МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
НАЦІОНАЛЬНА МЕТАЛУРГІЙНА АКАДЕМІЯ УКРАЇНИ

Н.В. НЕЧУХАЄВА, В. Г. РАСЧУБКІН, Г.А. ПАВЛЕНКО

ВИВЧЕННЯ МОВИ HTML З ВИКОРИСТАННЯМ КАСКАДУ СТИЛІВ CSS ТА ЕЛЕМЕНТІВ МОВИ JAVASCRIPT

Частина 1

Затверджено на засіданні Вченої ради академії
як навчальний посібник. Протокол № 1 від 30.01.2012

Дніпропетровськ НМетАУ 2012

УДК (004.43)
Нечухаєва Н.В., Расчубкін В.Г., Павленко Г.А. Вивчення мови HTML з використанням каскаду стилів CSS та елементів мови JAVASCRIPT. Частина 1: Навч. посібник (рос. мовою). – Дніпропетровськ: НМетАУ, 2012. – 72 с.

Викладені теоретичні відомості і практичні рекомендації щодо створення Web-документів та їх використання за допомогою провідних інтернет-технологій. Основна увага приділяється особливостям використання інструментальних засобів HTML та JavaScript у середовищі CSS.

Призначений для студентів напряму 6.020105 – документознавство та інформаційна діяльність.

Іл. 1. Бібліогр.: 6 найм.

Друкується за авторською редакцією.

Відповідальний за випуск Г.Г. Швачич, канд. техн. наук, проф.

Рецензенти: О.О. Івлєв, канд. техн. наук, проф. (ДХТУ)
А.Б. Устенко, канд. техн. наук, доц. (ДНУЗТ)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В **1989** году выпускник Оксфордского университета, бакалавр в области физики - сотрудник *Европейского центра ядерных исследований* **(CErN) Тим Бернес-Ли** разработал и всерьез приступил к созданию информационной службы **world wide web**. Он написал приложение **клиент/сервер** (**браузер**). В основу всей системы легло понятие **гипертекста** - т.е. множества отдельных текстов, имеющих ссылки друг на друга. Для работы с этими текстами был создан специальный протокол **HTTP - Hyper Text Transfer Protocol**, были обозначены основные элементы языка разметки **html**. Эта технология дала огромный толчок в развитии сети и сеть стала действительно интернациональной сетью.

Многие коммерческие компании подхватили эту инициативу и стали выпускать свои браузеры и к ним свои собственные расширения языка html.

1. ОСНОВНЫЕ ПОНЯТИЯ ОСОБЕННОСТИ ЯЗЫКА HTML

1.1 Набор символов

Формально, набор символов, используемых в html документе, должен включать ISO Latin 1, известную также как ISO 8859-1 кодировку, так как она принадлежит к набору стандартов ISO 8859, быть совместимым с ISO 10646 и Unicode.

В практической работе Вы должны использовать только ISO Latin 1 набор символов. Сейчас и в ближайшем будущем Вы можете твердо рассчитывать на его обширную поддержку приложениями. Поддержка ISO Latin 1 должна существовать во всех браузерах, однако иногда с этим существуют проблемы. Также Вы можете придерживаться ASCII набора символов, которые являются подмножеством ISO Latin 1, в особенности, если у Вас нет необходимости в написании символов с диакритическим знаком или символов, не входящих в английский алфавит (a - z).

Некоторые предупреждения. Набор символов Windows наиболее согласован с ISO Latin 1, однако есть некоторые кодовые позиции, которые зарезервированы в качестве управляющих символов в ISO Latin 1 и, тем не менее, используются для изображения видимых символов в наборе символов Windows. Наиболее

известные из них - два различных тире "en тире" и "em тире", которые не надо смешивать с дефисом (-) или подчеркиванием (_), принадлежащими к ISO Latin 1 (и даже к ASCII). Если Вы используете такие символы, пользователи Windows систем, вероятно, увидят их как положено, однако на всех других системах символы скорее всего будет выглядеть, как помарки. (Обычно, такие символы даже не выводятся совсем).

1.2. Теги HTML

Тег html состоит из следующих друг за другом в определенном порядке элементов: левой угловой скобки < (символ "меньше чем"), слэша (символ "/" необязательный элемент, который означает, что тег является конечным тегом, закрывающим некоторую структуру), атрибутов (необязательные элементы), правой угловой скобки > (символ "больше чем").

1.3. Элементы HTML

Большинство, но не все теги html спарены так, что за открывающим тегом следует соответствующий закрывающий тег, а между ними содержится текст или другие теги, например: **<H1>Foreword</H1>**. В таких случаях два тега и часть документа, отделенная ими, образуют блок, называемый html элементом или **контейнером**. Некоторые теги, например <Hr>, являются элементами html сами по себе, и для них соответствующий конечный тег неверен. Далее мы будем называть теги по их именам, опуская обязательные угловые скобки.

1.4. Атрибуты

Для каждого тега определяется множество возможных атрибутов. Большинство тегов допускает один или несколько атрибутов, однако атрибутов может и совсем не быть. Спецификация атрибута состоит из расположенных в следующем порядке:

- имени атрибута, например WIDTH;
- знака равенства (=);
- значения атрибута, которое задается строкой символов, например, "80".

Всегда полезно заключить значение атрибута в кавычки, используя либо одинарные ('80'), либо двойные кавычки ("80"). Строка в кавычках не должна содержать такие же кавычки внутри себя. Так, если дата заключена в двойные

кавычки, используйте одинарные кавычки для последующего заключения в кавычки, и наоборот. Предпочтительно использование двойных кавычек, так как для глаза человека бывает трудно отличить одинарные кавычки от символов, подобных символам акцентирования.

Вы можете также опустить кавычки для значений атрибутов, которые состоят только из следующих символов (обратитесь к технической концепции имени):

· символов английского алфавита (A - Z, a - z) · цифр (0 - 9);

· промежутков времени;

· дефисов (-).

1.5. Чувствительность к регистру

Что касается имен тегов, атрибутов и большинства значений атрибутов, html является не чувствительным к регистру языком. Вы можете, например, написать TITLE, или Title, или title, или даже tItLE, если Вам нравится. Но существуют и чувствительные к регистру конструкции языка, а именно: escape последовательности (более официально называемые символьными объектами), которые начинаются знаком & (например, <) UrL, так как он может содержать наименования файлов, которые являются case чувствительными во многих операционных системах (например, в Unix).

1.6. Разделение на строки и использование пробелов и символов табуляции

Когда документ выводится на экран, пробелы и пустые линии не сохраняются, за исключением текста, заключенного в теги Pre (предварительно отформатированный текст). То есть ***любая последовательность пробелов, символов табуляции и пустых линий эквивалентна единственному пробелу в файле HTML***. С другой стороны, пробел в файле html может быть представлен с использованием любого количества пробелов или новыми (пустыми) строками. Таким образом, не имеет значения, как разделен текст на строки, так как перевод строки эквивалентен пробелу. Заметим, однако, что Вы не должны в html разделять слово на две строки.

2. СТРУКТУРА HTML-ДОКУМЕНТА

Для того, чтобы текстовый файл превратился в HTML-файл, поменять его расширение с ".txt" на ".html" недостаточно. Надо соблюсти "правило первой строки".Каждый HTML-документ, отвечающий спецификации HTML какой-либо версии, обязан начинаться со строки декларации версии HTML **!DOCTYPE**, которая обычно выглядит так:

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

Эта строка поможет браузеру определить, как правильно интерпретировать данный документ. В данном случае мы говорим браузеру, что HTML соответствует международной спецификации версии 3.2, которая хоть и не отличается новизной, но, в отличие от более поздних версий, является полноценным, широко распространенным стандартом без каких-либо неопределенностей. Как видно из примера, самый короткий html-документ состоит буквально из одной строки.

Пример самого короткого HTML-документа:

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

Начало работы. После объявления версии и типа документа необходимо обозначить его начало и конец. Это делается с помощью тега-контейнера <HTML>. Необходимо отметить, что любой HTML-документ открывается тегом <HTML> и им же закрывается. Затем, между тегами <HTML> и </HTML> следует разместить заголовок и тело документа. Так должен выглядеть ваш базовый HTML-файл перед началом работы:

Правильный пример самого короткого HTML-документа:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML> <HEAD>
        <TITLE>Заголовок документа</TITLE>
    </HEAD>
    <BODY>
    Текст документа
    </BODY>
</HTML>
```

Если приведенный выше пример пояснить схематически, получится следующее:

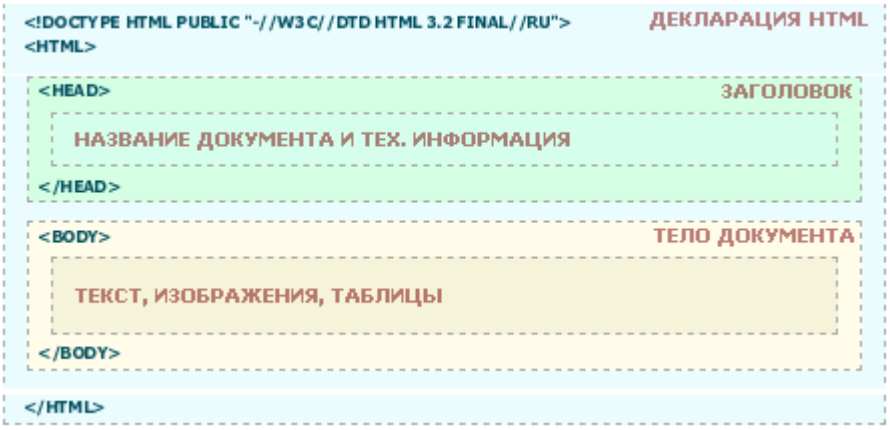


Рис. 2.1

Из схемы видно, что документ состоит из двух основных блоков – "***заголовка***" и "***тела документа***". Заголовок определяется с помощью элемента **HEAD**, а тело – элементом **BODY**.

Заголовок содержит "техническую" информацию о документе, хотя чаще всего используется только для обозначения его названия (см. элемент **TITLE**). В теле документа находится все то, что отображается на странице: текст, картинки, таблицы.

3. ВСТАВКА КОММЕНТАРИЕВ

<!-- -->

Используется для создания комментариев в любой части документа. Все, что находится внутри <!-- -->, будь то элемент или текст – будет проигнорировано браузером (не будет обрабатываться и выводиться на экран).

Рекомендуется комментировать все, что написали. Это волшебное правило сэкономит вам впоследствии немало времени.

4. ЗАГОЛОВОК HTML-ДОКУМЕНТА

Создается с помощью элемента **HEAD**, между тегами которого размещаются элементы, содержащие техническую информацию о документе.

Заголовок обычно располагается до тела документа (см. структуру HTML-документа).

Элементы, относящиеся к заголовку документа:

HEAD	Определяет начало и конец заголовка документа
TITLE	Определяет имя всего документа, которое отображается в заголовке окна браузера
BASE	Определяет базовый адрес, от которого отсчитываются относительные линки внутри документа
STYLE	Используется для вставки в документ таблицы стилей CSS
LINK	Описывает взаимосвязь документа с другими объектами
META	Используется для вставки метаданных

HEAD

Определяет начало и конец ***заголовка документа***. Является контейнером для элементов, содержащих техническую информацию о документе. (**TITLE**, **BASE**, **STYLE**, **LINK**, **META**).

Пример:

```
<HTML>
<!-- Начинаем заголовок... -->
<HEAD>
   <title>Справочник по HTML</title>
</HEAD>
<!-- ...Окончание написания заголовка. Дальше следует тело документа -->

<BODY>
Текст документа
</BODY>

</HTML>
```

TITLE

Определяет имя всего документа. Имя, как правило, отображается ***в заголовке окна*** браузера. Данный элемент *обязателен* для любого HTML-документа и может быть указан не более одного раза.

Пример:

```
...
<HEAD>
   <TITLE>Руководство по эксплуатации</TITLE>
</HEAD>
...
```

BASE

Указывает **базовый адрес** текущего документа (URL), который станет отправной точкой для расчета относительных адресов внутри документа. Элемент не имеет конечного тега. Обязательно присутствие хотя бы одного из атрибутов.

Атрибуты:

HREF – определяет базовый адрес (URL) текущего документа.

TARGET – определяет имя фрейма, которое будет использоваться в гиперссылках по умолчанию. Это может вам пригодиться, если вы хотите открывать все ссылки документа в другом фрейме.

STYLE

Используется для вставки в документ таблицы стилей (CSS – Cascade Style Sheet).

Атрибуты:

TYPE – обязательный атрибут. Определяет MIME-тип вставляемого блока стилей. Как правило, значением этого атрибута является "text/css".

TITLE – определяет имя создаваемой таблицы стилей. Необходим, если вы собираетесь использовать несколько элементов STYLE в одном документе.

В этом случае браузер должен спросить пользователя, какой из предложенных стилей будет применен к документу.

Пример:

```
<HEAD>
  <TITLE>Пример использования таблицы стилей</TITLE>
  <!—создаем табличку стилей -->
  <STYLE TYPE="text/css" TITLE="Красивая таблица стилей">
    <!--
    A {text-decoration : none;}
    P {color : blue; font-size : 12pt; font-family : Arial;}
    H1 {color : red; font-size : 18pt;}
    -->
  </STYLE>
  <!-- ... окончание создания таблички стилей -->
</HEAD>
```

Более подробно этот материал будет рассмотрен в дополнении к данному учебнику “Таблица Стилей CSS”

META

Элемент **META** используется для технического описания документа, которое представляет собой метаданные в виде пары "имя-значение". С помощью этого элемента в заголовок документа внедряется дополнительная полезная информация, невидимая для пользователя, но порой просто незаменимая для правильной индексации вашей страницы роботами поисковых серверов.

Элемент *не имеет конечного тега.*

Атрибуты:

NAME – определяет имя мета-записи. Существует множество предопределенных имен, некоторые из которых вы можете увидеть в указанном ниже примере.

Пример:

