**История создания HTML**

Язык HTML (Hyper Text Markup Language) – язык разметки гипертекста, предназначенный для разметки и оформления документов в Интернете. HTML был разработан ученым Европейского совета по ядерным исследованиям, Тимоти Джоном Бернерс-Ли, когда в 1991 году Европейский институт физики частиц ощутил потребность в механизме передачи гипертекстовой информации через Интернет. Тогда они выбрали в качестве основы язык SGML (который определял то, как должны выглядеть элементы разметки), как набор правил, на основе которых можно строить любые языки разметки, и на его базе был создан язык HTML (какие именно должны быть элементы и как они должны интерпретироваться браузерами.)

То-есть появление первых версий HTML можно отнести к 1986 году, а в 1991 году HTML был доработан и стал использоваться именно для передачи гипертекста по интернету.

Первая версия HTML создавалась для целей форматирования научных документов, а именно для структурного форматирования без элементов описания цветовых схем, параметров шрифта и тому подобных характеристик. Таким образом, изначально HTML позволял выделять в тексте заголовки, абзацы, списки и структурные элементы. В Результате обработки или воспроизведения, HTML не должен был зависеть от технических особенностей аппаратных средств его визуализации, поскольку не содержал в себе параметров этой визуализации.

**Структура документа HTML**

**HTML** - это язык тэгов.

**Тэг (дескриптор)** - специальный код, вставляемый в текст, для форматирования элементов HTML-документа. Это основной элемент кодирования, принятый в стандарте HTML. Тэг заключаются в угловые скобки <>. Существует два типа тэгов: контейнеры и одиночные тэги.

**Контейнер** - дескрипторная пара, состоящая из начального тега <TAG> (открывающего) и конечного тега </TAG>. (закрывающего). Контейнеры предназначены для хранения некоторой информации, например текста или других HTML - тэгов.

**Пустой дескриптор** – отличается от контейнера тем, что не содержит никакой информации. У него есть только начальный тэг. Пустой дескриптор обычно выполняет самостоятельную задачу, не связанную с конкретным текстом. Например, тэг <HR> создаёт горизонтальную линию.

**Элемент HTML - документа** - это начальный и конечный тэги контейнера вместе с заключённым между ними содержимым.

Самым главным из тэгов HTML является одноименный тэг <HTML>. Он должен всегда открывать документ, также как закрывающий тэг </HTML> должен обязательно стоять в последней его строке. Эти тэги означают, что находящиеся между ними строки представляют собой единый HTML документ. Это важно, так как сам по себе документ является обычным текстовым файлом ASCII. Без этих тэгов браузер или другая программа просмотра не в состоянии идентифицировать формат документа и правильно его интерпретировать.

Элементы, находящиеся внутри раздела **HEAD** (кроме названия документа, записываемого в разделе TITLE), не видны на экране. Элементы, содержащиеся внутри раздела HEAD документа, нужны для того, чтобы:

* Дать документу название
* Определить отношения между несколькими документами
* Дать указание браузеру создать форму для поиска
* Определить метод посылки специальных сообщений определенному браузеру или другой программе просмотра

Раздел HEAD открывается тэгом <HEAD>. Обычно этот тэг следует сразу же за тэгом <HTML>. Закрывающий тэг </HEAD>показывает конец этого раздела. Между упомянутыми тэгами располагаются остальные тэги раздела заголовка документа.

Раздел **TITLE** является единственным обязательным элементом заголовка документа и служит для того, чтобы дать документу название. Оно обычно показывается в заголовке окна браузера.

Название документа записывается между тэгами <TITLE> и </TITLE> и представляет собой текстовую строку.

Структура HTML по большому счету лежит в основе каждой web страницы. Правильная начальная структура HTML является залогом кросс-браузерности сайта (способность сайта выглядеть идентично во всех популярных браузерах).

**Структура документа:**

HTML документ должен содержать некоторые обязательные элементы, без которых он не будет правильно интерпретирован. Пример простейшего шаблона приведен в листинге.

<HTML>

<HEAD>      
  <TITLE>Заголовок</TITLE>  
      </HEAD>  
 <BODY>  
        <p>

<b>

Здесь вводится текст тела,

этот текст будет полужирным,

<i>а этот - ещё и курсивным</i>

</b>

</p>  
    </BODY>  
</HTML>

Шаблон открывается тэгом <HTML>, который, необходим для каждого HTML документа. Следующим идет тэг <HEAD>,начинающий заголовок документа. Заголовок содержит элемент <TITLE>, вводящий название документа. Заголовок закрывается тэгом </HEAD>. Далее идет тэг <BODY>, после которого помещается текст тела документа. Тэг </BODY> означает конец тела, а тэг </HTML> - конец всего документа.

**Версии HTML**

HTML версии 1.2 содержал около 40 тэгов и не подразумевал какого-либо описания физического представления документов. Все было приведено к логической и структурной разметке текста.

