Язык разметки гипертекста HTML

Язык разметки гипер текстов HTML(HyperText Markup Language), разработан Тимом Бернерсом-Ли на основе стандарта языка разметки печатных документов — **SGML** (**Standard Generalised Markup Language**, стандартный обобщенный язык разметки). Дэниел В. Конноли написал для него DDL(**Document Type Definition** — формальное описание синтаксиса HTML в терминах SGML).

Язык **HTML** позволяет размечать электронный документ, который отображается на экране с полиграфическим уровнем оформления; результирующий документ может содержать разнообразные метки, иллюстрации, аудио- и видеофрагменты и так далее.

Октябрь 1991

HTML Tags - неофициальный документ CERN, перечисляющий двенадцать HTML тегов, были сначала упомянуты публично. Ноябрь 1992.

Июль 1993

HTML был предложен IETF(Internet Engineering Task Force) как Интернет-проект (предложение о стандарте).

Ноябрь 1993

HTML + был предложен IETF как Интернет-проект.

Ноябрь 1995

HTML 2.0 был издан как IETF RFC 1866(RFC - Request for Comments) — документ из серии пронумерованных информационных документов Интернета, содержащих технические спецификации и Стандарты, широко применяемые во Всемирной сети). Дополнительные RFC добавили следующие возможности:

- RFC 1867 - загрузка файлов из форм (Ноябрь 1995);
- RFC 1942 - таблицы (Май 1996);
- RFC 1980 - ImageMap карты изображений (Август 1996);
- RFC 2070 - интернационализация(Январь 1997).

Январь 1997

HTML 3.2 был издан как Рекомендация W3C (World Wide Web Consortium — организация, разрабатывающая и внедряющая технологические стандарты для Всемирной Паутины). Это была первая версия стандарта, реализованная W3C (IETF закрыл Рабочую группу HTML в сентябре 1997).

Декабрь 1997

HTML 4.0 был издан как Рекомендация W3C. Предлагалось использование трех возможных вариантов стандарта:

Элементы разметки HTML задаются с помощью специальных **тегов** разметки. Все теги HTML начинаются с "**<**" (левой угловой скобки) и заканчиваются символом "**>**" (правой угловой скобки).

HTML-теги могут быть условно разделены на две категории:

- теги, определяющие, как будет отображаться WEB-браузером содержимое документа;
- теги, описывающие общие свойства документа, такие как заголовок или автор документа;

Как правило, существует стартовый тег и завершающий тег.

Например: *<TITLE> и </TITLE>*

HTML не реагирует на регистр символов, описывающих тег.

Дополнительные пробелы, символы табуляции и возврата каретки, добавленные в исходный текст HTML-документа для его лучшей читаемости, будут проигнорированы WEB-браузером при интерпретации документа.

Многие из открывающих HTML-тегов могут содержать дополнительные параметры, называемые атрибутами. Пары *атрибут=значение* помещаются перед закрывающей скобкой ">" начального тэга элемента. В начальном тэге элемента может быть любое число (допустимых) пар *атрибут=значение*, разделенных пробелами.

Например:

<P align=left >

Как правило, существует стартовый тег и завершающий тег.

Например: <TITLE> и </TITLE>

Структура HTML-документа:

Каждый документ *HTML* должен начинаться со строки типа:

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

DOCTYPE задает тип корневого элемента документа (<html>), *публичный идентификатор* и *системный идентификатор*.

Публичный идентификатор (-//W3C//DTD HTML 4.01//EN) показывает какой тип документа используется(W3C DTD), непосредственное название DTD (DTD HTML 4.01); и язык на котором написана DTD (EN, т.е. английский).

Системный идентификатор (www.w3.org/TR/html4/strict.dtd) — это путь к используемой DTD.

Варианты DOCTYPE для **HTML 4.01**:

Строгий (**Strict**): не содержит элементов, помеченных как «устаревшие» или «не одобряемые» (deprecated).

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

Переходный (**Transitional**): содержит устаревшие теги в целях совместимости и упрощения перехода со старых версий HTML.

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

С фреймами (**Frameset**): аналогичен переходному, но содержит также теги для создания наборов фреймов

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

DOCTYPE для **HTML5**:

<!DOCTYPE html>

Самый первый *тег*, который встречается в документе, должен быть тегом <HTML>.

Тег заголовочной части документа <HEAD> должен быть использован сразу после тега <HTML> и более нигде в теле документа. Данный тег содержит общее описание документа. За ним идет тело документа – <BODY>.

```
<HTML>

  <HEAD>
    «Заголовочная часть HTML-документа»
  </HEAD>

 <BODY>
    «Тело HTML-документа»
 </BODY>

</HTML>
```

Рассмотрим элементы разметки, включаемые в раздел *HEAD*:

TITLE (заглавие документа);

BASE (база URL);

META (метаинформация);

LINK (общие ссылки);

STYLE (описатели стилей);

SCRIPT (скрипты).

Синтаксис контейнера ***TITLE*** в общем виде выглядит следующим образом:

<TITLE>название документа</TITLE>

Элемент разметки **BASE** служит для определения базового URL для гипертекстовых ссылок документа, заданных в неполной (частичной) форме.

<BASE HREF=http://is.sevgu.sebastopol.ru/>

META содержит управляющую информацию, которую браузер использует для правильного отображения и обработки содержания тела документа.

<META HTTP-EQUIV="Content-type" CONTENT="text/html; CHARSET=windows-1251">

Для перезагрузки страницы:

<META HTTP-EQUIV="Refresh" CONTENT="1;URL=refresh.htm">

Для запрета кэширования достаточно вставить в *заголовок* *META*-тег вида:

<META HTTP-EQUIV="Pragma" CONTENT="no-cache"> (HTTP1.0)

<META HTTP-EQUIV="Cache-Control" CONTENT="no-cache"> (HTTP1.1)

LINK позволяет загружать внешние описатели стилей :

<LINK REL=stylesheet href="../css/css.htm" TYPE="text/css">

Атрибут **REL** определяет тип гипертекстовой связи, **HREF** (Hypertext REFerence) указывает адрес документа, идентифицирующего связь, а атрибут **TYPE** определяет тип содержания этого документа

Элемент разметки ***STYLE*** предназначен для размещения описателей стилей:
<STYLE TYPE=тип_описания_стилей> описание стиля/стилей </STYLE>

Элемент разметки ***SCRIPT*** служит для размещения кода JavaScript, VBScript или JScript:
<SCRIPT [TYPE=тип_языка_программирования] [SRC=URL]>
JavaScript/VBScript-код </SCRIPT>
<SCRIPT LANGUAGE="JavaScript" SRC="script.js">

