ГЛАВА

3

Начальный этап создания Web-страницы

В этой главе...

Имея общее представление о HTML, а также о типах элементов, составляющих HTML-документ, приступим к подробному изучению этих элементов. В этой главе рассматриваются главные характеристики основных элементов, а также демонстрируется, насколько просто использовать HTML при создании документов.

Основные правила HTML-кодирования

В создании HTML-документов действительно нет ничего сложного, поскольку они представляют собой простые текстовые файлы, включающие элементы описания разметки. Эти документы создаются с помощью любого редактора, способного экспортировать исходный текст. Кроме того, HTML-браузеры весьма "снисходительны" к пробелам, поэтому с дополнительными символами табуляции, перевода строк или пробелами проблем не возникает.

При создании простейших HTML-файлов очень важно приобрести навыки кодирования, которые впоследствии могут пригодиться при создании более сложных страниц. Рассмотрим примеры, представленные в следующих разделах.

Используйте пробелы

Используйте символы разрыва строк, чтобы отделить разделы кода, а с помощью пробелов сделайте отступы между последующими элементами. Это облегчает чтение и понимание кода. Обратите внимание на следующие два примера:

```
<!D0CTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Декларация Независимости</title>
<meta name="description" content="Our Nation's
Declaration of Independence"><meta name="keywords"
content="declaration, independence, revolutionary,
war, July, 4, 1776".></head><body><hl>
The Declaration of Independence</hl><p>IN CONGRESS, July 4,
1776. </p><p>The unanimous Declaration of the thirteen united
States of America, </p><p>When in the Course of human events,
it becomes necessary for one people to dissolve the political
bands which have connected them with another, and to
assume among the powers of the earth, the separate and equal
station to which the Laws of Nature and of Nature's God
entitle them, a decent respect to the opinions of mankind
requires that they should declare the causes which impel
them to the separation.</p> <p>We hold these truths to be
self-evident, that all men are created equal, that they
are endowed by their Creator with certain unalienable
Rights, that among these are Life, Liberty and the pursuit of Happiness...
```

и

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML  4.O1//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<html><head><title>Декларация Независимости</title>
<meta name="description"  content="Our Nation's Declaration
 of Independence">
<meta name="keywords" content="declaration, independence,
revolutionary, war, July,4, 1776">
</head><body>
<hl>The Declaration  of Independence</hl><p>IN CONGRESS, July
4, 1776.</p>
<p>The unanimous Declaration of the thirteen united States of
America ,</p><p>When in the Course of human events, it becomes
necessary for one people to dissolve the political bands which
have connected them with another, and to assume among
the powers of the earth, the separate and equal station to
which the Laws of Nature and of Nature's God entitle them,
a decent respect to the opinions of mankind requires that they
 should declare the causes which impel them to the
separation. </p><p>We hold these truths to be self-evident,
that all men are created equal, that they are endowed by
their Creator with certain unalienable Rights, that among
these are Life, Liberty and the pursuit of Happiness...
```

Как видно, второй пример читается намного проще, и, следовательно, проще обрабатывается.

Используйте формально корректный HTML-код

Формально корректный HTML-код подразумевает, что документы должны иметь следующие характеристики.

- Они должны включать дескриптор `<!DOCTYPE>`.
- Элементы должны внедряться, а не накладываться друг на друга. Это означает, что документы должны закрываться в порядке, противоположном тому, в котором их открывали. Обратите внимание на следующий неправильный пример:

```
<p>Последнее слово выделено <b>полужирным</p></b>
```

Обратите внимание на наложение дескрипторов полужирного текста и абзаца в конце блока. Во избежание этого следует закрыть дескриптор полужирного текста, как показано в следующем примере:

```
<p>Последнее слово выделено <b>полужирным</b></p>
```

- Наименования элементов и атрибутов следует записывать строчными буквами. Язык XHTML учитывает регистр клавиатуры; дескриптор `<HR>` отличается от дескриптора `<hr>`. Все дескрипторы в определениях типа XHTML-документа (DTD — Document Type Definitions) пишутся строчными буквами, поэтому так же должны записываться и дескрипторы создаваемого документа.
- Все непустые элементы должны быть завершены. Например, не допускается следующее написание:

```
Это один абзац <p>Это другой абзац<p>
```

Каждый открытый дескриптор абзаца должен быть закрыт.

- Все значения атрибутов следует заключать в кавычки. Рассмотрим, например, следующие два дескриптора:

```
<table border=0>
```

и

```
<table border="0">
```

Первый дескриптор записан неправильно, поскольку значение атрибута не заключено в кавычки. Второй дескриптор правильный, поскольку атрибут заключен в кавычки.

- Нельзя использовать минимизированные атрибуты, т.е. атрибуты без значений. Рассмотрим в качестве примера следующие два дескриптора:

```
<input type="checkbox" checked>
```

и

```
<input type="checkbox" checked="checked">
```

В первом дескрипторе используется минимизированный атрибут; у атрибута `checked` есть имя, но нет значения.

- Любой пустой дескриптор должен иметь соответствующий закрывающий дескриптор, или открывающий дескриптор должен завершаться косой чертой (/).

Комментируйте код

Корректно сформированный код должен говорить сам за себя. Тем не менее, существует множество случаев, когда есть все основания для включения в код комментариев. Например, из глав 22 и 23 можно узнать об использовании вложенных таблиц для создания сложных текстовых структур. В результате таких структур часто создаются следующие типы кода:

```
        </table >
    </table >
</table >
```

Бесспорно, вложенные таблицы не так просто отслеживаются. Тем не менее, добавление некоторых комментариев позволяет довольно просто определить назначение вложенных элементов:

```
        </table> <!-- /Верхний заголовок -->;
    </table> <!-- /Основной текст   -->
</table> <!-- /Плавающая страница  -->
```

Базовая структура

Базовая структура всех HTML-документов идентична и включает следующий минимальный набор элементов и дескрипторов:

- <!DOCTYPE> — объявленный тип документа;
- <html> — основной контейнер HTML-страницы;
- <head> — контейнер информации заголовка страницы;
- <title> — название страницы;
- <body> — основное содержимое страницы.

Все эти элементы соотносятся друг с другом, как показано в следующем шаблонном формате:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
   <meta... метадескрипторы...>
<title>Заголовок страницы/документа</title>
<LINK rel="stylesheet" href="имя внешней таблицы стилей"
   type="text/css">
<style>
...стили, специфичные для документа...
</style>
<script>
...клиентские сценарии...
</script>
</head>
<body>
...тело документа, абзацы, модифицированные с помощью
блочных элементов, символы, слова и предложения, модифицированные
с помощью встроенных элементов...
</body>
</html>
```

В следующих разделах эти элементы рассматриваются более подробно.

Объявление типа документа

Дескриптор <!DOCTYPE> определяет формат страницы и используемые ею стандарты. Это выполняется путем указания определений DTD, которых "придерживается" документ. Например, следующий дескриптор <!DOCTYPE> задает строгое определение HTML 4.01 DTD:

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

Формат и параметры дескриптора <!DOCTYPE> более подробно рассматриваются в главе 2. Список определений DTD можно найти на Web-сайте консорциума W3C по адресу:

```
http://www.w3.org/QA/2002/04/valid-dtd-list.html
```

Если не упоминается что-либо иное, в книге рассматривается строгая версия HTML 4.01 DTD.

