИД ТАБЛИЦЫ БЕЗ ФОРМАТИРОВАНИЯ CSS-СВОЙСТВАМИ

По умолчанию таблица и ячейки не имеют видимых границ. **Границы** задаются с помощью свойства border:

/\* внешние границы таблицы серого цвета толщиной 1px \*/

table {

border: 1px solid grey;

}

/\* границы ячеек первого ряда таблицы \*/

th {

border: 1px solid grey;

}

/\* границы ячеек тела таблицы \*/

td {

border: 1px solid grey;

}

CSS

**Промежутки между ячейками таблицы** убираются с помощью свойства table {border-collapse: collapse;}.

**Ширина** таблицы по умолчанию равна ширине её внутреннего содержимого. Чтобы установить ширину, нужно задать значение для свойства width:

/\* сделает ширину таблицы равной ширине блока контейнера, в котором она находится \*/

table {

width: 100%;

}

/\* задаст фиксированную ширину для таблицы \*/

table {

width: 600px;

}

CSS

Если для ячеек таблицы заданы внутренние отступы и границы, то ширина таблицы будет включать в себя следующие значения: padding-left и padding-right, ширина border-left плюс ширина border-right последней ячейки в ряду.

Если заданы ширина и границы ячеек, то ширина таблицы будет складываться из ширины ячеек плюс ширина border-left и ширина border-right последней ячейки в ряду.

### 2. Как создать строки (ряды) таблицы

Строки или ряды таблицы создаются с помощью элемента <tr>. Количество горизонтальных строк таблицы определяется количеством элементов <tr></tr>.

### 3. Как сделать ячейку заголовка столбца таблицы

Элемент <th> создаёт заголовок столбца — специальную ячейку, текст в которой выделяется полужирным. Количество ячеек заголовка определяется количеством элементов <th></th>. Для элемента доступны атрибуты colspan, rowspan, headers.

<table>

<tr><th>ячейка заголовка</th><th>ячейка заголовка</th></tr>

</table>

HTML

### 4. Как сделать ячейку тела таблицы

Элемент <td> создаёт ячейки таблицы, внутрь которых помещаются данные таблицы. Элементы <td></td>, расположенные в одном ряду, определяют количество ячеек в строке таблицы.

Количество пар ячеек <td> должно быть равно количеству пар ячеек <th>. Для элемента доступны атрибуты colspan, rowspan, headers.

<table>

<tr><th>ячейка заголовка</th><th>ячейка заголовка</th></tr>

<tr><td>ячейка тела таблицы</td><td>ячейка тела таблицы</td></tr>

</table>

HTML

### 5. Как добавить подпись (заголовок) к таблице

Элемент <caption> создает подпись таблицы. Добавляется непосредственно после тега <table>, вне строки или ячейки.

<table>

<caption>Перечень продуктов</caption>

<tr>

<th>№ п/п</th>

<th>Наименование товара</th>

<th>Ед. изм.</th>

<th>Количество</th>

<th>Цена за ед. изм., руб.</th>

<th>Стоимость, руб.</th>

</tr>

<tr>

<td>1.</td>

<td>Томаты свежие</td><td>кг</td><td>15,20</td><td>69,00</td><td>1048,80</td>

</tr>

<tr>

<td>2.</td>

<td>Огурцы свежие</td><td>кг</td><td>2,50</td><td>48,00</td><td>120,00</td>

</tr>

<tr>

<td colspan="5" style="text-align:right">ИТОГО:</td><td>1168,80</td>

</tr>

</table>

HTML

### 6. Группирование строк и столбцов таблицы

Элемент <colgroup> создает структурную группу столбцов, выделяя логически однородные ячейки. Группирует один или более столбцов для единого форматирования, позволяя применить стили к столбцам вместо того, чтобы повторять стили для каждой ячейки и для каждой строки.

Добавляется непосредственно после тегов <table> и/или <caption>.

Элемент <col> формирует группы столбцов, которые делят таблицу на разделы, не относящиеся к общей структуре, т.е. не содержащие информацию одного типа. Позволяет задавать свойства столбцов для каждого столбца в пределах элемента <colgroup>.

С помощью атрибута style можно изменить основной цвет фона ячеек. Для элемента <col> доступен атрибут span, задающий количество столбцов для объединения.

<table>

<colgroup>

<col span="2" style="background:Khaki"><!-- С помощью этой конструкции задаем цвет фона для первых двух столбцов таблицы-->

<col style="background-color:LightCyan"><!-- Задаем цвет фона для следующего (одного) столбца таблицы-->

</colgroup>

<tr>

<th>№ п/п</th>

<th>Наименование</th>

<th>Цена, руб.</th>

</tr>

<tr>

<td>1</td>

<td>Карандаш цветной</td>

<td>20,00</td>

</tr>

<tr>

<td>2</td>

<td>Линейка 20 см</td>

<td>30,00</td>

</tr>

</table>

HTML

### 7. Группировка разделов таблицы

Элемент <thead> создает группу заголовков для строк таблицы с целью задания единого оформления. Используется в сочетании с элементами <tbody> и <tfoot> для указания каждой части таблицы.