```
<HEAD>
...
<META HTTP-EQUIV="Expires" CONTENT="Sat, 26 Jun 1999 17:38:15 GMT">
<META NAME="GENERATOR" CONTENT="Greenback">
<META NAME="Publisher-Email" CONTENT="green@ua.fm">
<META NAME="Publisher-URL" CONTENT="Idea GraFix &#150; http://www.igf.ru/">
<META NAME="Keywords" CONTENT="OpenGL,3D,graphics,3Dfx,Permedia,Diamond,графика">
...
</HEAD>
```

HTTP-EQUIV – определяет имя мета-записи. Практически аналогичен атрибуту NAME, но используется лишь в случае необходимости передачи дополнительной информации в HTTP-заголовке.

CONTENT – присваивает значение мета-записи, определенной в атрибуте NAME (или HTTP-EQUIV).

5. ФРЕЙМЫ

Фреймы (frames) используются для разбивки окна браузера на несколько областей, каждая из которых представляет собой отдельный HTML-документ (фрейм). Как правило, фреймы используются для облегчения навигации по сайту, создания навигационного меню.

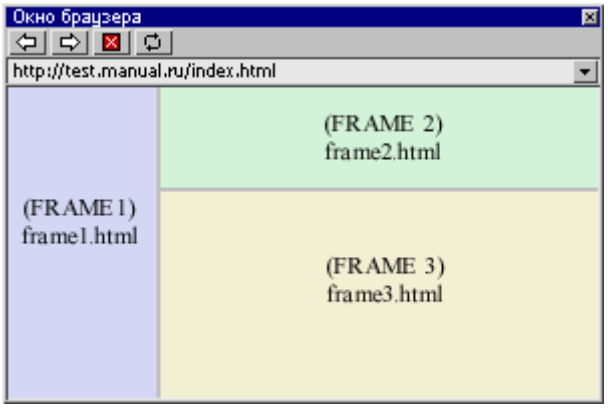


Рис. 5.1

Элементы для создания фреймов и работы с ними:

FRAMESET	Определяет фреймовую (оконную) структуру документа: размеры и расположение фреймов на странице
FRAME	Определяет фрейм и его свойства внутри FRAMESET-структуры
NOFRAMES	Определяет, что показывать, если браузер не поддерживает фреймы

Фреймы не относятся к телу документа. Внимательно следите, чтобы все вышеперечисленные элементы находились вне элемента BODY.

FRAMESET определяет фреймовую (оконную) структуру документа: размеры и расположение фреймов на странице. Создаётся *вместо* тела документа (то есть элемент **BODY** в документе *не используется*). Открывает и закрывает список фреймов, определяемых с помощью элемента **FRAME**. Между начальным и конечным тегами кроме элементов **FRAME** и **NOFRAMES** могут находиться другие элементы **FRAMESET**. То есть элемент **FRAMESET** поддерживает вложенные конструкции фреймов.

Атрибуты:

ROWS – определяет количество и размеры горизонтальных фреймов (фреймов-строк) в окне браузера. В качестве значения задается список размеров фреймов через запятую. Способы задания размеров:

- в процентах от высоты рабочей области окна браузера. Например: "30%,30%,40%";
- в виде знака "*" (звездочка), говорящего о том, что фрейм занимает все свободное пространство окна браузера, незанятое другими фреймами с явно указанными размерами. Например, звездочка в записи "25%,25%,*" равносильна 50%;
- в пикселах. Например: "75,*";

Все три способа можно совмещать. Например, "25%,40,*" разобьет экран на три горизонтальных фрейма, первый будет высотой в четверть окна браузера, второй – в 40 пикселов, а третий займет всю оставшуюся площадь.

COLS – определяет количество и размеры вертикальных фреймов (фреймов-столбцов) в окне браузера. В качестве значения задается список

размеров фреймов через запятую. Размеры задаются так же, как и в предыдущем атрибуте ROWS.

BORDER – определяет ширину рамок фреймов в пикселях. Данный атрибут действует только в браузерах Netscape;

FRAMEBORDER – определяет наличие рамок у содержащихся внутри FRAMESET фреймов. Возможные значения: Yes – отображать рамки; No или 0 – не отображать рамки;

Браузеры Netscape не поддерживают данный атрибут в полной мере, и для глобального определения ширины рамок используют атрибут BORDER.

FRAMESPACING – определяет расстояние (так называемую "серую область") между фреймами в пикселях. Данный атрибут необходим для создания фреймов без рамок.

Обратите внимание на то, что FRAMESET-структура создаётся вместо элемента BODY. FRAMESET-структура, расположенная в заголовке (внутри элемента HEAD) считается ошибкой.

FRAME - определяет фрейм и его свойства внутри FRAMESET-структуры (см. элемент FRAMESET) .

Атрибуты:

SRC – обязательный атрибут. Указывает адрес (URL) HTML-файла, отображаемого в данном фрейме.

NAME – определяет имя данного фрейма, которое будет в дальнейшем использоваться для ссылки на него из других документов с помощью атрибута TARGET (см. элемент A). В качестве значения нужно указать любое имя без пробелов с использованием латинских символов и цифр. Имя не должно начинаться с цифр и специальных символов.

MARGINWIDTH – определяет ширину (в пикселях) левого и правого полей фрейма. Если атрибут не указан, браузер самостоятельно определит оптимальный размер отступа.

MARGINHEIGHT – определяет ширину (в пикселях) верхнего и нижнего полей фрейма. Если атрибут не указан, браузер самостоятельно определит оптимальный размер отступа.

SCROLLING – определяет наличие линеек прокрутки содержимого фрейма. Возможные значения:

yes – отображать линейки прокрутки.

no – не отображать линейки прокрутки.

auto – отображать линейки прокрутки при необходимости (если документ, указанный в атрибуте SRC, не умещается во фрейме).

NORESIZE – не позволяет изменять размеры фрейма. Данный атрибут является флагом и не требует указания значения.

FRAMEBORDER – определяет наличие рамок у фрейма. Возможные значения: yes – отображать рамки; no или 0 – не отображать рамки;

Браузеры Netscape не поддерживают данный атрибут в полной мере и для глобального определения ширины рамок используют атрибут BORDER элемента FRAMESET.

Пример (файл index.html):

```
...
<FRAMESET FRAMEBORDER="0" FRAMESPACING="0"
        BORDER="0" COLS="265,*">
<FRAME SRC="frame1.html" NAME="page">
<FRAMESET ROWS="165,*">
<FRAME SRC="frame2.html" NAME="menu1" MARGINWIDTH="0">
<FRAME SRC="frame3.html" NAME="menu2" MARGINWIDTH="0">
</FRAMESET>
 <NOFRAMES>Ваш браузер не поддерживает фреймы</NOFRAMES>
</FRAMESET>
<BODY>
</BODY>
...
```

В результате окно браузера разделится на три фрейма, как показано ниже. Причем **frame1.html** будет иметь ширину 265 пикселов, а **frame2.html** – высоту 165 (смотри рис.2.1).

NOFRAMES

Все, что находится между начальным и конечным тегами данного элемента, будет отображено браузером, если он не поддерживает фреймы. Элемент **NOFRAMES** не имеет атрибутов и должен находиться внутри элемента **FRAMESET**.

Пример:

```
<FRAMESET ROWS="*,*">
<NOFRAMES>Ваш браузер не поддерживает фреймы</NOFRAMES>
   <FRAME SRC="frame1.html">
   <FRAME SRC="frame2.html">
</FRAMESET>
```

6. СКРИПТЫ

Скрипты – это включения в HTML не-html кода, дополняющего его возможности. С помощью скриптов можно создавать анимированные кнопки меню, осуществлять автоматическое перенаправление на другие документы и т.д. Большинство скриптов пишется на языке JavaScript.

Элементы для работы со скриптами:

SCRIPT	Вставляет скрипт в HTML-документ
NOSCRIPT	Определяет текст, который будет отображен, если браузер не поддерживает скрипты

Более подробно этот материал будет рассмотрен в дополнении к данному учебнику “Элементы языка JavaScript”

BODY

Указывает начало и конец тела HTML-документа. Между начальным и конечным тегами содержится текст документа, изображения и таблицы. Одним словом, все HTML-элементы, отвечающие за отображение документа, управление им и гипертекстовые ссылки. Элемент **BODY** должен встречаться в документе не более одного раза.

Атрибуты:

MARGINHEIGHT – определяет ширину (в пикселях) верхнего и нижнего полей документа. Работает только в браузерах Netscape.

TOPMARGIN – определяет ширину (в пикселях) верхнего и нижнего полей документа. Работает только в браузерах Internet Explorer.

MARGINWIDTH – определяет ширину (в пикселях) левого и правого полей документа. Работает только в браузерах Netscape.

LEFTMARGIN – определяет ширину (в пикселях) левого и правого полей документа. Работает только в браузерах Internet Explorer.

BACKGROUND – определяет изображение для "заливки" фона. Значение задается в виде полного URL или имени файла с картинкой в формате GIF или JPG.

BGCOLOR – определяет цвет фона документа.

TEXT – определяет цвет текста в документе.

LINK – определяет цвет гиперссылок в документе.

ALINK – определяет цвет подсветки гиперссылок в момент нажатия.

VLINK – определяет цвет гиперссылок на документы, которые вы уже просмотрели.

Значения атрибутов BGCOLOR, TEXT, LINK, ALINK и VLINK задаются либо RGB-значением в шестнадцатеричной системе, либо одним из 16 базовых цветов.

Пример:

```
<HTML>
<BODY BACKGROUND="images/bricks.jpg" BGCOLOR="#202020"
 TEXT="#FFFFFF" LINK="#FF0000" VLINK="#505050" MARGINHEIGHT="30"
 TOPMARGIN="30" LEFTMARGIN="40" MARGINWIDTH="40">
...
Гекст документа.
...
</BODY>
</HTML>
```

При задании ширины полей для обеспечения совместимости со всеми видами браузеров используйте одновременно атрибуты MARGINWIDTH/MARGINHEIGHT и TOPMARGIN/LEFTMARGIN, как показано в примере.

Всегда указывайте атрибуты BGCOLOR и TEXT одновременно. Если один из этих атрибутов не указан, браузер по умолчанию будет использовать цвет из текущей цветовой схемы Windows.

7. ГИПЕРССЫЛКИ

Ссылки на другие документы в HTML создаются либо с помощью элемента **A**, либо с помощью навигационных карт.

Элемент A (Anchor в переводе означает «якорь») применяется, если ссылкой планируется сделать часть текста или целое изображение. Навигационные карты имеет смысл применять, если ссылкой будет часть изображения.

A (Anchor)

Самый необходимый элемент, без которого работа в Интернете невозможна. Используется для создания и использования гипертекстовых ссылок.

Атрибуты:

HREF – определяет находящийся между начальным и конечным тегами текст или изображение как гипертекстовую ссылку (URL, или линк) на документ (и/или область документа), указанный в значении данного атрибута. Возможные значения:

http://... – создает ссылку на www-документ;

ftp://... – создает ссылку на ftp-сайт или расположенный на нем файл;

mailto:... – запускает почтовую программу-клиент с заполненным полем имени получателя. Если после адреса поставить знак вопроса, то можно указать дополнительные атрибуты, разделенные знаком "&";

news:.. – создает ссылку на конференцию сервера новостей;

gopher://... – создает ссылку на Gopher – сервер;

Если тип соединения и адрес машины не указаны, в качестве отправной точки используется адрес текущего документа. Это позволяет использовать относительные ссылки.

Например, <A HREF="docs/title.html">Документация</A> будет ссылаться на файл *title.html* в подкаталоге *docs* (относительно текущего).

NAME – помечает находящуюся между начальным и конечным тегами область документа как возможный объект для ссылки. В качестве значения нужно латиницей написать любое слово-указатель, уникальное для данного документа.

Например: <A NAME="part">Раздел1</A>. Теперь вы можете ссылаться на помеченную область простым указанием ее имени после имени документа. Например, <A HREF="document.html#part">Раздел1</A> отправит вас в раздел *"part"* файла *document.html*, а линк <A HREF="#bottom">В конец документа</A> – в раздел *"bottom"* текущего документа (см. Пример 1)

TARGET – определяет окно (фрейм), на которое указывает гипертекстовая ссылка. Этот атрибут используется только совместно с атрибутом HREF. В качестве значения необходимо задать либо имя одного из существующих фреймов, либо одно из следующих зарезервированных имен:

_self – указывает, что определенный в атрибуте HREF документ должен отображаться в текущем фрейме;

_parent – указывает, что документ должен отображаться во фрейме-родителе текущего фрейма. Иначе говоря, _parent ссылается на окно, содержащее FRAMESET, включающий текущий фрейм;

_top – указывает, что документ должен отображаться в окне-родителе всей текущей фреймовой структуры;

_blank – указывает, что документ должен отображаться в новом окне.

Пример 1:

<!-- *Использование атрибута NAME на примере создания онлайнового журнала Спорт:* -->
<A NAME="history">История бодибилдинга</A>
...
<A NAME="now">Спорт глазами современника</A>
...
Вернуться к разделу<A HREF="#history">истории</A>

В примере 2 обратите внимание, окно какой программы открывается при щелчке по гиперссылке и как располагаются элементы письма «Куда», «Кому», «От кого», «Тема» и само содержание.

Пример 2:

```
<!-- Создадим ссылку для письма с указанием нескольких атрибутов -->
<A HREF="mailto:grinko@mail.ru?subject=Приглашение
&cc=bg@microsoft.com&body=Приезжай на концерт нашего
факультета."> Отправить приглашение </A>.
<!-- или просто письмо : -->
<A HREF="mailto: grinko@mail.ru?subject=Привет">авторам</A>.
```

Элемент **A** не может быть вложенным в себе подобные элементы.

Если в атрибуте TARGET указать имя несуществующего окна или фрейма, создастся новое окно с указанным именем. Как мы видим, результат получится тот же, что и при задании нового окна: <A HREF="..." TARGET="_blank">, с той лишь разницей, что в последнем случае окно не будет иметь имени, и на него нельзя будет ссылаться.

8. НАВИГАЦИОННЫЕ КАРТЫ

Можно создать активный рисунок с гиперссылками, в котором разные части картинки имеют разные ссылки – такое средство принято называть активными картами. Это можно сделать путем включения информации об изображении в документ.

Для задания информации о гиперссылках в рисунке, которая включается в HTML документ, используется атрибут USERMAP= в теге IMG. Сама информация о гиперссылках определяется тегами MAP и AREA, как показано в примере 3.

В этом примере рисунок map1.gif был размечен равными прямоугольниками, при этом одна область является непомеченной, остальные три помечены документами h1.html, h2.html, h3.html соответственно. Тег AREA позволяет помечать области кругами, если используется параметр **circles** , или ломаными тогда применяют параметр **polygons**. Следует иметь в виду, что если две или более областей пересекаются, то браузер выбирает первую из тех, что

описаны в теге MAP, а непомеченные области никак не отвечают на действия пользователя

Пример 3:

```
<MAP NAME=”map1”>
< AREA SHAPE=”rect” coords=”0, 0, 20, 20” HREF=”h1.html”>
< AREA SHAPE=”rect” coords=”20, 0, 40, 20” NOHREF>
< AREA SHAPE=”rect” coords=”0, 20, 20, 40” HREF=”h2.html”>
< AREA SHAPE=”rect” coords=”20, 20, 40, 40” HREF=”h3.html”>
< AREA SHAPE=default HREF=”other.html”>.
</MAP>
<IMG BORDER=0   SRC=”map1.gif” USEMAP=”#map1”>
```

9. ТЕКСТОВЫЕ БЛОКИ

В этом разделе описаны элементы, разбивающие текст документа на блоки тем или иным способом. Типичными примерами текстовых блоков являются параграфы, абзацы и главы. Для отделения одной части текста от другой также используются разделительные горизонтальные линии и символы возврата каретки.

Элементы:

H1,H2,...H6	Используются для создания заголовков текста
P	Используется для разметки параграфов.
DIV	Отделяет блок HTML-документа от остальной его части
ADDRESS	Оформляет текст как почтовый адрес
BLOCKQUOTE	Оформляет текст как цитату
BR	Осуществляет перевод строки
HR	Вставляет в текст горизонтальную разделительную линию.
PRE	Включает в документ (моноширинным шрифтом) блок предварительно отформатированного текста
LISTING, PLAINTEXT, XMP	Включают в документ (моноширинным шрифтом) блок предварительно отформатированного текста (устаревшие элементы)

Существует шесть уровней заголовков, различающихся величиной шрифта. С их помощью можно разбивать текст на смысловые уровни – разделы и подразделы.

Атрибуты:

ALIGN – определяет способ выравнивания заголовка по горизонтали.

Возможные значения: left, right, center.

Пример:

```
<H1 ALIGN="center">Самый большой заголовок посередине</H1>
<H2>Заголовок поменьше</H2>
<H3>Еще меньше </H3>
...
<H6>Совсем маленький заголовок</H6>
```

P (Paragraph)

Используется для разметки параграфов.

Атрибуты:

ALIGN – определяет способ горизонтального выравнивания параграфа.

Возможные значения: left, center, right. По умолчанию имеет значение left.

Пример:

```
<P ALIGN="center">Это центрированный параграф.<BR>
Текст располагается в центре окна браузера</P>

<P ALIGN="right">А это параграф, выровненный по правому краю.</P>
```

DIV (Division)

Используется для логического выделения блока HTML-документа. Элемент группировки, как и элемент **SPAN**. В современном «сайтостроении» используется как удобный контейнер для объектов страницы, которым легко динамически манипулировать – перемещать, включать/выключать, создавать слои, регулировать отступы и т.п.

Находящиеся между начальным и конечным тегами текст или HTML-элементы по умолчанию оформляются как отдельный параграф.

Атрибуты:

ALIGN – определяет выравнивание содержимого элемента **DIV**. Атрибут может принимать значения: left, right, center.

Пример:

```
...Текст документа...
<DIV ALIGN="center">
...Текст, таблицы, изображения. Выравнивание по центру.
</DIV>
...Текст документа...
```

У атрибута ALIGN есть еще одно значение – justify, поддерживаемое современными браузерами. Оно позволяет выравнивать текст по ширине (одновременно по левому и правому краям документа). Не понимающие justify браузеры будут выравнивать текст по левому краю.

ADDRESS

Находящийся между начальным и конечным тегами ADDRESS текст оформляется ***как почтовый адрес***. Чаще всего оформление выражается в выделении строки адреса курсивом.

Пример:

```
Пишите по следующему адресу:
<ADDRESS>
   Днепропетровск 49000, пр. Карла Маркса,65,к.23 <BR>
   Издательство газеты ДНЕПР
</ADDRESS>
```

BLOCKQUOTE

Оформляет находящийся между начальным и конечным тегами текст ***как цитату.*** Используется для длинных цитат (в отличие от элемента **CITE**). Цитируемый текст отображается отдельным абзацем с увеличенным отступом.

Пример:

```
Редакция журнала "Домосед" выражает благодарность
Бухаресту Магарычу Шницелю за замечательное стихотворение:
<BLOCKQUOTE>
   Синели красные ромашки,<BR>
   Желтели в небе облака,<BR>
   Синицы в теплый край летели,<BR>
   К истоку двигалась река.<BR>
   ...
</BLOCKQUOTE>
```

BR (Break)

Данный элемент осуществляет перевод строки, то есть практически аналогичен нажатию Enter в текстовом редакторе. После того, как в браузерах появилась возможность обтекания изображения текстом (см. атрибут ALIGN элемента **IMG**), понадобился дополнительный атрибут CLEAR. Элемент **не имеет конечного тега.**

Атрибуты:

CLEAR – указывает на необходимость завершения обтекания изображения текстом. Может принимать следующие значения :

all – завершить обтекание изображения текстом.

left – завершить обтекание текстом изображения, выровненного по левому краю.

right – завершить обтекание текстом изображения, выровненного по правому краю.

Пример:

```
Первое предложение<BR>Второе предложение на следующей строке
```

Возьмите себе за правило всегда ставить
 после тега <img ...>. В противном случае все картинки будут иметь неприятный отступ.

HR (Horisontal Rule)

Вставляет в текст горизонтальную разделительную линию.

Атрибуты:

WIDTH – определяет длину линии в пикселах или процентах от ширины окна браузера.

SIZE – определяет толщину линии в пикселах.

ALIGN – определяет выравнивание горизонтальной линии. Атрибут может принимать следующие значения: left – выравнивание по левому краю документа; right – выравнивание по правому краю документа; center – выравнивание по центру документа (используется по умолчанию).

NOSHADE – определяет способ закраски линии как сплошной. Атрибут является флагом и не требует указания значения. Без данного атрибута линия отображается объемной.

COLOR – определяет цвет линии. Задается либо RGB-значением в шестнадцатиричной системе, либо одним из *16 базовых цветов*. Атрибут работает только в Internet Explorer.

Пример:

Вставьте любой текст и отделите его сплошной горизонтальной линией при помощи тега

<HR NOSHADE WIDTH="50%">.

PRE (Preformatted Text)

Используется для включения в документ ***уже отформатированного*** текста. Браузеры воспроизводят содержимое этого элемента с помощью моноширинного шрифта, сохраняя пробелы и символы конца строки.

Пример:

```
<PRE>
Один!
   Два!
      Три!
</PRE>
```

Желательно избегать использования символа горизонтальной табуляции внутри **PRE**, т.к. он может быть неадекватно интерпретирован некоторыми браузерами. Вместо символа табуляции рекомендуется использовать число пробелов, кратное четырем.

10. ФОРМАТИРОВАНИЕ ТЕКСТА

В этом разделе описаны элементы для оформления и смыслового выделения текста – подчеркивания, изменения шрифта, выделения курсивом, цитирования и т.д.

Элементы форматирования текста:

BASEFONT	Определяет основной шрифт, которым должен отображаться текст документа
FONT	Позволяет изменять цвет, размер и тип шрифта текста
I	Выделяет текст курсивом
EM	Используется для смыслового выделения текста (курсивом)
B	Выделяет текст жирным шрифтом
STRONG	Усиленное выделение текста (жирным)
U	Выделяет текст подчеркнутым
S, STRIKE	Выделяет текст перечеркнутым
BIG	Отображает текст увеличенным шрифтом (относительно текущего)
SMALL	Отображает текст уменьшенным шрифтом (относительно текущего)
SUP	Отображает текст со сдвигом вверх (верхний индекс)
SUB	Отображает текст со сдвигом вниз (нижний индекс)
CODE, SAMP	Оформляют текст как формулу или программный код
TT	Отображает текст моноширинным шрифтом
KBD	Выделяет текст, который предлагается набрать на клавиатуре
VAR	Используется для обозначения в тексте переменных
CITE	Оформляет текст как цитату или ссылку на источник

BASEFONT

Не имеет конечного тега. Определяет основной шрифт, которым должен отображаться текст документа.

Впоследствии вы можете легко изменить шрифт в любой части документа, используя элемент **FONT**. Действие элемента **BASEFONT** не распространяется на текст, заключенный в ячейки таблиц.

Атрибуты:

SIZE – обязательный атрибут. Определяет базовый размер шрифта. Возможные значения : целые числа от 1 до 7 включительно.

FACE – определяет используемый шрифт (гарнитуру).

Пример:

```
<BODY>
<BASEFONT SIZE="3">
...
Текст документа имеет шрифт 3 размера
...
   <FONT SIZE="+1">
   Слегка увеличиваем шрифт
   </FONT>
...
Продолжаем шрифтом 3 размера
...
</BODY>
```

FONT

Позволяет изменять *цвет, размер и тип шрифта* текста, находящегося между начальным и конечным тегами. Вне тегов <FONT> и </FONT> используется шрифт, указанный в элементе BASEFONT.

Атрибуты:

SIZE – определяет размер шрифта. Возможные значения:

– целое число от 1 до 7;

– относительный размер с указанием знака, вычисляется путем сложения с базовым размером, определенным с помощью атрибута SIZE элемента BASEFONT.

COLOR – определяет цвет текста. Задается либо RGB-значением в шестнадцатеричной системе, либо одним из 16 базовых цветов.

FACE – определяет используемый шрифт.

Пример:

```
<FONT SIZE="+2" COLOR="#AA0000">Увеличенный красный
шрифт</FONT>
<FONT SIZE="3" FACE="Courier New" COLOR="Magenta">Моноширинный
фиолетовый текст 3 размера</FONT>
```

Атрибут **FACE** старайтесь не использовать. Если на компьютере пользователя не окажется указанного шрифта – ошибки неизбежны. Если указать шрифт всё же необходимо, используйте только апробированные : Times New Roman, Arial, Tahoma, Courier, Courier New.

I (Italic)

Текст, заключенный между начальным и конечным тегами, будет выделен ***курсивом***.

Пример:

```
Текст с <I>курсивом</I>
```

EM (Emphasis)

Используется ***для смыслового выделения*** текста, стоящего между начальным и конечным тегами. Результат обычно отображается *курсивом*. То есть элемент **EM** по действию практически аналогичен элементу **I**.

Пример:

```
Порой в Украине встречаются <EM>действительно талантливые</EM> веб-
мастера. Но только не друг с другом.
```

STRONG

Усиленное выделение. Текст, заключенный между начальным и конечным тегами, будет выделен ***жирным*** шрифтом. То есть элемент **STRONG** практически аналогичен по действию элементу **B**.

Пример:

```
Я <STRONG>сильный</STRONG>, но легкий.
```

B (Bold)

Текст, заключенный между начальным и конечным тегами, будет выделен **жирным** шрифтом.

Пример:

Текст с <B>выделенным</B> словом

U (Underline)

Данный элемент отображает помещенный между начальным и конечным тегами текст как подчеркнутый.

Пример:

<U> Подчеркнутый текст </U>

S, STRIKE

Элемент **STRIKE** отображает помещенный между начальным и конечным тегами текст как перечеркнутый (~~перечеркнутый~~). В HTML 3.2 вместо <STRIKE> был предложен более краткий тег <S>.

Пример:

Лена + <STRIKE>Вася</STRIKE> Коля = Love

BIG

Текст, заключенный между начальным и конечным тегами, отображается ***увеличенным шрифтом*** (относительно текущего).

Пример:

Текст с <BIG>увеличенным</BIG> словом

Тег <BIG> аналогичен по действию тегу <FONT SIZE="+1"> (см. элемент **FONT**).

SMALL

Текст, заключенный между начальным и конечным тегами, отображается ***уменьшенным*** шрифтом (относительно текущего).

Пример:

Гулливер попал в страну <SMALL>лилипутов</SMALL>

Тег <**SMALL**> аналогичен по действию тегу <FONT SIZE="-1"> (см. элемент **FONT**).

SUP (Superscript)

Отображает текст со сдвигом вверх (верхний индекс) и уменьшением размера текущего шрифта на единицу.

Пример:

Microsoft ^{TM}

SUB (Subscript)

Отображает текст со сдвигом вниз (нижний индекс) и уменьшением размера текущего шрифта на единицу.

Пример:

X_i = B_i + C_i

CODE, SAMP

Данные элементы оформляют текст, находящийся между начальным и конечным тегами, ***как формулу или программный код***. Текст при этом как правило отображается моноширинным шрифтом.

Пример:

Формула: <CODE> n=((x*15-z/1.25)/4)^5 </CODE>

KBD (Keyboard)

Предназначен для отображения текста, который пользователь должен набрать на клавиатуре (например, при заполнении формы, введении пароля и т.п.). Как правило, текст, заключенный между начальным и конечным тегами, выделяется жирным моноширинным шрифтом.

Пример:

Чтобы войти в систему наберите <KBD>"GUEST"</KBD> заглавными буквами.

VAR (Variable)

Элемент **VAR** предназначен для обозначения в тексте переменных. Как правило, отображается *курсивом*.

Пример:

Переменная <VAR>IndexZ</VAR> равна 5.

CITE (Citation)

Оформляет находящийся между начальным и конечным тегами текст как цитату или ссылку на источник. Обычно используется для коротких цитат (в отличие от элемента **BLOCKQUOTE**). Цитируемый текст отображается курсивом.

Пример:

Как сказал классик:
<CITE>" HTML все возрасты покорны "</CITE>

11. СПИСКИ

Списки в HTML бывают двух видов: упорядоченные (пронумерованные) и неупорядоченные (непронумерованные). Отличаются они лишь способом оформления. Перед пунктами неупорядоченных списков обычно ставятся символы-буллеты (bullets), например, точки, ромбики и т.п., в то время как пунктам упорядоченных списков предшествуют их номера.

***Элементы списка*:**

UL	Создает неупорядоченный список
OL	Создает упорядоченный список
LI	Создает пункт меню внутри элементов OL или UL
MENU, DIR	Создает неупорядоченный список, подобный UL
DL	Открывает и закрывает список определений
DT	Создает термин в списке определений внутри элемента DL
DD	Создает определение термина внутри элемента DL

UL (Unsorted List)

Создает ***неупорядоченный*** список. Между начальным и конечным тегами должны присутствовать один или несколько элементов **LI**, обозначающих отдельные пункты списка.

Пример:

```
<UL>
   <LI> Первый пункт списка </LI>
   <LI> Второй пункт списка </LI>
   <LI> Третий пункт списка </LI>
</UL>
```

OL (Ordered List)

Создает упорядоченный список. Между начальным и конечным тегами должны присутствовать один или несколько элементов **LI**, обозначающих отдельные пункты списка.

Атрибуты:

START – определяет первое число, с которого начинается нумерация пунктов. (только целые числа)

TYPE – определяет стиль нумерации пунктов. Может иметь значения:

"A" – заглавные буквы A, B, C ...

"a" – строчные буквы a, b, c ...

"I" – большие римские числа I, II, III ...

"i" – маленькие римские числа i, ii, iii ...

"1" – арабские числа 1, 2, 3 ...

По умолчанию <UL TYPE="1">.

Пример:

```
<OL TYPE="I" START="2">
   <LI> Пункт два </LI>
   <LI> Пункт три </LI>
   <LI> Пункт четыре </LI>
</OL>
```

LI (List Item)

Создает ***пункт*** в списке. Располагается внутри элементов **OL** или **UL**.

Атрибуты:

VALUE – изменяет порядок нумерации элементов списка. Используется только если элемент LI находится внутри элемента OL. В качестве значения указывается порядковый номер элемента.

Помимо параметра TYPE="1" существует еще ряд других параметров. Посмотрите на следующую таблицу:

В этой таблице представлено 5 html-кодов и их результаты. По сути они одно и то же, но изменение параметра TYPE="..." дает знать о себе различными результатами.

В первом случае мы видим простой нумерованный список с арабскими цифрами.

Во втором – список маркированный заглавными латинскими буквами.

В третьем - маркировка строчными латинскими буквами.

В четвертом - маркировка большими римскими цифрами.

В пятом - маленькие римские цифры.

<ul type="1"> первая строка вторая строка третья строка 	<ul type="A"> первая строка вторая строка третья строка 	<ul type="a"> первая строка вторая строка третья строка 	<ul type="I"> первая строка вторая строка третья строка 	<ul type="i"> первая строка вторая строка третья строка
1. первая строка 2. вторая строка 3. третья строка	A. первая строка B. вторая строка C. третья строка	a. первая строка b. вторая строка c. третья строка	I. первая строка II. вторая строка III. третья строка	i. первая строка ii. вторая строка iii. третья строка

MENU, DIR (Menu, Directory List)

Данные теги предназначены для создания неупорядоченных списков, подобных **UL**. Эти элементы не вошли в спецификацию HTML 4.0, однако браузеры могут распознавать их в целях обратной совместимости.

DL (Definition List)

Открывает и закрывает список определений (терминов и их описаний). Определения задаются с помощью элементов **DT** и **DD**.

Атрибуты:

COMPACT – включает режим "компактного" отображения списка. Удобно использовать, если список определений очень велик. Данный атрибут является флагом и не требует присвоения значения.

Пример:

```
<DL COMPACT>
   <DT>Жапура<DD>Река, протекающая в Колумбии и Бразилии
   <DT>Зайцы<DD>Семейство млекопитающих отряда зайцеобразных
   ...
</DL>
```

12. ОБЪЕКТЫ

Объекты – это графические и мультимедийные вставки в HTML-документ, такие как картинки, Flash-анимация, Java-апплеты, звуки, музыка, VRML.

Элементы:

IMG	Используется для вставки в HTML изображений
EMBED	Используется для вставки в HTML различных объектов
NOEMBED	Используется, если браузер не поддерживает элемент EMBED
APPLET	Используется для вставки в HTML Java-апплетов
PARAM	Используется для передачи параметров Java-программе (см. элемент APPLET)

IMG

(HTML 2.0) – Image

Используется для вставки изображений в HTML-документ.

Это один из самых популярных элементов, незаменимый инструмент web-дизайнера. Элемент допускает вставку изображений в форматах JPEG (в том числе progressive jpeg) и Compuserve GIF (включая прозрачные и анимированные). Червертые версии браузеров позволяют также использовать формат PNG, но до тех пор, пока они не устареют, от применения PNG лучше воздержаться.

Элемент **IMG** не имеет конечного тега.

Атрибуты:

SRC – обязательный атрибут. Указывает адрес (URL) файла с изображением.

HEIGHT и WIDTH – определяют ширину и высоту изображения соответственно. Если указанные значения не совпадают с реальным размером изображения, изображение масштабируется (порой с заметной потерей качества).

HSPACE и VSPACE – определяют отступ картинки (в пикселах) по горизонтали и вертикали от других объектов документа. Просто необходимо при обтекании изображения текстом.

ALIGN – обязательный атрибут. Указывает способ выравнивания изображения в документе. Может принимать следующие значения:

LEFT – выравнивает изображение по левому краю документа. Прилегающий текст обтекает изображение справа.

RIGHT – выравнивает изображение по правому краю документа. Прилегающий текст обтекает изображение слева.

TOP и TEXTTOP – выравнивают верхнюю кромку изображения с верхней линией текущей текстовой строки.

MIDDLE – выравнивает базовую линию текущей текстовой строки с центром изображения.

ABSMIDDLE – выравнивает центр текущей текстовой строки с центром изображения.

BOTTOM и BASELINE – выравнивает нижнюю кромку изображения с базовой линией текущей текстовой строки.

ABSBOTTOM – выравнивает нижнюю кромку изображения с нижней кромкой текущей текстовой строки.

NAME – определяет имя изображения, уникальное для данного документа. Вы можете указать любое имя без пробелов с использованием латинских символов и цифр. Имя необходимо, если вы планируете осуществлять доступ к изображению, например, из JavaScript-сценариев.

ALT – определяет текст, отображаемый браузером на месте изображения, если браузер не может найти файл с изображением или включен в текстовый режим. В качестве значения задается текст с описанием изображения.

BORDER – определяет ширину рамки вокруг изображения в пикселах. Рамка возникает, только если изображение является гипертекстовой ссылкой. В таких случаях значение BORDER обычно указывают равным нулю.

LOWSRC – указывает адрес (URL) файла с альтернативным изображением более низкого качества (и, соответственно, меньшего объема), чем изображение, указанное в атрибуте SRC. Браузеры Netscape, поддерживающие данный атрибут, сначала загрузят картинку из LOWSRC, а затем заменят ее картинкой из SRC.

USEMAP – применяет к изображению навигационную карту (image map), заданную элементом MAP. В качестве значения задается имя карты с предшествующей ему решеткой. Например, если имя карты – *"map1"*, то ссылка на нее будет выглядеть как "#map1" (см. Пример 4). Обратите внимание:

прописные и строчные буквы в данном атрибуте трактуются браузером как разные.

ISMAP – определяет изображение как навигационную карту (image map), обрабатываемую сервером. Имеет смысл использовать только тогда, когда изображение является гиперссылкой. После клика мышкой на изображении серверу отправляются x, y- координаты нажатия. В зависимости от полученных координат, сервер (при наличии на нем соответствующего программного обеспечения) может показать вам тот или иной документ. Данный атрибут является флагом и не требует присвоения значения.

Пример 1:

<IMG src="/img/picture.gif" WIDTH="45" HEIGHT="53" ALT="Рысь" HSPACE="10" ALIGN="left"> Этот текст обтекает картинку справа и находится от нее на расстоянии 10 пикселов.

Пример 2. Использование изображения в качестве гиперссылки:

```
<A HREF="carlson.html">
<IMG src="/img/button.jpg" WIDTH="70" HEIGHT="30" ALIGN="right"
BORDER="0" ALT="Досье Карлсона"> </A>
```

Для просмотра досье нажмите на кнопку справа.

Пример 3. Использование ISMAP:

```
<A HREF="http://www.igf.ru/bin/imagemaps/map1">
    <IMG SRC="map.gif" ISMAP></A>');
```

Пример 4. Использование USEMAP:

```
<IMG src="/img/buttons.jpg" WIDTH="170" HEIGHT="120"ALIGN="middle"
BORDER="0" USEMAP="#ButtonsMap">');
```

Примечания (особо важно):

Золотое правило– всегда явно задавать размеры картинки в атрибутах HEIGHT и WIDTH, резервируя тем самым место в окне браузера еще до загрузки изображения. В противном случае документ при загрузке каждой картинки будет заново перерисовываться.

Второе золотое правило web-мастера: если на картинке изображено что-то разборчивое, нужно описать это словами в атрибуте ALT.

Всегда сразу после <IMG ...> ставьте вплотную <**BR**>! А то проблем не миновать – после картинки появится пустое пространство в несколько пикселей.

Для завершения обтекания изображения текстом используйте атрибут CLEAR элемента **BR**.

Указывайте значения атрибутов HSPACE и VSPACE, даже если вы хотите оставить поля нулевой ширины. Бывает, что некоторые браузеры по умолчанию присваивают им какое-то небольшое значение, не равное нулю.

APPLET

(HTML 3.2) – Applet

Имеет начальный и конечный теги. Используется для вставки в HTML-страницу Java-апплетов – программ на языке Java, исполняемых браузером на вашем компьютере. Java-апплет исполняется в специально отведенном для него месте, отображаясь в документе наподобие картинки. Поэтому многие атрибуты элемента **APPLET** сходны с атрибутами элемента **IMG**.

Если ваш браузер не имеет встроенной виртуальной Java-машины (и соответственно, не поддерживает элемента **APPLET**), то на месте окошка Java-апплета вы увидите текст, заключенный между начальным и конечным тегами.

Атрибуты:

CODE – обязательный атрибут. Определяет имя файла исполняемого Java-апплета.

CODEBASE – указывает базовый адрес (URL), по которому находится файл с кодом исполняемого Java-апплета. Если парамер CODEBASE опущен, используется URL текущего документа.

ALIGN – обязательный атрибут. Указывает способ выравнивания Java-апплета. Может принимать те же значения, что и аналогичный атрибут элемента IMG.

HEIGHT и WIDTH – обязательные атрибуты. Определяют ширину и высоту (в пикселах) окошка вывода программы.

HSPACE и VSPACE – определяют отступ (в пикселах) по горизонтали и вертикали от других объектов документа.

NAME – указывает имя Java-апплета, уникальное для данного документа. Позволяет Java-апплетам на данной странице находить себе подобных и взаимодействовать друг с другом.

ALT – определяет текст, отображаемый на месте апплета браузером, если браузер понимает элемент APPLET, но не может выполнять Java-апплеты.

SRC – указывает адрес (URL), ассоциированный с апплетом. Например, адрес сайта разработчика апплета.

Пример 1:

```
<APPLET CODE="JumpingGirl.class" WIDTH="30" HEIGHT="40"
ALIGN="left" ALT="Прыгающая девочка">
    Если вы видите этот текст, значит ваш браузер не поддерживает
Java.
    </APPLET>
```

В результате слева отображается окошко (размером 30x40 точек) с прыгающей девочкой. Само собой, у вас должен быть файл *JumpingGirl.class*, который должен лежать в той же директории, что и текущий документ.

Пример 2:

```
<APPLET CODEBASE="http://www.igf.ru/javagames"
CODE="CrazyTetris.class" WIDTH="300" HEIGHT="500" ALIGN="right"
SRC="http://www.igf.ru" ALT="Игра Тетрис">
</APPLET>
```

В данном примере справа отображается окошко (размером 300x500 точек) с игрой Тетрис. Браузер будет пытаться загрузить игру используя URL *"http://www.igf.ru/javagames/CrazyTetris.class"*.

Примечание: Для передачи Java-программе каких-либо параметров используется элемент **PARAM**.

PARAM (HTML 3.2) – Parameter

Располагается в начале элемента **APPLET**. Используется для передачи Java-программе каких-либо параметров. Элемент задает пару "имя – значение" переменной, которая будет передана Java-программе.

Пример:

```
<APPLET CODEBASE="http://www.igf.ru/applets"
CODE="JavaTetris.class" WIDTH="440" HEIGHT="475" ALIGN="center">
<PARAM NAME="width" VALUE="10">
<PARAM NAME="height" VALUE="20">
<PARAM NAME="name" VALUE="Cool Tetris">
   ...
</APPLET>
```

13. ТАБЛИЦЫ

Таблицы в HTML формируются нетрадиционным способом – построчно. Сначала с помощью элемента **TR** необходимо создать ряд таблицы, в который затем элементом **TD** помещаются ячейки.

Важно:

В HTML таблицы **используются не только для отображения таблиц как таковых**, но и **для дизайна**. С помощью таблиц можно создать невидимый "каркас" страницы, помогающий расположить текст и изображения так, как вам нравится.

Элементы для создания таблиц:

TABLE	Создает таблицу
CAPTION	Задает заголовок таблицы
TR	Создает новый ряд (строку) ячеек таблицы
TD и TH	Создает ячейку с данными в текущей строке

Пример:

```
<TABLE BORDER>
   <TR>
      <TD>A1</TD> <TD>B1</TD> <TD>C1</TD>
   </TR>
   <TR>
      <TD>A2</TD> <TD>B2</TD> <TD>C2</TD>
   </TR>
</TABLE>
```

Результат:

1	1	1
2	2	2

TABLE

(HTML 3.2) – Table

Элемент для создания таблицы. Обязательно должен иметь начальный и конечный теги. По умолчанию таблица печатается без рамки, а разметка осуществляется автоматически в зависимости от объема содержащейся в ней информации. Ячейки внутри таблицы создаются с помощью элементов **TR**, **TD**, **TH** и **CAPTION**.

Атрибуты:

ALIGN – определяет способ горизонтального выравнивания таблицы. Возможные значения: left, center, right. Значение по умолчанию – left.

VALIGN – должен определять способ вертикального выравнивания таблицы. Возможные значения: top, bottom, middle.

BORDER – определяет ширину внешней рамки таблицы (в пикселах). При BORDER="0" или при отсутствии этого атрибута рамка отображаться не будет.

CELLPADDING – определяет расстояние (в пикселах) между рамкой каждой ячейки таблицы и содержащимся в ней материалом.

CELLSPACING – определяет расстояние (в пикселах) между границами соседних ячеек.

WIDTH – определяет ширину таблицы. Ширина задается либо в пикселах, либо в процентном отношении к ширине окна браузера. По умолчанию этот атрибут определяется автоматически в зависимости от объема содержащегося в таблице материала.

HEIGHT – определяет высоту таблицы. Высота задается либо в пикселах, либо в процентном отношении к высоте окна браузера. По умолчанию этот атрибут определяется автоматически в зависимости от объема содержащегося в таблице материала.

BGCOLOR – определяет цвет фона ячеек таблицы. Задается либо RGB-значением в шестнадцатиричной системе, либо одним из 16 базовых цветов.

BACKGROUND – позволяет заполнить фон таблицы рисунком. В качестве значения необходимо указать URL рисунка.

Примеры таблиц

Пример 1:

```
<TABLE BORDER>
 <TR>
  <TH ROWSPAN=2>HDD</TH>
  <TD>WD Caviar 3.1Gb</TD><TD ALIGN="right">85$</TD>
 </TR>
 <TR>
  <TD>Quantum FB ST 6.4Gb</TD><TD ALIGN="right">110$</TD>
 </TR>
</TABLE>
```

Результат:

HDD	WD Caviar 3.1Gb	85$
	Quantum FB ST 6.4Gb	110$

Примечания:

Если атрибут HEIGHT указать равным 100%, то старые версии Internet Explorer будут создавать таблицу высотой, чуть превышающей высоту окна

браузера. Чтобы этого избежать, можно пожертвовать рамкой таблицы, указав **BORDER="0"**.

Старые версии Netscape работают некорректно, если ширина и высота таблицы равны 100%. В этом случае вокруг таблицы остается свободное пространство.

CAPTION

(HTML 3.2) – Caption

Задает заголовок таблицы. Содержание заголовка должно состоять только из текста. Использование блочных элементов в этом случае недопустимо.

Атрибуты:

ALIGN – определяет способ вертикального выравнивания заголовка таблицы. Возможные значения:

top – помещает заголовок над таблицей (значение по умолчанию);

bottom – помещает заголовок под таблицей.

Пример:

```
<TABLE BORDER="1">
  <CAPTION ALIGN="bottom">Заголовок таблицы</CAPTION>
  <TR>
    <TD>Ячейка таблицы</TD>
  </TR>
</TABLE>
```

TR

(HTML 3.2) – Table Row

Создает новый ряд (строку) ячеек таблицы. Ячейки в ряду создаются с помощью элементов **TD** и **TH**

Атрибуты:

ALIGN – определяет способ горизонтального выравнивания содержимого всех ячеек данного ряда. Возможные значения: left, center, right.

VALIGN – определяет способ вертикального выравнивания содержимого всех ячеек данного ряда. Возможные значения: top, bottom, middle.

BGCOLOR – определяет цвет фона для всех ячеек данного ряда. Задается либо RGB-значением в шестнадцатиричной системе, либо одним из 16 базовых цветов

TD и TH

(HTML 3.2) – Table Data & Table Head

Элемент **TD** создает ячейку с данными в текущей строке. Элемент **TH** также создает ячейку, но определяет ее как ячейку-заголовок.

Такое разграничение позволяет браузерам оформлять содержимое ячейки-заголовка и ячеек с данными разными шрифтами. Кроме того, должна улучшиться работа браузеров, использующих речевой интерфейс. В качестве содержимого ячейки можно использовать другие таблицы.

Атрибуты:

ALIGN – определяет способ горизонтального выравнивания содержимого ячейки. Возможные значения: left, center, right. По умолчанию способ выравнивания определяется значением атрибута ALIGN элемента TR. Если же и он не задан, то для TD выполняется выравнивание по левому краю, а для TH – центрирование.

VALIGN – определяет способ вертикального выравнивания содержимого ячейки. Возможные значения: top, bottom, middle. По умолчанию происходит выравнивание по центру (VALIGN="middle"), если значение этого атрибута не было задано ранее в элементе TR.

WIDTH – определяет ширину ячейки. Ширина задается в пикселах или в процентном отношении к ширине таблицы.

HEIGHT – определяет высоту ячейки. Высота задается в пикселах или в процентном отношении к высоте таблицы.

COLSPAN – определяет количество столбцов, на которые простирается данная ячейка. По умолчанию имеет значение 1.

ROWSPAN – определяет количество рядов, на которые простирается данная ячейка. По умолчанию имеет значение 1.

NOWRAP – блокирует автоматический перенос слов в пределах текущей ячейки. Обратите внимания на примечание, касающееся использования данного атрибута (далее, внизу страницы).

BGCOLOR – определяет цвет фона ячейки. Задается либо RGB-значением в шестнадцатиричной системе, либо одним из 16 базовых цветов.

BACKGROUND – заполняет ячейку фоновым рисунком. Необходимо указать URL рисунка. Данный атрибут не работает в старых версиях браузера Netscape (до 3.X включительно).

Примечания:

Границы ячейки отображаются только в том случае, когда она имеет некое содержание. Чтобы получить пустую ячейку с границами, достаточно поместить в нее специальный символ ** ,** обозначающий пробел, или маленькую прозрачную gif-картинку.

Если вы используете одновременно атрибуты NOWRAP и WIDTH="x", где *x* – маленькое число, то следует дополнительно вставлять внутрь ячейки <NOBR>. Дело в том, что в браузерах Internet Explorer для полной отмены переноса слов используется именно <NOBR>, а атрибут NOWRAP действует только при отсутствии WIDTH.

Атрибут HEIGHT, указанный в процентном отношении, работает нормально лишь в том случае, когда явно задана высота всей таблицы.

Один из наиболее распространенных приемов web-дизайнера – "таблица в таблице". Для этого достаточно расположить одну таблицу внутри ячейки другой. Так можно создавать даже очень сложные вложенные "каркасы" дизайна страницы.

14. ФОРМЫ

С помощью описанных ниже элементов вы можете создавать заполняемые анкеты, опросники и различные поля для ввода текста пользователем с возможностью последующей отправки заполненной формы на ваш сервер.

Элементы для создания форм:

FORM	Создает заполяемую форму
TEXTAREA	Создает поле для ввода нескольких строк текста
SELECT	Создает меню в заполняемой форме
OPTION	Создает отдельные пункты в меню (см. SELECT)
INPUT	Создает поле в форме

FORM

(HTML 2.0) – Form

Используется для создания заполняемой формы. Необходимо присутствие начального и конечного тегов. Внутри элемента **FORM** разрешается использовать большинство HTML-элементов.

Атрибуты:

NAME – определяет имя формы, уникальное для данного документа. Используется, если в документе присутствует несколько форм.

ACTION – обязательный атрибут. Определяет URL, по которому будет отправлено содержимое формы – путь к скрипту сервера, обслуживающему данную форму.

METHOD – определяет способ отправки содержимого формы. Возможные значения GET (по умолчанию) и POST.

ENCTYPE – определяет способ кодирования содержимого формы при отправке. По умолчанию используется "application/x-www-form-urlencoded".

TARGET – определяет имя окна, в которое возвращается результат обработки отправленной формы. Возможные значения : _self, _parent, _top, _blank или явно указанное имя окна. Подробное описание значений смотрите в атрибуте TARGET элемента A.

Пример:

```
<!-- Создаем форму -->
<FORM ACTION="/cgi-bin/thanks.pl" METHOD=GET
NAME="TestForm">

<!-- Внутри формы создаем поле ввода: -->
Фамилия:
<INPUT TYPE="text" name="lastname" SIZE="20"
VALUE="Пулкин"><br>

<!-- Кнопка "Отправить": -->
<INPUT TYPE="submit" VALUE="Отправить">
</FORM>
<!-- Все, конец форме -->'
```

Примечания:

Во время отладки скрипта, принимающего данные, удобнее всего использовать метод GET.

Метод GET не позволяет передать скрипту большой объём данных. Если предполагается, что пользователь будет заполнять очень большую форму или вводить объёмные текстовые данные, или пересылать файл – используйте METHOD="POST".

TEXTAREA

(HTML 2.0) – Text Area

Создает поле для ввода нескольких строк текста. Обычно содержит текст инициализации, который при загрузке документа изначально будет записываться в данное поле. Элемент **TEXTAREA** должен располагаться внутри элемента FORM.

Атрибуты:

NAME – обязательный атрибут. Определяет название, которое будет использоваться при идентификации заполненого поля сервером.

ROWS – определяет количество строк текста, видимых на экране.

COLS – определяет ширину текстового поля – в печатных символах.

WRAP – определяет способ переноса слов в заполняемой данной заполняемой форме. Возможные значения:

off – перенос слов не происходит (значение по умолчанию)

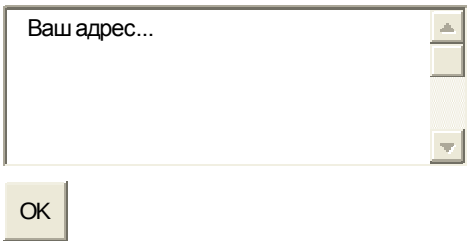
virtual – перенос слов только отображается, на сервер же поступает неделимая строка.

physical – перенос слов будет происходить во всех точках переноса.

Пример:

```
...
<FORM ACTION="receive.html" METHOD=POST>
   <TEXTAREA NAME="address" WRAP="virtual" COLS="40"  ROWS="3">
Ваш адрес...</TEXTAREA> <br>
   <INPUT TYPE="submit" VALUE="OK">
</FORM>
...
```

Результат:



Примечание:

Поле типа **TEXTAREA** позволяет пользователю набрать довольно большой блок текста. Поэтому, если в форме присутствует поле **TEXTAREA**, передавайте скрипту данные методом POST (см. элемент **FORM**), иначе есть вероятность потери данных.

SELECT

(HTML 2.0) – Select

Элемент **SELECT** создает в заполняемой форме меню типа "Выбор одного пункта из многих" или "Выбор нескольких пунктов из многих". Должен располагаться внутри элемента **FORM** и иметь как начальный, так и конечный теги. Содержит несколько элементов **OPTION**, иначе не имеет смысла.

Атрибуты:

MULTIPLE – дает возможность выбора нескольких пунктов меню при удержании клавиши Ctrl. По умолчанию можно выбрать только один пункт меню.

NAME – определяет имя меню, уникальное для данной формы, которое будет использовтся при передаче данных на сервер. Каждый выбранный пункт меню при передаче на сервер будет иметь вид: name/value. Значение (value) формируется элементом OPTION.

SIZE – определяет количество видимых пунктов в меню. Если значение этого атрибута больше единицы, то результатом будет список пунктов.

Пример:

```
...
<FORM ACTION="receive.cgi">
<SELECT NAME="OS" MULTIPLE>
   <OPTION VALUE="DOS">MS-DOS
   <OPTION VALUE="WinXP">MS Windows98
   <OPTION VALUE="Unix" SELECTED>UNIX
   <OPTION VALUE="WinNT">MS Windows NT
</SELECT>
<INPUT TYPE="submit" VALUE="Послать">
</FORM>
...
```

Результат:

OPTION

(HTML 2.0) – Option

Используется только с элементом **SELECT**. Элемент **OPTION** описывает отдельные пункты меню. Не имеет конечного тега.

Атрибуты:

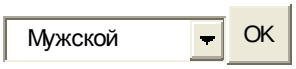
SELECTED – Определяет пункт меню, который будет выбран изначально при загрузке документа. Если меню имеет тип "один из многих", то флагом SELECTED может быть помечен лишь один из пунктов меню.

VALUE – Задает данному пункту значение, которое будет использовано наряду с другими сведениями о содержимом заполненной формы. При предоставлении информации на сервер это значение будет объединено со значением атрибута NAME в элементе SELECT.

Пример:

```
...
<FORM ACTION="script.cgi">
<SELECT NAME="gender">
   <OPTION VALUE="male" SELECTED>Мужской
   <OPTION VALUE="female">Женский
</SELECT>
<INPUT TYPE="submit" VALUE="OK">
</FORM>
...
```

Результат:



INPUT

(HTML 2.0) – Input

Элемент **INPUT** создает поле формы (кнопку, поле ввода, чекбокс и т.п.), содержание которого может быть изменено или активизировано пользователем. Элемент не имеет конечного тега. Элемент **INPUT** должен располагаться внутри элемента **FORM**.

Атрибуты:

NAME – определяет имя, используемое при передаче содержания данной формы на сервер. Этот атрибут необходим для большинства типов (атрибут TYPE – см. ниже) элемента INPUT и обычно используется для идентификации поля или для группы полей, связанных логически.

TYPE – определяет тип поля для ввода данных. По умолчанию – это "text". Возможные значения:

TEXT – создает поле ввода под одну строку текста. Как правило, используется совместно с атрибутами SIZE и MAXLENGTH.

TEXTAREA – создает поле ввода для текста в несколько строк. Но для этих целей лучше использовать элемент TEXTAREA

FILE – дает возможность пользователю приобщить файл к текущей форме. Возможно использование совместно с атрибутом ACCEPT.

PASSWORD – создает поле ввода под одну строку, однако текст, вводимый пользователем, отображается в виде значков "*", скрывая тем самым его содержание от любопытных глаз.

CHECKBOX – создает поле ввода для атрибутов типа Boolean ("да"/"нет") или для атрибутов, которые могут одновременно принимать несколько значений. Эти атрибуты представляют собой несколько полей checkbox, которые могут иметь одинаковые имена. Каждое выбранное поле chackbox создает отдельную пару *name/value* в информации, посылаемой на сервер, даже если результатом будут дублирующиеся имена. Поле этого типа обязательно должно иметь атрибуты NAME и VALUE, а также необязательный атрибут CHECKED, который указывает на то, что поле активизировано.

RADIO – создает поле ввода для атрибутов, которые принимают одно значение из нескольких возможных. Все кнопки (radio buttons) в группе должны иметь одинаковые имена, но только выбранная кнопка в группе создает пару *name/value*, которая будет послана на сервер. Как и для полей checkbox, атрибут CHECKED необязателен; он может быть использован для определения выделенной кнопки в группе кнопок (radio button).

SUBMIT – создает кнопку, при нажатии которой заполненная форма посылается на сервер. Атрибут VALUE в данном случае изменяет надпись на кнопке, содержание которой, заданное по умолчанию, зависит от браузера. Если атрибут NAME указан, то при нажатии данной кнопки к информации, посылаемой на сервер, добавляется пара *name/value*, указанная для для атрибута SUBMIT, в противном случае пара не добавляется.

IMAGE – создает графическую кнопку-картинку, инициализирующую передачу данных на сервер. Местонахождение графического изображения можно задать с помощью атрибута SRC. При передаче данных серверу сообщаются координаты x и y той точки на изображении, где был произведен щелчок клавишей мыши. Координаты измеряются из верхнего левого угла изображения. При этом информация о поле типа image записывается в виде двух пар значений *name/value*. Значение name получается посредством добавления к названию соответствующего поля суффиксов ".x" (абсциссы), и ".y" (ординаты).

RESET – создает кнопку, сбрасывающую значения полей формы к их первоначальным значениям. При нажатии кнопки данные на сервер не

отправляются. Надпись на кнопке может быть изменена с помощью атрибута VALUE. По умолчанию надпись на кнопке зависит от версии браузера.

HIDDEN – поля этого типа не отображаются на экране монитора, что позволяет разместить "секретную" информацию в рамках формы. Содержание этого поля посылается на сервер в виде *name/value* вместе с остальной информацией формы. Этот тип полей удобно использовать для передачи данных от скрипта скрипту незаметно для пользователя.

BUTTON – позволяет создать пользовательскую кнопку в HTML документе, что, при умелом использовании JavaScript, добавляет форме функциональность. Атрибут NAME позволяет задать имя данной кнопке, которое может быть использовано для какой-либо функции в скрипте. Атрибут VALUE позволяет задать текст, который будет отображен на кнопке в документе.

VALUE – задает текстовый заголовок для полей любого типа, в том числе и кнопок. Для таких полей как checkbox или radio, будет возвращено значение, заданное в атрибуте VALUE.

CHECKED – указывает, что поля типов checkbox и/или radio (см. выше атрибут TYPE) активизированы.

SIZE – определяет размер поля в символах. Например, чтобы определить поле с видимой шириной в 24 символа, надо указать SIZE="24".

MAXLENGTH – определяет максимальное количество символов, которые можно ввести в текстовом поле. Оно может быть больше, чем количество символов, указанных в атрибуте SIZE. По умолчанию количество символов не ограничено.

SRC – задает URL-адрес картинки, используемой при создании графической кнопки. Используется совместно с атрибутом TYPE="image".

ALIGN – определяет способ вертикального выравнивания для изображений. Используется совместно с атрибутом TYPE="image". Полностью аналогичен атрибуту ALIGN элемента IMG. По умолчанию имеет значение bottom.

ACCEPT – конкретизирует тип файла. Используется только совместно с параметром TYPE="file". Значение задается в виде MIME-типа.

Пример 1:

```
<FORM NAME="Form1" ACTION="http://www.igf.ru/cgi-bin/banya.pl">
   <INPUT TYPE="hidden" NAME="info" VALUE="Запись на
    поездку">
   <INPUT TYPE="radio" NAME="sex" VALUE="Male" CHECKED>
    Муж<BR>
   <INPUT TYPE="radio" NAME="sex" VALUE="Female"> Жен<BR>
   Имя:<BR>
   <INPUT TYPE="text" NAME="textfield" VALUE="Василий Пулкин"
    SIZE="30" MAXLENGTH="60"><BR>
   Пароль:<BR>
   <INPUT TYPE="password" WIDTH="10"
    NAME="passwd"><BR><BR>
   <INPUT TYPE="submit" VALUE="Отправить">
</FORM>
```

Пример 2:

```
Хочу получать следующие издания:<br>
<FORM NAME="Form2" ACTION="http://www.igf.ru/cgi-
bin/magazines.pl">
   <INPUT TYPE="checkbox" NAME="m1">Страшная газета<br>
   <INPUT TYPE="checkbox" NAME="m2">6 соток<BR>
   <INPUT TYPE="checkbox" NAME="m3" CHECKED>Экспресс<BR>
   <INPUT type="image" src="/img/button.gif" WIDTH="60"
HEIGHT="30">
</FORM>
```

Таблица базовых цветов

Это основные цвета, используемые в HTML. Именно эти цвета используются дизайнерами для создания страниц поисковых систем, а также в дизайне сайтов с огромной аудиторией, то есть используются там, где нужна максимальная совместимость. Все они прекрасно отображаются даже в 256-цветном режиме работы видеокарты. Несомненно, для оригинального цветового дизайна обычных сайтов их маловато.

Для получения любого другого оттенка применяются комбинации RGB-значений цвета, записанные в шестнадцатиричном формате. Приведенные ниже 216 цветов также считаются "безопасными" (browser-safe).

Для удобства "безопасные" цвета поделены на три таблицы по 216 цветов, отсортированные по разным цветовым компонентам.

Таблицу базовых цветов можно найти по ссылкам

http://depositfiles.od.ua/tags/color.php

http://grizun.org.ua/documents/offline/html.manual.ru/book/

Отображение специальных символов

Иногда у вас могут возникнуть проблемы при попытке отобразить некоторые символы на веб-странице. Такие символы как &, < >, ®, ©, ™, используются при написании некоторых тегов. Другие символы не содержатся в обычном алфавите. Кроме того, существуют символы, используемые в иностранных языках, но не содержащиеся в английском -например, чтобы правильно отобразить на веб-странице слово français вам понадобится набрать **fran** (буквы до седиля), набрать **ç**, и **ais** (буквы после седиля).

Пример

français

Вот что мы получим в результате:

fran**ç**ais

Таким же образом можно вставлять любые символы. Ниже приведена только часть из всех существующих.

Символ	**HTML-тег**	**Символ**	**HTML-тег**
<	<	>	>
&	&	"	"
©	©	®	®
™	™	¢	¢
£	£	¤	¤
¥	¥	§	§
¨	¨	ª	ª

15. ЗАДАНИЯ

Задание 1. Маркированные списки, цвет фона

Создать страничку, код которой показан ниже. Изменить ДИ-О5 на название вашей группы. Изменить фон. Заполнить список фамилиями студентов своей группы.

Html-код:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
<TITLE> Группа ДИ-05 </TITLE>
<BODY BGCOLOR="red" TEXT="black" LINK="#FF0000" VLINK="#505050"
      MARGINHEIGHT="30" TOPMARGIN="30" LEFTMARGIN="40"
      MARGINWIDTH="40">
<!--текст с несколькими уровнями заголовков. Фон красный.  Цвет шрифта черный. Отступы снизу и сверху 30 слева и справа 40-->

<h1> Моя группа<h1>
 <h2> ДИ-05 <h2>
<HR Align="left" NOSHADE WIDTH="50%">
<!-- черта без тени слева-->
  <h3> мы - это: <h3>
```

```
<!-- пример вставки списка-->
  <Ul>
      <li> Смок </li>
      <li> Курочка </li>
          <li> Курочка </li>
      <li> Черная </li>
   <ul>
</HTML>
```

Задание 2. Нумерованные списки

START – определяет первое число, с которого начинается нумерация пунктов. (только целые числа) ,TYPE – определяет стиль нумерации пунктов

Html-код:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
<BODY>
<h>Список 1<h>
```

```
<OL TYPE="I" START="2">
   <LI> Пункт два </LI>
   <LI> Пункт три </LI>
   <LI> Пункт четыре </LI>
</OL>

<!-- А теперь другой список-->

<h1>Список 2</h1>
  <h2>Доклад</h2>
<OL TYPE="a" START="1">
   <LI> Прежде всего... </LI>
   <LI> Надо также отметить... </LI>
   <LI> В заключение следует назвать... </LI>
</OL>
</BODY>
</HTML>
```

Задание 3. Вставка изображений в документ HTML

Создание фона из элементов графики (файла зима.jpg) и вставка рисунка с текстом в web-страницу. Рисунок “Тигренок” предварительно разместить в папке C:\Documents and Settings\BOSS\Рабочий стол\ДИ

```
<BODY background="C:\Documents and Settings\All Users\Документы\Мои рисунки\Образцы рисунков\зима.jpg"
<IMG src="C:\Documents and Settings\ Рабочий стол\ДИ\тигренок.jpg" WIDTH="45" HEIGHT="53" ALT="Мой тигренок" HSPACE="10" ALIGN="left"> Этот текст обтекает картинку справа и находится от нее на расстоянии 10 пикселов.
</BODY
</HTML>
```

Задание 4. Создание гиперссылок в html

1. Следующий отрывок HTML содержит две ссылки, якорь одной из которых указывает на документ HTML с названием "chapter2.html", а якорь другой - на GIF-изображение в файле "forest.gif". Активировав эту ссылку, пользователь может посетить этот ресурс. Заметьте, что атрибут href в каждом якоре специфицирует адрес якоря назначения с применением URL.