В 1994 году был создан консорциум W3C (Ц3С), который унаследовал право главенствовать в мире Интернета от Европейского института физики частиц. Эта организация сразу же принялась за создание следующей спецификации HTML версии 2.0. но окончательный стандарт HTML 2.0 был принят только в 1995 году, когда уже обсуждали и разрабатывали HTML 3.0. Единственным существенным усовершенствованием HTML 2.0 было введение в язык форм - средств отправки информации от пользователя на сервер. Считается, что HTML версии 3.0 - это самый большой прорыв в HTML-технологиях. Первоначальный вариант стандарта включал в себя много интересных нововведений: теги для создания таблиц, разметки математических формул, вставки обтекаемых текстом рисунков, примечаний и др. Но самое главное, в новом стандарте впервые была предпринята попытка найти компромисс между суровыми требованиями идеологии структурной разметки и таким простым и понятным желанием пользователей "сделать красиво".

В 1996 году Microsoft взял под свою опеку W3C. В сжатые сроки был создан стандарт HTML версии 3.2, который был полностью ориентирован на Internet Explorer, так как Microsoft перестал быть просто наблюдателем на рынке браузеров, и создала свой браузер версии 2.0 и позднее 3.0.

HTML 3.2 считался стандартом этого развивающегося языка WEB-строительства. Эта версия HTML навела порядок в плане поддержки элементов разметки всеми браузерами.

В HTML 4.0 вводятся механизмы таблиц стилей, скриптов, кадров, внедрения объектов, улучшенная поддержка разных направлений письма и направления справа налево, таблицы с большим количеством возможностей и новые свойства форм, обеспечивая лучшие возможности доступа для людей с физическими недостатками.

В последние годы (2004 год) была принята HTML версии 4.01. Она также обеспечивает достаточно высокую кросс - браузерность и кросс - платформенность.

Далее появляется язык XHTML - это основанный на [XML](https://htmlweb.ru/xml/xml1.php) язык разметки гипертекста, максимально приближенный к текущим стандартам [HTML](https://htmlweb.ru/html/bhtml.php). XHTML отличается от HTML строгостью написания кода. Если HTML позволял писать практически любые конструкции и браузер их корректно распознавал, то теперь, с появлением XHTML, это стало невозможным, так как XHTML требует строгого соблюдения всех правил, предъявляемых консорциумом W3C. Строгие требования к оформлению XHTML-кода позволяют избежать многих ошибок ещё на стадии написания и отладки.

XHTML — это тот же HTML с точки зрения изучения его функциональных возможностей, никакой разницы, это просто стандарт, который заставляет не допускать ошибок при создании HTML документов и запрещает некоторые необязательные конструкции в HTML.

XHTML - это новый язык, который пришёл на смену старому HTML. В итоге все браузеры, как предполагается, перейдут на XHTML, сохраняя совместимость со старым HTML

Элемент <!DOCTYPE> предназначен для указания типа текущего документа — DTD (описание типа документа). Это необходимо, чтобы браузер понимал, как следует интерпретировать текущую веб-страницу, поскольку HTML существует в нескольких версиях, кроме того, имеется XHTML, похожий на HTML, но различающийся с ним по синтаксису. Чтобы браузер не путался и понимал, согласно какому стандарту отображать веб-страницу, необходимо в первой строке кода задавать <!DOCTYPE>.

Существует несколько видов <!DOCTYPE>, они различаются в зависимости от версии языка, на которого ориентированы.

Пример:

<!DOCTYPE[Элемент верхнего уровня] [Публичность] "[Регистрация]//[Организация]//[Тип] [Имя]//[Язык]" "[URL]">

**Формы**

Формы дают возможность пользователям вводить информацию, например тесты, опросы, голосования. Для того, чтобы все это сделать на своих web-страницах и нужны формы.

HTML формы сами по себе только позволяют вводить информацию, а вот обрабатывать ее HTML не умеет, так как это язык разметки, а не язык программирования программирования. Для обработки самой информации используются такие языки, как javascript, php и другие.

В HTML форма задается тегами <form></form>. Все остальные элементы формы располагаются между этими тегами.

У тега <form> есть несколько параметров:

* name - имя формы. Необходимо, если на странице несколько форм;
* action - определяет URL-адрес, по которому будет отправлена информация введенная пользователем;
* method - определяет способ отправки информации;
* target - указывает имя окна, в котором будут отображаться результаты обработки отправленной формы.
* Значения этих параметром нужны для языков обработки данных.

<form name=“forma1”></form>

**Текстовое поле –** простое однострочное текстовое поле в котором можно вводить и редактировать текст. Задается тегом <input>.

<form name=”forma1”>

<input type="text" name="text1" size="20"

Maxlength="50" value="Введите текст"><br>

</form>

Параметры:

* name - имя элемента,
* type - тип элемента (в данном случае - text),
* size - размер текстового поля в символах, которые одновременно будут видны, при вводе большего количества символов, они будут прокручиваться,
* maxlength - максимальное количество символов, которое можно ввести в поле, если опустить этот параметр, то число символов будет неограниченным,
* value - текст, который будет отображаться (его можно стереть), при отсутствии этого параметра поле будет пустым.

Возможны еще два параметра:

* disabled - блокирует поле от любых изменений,
* readonly - делает поле доступным только для чтения.