Элемент должен быть использован в следующем порядке: как дочерний элемент <table>, после <caption> и <colgroup>, и перед <tbody>, <tfoot> и <tr> элементами. В пределах одной таблицы можно использовать один раз.

Элемент <tbody> группирует основное содержимое таблицы. Используется в сочетании с элементами <thead> и <tfoot>.

Элемент <tfoot> создает группу строк для представления информации о суммах или итогах, расположенную в нижней части таблицы. Используется в таблице один раз. Располагается после элемента <thead>, перед элементами <tbody> и <tr>.

<table>

<thead>

<tr>

<th>№ п/п</th>

<th>Наименование товара</th>

<th>Ед. изм.</th>

<th>Количество</th>

<th>Цена за ед. изм., руб.</th>

<th>Стоимость, руб.</th>

</tr>

</thead>

<tfoot>

<tr>

<td colspan="5" style="text-align:right">ИТОГО:</td><td>1168,80</td>

</tr>

</tfoot>

<tbody>

<tr>

<td>1.</td>

<td>Томаты свежие</td><td>кг</td><td>15,20</td><td>69,00</td><td>1048,80</td>

</tr>

<tr>

<td>2.</td>

<td>Огурцы свежие</td><td>кг</td><td>2,50</td><td>48,00</td><td>120,00</td>

</tr>

</tbody>

</table>

HTML

Такая группировка строк была заложена в стандарте в расчете на то, что обозреватели при отображении длинных таблиц обеспечат прокрутку строк данных при сохранении надзаголовка и подзаголовка неподвижными, а при их выводе на принтер смогут использовать надзаголовок и подзаголовок в качестве колонтитулов страницы. Однако, современные обозреватели этого не делают и либо просто отображают *<thead>* и *<tfoot>* как строки данных, либо, в лучшем случае, просто помещают соответствующие строки в начало и конец таблицы.

### 8. Как объединить ячейки таблицы

Атрибуты colspan и rowspan объединяют ячейки таблицы. Атрибут colspan задает количество ячеек, объединенных по горизонтали, а rowspan — по вертикали.

<table>

<tr>

<th>№ п/п</th>

<th>Наименование товара</th>

<th>Ед. изм.</th>

<th>Количество</th>

<th>Цена за ед. изм., руб.</th>

<th>Стоимость, руб.</th>

</tr>

<tr>

<td>1.</td>

<td>Томаты свежие</td><td>кг</td><td>15,20</td><td>69,00</td><td>1048,80</td>

</tr>

<tr>

<td>2.</td>

<td>Огурцы свежие</td><td>кг</td><td>2,50</td><td>48,00</td><td>120,00</td>

</tr>

<tr>

<td colspan="5" style="text-align:right">ИТОГО:</td><td>1168,80</td><!-- Задаем количество ячеек по горизонтали для объединения-->

</tr>

</table>

HTML

РИС. 3. ПРИМЕР ОБЪЕДИНЕНИЯ ЯЧЕЕК ТАБЛИЦЫ ПО ГОРИЗОНТАЛИ ПРИ ПОМОЩИ АТРИБУТА COLSPAN

### 9. Атрибуты элементов таблицы

|  |  |
| --- | --- |
| ТАБЛИЦА 1. АТРИБУТЫ ЭЛЕМЕНТОВ ТАБЛИЦЫ | |
| **Атрибут** | **Описание, принимаемое значение** |
| colspan | Количество ячеек в строке для объединения по горизонтали. <td colspan="3"> Возможные значения: число от 1 до 999. |
| headers | Задает список ячеек заголовка, содержащих информацию о заголовке текущей ячейки данных. Предназначен для речевых браузеров. <th id="идентификатор">...</th> <th headers="идентификатор">...</th> Принимаемые значения: список имен ячеек, разделенных пробелами; эти имена должны быть присвоены ячейкам через их атрибут id. |
| rowspan | Количество ячеек в столбце для объединения по вертикали. <td rowspan="2"> Возможные значения: число от 1 до 999. |
| span | Количество колонок, объединяемых для задания единого стиля, по умолчанию равно 1. <col span="2"> Принимаемые значения: любое целое положительное число. |