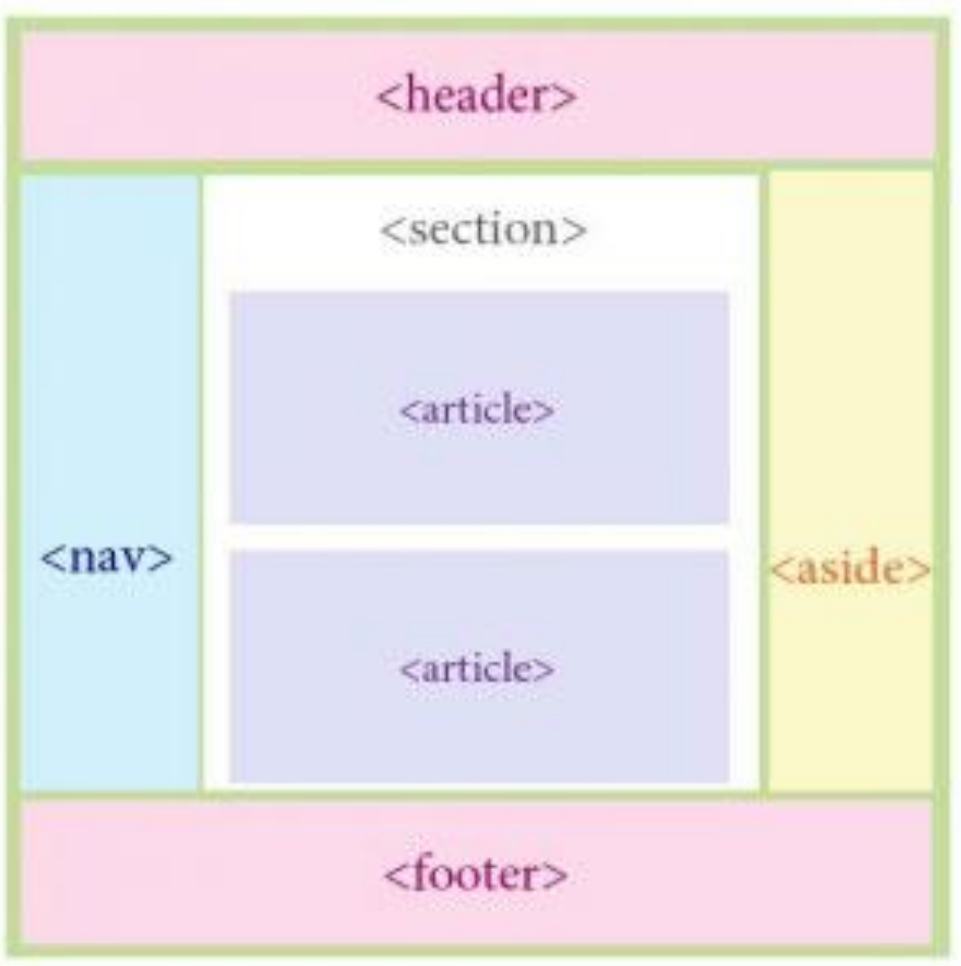
Пример заголовочной части документа:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html >
  <head>
<meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
<title>The W3C Markup Validation Service</title>
<link rel="shortcut icon" href="http://www.w3.org/2008/site/images/favicon.ico" type="image/x-icon" />
<style type="text/css" media="all“>      @import "./style/base.css“; </style>
<script type="text/javascript" src="scripts/w3c-validator.js"></script>
<meta name="keywords" content="HTML, HyperText Markup Language, Validation,       W3C Markup Validation Service" />
<meta name="description" content="W3C's easy-to-use       HTML validation service, based on an SGML parser." />
   </head>
…
```

Теги тела документа

Стандарт HTML5 предоставил новые элементы для структурирования содержимого веб-страниц – *семантические теги*:

- <header>
- <footer>
- <nav>
- <section>
- <aside>
- <article>
- <main>.



Теги тела документа

```
<header>
   <h1>Название сайта</h1>
   <nav>
      <ul>
         <li><a href="page1.html">Страница 1</a></li>
         <li><a href="page2.html">Страница 2</a></li>
         <li><a href="page3.html">Страница 3</a></li>
      </ul>
   </nav>
</header>
<section>
   <h2>Свежие статьи</h2>
   <article>
      <h2>Заголовок статьи 1</h2>
      <p>Текст статьи</p>
   </article>
   <article>
      <h2>Заголовок статьи 2</h2>
      <p>Текст статьи</p>
      <aside>Дополнительная информация, статьи 2</aside>
   </article>
</section>
<aside>
   <section>
      <h3>Заголовок 3</h3>
   </section>
   <section>
      <h3>Реклама</h3>
   </section>
```

Комментарии в HTML:

<!-- Это HTML
комментарий -->

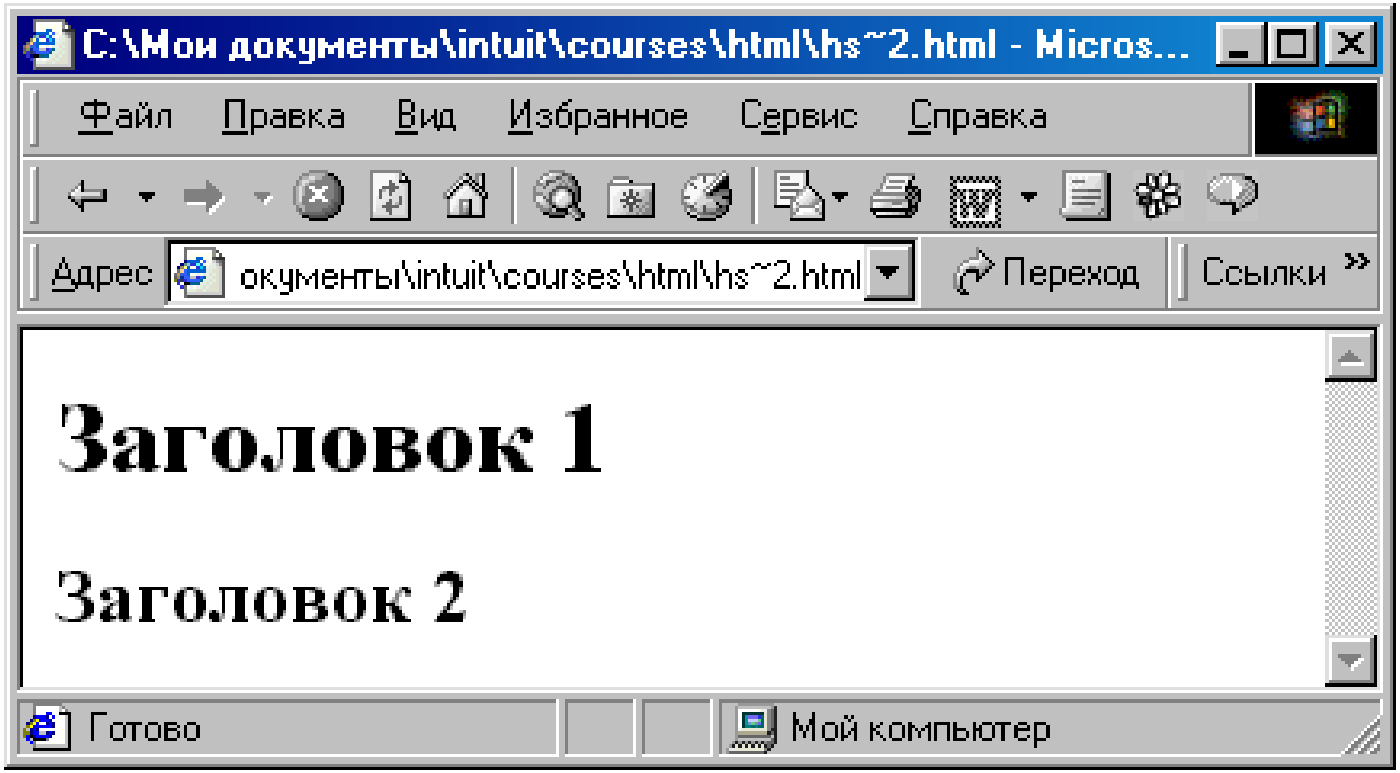
Спец символы:

Числовой код	Именная замена	Символ	Описание
&#034;	&quot;	"	Кавычка
&#038;	&amp;	&	Амперсант
&#060;	&lt;	<	Меньше
&#062;	&gt;	>	Больше
			Неразрывный пробел
&#161;	&iexcl;	¡	Перевернутый восклицательный знак
&#162;	&cent;	¢	Цент
&#163;	&pound;	£	Фунт
&#164;	&curren;	¤	Валюта
&#165;	&yen;	¥	Йена
&#168;	&uml;	¨	Умляут
&#169;	&copy;	©	Копирайт
&#171;	&laquo;	«	Левая угловая кавычка
&#174;	&reg;	®	Зарегистрированная торговая марка
&#177;	&plusmn;	±	Плюс или минус
&#187;	&raquo;	»	Правая угловая кавычка

Заголовки

Заголовок обозначает начало раздела документа. В стандарте определено 6 уровней заголовков: от **H1** до **H6**. Текст, окруженный тегами <H1></H1>, получается большим — это основной заголовок. Если текст окружен тегами <H2></H2>, то он выглядит несколько меньше (подзаголовок); текст внутри <H3></H3> еще меньше и так далее до <H6></H6>.

<H1>Заголовок 1</H1>
<H2>Заголовок 2</H2>



Тег <P>

Тег <P> применяется для разделения текста на параграфы. В нем используются те же атрибуты, что и в заголовках.

**Тег
**

Принудительный перевод строки используется для того, чтобы нарушить стандартный порядок отображения текста.

Тег <NOBR>

Неразрывная строка

Например:

<NOBR> Данная строка является самой длинной строкой документа,<WBR> которая не допускает какого-либо разбиения где бы то ни было </NOBR>

Цитата <BLOCKQUOTE>

Данный тег предназначен для обозначения в документе цитаты из другого источника. Текст, обозначенный тегом <BLOCKQUOTE>, отступает от левого края документа на 8 пробелов.

Например:

```
<HTML>
<HEAD>
   <title>Web программирование. Пример BLOCKQUOTE.</title>
</HEAD>
<BODY>
На открытии данной конференции глава представительства произнес:
<BLOCKQUOTE> Сегодня один из величайших дней для нашей компании.
<BR>
Мы открыли новую технологию, позволяющую нашим клиентам повысить
производительность их систем в несколько раз.</BLOCKQUOTE>
</BODY>
</HTML>
```

Списки в HTML

Существует три основных вида списков в HTML-документе:

- ***нумерованный;***
- ***ненумерованный;***
- ***список определений.***

Нумерованные списки

Нумерованный список начинается стартовым тегом <OL> и завершается тегом </OL>. Каждый элемент списка начинается с тега <LI>. Например:

```
<OL>
   <LI> Нумерованный
   <LI> Ненумерованный
   <LI> Список определений
</OL>
```

Тег <OL> может иметь атрибуты:
<OL TYPE=A|a|I|i|1 START=n>,где:
TYPE - вид счетчика:
A - большие латинские буквы (A,B,C...)
a - маленькие латинские буквы (a,b,c...)
I - большие римские цифры (I,II,III...)
i - маленькие римские цифры (i,ii,iii...)
1 - обычные цифры (1,2,3...)
START=n - число, с которого начинается отсчет.

Ненумерованные списки

Для непронумерованных списков браузер обычно использует маркеры для пометки элемента списка. Вид маркера, как правило, настраивает пользователь браузера.

Пронумерованный список начинается стартовым тегом <UL> и завершается тегом </UL>. Каждый элемент списка начинается с тега <LI>. Например:

```
<UL>
   <LI> Нумерованный
   <LI> Ненумерованный
   <LI> Список определений
</UL>
```

Тег <UL> может иметь атрибут:

<UL TYPE=disc|circle|square>

Атрибут TYPE тега <UL> определяет внешний вид маркера как вид по умолчанию (disc), круглый (circle) или квадратный (square).

Тег элемента списка <LI> может иметь атрибуты:

<LI TYPE=disc|circle|squade> или <LI TYPE=A|a|I|i|1 VALUE=n>

в зависимости от того, в списке какого вида находится данный элемент.

Вложенные списки

Пример

Списки определений

Список определений начинается с тега <DL> и завершается тегом </DL>. Данный список служит для создание списков типа "термин"-"описание". Каждый термин начинается тегом <DT> , а описание - тегом <DD>. Пример.

Графика в HTML-документах

Внедрение графических образов в документ позволяет пользователю видеть изображения непосредственно в контексте других элементов документа.

Тег <IMG>

Для внедрения изображения в HTML-документ используется тег <IMG>. Синтаксис тега:

<IMG SRC="*URL*" ALT="*text*" HEIGHT=n1 WIDTH=n2 ALIGN=*top*|*middle*|*bottom*|*texttop* ISMAP>

Атрибуты тега:

URL - указывает браузеру где находится рисунок. Рисунок должен храниться в графическом формате, поддерживаемом браузером (например, GIF или JPG).

ALT="text" - данный необязательный атрибут задает текст, который будет отображен браузером, не поддерживающим отображение графики или с отключенным отображением изображений. Обычно, это короткое описание изображения, которое пользователь сможет увидеть на экране в виде всплывающей подсказки.

HEIGTH=n1 - данный необязательный атрибут используется для указания высоты рисунка в пикселях. Если данный атрибут не указан, то используется оригинальная высота рисунка.

WIDTH=n2 - Позволяет задать абсолютную ширину рисунка в пикселях.

ALIGN - данный атрибут используется, чтобы сообщить браузеру, куда поместить следующий блок текста. Это позволяет более строго задать расположение элементов на экране.
ISMAP – этот атрибут сообщает браузеру, что данное изображение позволяет пользователю выполнять какие-либо действия, щелкая мышью на определенном месте изображения.
BORDER - данный атрибут позволяет определить ширину рамки вокруг рисунка.
VSPACE - позволяет установить размер в пикселах пустого пространства над и под рисунком, чтобы текст не наезжал на рисунок. Особенно это важно для динамически формируемых изображений, когда нельзя заранее увидеть документ.
HSPACE - то же самое, что и VSPACE, но только по горизонтали.

ПРИМЕР

Таблицы в HTML

Таблицы в HTML организуются как набор строк и столбцов. Ячейки таблицы могут содержать любые HTML-элементы, такие, как заголовки, списки, абзацы, фигуры, графику, а также элементы форм. Таким образом, можно использовать таблицы для равномерного размещения элементов на странице.

Основные теги таблиц

Все элементы таблицы должны находиться внутри двух тегов **<TABLE>…</TABLE>**.

Строка таблицы задается парой тегов **<TR>…</TR>**.

Пара тегов **<TD>…</TD>** описывает стандартную ячейку таблицы.

Ячейка заголовка таблицы определяется тегами **<TH>…</TH>** и имеет ширину всей таблицы; текст в данной ячейке имеет атрибут BOLD и ALIGN=center.

Теги **<CAPTION>…</CAPTION>** описывают название таблицы (подпись). Тег **<CAPTION>** может присутствовать внутри **<TABLE>…</TABLE>**, но снаружи описания какой-либо строки или ячейки.

Основные атрибуты таблиц

BORDER - данный атрибут используется в теге TABLE. Если данный атрибут присутствует, граница таблицы прорисовывается для всех ячеек и для таблицы вцелом. BORDER может принимать числовое значение, определяющее ширину границы, например BORDER=3.

ALIGN - Если атрибут ALIGN присутствует внутри тегов <CAPTION> и </CAPTION>, то он определяет положение подписи для таблицы (сверху или снизу). По умолчанию ALIGN=top. Если атрибут ALIGN встречается внутри <TR>, <TH> или <TD>, он управляет положением данных в ячейках по горизонтали. Может принимать значения left (слева), right (справа) или center (по центру). Пример.

VALIGN - данный атрибут встречается внутри тегов <TR>, <TH> и <TD>. Он определяет вертикальное размещение данных в ячейках. Может принимать значения top (верх), bottom (низ), middle (по середине) и baseline (все ячейки строки прижаты вверх). Пример.

NOWRAP - данный атрибут говорит о том, что данные в ячейке не могут логически разбиваться на несколько строк и должны быть представлены одной строкой.

COLSPAN - указывает, какое количество ячеек будет объединено по горизонтали для указанной ячейки. По умолчанию - 1.

ROWSPAN - указывает, какое количество ячеек будет объединено по вертикали для указанной ячейки. По умолчанию - 1.

COLSPEC - данный параметр позволяет задавать фиксированную ширину колонок либо в символах, либо в процентах, например COLSPEC="20%".
CELLPADDING - данный атрибут определяет ширину пустого пространства между содержимым ячейки и ее границами, то есть задает поля внутри ячейки. Пример.
CELLSPACING - определяет ширину промежутков между ячейками в пикселях. Если этот атрибут не указан, по умолчанию задается величина, равная двум пикселям. Пример.

Пример таблицы с использованием стилей

Использование каскадных таблиц стилей CSS в HTML:

Каскадные таблицы стилей - CSS (Cascading Style Sheets) позволяют управлять размером и стилем шрифтов, интервалами между строками текста, отступами, цветами, используемыми для текста и фона и многими другими параметрами отображения HTML-документов.

Размещение информации о стилях:

Информация о стилях может располагаться либо **в отдельном файле**, либо **в заголовочной части Web-страницы**, либо непосредственно **внутри тега элемента**.

1)Ссылки на таблицы стилей выполняются с помощью тега <LINK>, располагающегося внутри тега <HEAD>:
<LINK REL=STYLESHEET TYPE="text/css" HREF="URL">

2) внутри Web-страницы, внутри тега <HEAD> или<BODY> используется тег:
<STYLE type="text/css">...описание стилей...</STYLE>

2а) Импорт описателей стилей внутри тега STYLE:
<STYLE type="text/css">...
@import "./style/base.css“;
@import:url(http://is.sevntu.sebastopol.ua/style.css)
</STYLE>

3) Описание стиля располагается непосредственно внутри тега элемента, который описывается с помощью атрибута STYLE:
STYLE="СВОЙСТВО: ЗНАЧЕНИЕ; СВОЙСТВО: ЗНАЧЕНИЕ;”

Синтаксис CSS:

Синтаксис описания стилей в общем виде представляется следующим образом:

selector[, selector[, ...]]
{ attribute:value;
[atribute:value;...] }

или

selector selector [selector ...]
{ attribute:value;
[atribute:value;...] }

В первом варианте перечислены селекторы, для которых действует данное описание стиля. Второй вариант задает иерархию вложенности селекторов, для совокупности которых определен стиль.

В качестве селектора можно использовать:

- **имя элемента разметки**;
- **имя класса**;
- **идентификатор объекта** на HTML-странице.

Атрибут (**attribute**) определяет свойство отображаемого элемента, например левый отступ параграфа (**margin-left**), а значение (**value**) — значение этого атрибута, например, 10 типографских пунктов (**10 pt**).

Селектор — имя элемента разметки:

I, EM { color:#003366,font-style:normal }
A I { font-style:normal;font-weight:bold; text-decoration:line-through }

В первой строке этого описания перечислены селекторы-элементы, которые будут отображаться одинаково:

<I>Это курсив</I> и это тоже <EM>курсив</EM>

Последняя строка определяет стиль отображения вложенного в гипертекстовую ссылку курсива:

<A NAME=empty><I>SevNTU</I></A>

В данном случае переопределение состоит в том, что текст отображается внутри гипертекстовой ссылки перечеркнутым, причем жирным шрифтом.

Селектор — имя класса:

Имя класса не является каким-либо стандартным именем элемента HTML-разметки. Оно определяет описание класса элементов разметки, которые будут отображаться одинаково. Для того, чтобы отнести элемент разметки к тому или иному классу, нужно воспользоваться его атрибутом CLASS:

<STYLE>
.test {color:white;background-color:black;}
</STYLE>