```
<BODY>
...текст...
<P>Вы найдёте многое на  <A href="chapter2.html">chapter two</A>.
См. также здесь <A href="../images/forest.gif">карту леса.</A>
</BODY>
```

2. Создание оглавления, вхождения которого связаны с элементами H2, H3 и т.д. в этом же документе. Используется элемент A для создания якоря

```
<H1>Оглавление</H1>
<P><A href="#section1">Введение</A><BR>
<A href="#section2">Фон</A><BR>
<A href="#section2.1"> Несколько заметок </A><BR>
...остальная часть оглавления...
...тело документа...
<H2><A name="section1">Введение</A></H2>
...раздел 1...
<H2><A name="section2">Фон</A></H2>
...раздел 2...
<H3><A name="section2.1">Несколько заметок</A></H3>
...раздел 2.1...
```

3. Преобрзование элеменов-заголовков в якоря:

```
<H1>Оглавление</H1>
<P><A href="#section1">Введение</A><BR>
<A href="#section2">Фон</A><BR>
<A href="#section2.1">Несколько заметок</A><BR>
```

...остальная часть оглавления...

...тело документа...

<H2 id="section1">Introduction</H2>

...раздел 1...

<H2 id="section2">Фон</H2>

...раздел 2...

<H3 id="section2.1">Несколько заметок</H3>

...раздел 2.1...

4. Ссылка с картинки (изображения) в html теге A

В качестве якоря для html тега ссылки, вместо текста, может использоваться изображение. В этом случае тег Img заключается в открывающий и закрывающий теги ссылки A. Изображение-гиперссылка обычно обводится браузером рамкой такого же цвета, что и цвет текста обычной гиперссылки в соответствующем состоянии (посещенная, непосещенная или активная).

<A href = «info.html»><IMG SCR = «info.gif»></A>

Задание 5 . Таблицы в HTML. Создание расписания группы

```
<head>
<title> Лабораторная работа, создание таблиц в HTML-документе. </title>
</head>
<body BACKGROUND="back.jpg"> <font face ="Arial">
<p align=center><FONT COLOR="red" SIZE="5"><b>V курс
гр.8501</b></font></p>
<p><FONT  SIZE="5"><b>Четная неделя</b></p>
<p><FONT  SIZE="4"><b>Понедельник </b></font></p>
<table border=1 cellpadding=0 width="100%" align = center>
 <tr bgcolor=#b2b2b2>
        <td width="6%" align = center> <i>Пара</i>   </td>
```

```
        <td width="20%" align = center> <i>Время</i> </td>
        <td width="50%" align = center> <i>Название предмета</i> </td>
        <td align=center> <i>Аудитория</i> </td>  </tr>
 <tr bgcolor=#ccffcc align = center>
        <td><p><FONT COLOR="blue">1</p> </td>
        <td> <p><FONT COLOR="blue">8:30 - 10:05</p>  </td>
        <td rowspan=4> <p><FONT COLOR="blue">-</p>  </td>
        <td rowspan=4> <p><FONT COLOR="blue">-</p>  </td> </tr>
 <tr bgcolor=#ccffcc align = center>
        <td> <p><FONT COLOR="blue">2</p> </td>
        <td> <p><FONT COLOR="blue">10:25 - 12:00</p> </td>  </tr>
 <tr bgcolor=#ccffcc align = center>
        <td> <p><FONT COLOR="blue">3</p></td>
        <td><p><FONT COLOR="blue">12:20 - 13:55</p>  </td> </tr>
 <tr bgcolor=#ccffcc align = center>
        <td><p><FONT COLOR="blue">4</p> </td>
        <td> <p><FONT COLOR="blue">14:15 - 15:50</p> </td> </tr>
 <tr bgcolor=#ccffcc align = center>
        <td><p><FONT COLOR="blue">5</p> </td>
        <td><p><FONT COLOR="blue">16:10 - 17:45</p>  </td>
        <td> <p><FONT COLOR="blue">Проектирование Интернет –
приложений (ЛК)</p>  </td>
        <td><p><FONT COLOR="blue">19/140</p></td> </tr>
 <tr bgcolor=#ccffcc align = center>
        <td><p><FONT COLOR="blue">6</p> </td>
        <td> <p><FONT COLOR="blue">18:05 - 19:40</p> </td>
        <td> <p><FONT COLOR="blue">
              Проектирование Интернет – приложений (ЛБ)</p> </td>
        <td> <p><FONT COLOR="blue">КЦ/417</p></td> </tr>
</table>

<p><FONT  SIZE="4"><b>Вторник</b></font></p>
<table border=1 cellpadding=0 width="100%" align = center>
 <tr bgcolor=#b2b2b2>
        <td width="6%" align = center><i>Пара</i>    </td>
```

```
        <td width="20%" align = center><i>Время</i></td>
        <td width="50%" align = center> <i>Название предмета</i></td>
        <td align=center> <i>Аудитория</i></td> </tr>
<tr bgcolor=#ccffcc align = center>
        <td> <p><FONT COLOR="blue">1</p></td>
        <td> <p><FONT COLOR="blue">8:30 - 10:05</p>  </td>
        <td><p><FONT COLOR="blue">Компьютерная графика (ЛК)</p></td>
        <td> <p><FONT COLOR="blue">КЦ/214</p> </td> </tr>
<tr bgcolor=#ccffcc align = center>
        <td><p><FONT COLOR="blue">2</p> </td>
        <td> <p><FONT COLOR="blue">10:25 - 12:00</p> </td>
        <td> <p><FONT COLOR="blue">-</p> </td>
        <td> <p><FONT COLOR="blue">-</p> </td> </tr>
<tr bgcolor=#ccffcc align = center>
        <td> <p><FONT COLOR="blue">2</p> </td>
        <td><p><FONT COLOR="blue">10:25 - 12:00</p>  </td>
        <td><p><FONT COLOR="blue">Реинжиниринг БП (ЛК)</p> </td>
        <td> <p><FONT COLOR="blue">КЦ/214</p> </td>  </tr>
<tr bgcolor=#ccffcc align = center>
        <td> <p><FONT COLOR="blue">2</p> </td>
        <td> <p><FONT COLOR="blue">10:25 - 12:00</p> </td>
        <td> <p><FONT COLOR="blue">Реинжиниринг БП (ЛБ)</p>  </td>
        <td> <p><FONT COLOR="blue">ВЦ/421</p> </td> </tr>
<tr bgcolor=#ccffcc align = center>
        <td> <p><FONT COLOR="blue">5</p></td>
        <td><p><FONT COLOR="blue">16:10 - 17:45</p>  </td>
        <td rowspan=2><p><FONT COLOR="blue">-</p> </td>
        <td rowspan=2> <p><FONT COLOR="blue">-</p>  </td> </tr>
<tr bgcolor=#ccffcc align = center>
        <td> <p><FONT COLOR="blue">6</p> </td>
        <td> <p><FONT COLOR="blue">18:05 - 19:40</p> </td>  </tr>
</table>
</body>
</html>
```

Результат

Четная неделя

Понедельник

Пара	*Время*	*Название предмета*	*Аудитория*
1	8:30 - 10:05	-	-
2	10:25 - 12:00		
3	12:20 - 13:55		
4	14:15 - 15:50		
5	16:10 - 17:45	Проектирование Интернет – приложений (ЛК)	19/140
6	18:05 - 19:40	Проектирование Интернет – приложений (ЛБ)	КЦ/417

Вторник

Пара	*Время*	*Название предмета*	*Аудитория*
1	8:30 - 10:05	Компьютерная графика (ЛК)	КЦ/214
2	10:25 - 12:00	-	-
2	10:25 - 12:00	Реинжиниринг БП (ЛК)	КЦ/214
2	10:25 - 12:00	Реинжиниринг БП (ЛБ)	КЦ/421
5	16:10 - 17:45	-	-
6	18:05 - 19:40		

Задание 6. Параллельные тексты

Если у Вас есть логически параллельный текст, например, документ на нескольких языках, или несколько вариантов одного текста, элемент TABLE - наилучший способ его представления.

Вы должны разделить параллельный текст на логические части (абзацы), и сделать каждую часть ячейкой таблицы. Вы должны уделить внимание

правильной последовательности работы с текстом: после первой части первого текста Вы должны работать с первой частью второго текста и т.д.

Нижеследующий пример представляет фрагмент текста из Библии в трех вариантах и переводах. Заметим, что атрибуты ALIGN и VALIGN могут существенно влиять на качество отображения. Браузеры не могут определять тип таблиц по их содержанию, поэтому автор документа может подправить формат документа с помощью средств выравнивания.

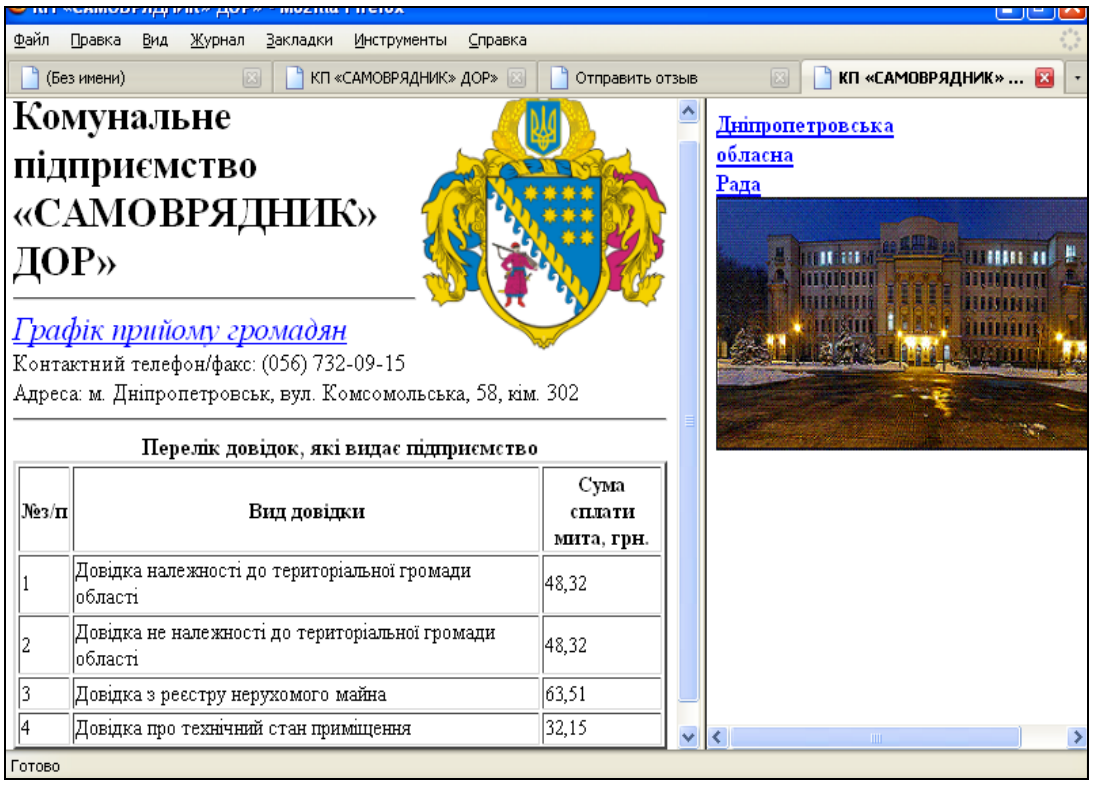
Здесь приведены три текста, написанных на английском, латыни и финском языках.

```
<TABLE>
<CAPTION><STRONG>The beginning of Genesis
        in three languages</STRONG></CAPTION>
<TR ALIGN=LEFT VALIGN=TOP>
        <TH><TH>Latin (Vulgate) </TH>
        <TH>English (King James version)</TH>
        <TH>Finnish (1992 version)</TH> </TR>
<TR ALIGN=LEFT VALIGN=TOP>
        <TH>1</TH> <TD>In principio creavit Deus caelum et terram.</TD>
        <TD>In the beginning God created the heaven and the earth.</TD>
        <TD>Alussa Jumala loi taivaan ja maan.</TD> </TR>
<TR ALIGN=LEFT VALIGN=TOP>
        <TH>2</TH>
        <TD>Terra autem erat inanis et vacua et tenebrae super faciem
        abyssi et spiritus Dei ferebatur super aquas.</TD>
        <TD>And the earth was without form, and void;
        and darkness was upon the face of the deep.
        And the Spirit of God moved upon the face of the waters.</TD>
        <TD>Maa oli autio ja tyhjд, pimeys peitti syvyydet,
        ja Jumalan henki liikkui vetten yllд. </TD> </TR>
<TR ALIGN=LEFT VALIGN=TOP>
        <TH>3</TH>
        <TD>Dixitque Deus "Fiat lux" et facta est lux.</TD>
        <TD>And God said, Let there be light: and there was light.</TD>
```

<TD>Jumala sanoi: "Tulkoon valo!" Ja valo tuli.</TD> </TR>
</TABLE>

Задание 7. Создание фреймов

Вставить в web-страницу два вертикальных фрейма, предварительно создав таблицу и загрузив логотип и изображение днепропетровского горсовета.



Перелік довідок, які видає підприємство

№з/п	Вид довідки	Сума сплати мита, грн.
1	Довідка належності до територіальної громади області	48,32
2	Довідка не належності до територіальної громади області	48,32
3	Довідка з реєстру нерухомого майна	63,51
4	Довідка про технічний стан приміщення	32,15

Html-код:

Page index