### 10. Пример создания таблицы

РИС. 4. СОЗДАНИЕ МЕНЮ РЕСТОРАНА С ПОМОЩЬЮ HTML-ТАБЛИЦЫ

<table>

<caption>Меню ресторана "Ромашка"</caption>

<tr>

<th rowspan="2" class="first">Кухня</th>

<th colspan="2">Холодные блюда</th>

<th colspan="2">Горячие блюда</th>

<th rowspan="2">Десерты</th>

</tr>

<tr>

<td class="first">Салаты</td>

<td class="first">Закуски</td>

<td class="first">Первые блюда</td>

<td class="first">Вторые блюда</td>

</tr>

<tr>

<td rowspan="3" class="first">Русская</td>

<td>Винегрет</td>

<td>Язык с хреном</td>

<td>Щи с квашеной капустой</td>

<td>Вареники с картошкой</td>

<td>Печеные яблоки с медом</td>

</tr>

<tr>

<td>Оливье</td>

<td>Студень говяжий</td>

<td>Рассольник домашний</td>

<td>Караси запеченые в сметане</td>

<td>Блинчатый пирог</td>

</tr>

<tr>

<td>Сельдь под "шубой"</td>

<td>Судак заливной</td>

<td>Мясная солянка</td>

<td>Котлеты "Пожарские"</td>

<td>Пирожное "Картошка"</td>

</tr>

<tr>

<td rowspan="3" class="first">Испанская</td>

<td>Севиче из гребешков</td>

<td>Эмпанадас</td>

<td>Хлебный суп с чесноком</td>

<td>Паэлья с морепродуктами</td>

<td>Чуррос</td>

</tr>

<tr>

<td>Тимбал из авокадо и тунца</td>

<td>Ахотомате</td>

<td>Астурийская фабада</td>

<td>Свиное раксо</td>

<td>Альмойшавена</td>

</tr>

<tr>

<td>Фасоль с ветчиной</td>

<td>Чанфайна</td>

<td>Рыбный суп с манными клецками </td>

<td>Тортилья картофельная</td>

<td>Бунуэлос</td>

</tr>

<tr>

<td rowspan="3" class="first">Французская</td>

<td>Вогезский салат</td>

<td>Рийет из курицы</td>

<td>Баклажанный крем-суп "Ренуар"</td>

<td>Картофель огратен</td>

<td>Бриоши</td>

</tr>

<tr>

<td>Салат "Панзанелла"</td>

<td>Делисьез из сыра</td>

<td>Французский тыквенный суп</td>

<td>Гратин из птицы</td>

<td>Лигурийский лимонный пирог</td>

</tr>

<tr>

<td>Тар-тар</td>

<td>Маринованный лосось</td>

<td>Суп "Конти"</td>

<td>Тартифлетт</td>

<td>Саварен "Триумф"</td>

</tr>

</table>

HTML

body {

margin: 0;

background: #F4F1F8;

}

table {

border-collapse: collapse;

line-height: 1.1;

font-family: "Lucida Sans Unicode", "Lucida Grande", sans-serif;

background: radial-gradient(farthest-corner at 50% 50%, white, #DCECF8);

color: #0C213B;

}

caption {

font-family: annabelle, cursive;

font-weight: bold;

font-size: 2em;

padding: 10px;

color: #F3CD26;

text-shadow: 1px 1px 0 rgba(0,0,0,.3);

}

caption:before, caption:after {

content: "\274B";

color: #A9E2CC;

margin: 0 10px;

}

th {

padding: 10px;

border: 1px solid #A9E2CC;

}

td {

font-size: 0.8em;

padding: 5px 7px;

border: 1px solid #A9E2CC;

}

.first {

font-size: 1em;

font-weight: bold;text-align: center;}

# HTML5-формы

**HTML-формы** являются элементами управления, которые применяются для сбора информации от посетителей веб-сайта.

Веб-формы состоят из набора текстовых полей, кнопок, списков и других элементов управления, которые активизируются щелчком мыши. Технически формы передают данные от пользователя удаленному серверу.

Для получения и обработки данных форм используются языки веб-программирования, такие как **PHP**, **Perl**.

До появления HTML5 веб-формы представляли собой набор нескольких элементов <input type="text">, <input type="password">, завершающихся кнопкой <input type="submit">. Для стилизации форм в разных браузерах приходилось прилагать немало усилий. Кроме того, формы требовали применения JavaScript для проверки введенных данных, а также были лишены специфических типов полей ввода для указания повседневной информации типа дат, адресов электронной почты и URL-адресов.

**HTML5-формы** решили большинство этих распространенных проблем благодаря наличию новых атрибутов, предоставив возможность изменять внешний вид элементов форм за счет **CSS3**.