```
<STYLE>
.test {color:white;background-color:black;}
</STYLE>
...
<P CLASS="test">
Этот параграф мы отобразим белым цветом по
черному фону
</P>
...
<P>
Эту <A CLASS="test">гипертекстовую ссылку</A>
мы отобразим белым цветом по черному фону.
</P>
```

Селектор — идентификатор объекта

Вместо двух описаний классов, которые отличаются только одним из параметров, можно создать одно описание класса и описание идентификатора объекта. Описание стиля для объекта задается строкой, в которой селектор представляет собой имя этого объекта с лидирующим символом "#":

```
a.mainlink { color:darkred;
        text-decoration:underline;
        font-style:italic; }
#blue { color:#003366 }
...
<A CLASS=mainlink>основная гипертекстовая
ссылка</A>
<A CLASS=mainlink ID=blue>модифицированная
гипертекстовая ссылка</A>
```

В следующей таблице приводятся краткие сведения о синтаксисе селекторов в CSS2:

Шаблон	Значение
*	Сопоставляется любому элементу.
E	Сопоставляется любому элементу E (т.е. элементу типа E.
E F	Сопоставляется любому элементу F, который является потомком элемента E.
E > F	Сопоставляется любому элементу F, который является дочерним элементом элемента E.
E:first-child	Сопоставляется элементу E, если он является первым дочерним элементом своего родительского элемента.
E:link E:visited	Сопоставляется элементу E, если он является привязкой гиперссылки, направляющей к документу, которой еще не был просмотрен (:link) или уже был просмотрен (:visited).
E:active E:hover E:focus	Сопоставляется элементу E во время определенных действий пользователя.
E:lang(c)	Сопоставляется элементу E, если он присутствует в разговорном языке (язык документа указывает, каким образом определяется разговорный язык).
E + F	Сопоставляется любому элементу F, которому непосредственно предшествует элемент E.
E[foo]	Сопоставляется любому элементу E с набором атрибутов "foo" (независимо от значения).
E[foo="warning"]	Сопоставляется любому элементу E, у которого значение атрибута "foo" в точности равно "warning".
E[foo~="warning"]	Сопоставляется любому элементу E, у которого значением атрибута "foo" является список значений, разделенных пробелами, и одно из этих значений в точности равно "warning".
E[lang|="en"]	Сопоставляется любому элементу E, атрибут "lang" которого имеет список значений, разделенных знаками дефиса, начинающийся (слева) со значения "en".
DIV.warning	*Только в HTML*. Значение аналогично значению DIV[class~="warning"].
E#myid	Сопоставляется любому элементу E, атрибут ID которого равен "myid".

При использовании стилей действуют следующие правила старшинства стилей:

- сначала применяются стили браузера по умолчанию;
- стили браузера по умолчанию переопределяются прилинкованными стилями (элемент LINK заголовка документа);
- прилинкованные стили переопределяются описаниями стилей в элементе STYLE;
- стили элемента STYLE переопределяются атрибутом STYLE в любом из элементов разметки.

Элемент DIV

DIV играет роль универсального блока. Блочный элемент всегда отделен от прочих элементов страницы (контекста) пустой строкой. DIV не несет никакой смысловой нагрузки. Часто говорят, что DIV — это раздел страницы. Но на самом деле его применение имеет смысл только в контексте CSS. Никаких правил по умолчанию для отображения DIV не существует.

DIV позволяет применить атрибуты стиля, связанные с границей блока и отступами блока от границ старшего элемента, а также "набивку", т.е. отступ от границы блока до границы вложенного элемента: ПРИМЕР

Элемент SPAN

Элемент разметки SPAN — это обобщенный строковый элемент разметки, применение которого не приводит к образованию блока текста. Он может заменить элементы FONT, I, B, U, SUB, SUP и т.п. Приведем примеры таких соответствий:

Тэг конца элемента строкового типа закрывает ближайший элемент, а не тот, который открыт тэгом начала данного строкового стиля. Также и в случае применения элемента SPAN, где тэг конца можно соотнести только с ближайшим тэгом начала элемента SPAN:

<B>предложение <I>с пересекающимися</B> стилями</I>

предложение *с пересекающимися* *стилями*

<SPAN STYLE="font-weight:bold;">предложение <SPAN STYLE="font-style:italic;">с пересекающимися</SPAN> стилями</SPAN>

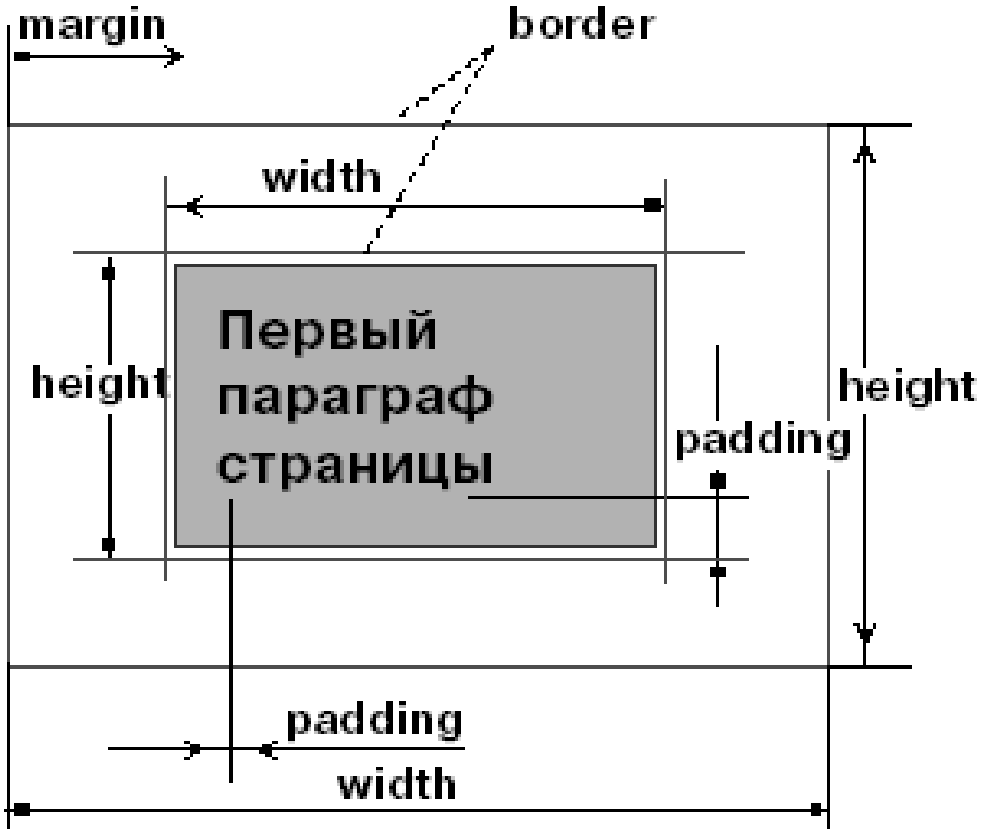
предложение *с пересекающимися* стилями

Применение элемента SPAN ограничено браузерами, которые поддерживают CSS. При этом не все атрибуты спецификации CSS поддерживаются в браузерах. Например, атрибут vertical-align, который призван заменить элементы SUP и SUB, не поддерживается ни одним из браузеров.

Свойства блоков

Блочные элементы(блоки текста или box) позволяют оперировать с текстом в терминах прямоугольников, которые этот текст занимает. При этом блок текста становится элементом дизайна страницы с теми же свойствами, что и картинка, таблица или прямоугольная область приложения.

Блок текста обладает свойствами: высоты (height), ширины (width), границы (border), отступа (margin), набивки (padding), произвольного размещения (float), управления обтеканием (clear).



Позиционирование блоков

При использовании "абсолютных" координат точка отсчета помещается в верхний левый угол окна браузера, а оси X и Y направлены вправо по горизонтали и вниз по вертикали, соответственно:

```
.example { position:absolute;top:10px; left:20px; }
```

При использовании «относительных» координат блоки располагаются на странице в координатах охватывающего их блока. Это позволяет сохранять взаимное расположение элементов разметки при любом размере окна браузера и его настройках по умолчанию. Пример.

Порядок перекрытия блоков определяется атрибутом z-index. Чем больше значение z-index, тем ближе к наблюдателю находится слой. Пример.

СВОЙСТВА ШРИФТА

font-family	Используется для указания шрифта или шрифтового семейства, которым будет отображаться элемент. **P {font-family: Times New Roman;}**
font-weight	Определяет степень жирности шрифта с помощью трех параметров: lighter, bold, bolder **B {font-weight: bolder;}**
font-size	Устанавливает размер шрифта. Параметр может указываться как в относительной (проценты), так и абсолютной величине (пункты, пикселы, сантиметры) **H1 {font-size: 200%;}** **H2 {font-size: 150px;}** **H3 {font-size: 400pt;}**

ЦВЕТ ЭЛЕМЕНТА И ЦВЕТ ФОНА

color	Определяет цвет элемента I {color: yellow;}
background-color	Устанавливает цвет фона для элемента – именно для элемента, а не для страницы. Браузеры отображают это свойство по-разному: Microsoft IE отводит под фон элемента всю доступную ширину страницы, а Netscape Navigator – лишь ширину, занимаемую этим элементом.

СВОЙСТВА ТЕКСТА

text-decoration	Устанавливает эффекты оформления шрифта, такие, как подчеркивание или зачеркнутый текст **H4 {text-decoration: underline;}** **A {text-decoration: none;}** **.wrong {text-decoration: line-through;}**
text-align	Определяет выравнивание элемента. **P {text-align: justify}** **H1 {text-align: center}**
text-indent	Устанавливает отступ первой строки текста. Чаще всего используется для создания параграфов с табулированной первой строкой. **P {text-indent: 50pt;}**
line-height	Управляет интервалами между строками текста. **P {line-height: 50 %}**

СВОЙСТВА ГРАНИЦ

margin-left	Устанавливают значения отступов вокруг элемента. **IMG { margin-right: 20pt}** **P { margin-left: 2cm}**
margin-right margin-right margin-top	Устанавливают значения отступов вокруг элемента. **IMG { margin-right: 20pt}** **P { margin-left: 2cm}**

ЕДИНИЦЫ ИЗМЕРЕНИЯ:

px – Пикселы; **cm** –Сантиметры; **mm** – Миллиметры;
pt - Пункты (типограф.) **%** -Проценты

Использование CSS фреймворков

CSS фреймворк — это набор стилей, который позволяет быстрее и проще создавать макеты web-страниц. При использовании CSS фреймворка нет необходимости создавать таблицы стилей — достаточно лишь изменить предлагаемые стили, что существенно экономит время на разработку сайта.

Например, для использования CSS фреймворка **BluePrint (http://blueprintcss.org/)** необходимы только 3 файла:
screen.css
print.css
ie.css

screen.css – основной набор стилей, содержащий:
* reset.css – сброс стилей браузера по умолчанию.
* typography.css – установка стилей шрифтов.
* forms.css –стили для использования в формах.
* grid.css –«сеточная» система блоков. Здесь определяется сетка из 24 колонок, под которую заранее заготовлены все возможные классы DIV, имеющие разную ширину: span-1, span-2 … span-24. Последний по горизонтали блок требует добавления стиля класса «LAST»(см. ниже).

print.css – стиль отображения, который будет подключен при отправке на принтер.

<link rel="stylesheet" href="/css/blueprint/print.css" type="text/css" media="print">

ie.css – содержит набор стилей специально предназначенных для «обхода» особенностей Microsoft Internet Explorer. Для его подключения необходимо использовать условный комментарий вида:

<!--[if IE]><link rel="stylesheet" href="../blueprint/ie.css" type="text/css" media="screen, projection"><![endif]-->

Рассмотрим примеры использования BluePrint

Еще один пример CSS фрейморка – YAML(http://www.yaml.de/)

Гипертекстовые ссылки

Гипертекстовые ссылки являются одним из ключевых элементов, делающим WEB привлекательным для пользователей.

Структура ссылок в HTML-документе

Выше был рассмотрен формат URL. Для того, чтобы браузер отобразил ссылку на URL в HTML-документе, необходимо отметить URL специальными тегами. Синтаксис HTML, позволяющий это сделать следующий:

<A HREF="URL"> текст-который-будет-ссылкой</A>

Любой текст, находящийся между данными двумя тегами подсвечивается специальным образом Web-браузером. Пример ссылки:

<A HREF="www.is.sevntu.sebastopol.ua/webprogramming/first.html">
Ваша первая Web-страница </A>

Ссылки на разделы внутри документа

Возможно делать ссылки на различные участки или разделы одного и того же документа, используя специальных скрытый маркер(***якорь***) для этих разделов.

Для якоря, как и для ссылки, используется тег <A>, но без атрибута HREF, а с атрибутом NAME:

<A NAME="named_anchor"> Текст-который-отобразится-в-первой-строке-браузера </A>

Для создания ссылки на этот якорь из этой же страницы в атрибуте HREF ссылки укажем имя якоря после символа #:

<A HREF="#named_anchor"> Текст </A>

Например:

Формы в HTML

Для организации обратной связи между пользователем и WEB-сервером в HTML используются формы.

Когда HTML-документ, содержащий форму, интерпретируется WEB-браузером, на экране браузер создает специальные экранные элементы ввода, такие, как поля ввода, checkboxes, radiobuttons, выпадающие меню, скроллируемые списки, кнопки и т.д. Когда пользователь заполняет форму и нажимает кнопку "Подтверждение" (SUBMIT - специальный тип кнопки, который задается при описании HTML-документа внутри формы), информация, введенная пользователем, посылается WEB-серверу для передачи указанному CGI-скрипту или отправки по E-mail.

Все теги формы помещаются между тегами ***<FORM>*** *и* ***</FORM>:***

<FORM METHOD="GET | POST" ACTION="URL">

Элементы_формы_и_другие_элементы_HTML

</FORM>

Атрибуты формы

Атрибут ACTION

ACTION описывает URL, который будет вызываться для обработки формы. Данный URL почти всегда указывает на CGI-программу, обрабатывающую данную форму. В том случае если необходимо отправить данные формы на электронный адрес в атрибуте ACTION необходимо указать “mailto: EMAIL”, где EMAIL –электронный адрес для отправки.

Атрибут METHOD

Метод посылки сообщения с данными из формы определяется атрибутом **METHOD**. В зависимости от используемого метода вы можете посылать результаты ввода данных в форму двумя путями:

GET: Информация из формы добавляется в конец URL, который был указан в описании заголовка формы. Ваша CGI-программа (CGI-скрипт) получает данные из формы в виде параметра переменной среды QUERY_STRING. Использование метода GET не рекомендуется.

POST: Данный метод передает всю информацию о форме в теле HTTP-сообщения. Ваша CGI-программа получает данные из формы в стандартный поток ввода. Сервер не будет пересылать вам сообщение об окончании пересылки данных в стандартный поток ввода; вместо этого используется переменная среды **CONTENT_LENGTH** для определения, какое количество данных необходимо считать из стандартного потока ввода. Данный метод рекомендуется к использованию.

Например:

<FORM METHOD=post
ACTION=mailto:yourname@your.email.address>

Теги форм

Тег <TEXTAREA>

Тег **<TEXTAREA>** используется для того, чтобы позволить пользователю вводить многострочную текстовую информацию. Вот пример использования тега <TEXTAREA>:

<TEXTAREA NAME="IS" ROWS=10 COLS=50></TEXTAREA>

Атрибуты, используемые внутри тега **<TEXTAREA>**, описывают внешний вид и имя вводимого значения. Тег **</TEXTAREA>** необходим даже тогда, когда поле ввода изначально пустое. Описание основных атрибутов:

NAME - имя поля ввода;

ROWS - высота поля ввода в символах (количество строк ввода);

COLS - ширина поля ввода в символах (количество символов в строке).

Если вы хотите, чтобы в поле ввода по умолчанию выводился какой-либо текст, то его необходимо вставить внутри тегов **<TEXTAREA>** и **</TEXTAREA>**. Например.

Тег <INPUT>

Тег <INPUT> используется как для ввода одной строки текстовой информации, так и для организации CHECKBOX, RADIOBUTTON, кнопок и др.

Атрибуты тега:

- ***TYPE*** - определяет тип поля ввода. По умолчанию это простое поле ввода для одной строки текста. Остальные типы должны быть указаны явно:

TYPE=TEXT - данный тип поля ввода используется по умолчанию и описывает однострочное поле ввода(атрибуты MAXLENGTH и SIZE для определения максимальной длинны вводимого значения в символах и размера отображаемого поля ввода на экране по умолчанию - 20 символов).

TYPE=CHECKBOX - используется для простых логических (BOOLEAN) значений. Значение, ассоциированное с именем данного поля, которое будет передаваться в вызываемую CGI-программу, может принимать значение ON или OFF.

TYPE=HIDDEN - поля данного типа не отображаются браузером и не дают пользователю изменять присвоенные данному полю по умолчанию значение. Это поле может использоваться для передачи в CGI-программу статической служебной информации.

TYPE=IMAGE - данный тип поля ввода позволяет вам связывать графический рисунок с именем поля. При нажатии мышью на какую-либо часть рисунка будет немедленно вызвана ассоциированная форме CGI-программа. Значения, присвоенные переменной NAME, будут выглядеть так - создается две новых переменных: первая имеет имя, обозначенное в поле NAME с добавлением .x в конце имени, а переменная с именем, содержащимся в NAME и добавленным .y, будет содержать Y-координату.

TYPE=PASSWORD - то же самое, что и атрибут TEXT, но вводимое пользователем значение не отображается браузером на экране (предназначается для ввода пароля).

TYPE=RADIO - данный атрибут позволяет организовать элемент для ввода одного значения из нескольких альтернатив. Для создания набора альтернатив необходимо создать несколько полей ввода с атрибутом TYPE="RADIO" с разными значениями атрибута VALUE, но **с одинаковыми значениями атрибута NAME.** В CGI-программу при этом будет передано значение типа **NAME='VALUE'**.

TYPE=RESET - данный тип обозначает кнопку, при нажатии которой все поля формы примут значения, описанные для них по умолчанию. Такая кнопка используется для очистки формы.

TYPE=SUBMIT - данный тип обозначает кнопку, при нажатии которой данные введенные в форму будут отправлены на сервер для обработки в соответствии с атрибутами METHOD и ACTION, указанными в заголовке формы. Атрибут VALUE в этом случае может содержать текстовую строку, которая будет высвечена на кнопке.

Атрибуты тега *<INPUT>*(продолжение):

- **CHECKED -** означает, что CHECKBOX или RADIOBUTTON будет в установленном состоянии.

- ***MAXLENGTH -*** определяет количество символов, которое пользователь может ввести в поле ввода.

- ***SIZE*** - определяет визуальный размер поля ввода на экране в символах.

- **NAME** - имя поля ввода. Данное имя используется как уникальный идентификатор поля, по которому, впоследствии, вы сможете получить данные, помещенные пользователем в это поле.

- **SRC** - URL, указывающий на картинку (используется совместно с атрибутом IMAGE).

- **VALUE** - присваивает полю значение по умолчанию или значение, которое будет выбрано при использовании типа RADIO (для типа RADIO данный атрибут обязателен).

Пример

Списки выбора в формах

Для организации в формах списков выбора с прокруткой и выпадающих списков выбора используют тег **<SELECT>**. Тег <**SELECT**> имеет один или более элементов выбора между стартовым тегом <**SELECT**> и завершающим <**/SELECT**>. По умолчанию, первый элемент отображается в строке выбора.

Тег <**SELECT**> поддерживает три необязательных атрибута: **MULTIPLE**, **NAME** и **SIZE**.

Указание атрибута **MULTIPLE** в теге <***SELECT***> указывает, что возможен выбор более чем одного значения (множественный выбор производиться при удерживании клавиш SHIFT или CTRL).

Атрибут **NAME** определяет наименование объекта.

Атрибут **SIZE** определяет число видимых пользователю пунктов списка. Если в форме установлено значение атрибута SIZE=1, то браузер выводит на экран список в виде выпадающего меню. В случае SIZE > 1 браузер представляет на экране обычный список.

Для определения каждого из пунктов списка выбора используют тег **<OPTION>**. Тег **<OPTION>** имеет атрибуты **SELECTED** и **VALUE:**

Атрибут **SELECTED** используется для указания выбранных по умолчанию значений.

Атрибут **VALUE** указывает на значение, возвращаемое формой после выбора пользователем данного пункта (по умолчанию значение поля соответствует тексту тега **<OPTION>**).