```
<HTML>
<!-вставка FRAMEs-->
<HEAD>
<TITLE> КП «САМОВРЯДНИК» ДОР» </TITLE>
</HEAD>
<FRAMESET COLS="225,75">
<FRAME SRC="first.htm">
```

<FRAME SRC="link.htm">
</FRAMESET>
</HTML>

Page first

<HTML>
<HEAD>
<TITLE> КП «САМОВРЯДНИК» ДОР» </TITLE>
</HEAD>
<BODY> <IMG SRC="pic2.gif"ALIGN="right">
<ALIGN=CENTER><FONT SIZE=6> <STRONG> Комунальне підприємство
</STRONG>

 <ALIGN=CENTER><FONT SIZE=6> <STRONG> «САМОВРЯДНИК»
ДОР»</STRONG>
<HR> <ALIGN=LEFT> <FONT SIZE=5> <A HREF="list.htm">
<I>Графік прийому громадян </I> </A>
<FONT SIZE=4> Контактний телефон/факс: (056) 732-09-15
Адреса: м. Дніпропетровськ, вул. Комсомольська, 58, кім. 302 <HR>
<ALIGN=CENTER> <TABLE BORDER=3 WIDHT="100%"> <CAPTION
ALIGN=TOP> <FONT SIZE=4> <B> Перелік
довідок, які видає підприємство <CAPTION>
<TR ALIGN="CENTER">
<TH ROWSPAN=1>№з/п
<TH ROWSPAN=1> Вид довідки
<TH ROWSPAN=1> Сума сплати
 мита, грн. </B>
<TR><TD>1<TD> Довідка належності до територіальної громади області <TD>
48,32
<TR><TD>2<TD> Довідка не належності до територіальної громади області
<TD> 48,32
<TR><TD>3<TD> Довідка з реєстру нерухомого майна <TD> 63,51
<TR><TD>4<TD> Довідка про технічний стан приміщення <TD> 32,15
</TABLE>
<HR>
</BODY>
</HTML>

Page first1

```
<HTML>
<HEAD>
<TITLE> Дніпропетровська облрада </TITLE>
</HEAD>
<BODY> <FONT SIZE=8> <I> <STRONG> Дніпропетровська обласна Рада
</STRONG> <BR>
<IMG SRC="pic.gif" ALIGN="CENTER"> </BODY>
</HTML>
```

Page list

```
<HTML>
<HEAD>
<TITLE> КП «САМОВРЯДНИК» ДОР» </TITLE>
</HEAD>
<BODY>
<H1> ГРАФІК </H1>
<P> <H1> прийому громадян </H1>
<DL> <DD> Пн. 12.00 – 13.30
<DD> Ср. 15.00 – 17.00
<DD> Птн. 14.30 – 16.30
</BODY>
</HTML>
```

Page link

```
<HTML>
<HEAD>
<TITLE> КП «САМОВРЯДНИК» ДОР» </TITLE>
</HEAD>
<BODY> <FONT SIZE=4> <STRONG> <A HREF=“first1.htm”>
Дніпропетровська <BR>обласна <BR>Рада
</STRONG>  </A> <BR> <IMG SRC="pic1.gif"ALIGN="CENTER">
</BODY>
</HTML>
```

Задание 8. Создание элементов формы с элементом SELECT, создающем в заполняемой форме меню типа «Выбор одного пункта из многих» или «Выбор нескольких пунктов из многих».

```
<FORM ACTION=»receive.cgi»>
<SELECT NAME=»OS» MULTIPLE>
  <OPTION VALUE=»DOS»>MS-DOS
   <OPTION VALUE=»WinXP»>MS Windows98
   <OPTION VALUE=»Unix» SELECTED>UNIX
   <OPTION VALUE=»WinNT»>MS Windows NT
</SELECT>
<INPUT TYPE=»submit» VALUE=»Послать»>
</FORM>
```

Задание 9. Создание элементов формы с атрибутом INPUT , создающего поле формы (кнопку, поле ввода, чекбокс и т.п.), содержание которого может быть изменено или активизировано пользователем.

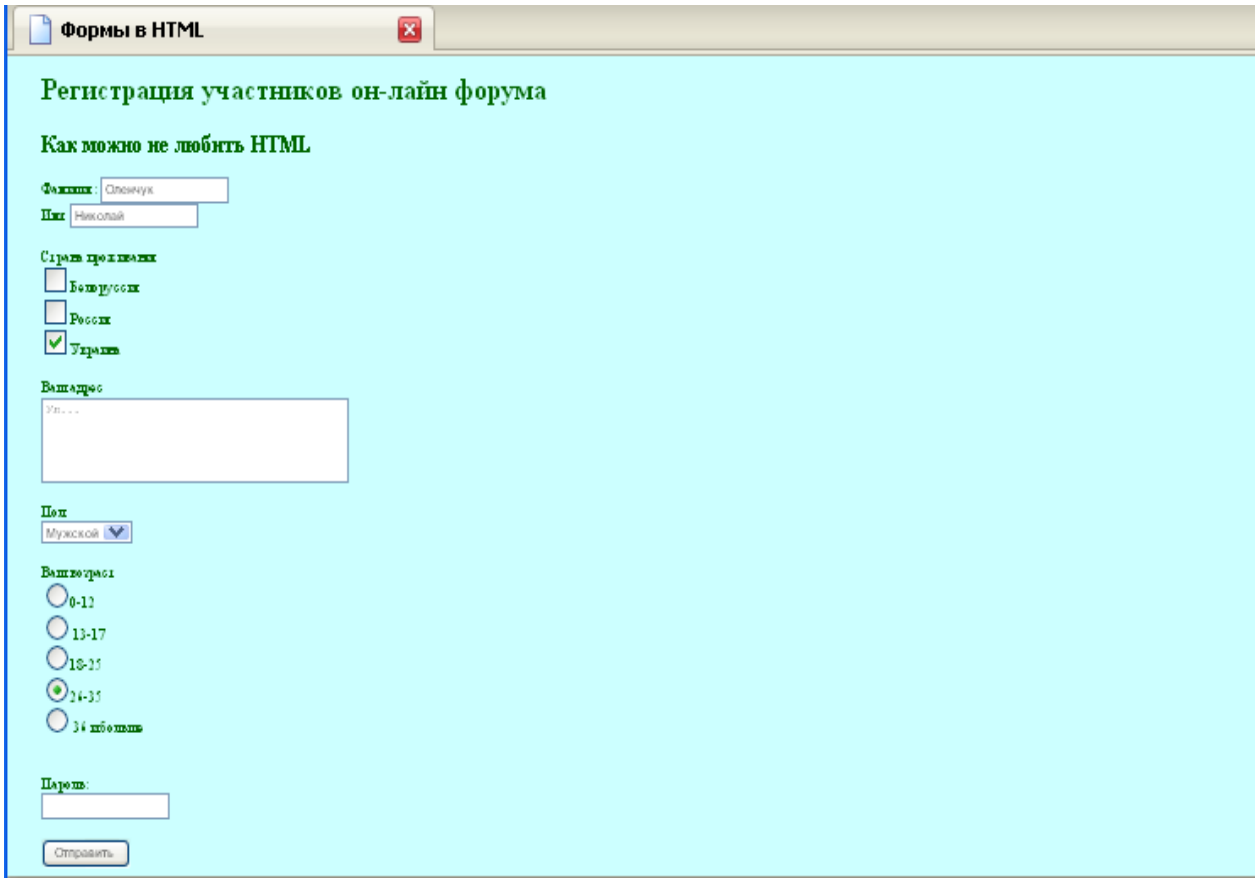
```
…
Хочу получать следующие издания:<br>
<FORM NAME="Form1" ACTION="http://www.igf.ru/cgi-bin/magazines.pl">
   <INPUT TYPE="checkbox" NAME="m1">Elle <br>
   <INPUT TYPE="checkbox" NAME="m2">Коммерсант <BR>
   <INPUT TYPE="checkbox" NAME="m3" CHECKED>Женский Журнал <BR>
   <INPUT TYPE="checkbox" NAME="m4" >Спорт <BR>
   <INPUT type="image" src="/img/button.gif" WIDTH="60" HEIGHT="30">
</FORM>
…
```

Задание 10. Создание больших форм для регистрации

```
<HTML>
<TITLE> Формы в HTML</TITLE>

<BODY BGCOLOR="#ccffff" TEXT="#006600" VLINK="#505050"
MARGINHEIGHT="15" TOPMARGIN="10" LEFTMARGIN="20"
MARGINWIDTH="40">

<h1>Регистрация участников он-лайн форума</h1>
 <h2> Как можно не любить HTML</h2>
 <h3> </h3>

<!-- Создаем форму -->
<FORM  METHOD=GET  NAME="TestForm">

<!-- Внутри формы создаем поля ввода: -->
   <!-- для одной строки-->
Фамилия:
    <INPUT TYPE="text" name="lastname" SIZE="20" VALUE="Пупкин"><br>
Имя
<INPUT TYPE="text" name="name" SIZE="20" VALUE="Коля"><br><br>

 <!-- Для флага (отметить можно и несколько пунктов) -->
Страна проживания<br>
<INPUT TYPE="checkbox" NAME="m1">Белоруссия<br>
    <INPUT TYPE="checkbox" NAME="m2">Россия<BR>
    <INPUT TYPE="checkbox" NAME="m3" CHECKED>Украина<BR><br>

<!-- Для большого блока информации, например, для адреса-->
Ваш адрес<br>
<TEXTAREA NAME="address" WRAP="virtual" COLS="40"
ROWS="3">Ул...</TEXTAREA><br><br>

 Пол <br>

<!--Select определяет количество видимых пунктов-->

<SELECT NAME="gender">
```

```
   <OPTION VALUE="male" SELECTED>Мужской
   <OPTION VALUE="female">Женский
</SELECT><br><br>

<!--Radio Используется для атрибута, который может принимать единственное значение из множества. Радиокнопки требуют явного атрибута VALUE. Единственная нажатая радиокнопка в группе генерирует пару имя/значение в формируемых данных. Одна радиокнопка в группе атрибутом CHECKED должна быть предварительно установлена по умолчанию-->

Ваш возраст<br>
<INPUT TYPE=RADIO NAME=age VALUE=" 0-12">0-12 <br>
<INPUT TYPE=RADIO NAME=age VALUE="13-17"> 13-17<br>
<INPUT TYPE=RADIO NAME=age VALUE="18-25">18-25 <br>
<INPUT TYPE=RADIO NAME=age VALUE=" 26-35" CHECKED>26-35<br>
<INPUT TYPE=RADIO NAME=age VALUE="36 -"> 36  и больше<br><br><br>


<!--Введите пароль (до 10 символов)-->
Пароль:<BR>
   <INPUT TYPE="password" WIDTH="10" NAME="passwd"><BR><BR>

<!--Определяет кнопку, которую пользователь может нажать, чтобы передать содержимое формы серверу-->
   <INPUT TYPE="submit" VALUE="Отправить "><br><br>
</Form>
<!-- А что это- определите самостоятельно-->
<marquee height="30" width="700" bgcolor="#ff66cc"  align="center" loop="25">
 Ну, разобрались с формами? Поздравления!!!
</marquee>
```

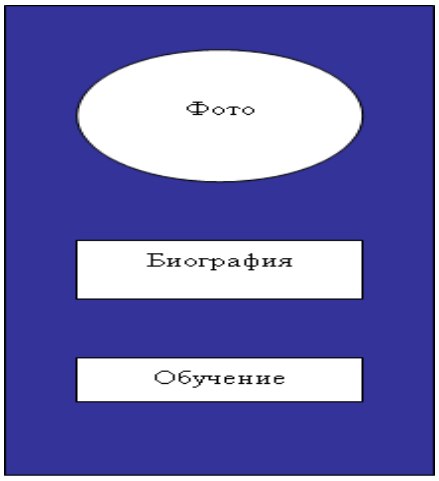
Индивидуальное задание: изменить поля формы так, чтобы имя и фамилия находились на равном отступе слева, и между указанными строчками должна быть как минимум одна пустая строка.



Задание 11. Навигационная карта

Создать навигационную карту, приведенную на рисунке, с активными зонами в виде окружности и прямоугольников.

Индивидуальное задание: создать файлы фото.bmp, family.bmp, nmetau.bmp, file1.htm и указать в гиперссылках их расположение



```
<HTML>
<HEAD>
  карта
<TITLE> карта</TITLE>
</HEAD>
<BODY bgcolor="yellow">
<p>
<img border=5 src="d:\HTML\карта.bmp"  usemap="#map1"
                    title="Навигационная карта"
                     width=167  height=400>
<map name="map1">
    <area shape="circle" coords="80,110,50"  href="фото.bmp">
    <area shape="rect" coords=
                          "20,175,160,230"  href="family.bmp">
 <area shape="rect" coords=
                          "20,275,160,330"  href="nmetau.bmp">
     <area shape default href="file1.htm">
</map>
</BODY>
</HTML>
```

СПИСОК ЛИТЕРАТУРЫ

1. Чак Муссиано и Билл Кеннеди. HTML и XHTML. Подробное руководство. 6-е издание. – М.: Издательство «Символ-Плюс», 2008.
2. Эрик А. Мейер. CSS. Каскадные таблицы стилей. Подробное руководство. 3-е издание. – М.: Издательство «Символ-Плюс», 2008.
3. О. Н. Рева. Использование HTML, JavaScript и CSS. Руководство Web-дизайнера. – М.: Издательство «Эксмо», 2008.
4. www.echoecho.com .
5. www.artlebedev.ru/kovodstvo.
6. www.scriptsearch.com.

Навчальне видання

Нечухаєва Наталія Вікторівна
Расчубкін Віталій Геннадійович
Павленко Ганна Анатоліївна

ВИВЧЕННЯ МОВИ HTML З ВИКОРИСТАННЯМ КАСКАДУ СТИЛІВ CSS ТА ЕЛЕМЕНТІВ МОВИ JAVASCRIPT

Частина 1

Навчальний посібник

(російською мовою)

Тем. план 2012, поз.245

Підписано до друку 26.09.2012. Формат 60х84 1/16. Папір друк. Друк плоский. Облік.-вид. арк. 4,24. Умов. друк. арк. 4,18. Тираж 100 пр. Замовлення №

Національна металургійна академія України
49600, м. Дніпропетровськ-5, пр. Гагаріна,4

Редакційно-видавничий відділ НМетАУ