РИС. 1. УЛУЧШЕННЫЕ ВЕБ-ФОРМЫ С ПОМОЩЬЮ HTML5

## Создание HTML5-формы

* **СОДЕРЖАНИЕ:**
* [1. Элемент <form>](https://html5book.ru/html5-forms/#form)
* [2. Группировка элементов формы <fieldset>](https://html5book.ru/html5-forms/#fieldset)
* [3. Создание полей формы <input>](https://html5book.ru/html5-forms/#input)
* [4. Текстовые поля ввода <textarea>](https://html5book.ru/html5-forms/#textarea)
* [5. Раскрывающийся список <select>](https://html5book.ru/html5-forms/#select)
* [6. Надписи к полям формы <label>](https://html5book.ru/html5-forms/#label)
* [7. Кнопки <button>](https://html5book.ru/html5-forms/#button)
* [8. Флажки и переключатели в формах](https://html5book.ru/html5-forms/#part8)

### 1. Элемент <form>

Основу любой формы составляет элемент <form>...</form>. Он не предусматривает ввод данных, так как является контейнером, удерживая вместе все элементы управления формы – **поля**. Атрибуты этого элемента содержат информацию, общую для всех полей формы, поэтому в одну форму нужно включать поля, объединенные логически.

|  |  |
| --- | --- |
| ТАБЛИЦА 1. АТРИБУТЫ ЭЛЕМЕНТА <FORM> | |
| **Атрибут** | **Значение / описание** |
| accept-charset | Значение атрибута представляет собой разделенный пробелами **список кодировок символов**, которые будут использоваться для отправки формы, например, <form accept-charset="ISO-8859-1">. |
| action | **Обязательный атрибут**, который указывает url обработчика формы на сервере, которому передаются данные. Представляет из себя файл (например, action.php), в котором описано, что нужно делать с данными формы. Если значение атрибута не будет указано, то после перезагрузки страницы элементы формы примут значения по умолчанию. В случае, если вся работа будет выполняться на стороне клиента сценариями JavaScript, то для атрибута action можно указать значение #. Также можно сделать так, чтобы заполненная посетителем форма приходила вам на почту. Для этого нужно внести следующую запись: <form action="mailto:адрес вашей электронной почты" enctype="text/plain"></form> |
| autocomplete | Отвечает за запоминание введенных в текстовое поле значений и автоподстановку их при последующем вводе: on — означает, что поле не защищено, и его значение можно сохранять и извлекать, off — отключает автозаполнение для полей форм. |
| enctype | Используется для указания **MIME**-типа данных, отправляемых вместе с формой, например, enctype="multipart/form-data". Указывается только в случае method="post". application/x-www-form-urlencoded — тип содержимого по умолчанию, указывает на то, что передаваемые данные представляют список URL-кодированных переменных формы. Символы пробела (ASCII 32) будут закодированы как +, а специальный символ, например, такой как ! будет закодирован шестнадцатиричной форме как %21. multipart/form-data — используется для отправки форм, содержащих файлы, не-ASCII данные и бинарные данные, состоит из нескольких частей, каждая из которых представляет содержимое отдельного элемента формы. text/plain — указывает на то, что передается обычный (не html) текст. |
| method | Задает способ передачи данных формы. Метод get передает данные на сервер через адресную строку браузера. При формировании запроса к серверу все переменные и их значения формируют последовательность вида www.anysite.ru/form.php?var1=1&var2=2. Имена и значения переменных присоединяются к адресу сервера после знака ? и разделяются между собой знаком &. Все специальные символы и буквы, отличные от латинских, кодируются в формате %nn, пробел заменяется на +. Этот метод нужно использовать, если вы не передаете больших объемов информации. Если вместе с формой предполагается отправка какого-либо файла, этот метод не подойдет. Метод post применяется для пересылки данных больших объемов, а также конфиденциальной информации и паролей. Данные, отправляемые с помощью этого метода, не видны в заголовке URL, так как они содержатся в теле сообщения. <form action="action.php" enctype="multipart/form-data" method="post"></form> |
| name | Задает **имя формы**, которое будет использоваться для доступа к элементам формы через сценарии, например, name="opros". |
| novalidate | Отключает проверку в кнопке для отправки формы. Атрибут используется без указания значения |
| target | Указывает окно, в которое будет направлена информация: \_blank — новое окно \_self — тот же фрейм \_parent — родительский фрейм (если он существует, если нет — то в текущий) \_top — окно верхнего уровня по отношению к данному фрейму. Если вызов происходит не из дочернего фрейма, то в тот же фрейм. |

### 2. Группировка элементов формы

Элемент <fieldset>...</fieldset> предназначен для группировки элементов, связанных друг с другом, разделяя таким образом форму на логические фрагменты.