```
<FORM>
   <SELECT NAME=AGE>
     <OPTION value=0> меньше 15
     <OPTION value=1> от 16 до 20
     <OPTION value=2> от 21 до 30
     <OPTION value=3> больше 30
   </SELECT>
</FORM>
```

Пример 2:

```
<FORM>
  <SELECT  NAME=AGE SIZE=3>
     <OPTION> меньше 15
     <OPTION SELECTED> от 16 до 20
     <OPTION> от 21 до 30
     <OPTION> больше 30
  </SELECT>
</FORM>
```

Пример №3:

```
<FORM>
  <SELECT NAME=group SIZE=4 MULTIPLE>
     <OPTION value=21>И-21
     <OPTION value=22 SELECTED>И-22

     <OPTION value=24>И-24
  </SELECT>
</FORM>
```

Пример 1:

Для выполнения группировки элементов списка выбора используется тег **<OPTGROUP>** (закрывающий тег обязателен). Обычно это полезно, если пользователь должен делать выбор в длинном списке вариантов; группы связанных вариантов проще просматривать и запоминать, чем один длинный список вариантов.

```
<SELECT name="Groups">
    <OPTGROUP label="4 курс">
      <OPTION value="41">И-41
      <OPTION value="42">И-42
      <OPTION value="43">И-43
      <OPTION value="44">И-44
    </OPTGROUP>
    <OPTGROUP LABEL="5 курс">
      <OPTION value="51">И-51
      <OPTION value="52">И-52
      <OPTION value="53">И-53
      <OPTION value="54">И-54
    </OPTGROUP>
    <OPTGROUP LABEL="Магистры">
      <OPTION SELECTED value="MAG1">Маг-51
    </OPTGROUP>
 </SELECT>
```

Пример

следующими свойствами:

- каждый фрейм может иметь свой собственный URL, что позволяет загружать его независимо от других фреймов;
- каждый фрейм может иметь собственное имя (параметр NAME), позволяющее обращаться к нему из другого фрейма;
- размер фрейма может быть изменен пользователем прямо на экране при помощи мыши.

Данные свойства фреймов позволяют создавать такие интерфейсные решения как:

- размещение статической информации, которую автор считает необходимым постоянно показывать пользователю, в одном статическом фрейме (это может быть графический логотип, набор управляющих кнопок и тп), тогда как во втором фрейме осуществляется просмотр остального содержимого со скроллингом;
- помещение в статическом фрейме оглавления всех или части WEB-документов, содержащихся на WEB-сервере, что позволяет пользователю быстро находить интересующую его информацию;
- создание окна результатов запросов, когда в одном фрейме находится собственно запрос, а в другом результаты запроса.

Вместо тега BODY во фреймовых документах используется контейнер **FRAMESET**:

```
<HTML>
 <HEAD>...</HEAD>
  <FRAMESET>...</FRAMESET>
</HTML>
```

Общий синтаксис фреймов:

```
<FRAMESET COLS="value" | ROWS="value">
  <FRAME SRC=" URL">
     <FRAME ...>
      ...
</FRAMESET>
```

Контейнер **FRAMESET** описывает все фреймы, на которые делится экран. Вы можете разделить экран на несколько вертикальных или несколько горизонтальных фреймов. Тег **FRAME** описывает каждый фрейм в отдельности. Пример.

Тег <NOFRAMES >

Данный тег используется в случае, когда создается документ, который может просматриваться как браузерами, поддерживающими фреймы, так и браузерами, их не поддерживающими. Данный тег помещается внутри контейнера FRAMESET. Браузер, не поддерживающий фреймы, проигнорирует все внутри FRAMESET, кроме <NOFRAMES>, а то, что находится внутри тегов <NOFRAMES> и </NOFRAMES> наоборот игнорируется браузерами, поддерживающими фреймы.

Следующий тег <FRAMESET> создает экран, на котором верхняя строка занимает 10% высоты экрана, средняя — 60%, а нижняя — оставшиеся 30%:

<FRAMESET ROWS="10%, 60%, 30%">

Можно задать относительные значения в комбинации с фиксированными, выраженными в процентах или пикселях. Например, следующий тег создает экран, на котором верхняя строка имеет высоту 20 пикселей, средняя — 80 пикселей, а нижняя занимает все оставшееся место:

<FRAMESET ROWS="20, 80, *">

Атрибут **COLS** - столбцы задаются так же, как строки. Для них применимы те же атрибуты.

Атрибут **BORDER** - данный атрибут используется для задания толщины границы между фреймами. Значение BORDER=0 приведет к отсутствию границ.

Атрибуты тега <FRAME>

Тег **<FRAME>** определяет внешний вид и поведение фрейма. Этот тег не имеет закрывающего тега. Атрибутов шесть: NAME, MARGINWIDTH, MARGINHEIGHT, SCROLLING, NORESIZE и SRC.

Атрибут NAME – именование фрейма позволяет открывать ссылки из других фреймов в поименованном. Фрейм, в котором отображаются страницы, называется целевым (target). Фреймы, которые не являются целевыми, именовать не обязательно.

Внимание! Имя окна (фрэйма), используемого в параметре TARGET должно начинаться с латинской буквы или цифры.

Атрибут MARGINWIDTH - задает горизонтальный отступ между содержимым кадра и его границами. Наименьшее значение этого атрибута равно 1. Нельзя указать 0. По умолчанию равен 6.

Атрибут MARGINHEIGHT - атрибут MARGINHEIGHT действует так же, как и MARGINWIDTH. Он задает поля в верхней и нижней частях фрейма.

Атрибут SCROLLING - атрибут SCROLLING дает возможность пользоваться прокруткой во фрейме. Возможные варианты: SCROLLING=yes, SCROLLING=no, SCROLLING=auto.

По умолчанию SCROLLING=auto.

Атрибут NORESIZE - когда необходимо сделать границы фрейма неподвижными используется атрибут NORESIZE.

Атрибут SRC - Атрибут SRC применяется в теге FRAME при разработке фреймовой структуры для того, чтобы определить, какая страница появится в том или ином фрейме.

Атрибут TARGET тегов <A> - позволяет открывать ссылку в указанном в данном атрибуте фрейме(по имени фрейма).

Вложенные и множественные фреймовые структуры

Задав внутри одной объемлющей фреймовой структуры две независимых подструктуры, можно поместить в левой части экрана столбец из двух, а в правой — из трех фреймов. Пример.

XHTML

XHTML (англ. Extensible Hypertext Markup Language — расширяемый язык гипертекстовой разметки) — семейство языков разметки веб-страниц на основе XML, повторяющих и расширяющих возможности HTML 4. Спецификации XHTML 1.0 и XHTML 1.1 являются рекомендациями консорциума Всемирной паутины, однако на данный момент его развитие остановлено с рекомендацией использовать HTML. Новые версии XHTML не выпускаются.

Главное отличие XHTML от HTML заключается в обработке документа. Документы XHTML обрабатываются своим модулем (парсером) аналогично документам XML. В процессе этой обработки ошибки, допущенные разработчиками, не исправляются.

XHTML соответствует спецификации SGML, поскольку XML является её подмножеством. HTML обладает множеством особенностей в процессе обработки и фактически перестал относиться к семейству SGML, что и закреплено в черновике спецификации HTML 5.

XHTML

Различия между XHTML и HTML

Все элементы должны быть закрыты. Теги, которые не имеют закрывающего тега (например, <img> или <br>), должны иметь на конце / (например, <br />).

Логические атрибуты записываются в развёрнутой форме. Например, следует писать <option selected="selected"> или <td nowrap="nowrap">.

Имена тегов и атрибутов должны быть записаны строчными буквами (например, вместо <IMG ALT="" />).