**Флажки**

Элемент флажков задается тем же тегом <input>, причем один тег задает один флажок.

<form name=”forma1”>

Какими языками вы владеете:

<input name="lan1" type="checkbox" value=”english”>английский

<input name="lan2" type="checkbox" value=”german”>немецкий

</form>

Параметры:

* type - тип элемента (в данном случае - checkbox),
* name - имя элемента, указывает программе обработчику формы, какой пункт выбрал пользователь,
* value - значение элемента, указывает программе обработчику формы значение пункта, который выбрал пользователь. В нашем примере выбран пункт английский, следовательно, программа-обработчик получит: lan1="english",
* checked - им обычно помечают наиболее вероятные для выбора пункты, пользователь щелчком мыши может выбрать другие пункты.

**Переключатели**

В отличии от флажков, можно выбрать только один пункт. В связи с этим значения параметра name должны быть одинаковы для всех элементов группы. Параметр type="radio", все остальные такие же, как у флажков.

<form name=”forma1”>

Укажите ваш пол:<br>

<input name="mycolor" type="radio" value=”man”> мужской

` <input name="mycolor" type="radio" value=”woman”>женский

</form>

**Кнопки**

Существует четыре вида кнопок:

submit - кнопка отправки содержимого формы web-серверу. Ее параметры:

* type="submit" - тип кнопки,
* name - имя кнопки,
* value - надпись на кнопке.

image - графическая кнопка отправки содержимого формы web-серверу. Для ее использования необходимо подготовить картинку кнопки, а потом использовать ее в виде кнопки. Ее параметры:

* type="image" - тип графической кнопки,
* name - имя кнопки,
* src - адрес картинки для кнопки.

reset - кнопка, позволяющая восстановить все значения по умолчанию в форме. Ее параметры:

* type="reset" - тип кнопки очищения, name - имя кнопки,
* value - надпись на кнопке.

button - произвольная кнопка, ее действия назначаются вами, т.е. сама она делать ничего не умеет. Ее параметры:

* type="button" - тип произвольной кнопки,
* name - имя кнопки,
* value - надпись на кнопке.
* onclick - указывает, что делать при щелчке по кнопке. Вообще, у этого типа кнопок есть и другие события (например, двойной щелчок), но здесь мы не будем их рассматривать.

Если на форме несколько кнопок, то они должны иметь разные названия.

<from name=”forma1”>

<input type="submit" name="submit" value="Отправить">

<input type="image" name="image" src="http://www.site- do.ru/images/but.gif">

<input type="reset" name="reset" value="Очистить форму">

</form>

Кнопки можно задавать и по другому, при помощи тегов </button></botton>. Возможности у таких кнопок несколько шире, они могут иметь содержимое в виде текста или картинки.

**Многострочное текстовое поле**

Для объемных текстов, например для почтовых сообщений, удобно использовать именно этот элемент. Он создается тегами <textarea></textarea> и имеет следующие параметры:

name - имя поля,

cols - ширина поля в символах,

rows - количество строк текста, видимых на экране,

wrap - способ переноса слов:

* off - переноса не происходит,
* virtual - перенос отображается, но на сервер поступает неделимая строка,
* physical - перенос и на экране и при поступлении на сервер.

disabled - неактивное поле,

readonly - разрешено только чтение.

<form name=”forma1>

<textarea cols=”20” rows=”3” wrap=”off”> </textarea>

</form>

**Раскрывающиеся списки**

Списки бывают с возможностью выбора одного элемента и с множественным выбором. Задаются и те, и другие с помощью тегов <select></select>, внутри которых располагаются элементы значений, заданных тегом:

<select>:

* name - имя списка. Каждый выбранный элемент списка при передаче на сервер будет иметь вид: name.value, где значение (value) берется из тега option.
* size - определяет количество видимых элементов в списке: 1 - простой раскрывающийся список, больше 1 - список с полосой прокрутки.
* multiple - разрешает выбор нескольких элементов списка.

<option>:

* selected - им помечают наиболее вероятный для выбора элемент списка, если список со множественным выбором, то можно пометить несколько пунктов.
* value - значение, которое будет отправлено серверу, если пункт выбран.

<form name=”forma1”>

Какой язык вы хотите изучать:

<select name=”language” size=”1”>

<option selected value=”html”>html

<option value=”php”>php

<option value=”java”>java

</select><br>

</form>

Существуют еще теги <optgroup></optgroup> , позволяющие группировать элементы списка по каким-либо признакам. Например, мы хотим задать каталог сайтов в виде списка, тогда удобнее разбить его на группы по интересам.

**Кнопка для загрузки файлов**

Служит для реализации загрузки файлов на сервер. Объект browse начитается с парных тегов *<form></form>*. Начинающий тэг *<form>* содержит необходимый атрибут encrypt. Атрибут *encrypt* принимает значение *multipart/form-data*, который извещает сервер о том, что вместе с обычной информацией посылается и файл. При создании текстового поля также необходимо указать тип файла – “*file*”.

<form name=”forma1” enctype="multipart/form-data" action="upload.php" method="post">

Загрузить файл:

<input name="my\_file" type="file">

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

</form>