Каждой группе элементов можно присвоить название с помощью элемента <legend>, который идет сразу за открывающим тегом элемента <fieldset>. Название группы проявляется слева в верхней границе <fieldset>. Например, если в элементе <fieldset> хранится контактная информация:

<form>

<fieldset>

<legend>Контактная информация</legend>

<p><label for="name">Имя <em>\*</em></label><input type="text" id="name"></p>

<p><label for="email">E-mail</label><input type="email" id="email"></p>

</fieldset>

<p><input type="submit" value="Отправить"></p>

</form>

HTML

РИС. 2. ГРУППИРОВКА ЭЛЕМЕНТОВ ФОРМЫ С ПОМОЩЬЮ <FIELDSET>

|  |  |
| --- | --- |
| ТАБЛИЦА 2. АТРИБУТЫ ЭЛЕМЕНТА <FIELDSET> | |
| **Атрибут** | **Значение / описание** |
| disabled | Если атрибут присутствует, то группа связанных элементов формы, находящихся внутри контейнера <fieldset>, отключены для заполнения и редактирования. Используется для ограничения доступа к некоторым полям формы, содержащих ранее введенные данные. Атрибут используется без указания значения — <fieldset disabled>. |
| form | Значение атрибута должно быть равно атрибуту id элемента <form> в этом же документе. Указывает на одну или несколько форм, к которым принадлежит данная группа элементов. На данный момент атрибут не поддерживается ни одним браузером. |
| name | Определяет **имя**, которое будет использоваться для ссылки на элементы в JavaScript, или для ссылки на данные формы после заполнения и отправки формы. Является аналогом атрибута id. |

### 3. Создание полей формы

Элемент <input> создает большинство полей формы. Атрибуты элемента отличаются в зависимости от типа поля, для создания которого используется этот элемент.

С помощью css-стилей можно изменить размер шрифта, тип шрифта, цвет и другие свойства текста, а также добавить границы, цвет фона и фоновое изображение. Ширина поля задается свойством width.