XHTML гораздо строже относится к ошибкам в коде; < и & везде, даже в URL, должны замещаться < и & соответственно. По рекомендации W3C браузеры, встретив ошибку в XHTML, должны сообщить о ней и не обрабатывать документ. Для HTML браузеры должны были попытаться понять, что хотел сказать автор.

Кодировкой по умолчанию является UTF-8 (в отличие от HTML, где кодировкой по умолчанию является ISO 8859-1).

Для XHTML-страниц рекомендуется задавать MIME-тип — application/xhtml+xml, но это не является обязательным, более того — браузер Internet Explorer 8 и младшие версии не смогут обрабатывать страницу, поэтому с XHTML 1.0 традиционно используется MIME-тип для HTML — text/html.

HTML 5

Спецификация HTML5 доступна с 2008 года на официальном сайте W3C. В настоящий момент статус – кандидат в рекомендации(может быть изменена, дополнена и тп).

HTML5 добавляет много новых тегов - <video>, <audio>, <header> и <canvas> . Эти элементы разработаны для того, чтобы сделать проще внедрение и управление графическими и мультимедийными объектами в вебе без необходимости обращаться к собственным плагинам и API. Другие новые элементы, такие как <section>, <article>, <header> и <nav> разработаны для того, чтобы обогащать семантическое содержимое документа (страницы).

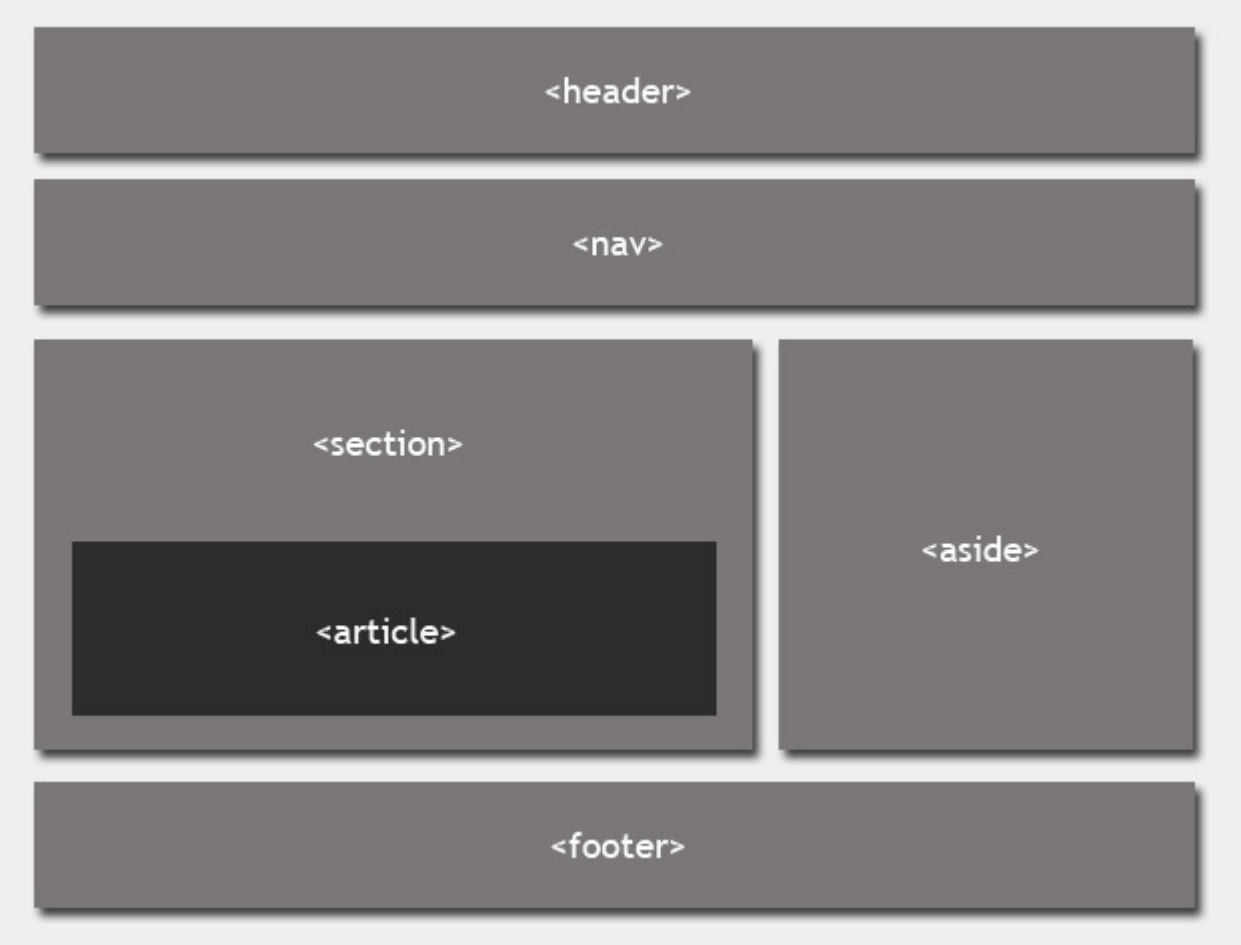
Новые теги HTML5

1. Теги разделов (article, aside, footer, header, hgroup, nav, section)
2. Теги группировки контента (figure, figcaption)
3. Теги для семантического выделения текста (bdo, mark, time, ruby, rt, rp, wbr)
4. Теги для вставки контента (audio, video, canvas, embed, source)
5. Теги для элементов форм (datalist, keygen, output, progress, meter)
6. Интерактивные элементы (details, summary, command, menu)

HTML 5

<article> Определяет статью
<aside> Определяет контент в стороне от основного содержимого страницы
<audio> Определяет аудио контент
<canvas> Определяет графики
<command> Определяет командную кнопку
<datagrid> Определяет данные в упорядоченный список
<datalist> Определяет выпадающий список
<datatemplate> Определяет шаблон данных
<details> Определяет детали элемента
<dialog> Определяет диалог (разговор)
<eventsource> Определяет цель события, отправляемого по серверу
<figure> Определяет группу медиа-контента, и их подписи
<footer> Определяет нижний колонтитул для раздела или страницы
<header> Определяет область заголовка раздела или страницы
<mark> Определяет выделенный текст
<meter> Определяет измерения в течение заранее определенного диапазона
<nav> Определяет навигационные ссылки
<nest> Определяет вложенную точку в шаблоне данных
<output> Определяет некоторые виды результата
<progress> Определяет ход выполнения задачи любого рода
<rule> Определяет правила для обновления шаблонов
<section> Определяет раздел (секцию)
<source> Определяет медиа-ресурсы
<time> Определяет дату/время
<video> Определяет видео

HTML 5



HTML 5

Неподдерживаемые теги

<acronym> Не поддерживается. Определяет акроним

<applet> Не поддерживается. Определяет апплет
<basefont> Не поддерживается. Используетя вместо CSS для задания шрифта
<big> Не поддерживается. Определяет большой текст
 Не поддерживается. Определяет текст по центру
<dir> Не поддерживается. Определяет список директорий
<frame> Не поддерживается. Определяет фрейм
<frameset> Не поддерживается. Определяет набор фреймов
<isindex> Не поддерживается. Определяет поисковый индекс в документе
<noframes> Не поддерживается. Определяет секцию, не поддерживающую фрейм
<s> Не поддерживается. Определяет зачеркнутый текст
<strike> Не поддерживается. Определяет зачеркнутый текст
<tt> Не поддерживается. Определяет телетайп текст
<u> Не поддерживается. Определяет подчеркнутый текст
<xmp> Не поддерживается. Определяет выровненный текст

HTML 5

Новые элементы форм

Числа
Новые элемент number, позволяет добавить на форму элемент для ввода чисел, а указав свойства max и min можно указать пределы значений.
<input type=""number"/">

Диапазон
Выводит на экран ползунок, с помощью которого можно выбрать необходимое численное значение. Так же как и number имеет свойства min и max.
<input type=""range"/">

Дата и время
<input type=""date"/">
<input type=""time"/">
<input type=""datetime"/">
<input type=""month"/">
<input type=""week"/">

Выбор цвета
<input type=""color"/">

Поле поиска
По сути обычное поле для ввода, но оформленное немного в другом стиле.
<input type=""search"/">