|  |  |
| --- | --- |
| ТАБЛИЦА 3. АТРИБУТЫ ЭЛЕМЕНТА <INPUT> | |
| **Атрибут** | **Значение / описание** |
| accept | Определяет тип файла, разрешенных для отправки на сервер. Указывается только для <input type="file">. Возможные значения: file\_extension — разрешает загрузку файлов с указанным расширением, например, accept=".gif", accept=".pdf", accept=".doc" audio/\* — разрешает загрузку аудиофайлов video/\* — разрешает загрузку видеофайлов image/\* — разрешает загрузку изображений media\_type — указывает на медиа-тип загружаемых файлов. |
| alt | Определяет **альтернативный текст** для изображений, указывается только для <input type="image">. |
| autocomplete | Отвечает за запоминание введенных в текстовое поле значений и автоподстановку их при последующем вводе: on — означает, что поле не защищено, и его значение можно сохранять и извлекать, off — отключает автозаполнение для полей форм. |
| autofocus | Позволяет сделать так, чтобы в загружаемой форме то или иное поле ввода уже имело фокус (было выбрано), являясь готовым к вводу значения. |
| checked | Атрибут проверяет, установлен ли флажок по умолчанию при загрузке страницы для полей типа type="checkbox" и type="radio". |
| disabled | Отключает возможность редактирования и копирования содержимого поля. |
| form | Значение атрибута должно быть равно атрибуту id элемента <form> в этом же документе. Определяет форму, которой принадлежит данное поле. |
| formaction | Задает url файла, который будет обрабатывать введенные в поля данные при отправке формы. Задается только для полей типа type="submit" и type="image". Атрибут переопределяет значение атрибута action самой формы. |
| formenctype | Определяет, как будут кодироваться данные полей формы при отправке на сервер. Переопределяет значение атрибута enctype формы. Задается только для полей типа type="submit" и type="image". Варианты: application/­x-www-form-urlencoded — значение по умолчанию. Все символы кодируются перед отправкой (пробелы заменяются на символ +, специальные символы преобразуются в значения ASCII HEX) multipart/form-data — символы не кодируются text/plain — пробелы заменяются на символ +, а специальные символы не кодируются. |
| formmethod | Атрибут определяет метод, который браузер будет использовать для отправки данных формы на сервер. Задается только для полей типа type="submit" и type="image". Переопределяет значение атрибута method формы. Варианты: get — значение по умолчанию. Данные из формы (пара имя/значение) добавляются в url-адрес и отправляются на сервер: URL?имя=значение&имя=значение post — данные формы отправляются в виде http-запроса. |
| formnovalidate | Определяет, что данные полей формы не должны проверяться при отправке формы. Переопределяет значение атрибута novalidate формы. Можно использовать без указания значения атрибута. |
| formtarget | Определяет, где выводить ответ, полученный после отправки формы. Задается только для полей типа type="submit" и type="image". Переопределяет значение атрибута target формы. \_blank — загружает ответ в новое окно/вкладку \_self — загружает ответ в то же окно (значение по умолчанию) \_parent – загружает ответ в родительский фрейм \_top – загружает ответ во весь экран framename – загружает ответ во фрейм с указанным именем. |
| height | Значение атрибута содержит количество пикселей без указания единицы измерения. Устанавливает высоту поля формы типа type="image", например, <input type="image" src="img\_submit.gif" height="50">. Рекомендуется одновременно устанавливать как высоту, так и ширину поля. |
| list | Является ссылкой на элемент <datalist>, содержит его id.Позволяет предоставить пользователю несколько вариантов на выбор, когда он начинает вводить значение в соответствующем поле. |
| max | Позволяет ограничить допустимый ввод числовых данных максимальным значением, значение атрибута может содержать целое или дробное число. Рекомендуется использовать этот атрибут вместе с атрибутом min. Работает со следующими типами полей: number, range, date, datetime, datetime-local, month, time и week. |
| maxlength | Атрибут задает максимальное количество символов, вводимых в поле. Значение по умолчанию 524288 символов. |
| min | Позволяет ограничить допустимый ввод числовых данных минимальным значением. |
| multiple | Позволяет пользователю ввести несколько значений атрибутов, разделяя их запятой. Применяется в отношении файлов и адресов электронной почты. Указывается без значения атрибута. |
| name | Определяет имя, которое будет использоваться для доступа к элементу <form>, к примеру, в таблицах стилей css. Является аналогом атрибута id. |
| pattern | Позволяет определять с помощью **регулярного выражения** синтаксис данных, ввод которых должен быть разрешен в определенном поле. Например, pattern="[a-z]{3}-[0-9]{3}" — квадратные скобки устанавливают диапазон допустимых символов, в данном случае — любые строчные буквы, число в фигурных скобках указывает, что нужны три строчные буквы, после которых следует тире, далее — три цифры в диапазоне от 0 до 9. |
| placeholder | Содержит текст, который отображается в поле ввода до заполнения (чаще всего это подсказка). |
| readonly | Не позволяет пользователю изменять значения элементов формы, выделение и копирование текста при этом доступно. Указывается без значения атрибута. |
| required | Выводит сообщение о том, что данное поле является обязательным для заполнения. Если пользователь попытается отправить форму, не введя в это поле требуемое значение, то на экране отобразится предупреждающее сообщение. Указывается без значения атрибута. |
| size | Задает видимую ширину поля в символах. Значение по умолчанию — 20. Работает со следующими типами полей: text, search, tel, url, email и password. |
| src | Задает url изображения, используемого в качестве кнопки отправки данных формы. Указывается только для поля <input type="image">. |
| step | Используется для элементов, предполагающих ввод числовых значений, указывает величину увеличения или уменьшения значений в процессе регулировки диапазона (шаг). |
| type | button — создает кнопку. |
| checkbox — превращает поле ввода во флажок, который можно установить или очистить, например,  У меня есть автомобиль |
| color — генерирует палитры цветов в поддерживающих браузерах, давая пользователям возможность выбирать значения цветов в шестнадцатеричном формате. |
| date — позволяет вводить дату в формате дд.мм.гггг. День рождения: |
| datetime-local — позволяет вводить дату и время, разделенные прописной английской буквой Т по шаблону дд.мм.гггг чч:мм. День рождения — день и время: |
| email — браузеры, поддерживающие данный атрибут, будут ожидать, что пользователь введет данные, соответствующие синтаксису адресов электронной почты. E-mail: |
| file — позволяет загружать файлы с компьютера пользователя. Выберите файл: |
| hidden — скрывает элемент управления, который не отображается браузером и не дает пользователю изменять значения по умолчанию. |
| image — создает кнопку, позволяя вместо текста на кнопке вставить изображение. |
| month — позволяет пользователю вводить год и номер месяца по шаблону гггг-мм. |
| number — предназначено для ввода целочисленных значений. Его атрибуты min, max и step задают верхний, нижний пределы и шаг между значениями соответственно. Эти атрибуты предполагаются у всех элементов, имеющих численные показатели. Их значения по умолчанию зависят от типа элемента. Укажите количество (от 1 до 5): |
| password — создает текстовые поля в форме, при этом вводимые пользователем символы заменяются на звездочки, маркеры, либо другие, установленные браузером значки. Введите пароль: |
| radio — создает переключатель — элемент управления в виде небольшого кружка, который можно включить или выключить. Вегетарианец: |
| range — позволит создать такой элемент интерфейса, как ползунок, min / max — позволят установить диапазон выбора |
| reset — создает кнопку, которая очищает поля формы от введенных пользователем данных. |
| search — обозначает поле поиска, по умолчанию поле ввода имеет прямоугольную форму. Поиск: |
| submit — создает стандартную кнопку, активизируемую щелчком мыши. Кнопка собирает информацию с формы и отправляет ее для обработки. |
| text — создает текстовые поля в форме, выводя однострочное текстовое поле для ввода текста. |
| time — позволяет вводить время в 24-часовом формате по шаблону чч:мм. В поддерживающих браузерах оно отображается как элемент управления в виде числового поля ввода со значением, изменяемым с помощью мыши, и допускает ввод только значений времени. Укажите время: |
| url — поле предназначено для указания URL-адресов. Главная страница: |
| week — соответствующий инструмент-указатель позволяет пользователю выбрать одну неделю в году, после чего обеспечит ввод данных в формате нн-гггг. В зависимости от года число недель может быть 52 или 53. Укажите неделю: |
| value | Определяет текст, отображаемый на кнопке, в поле или связанный текст. Не указывается для полей типа file. |
| width | Значение атрибута содержит количество пикселей. Позволяет задать ширину полей формы. |

### 4. Текстовые поля ввода

Элемент <textarea>...</textarea> используется вместо элемента <input type="text">, когда нужно создать большие текстовые поля. Текст, отображаемый как исходное значение, помещается внутрь.

Размеры поля устанавливаются при помощи атрибутов cols – размеры по горизонтали, rows – размеры по вертикали. Высоту поля можно задать свойством height. Все размеры считаются исходя из размера одного символа моноширинного шрифта.

|  |  |
| --- | --- |
| ТАБЛИЦА 4. АТРИБУТЫ ЭЛЕМЕНТА <TEXTAREA> | |
| **Атрибут** | **Значение / описание** |
| autofocus | Устанавливает фокус на нужном начальном текстовом поле автоматически. |
| cols | Устанавливает ширину через количество символов. Если пользователь вводит больше текста, появляется полоса прокрутки. |
| disabled | Отключает возможность редактирования и копирования содержимого поля. |
| form | Значение атрибута должно быть равно значению атрибута id элемента <form> в этом же документе. Определяет одну или несколько форм, которым принадлежит данное текстовое поле. |
| maxlength | Значение атрибута задает максимальное число символов для ввода в поле. |
| name | Задает имя текстового поля. |
| placeholder | Определяет короткую текстовую подсказку, которая описывает ожидаемое вводимое значение. |
| readonly | Отключает возможность редактирования содержимого поля. |
| required | Выводит сообщение о том, что данное поле является обязательным для заполнения. |
| rows | Указывает число, которое означает, сколько строк должно отображаться в текстовой области. |
| wrap | Определяет, должен ли текст сохранять переносы строк при отправке формы. Значение hard сохраняет перенос, а значение soft не сохраняет. Если используется значение hard, то должно указываться значение атрибута cols. |

### 5. Раскрывающийся список

Списки дают возможность расположить большое количество пунктов компактно. Раскрывающиеся списки создаются при помощи элемента <select>...</select>. Они позволяют выбрать одно или несколько значений из предложенного множества. По умолчанию в поле списка отображается его первый элемент.

Для добавления в список пунктов используются элементы <option>...</option>, которые располагаются внутри <select>.

Для систематизации списков применяется элемент <optgroup>...</optgroup>, который создает заголовки в списках.

Для списков возможно изменить размер шрифта, тип шрифта, цвет и другие свойства текста, а также добавить границы, цвет фона и фоновое изображение.

|  |  |
| --- | --- |
| ТАБЛИЦА 5. АТРИБУТЫ ЭЛЕМЕНТА <SELECT> | |
| **Атрибут** | **Значение / описание** |
| autofocus | Устанавливает автоматический фокус на элементе при загрузке страницы. |
| disabled | Отключает раскрывающийся список. |
| form | Определяет форму, которой принадлежит данный список. В качестве значения атрибута указывается идентификатор формы. |
| multiple | Дает возможность выбора одного или нескольких пунктов, для этого при выборе нужно нажать и удерживать нажатой клавишу Ctrl. |
| name | Определяет имя для выпадающего списка. Значение атрибута содержит название, отражающее тематику списка. |
| required | Выводит сообщение о том, что пользователь должен выбрать значение из раскрывающегося списка перед отправкой формы. |
| size | Задает количество одновременно видимых на экране элементов списка. Если количество элементов списка превышает установленное количество, появляется полоса прокрутки. Значение атрибута задается целым положительным числом. |

|  |  |
| --- | --- |
| ТАБЛИЦА 6. АТРИБУТЫ ЭЛЕМЕНТА <OPTION> | |
| **Атрибут** | **Значение / описание** |
| disabled | Делает недоступным для выбора элемент списка. |
| label | Задает укороченную версию для элемента, которая будет отражаться в выпадающем списке. Значение атрибута содержит текст, описывающий соответствующий пункт выпадающего списка. |
| selected | Отображает выбранный элемент списка по умолчанию при загрузке страницы браузером. |
| value | Указывает значение, которое будет отправлено на сервер при отправке формы. |

|  |  |
| --- | --- |
| ТАБЛИЦА 7. АТРИБУТЫ ЭЛЕМЕНТА <OPTGROUP> | |
| **Атрибут** | **Значение / описание** |
| disabled | Отключает данную группу элементов списка для выбора. |
| label | Задает заголовок для группы элементов выпадающего списка. Значение атрибута содержит текст, недоступный для выбора, который будет располагаться над соответствующими пунктами списка. Текст выделяется в браузере жирным начертанием. |

### 6. Надписи к полям формы

Надписи к элементам формы создаются с помощью элемента <label>...</label>. Существует два способа группировки надписи и поля. Если поле находится внутри элемента <label>, то атрибут for указывать не нужно.

<!-- с указанием атрибута for -->

<label for="comments">Когда вы последний раз летали на самолете?</label>

<textarea id="comments"></textarea>

<!-- без атрибута for -->

<p><label>Кошка<input id="cat" type="checkbox"></label></p>

HTML

|  |  |
| --- | --- |
| ТАБЛИЦА 8. АТРИБУТЫ ЭЛЕМЕНТА <LABEL> | |
| **Атрибут** | **Значение / описание** |
| for | Определяет, к какому полю формы привязан данный элемент. Можно создавать поясняющие надписи к следующим элементам формы: <input>, <textarea>, <select>. Значение атрибута содержит идентификатор поля формы. |

### 7. Кнопки

Элемент <button>...</button> создает кликабельные кнопки. В отличие от кнопок, созданных <input> (<input type="submit"></input>, <input type="image">, <input type="reset">, <input type="button">), внутрь элемента <button> можно поместить контент — текст или изображение.

Для корректного отображения элемента <button> разными браузерами нужно указывать атрибут type, например, <button type="submit"></button>.

Кнопки позволяют пользователям передавать данные в форму, очищать содержимое формы или предпринимать какие-либо другие действия. Можно создавать границы, изменять фон и выравнивать текст на кнопке.

|  |  |
| --- | --- |
| ТАБЛИЦА 9. АТРИБУТЫ ЭЛЕМЕНТА <BUTTON> | |
| **Атрибут** | **Значение / описание** |
| autofocus | Устанавливает фокус на кнопке при загрузке страницы. |
| disabled | Отключает кнопку, делая ее некликабельной. |
| form | Указывает на одну или несколько форм, которым принадлежит данная кнопка. Значение атрибута — идентификатор соответствующей формы. |
| formaction | Значение атрибута содержит url-адрес обработчика данных формы, отправляемых при нажатии на кнопку. Только для кнопки типа type="submit". Переопределяет значение атрибута action, указанного для элемента <form>. |
| formenctype | Задает тип кодировки данных формы перед отправкой на сервер при нажатии на кнопки типа type="submit". Переопределяет значение атрибута enctype, указанного для элемента <form>. Возможные значения: application/x-www-form-urlencoded — значение по умолчанию. Все символы перед отправкой будут закодированы. multipart/form-data — символы не кодируются. Используется в случае, когда с помощью формы загружаются файлы. text/plain — символы не кодируются, а пробелы заменяются на символ +. |
| formmethod | Атрибут определяет метод, который браузер будет использовать для отправки формы. Переопределяет значение атрибута method, указанного для элемента <form>. Указывается только для кнопок типа type="submit". Возможные значения: get — данные из формы (пара имя/значение) добавляются в url-адрес и отправляются на сервер. Данный способ имеет ограничения по размеру отправляемых данных и не подходит для отправки паролей и конфиденциальной информации. post — данные из формы добавляются в виде http-запроса. Метод является более надежным и безопасным, чем get и не имеет ограничений по размеру. |
| formnovalidate | Атрибут задает, что данные формы не должны проверяться при отправке. Указывается только для кнопок типа type="submit". |
| formtarget | Атрибут задает, в каком окне выводить результат после отправки формы. Указывается только для кнопок типа type="submit". Переопределяет значение атрибута target, указанного для элемента <form>. \_blank — загружает ответ в новое окно/вкладку \_self — загружает ответ в то же окно (значение по умолчанию) \_parent — загружает ответ в родительский фрейм \_top — загружает ответ во весь экран framename — загружает ответ во фрейм с указанным именем. |
| name | Задает имя кнопки, значение атрибута — текст. Используется для ссылки на данные формы, после того как форма была отправлена, или для ссылки на данную кнопку (кнопки) в JavaScript. |
| type | Определяет тип кнопки. Возможные значения: button — кликабельная кнопка reset — кнопка сброса, возвращает первоначальное значение submit — кнопка для отправки данных формы. |
| value | Задает значение по умолчанию, отправляемое при нажатии на кнопку. |

### 8. Флажки и переключатели в формах

Флажки в формах задаются с помощью конструкции <input type="checkbox">, а переключатель — при помощи <input type="radio">.

Флажков, в отличие от переключателей, в одной форме может быть установлено несколько. Если для флажков указан атрибут checked, то при загрузке станицы на соответствующих полях формы флажки уже будут установлены.

Элемент <label> применяется при реализации выбора с помощью переключателей и флажков. Можно выбрать нужный пункт, просто щелкая кнопкой мыши на тексте, связанном с ним. Для этого нужно поместить <input> внутрь элемента <label>.