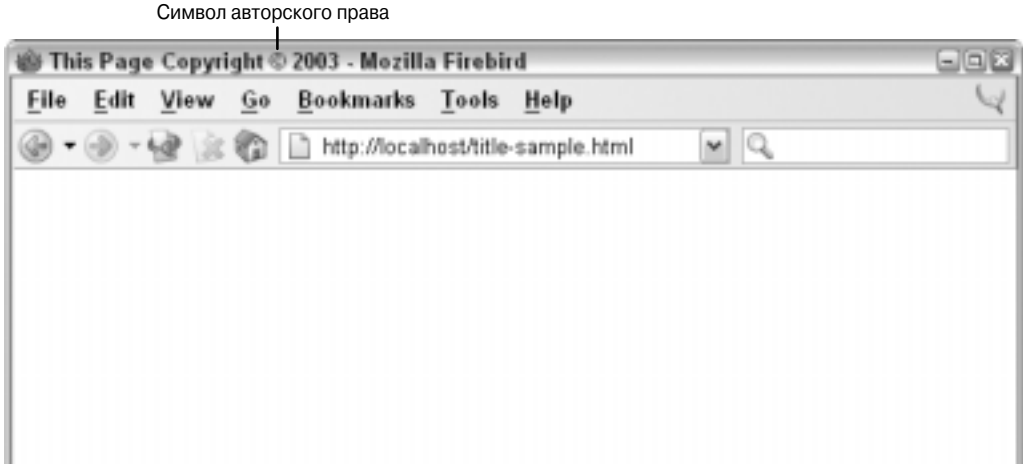


Рис. 3.1. Сущности корректно отображаются в заголовках документов

Указание названия документа

Дескриптор `<head>` определяет ряд других элементов, включая название документа. Название документа заключается между дескрипторами `<title>` и может включать любой символ или объект. Рассмотрим, например, следующий дескриптор `<title>`, который включает символ авторского права:

```
<title>Эта страница защищена авторским правом &copy; 2003</title>
```

Это название отображается в строке заголовка браузера Internet Explorer (рис. 3.1).

Помимо того, что название документа отображается в строке заголовка клиентского браузера, оно также используется и в некоторых других целях: в качестве названия ярлыка/закладки, заданного по умолчанию для браузеров, в качестве ссылки в поисковых машинах и т.д. В связи с этим необходимо всегда включать название документа и по возможности делать его более описательным (и в то же время кратким).

Предоставление информации для поисковых машин

Раздел документа `<head>` может также включать дескрипторы `<meta>`, которые используются для передачи информации и команд клиентскому браузеру.

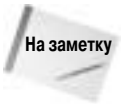
Как видно из названия, дескриптор `<meta>` содержит *метаинформацию* о документе. Под этим подразумевается информация о самом документе, а не о его содержимом. Большая часть метаинформации о документе генерируется Web-сервером, который предоставляет документ в распоряжение пользователей. Тем не менее, с помощью дескрипторов `<meta>` можно предоставить различную дополнительную информацию о документе.

Используя дескрипторы `<meta>`, можно указывать довольно обширный объем информации. Включая параметр `HTTP-EQUIV`, можно отобразить или заменить информацию HTTP-заголовка. Например, следующий дескриптор `<meta>` определяет тип содержимого документа как HTML-код с латинским набором символов (ISO-8859-1):

```
<meta http-equiv="Content-Type" content="text/html;
charset=IS0-8859-l">
```

Кроме того, можно управлять некоторыми аспектами обработки документа клиентским браузером. Например, указывать время кэширования документа (если он кэшируется), обновлять отображение в окне браузера с помощью другой страницы в случае задержки и т.д. Например, следующие два дескриптора `<meta>` дают команду браузеру не кэшировать текущую страницу (`pragma`, `no-cache`) и обновлять окно браузера, в котором отображается другая страница, спустя 3 секунды (`refresh`):

```
<meta http-equiv="pragma"  content="no-cache">
<meta http-equiv="refresh"
content="3;URL=http://www.example.com/newpage.html">
```



Исчерпывающий список заголовков HTTP 1.1 можно найти в определении HTTP 1.1 на Web-сайте W3C по адресу:

`http://www.w3.org/Protocols/rfc2616/rfc2616.html`

Всегда добавляйте в документы хотя бы минимальный объем информации, что позволит поисковым машинам правильно сгруппировать документы. Двумя основными видами мета-информации являются описание документа и ключевые слова, характеризующие его содержимое. Описание и ключевые слова задаются следующими двумя дескрипторами `<meta>`:

```
<meta name="description" content="The latest movie news">
<meta name="keywords" content="movie, movies, production,
genre, sci fl, horror, drama, comedy, anima, manga, news,
chat, bbs, discuss, review, recent">
```

Такие поисковые машины, как Google (`www.google.com`), включают предоставленное описание и ключевые слова в содержимое сайта.

Указание пути, заданного по умолчанию

При определении связей и ссылок в HTML-документе следует по возможности точнее указывать ссылки. Например, при ссылке на изображение с помощью дескриптора `<img>` необходимо выработать привычку включать протокол и полный путь к изображению, как показано в следующей строке кода:

```
<img src="http://www.example.com/images/sailboat.gif">
```

Тем не менее, нецелесообразно указывать полный путь к каждому локальному элементу, на который имеется ссылка в документе. По сути, документ, размещаемый на сервере `example.com`, мог бы ссылаться на то же изображение с помощью следующего кода:

```
<img src="images/sailboat.gif">
```

Что же произойдет при повторном размещении документа? Каталог изображения уже не будет подкаталогом каталога, в котором размещается документ. Изображение обязательно должно находиться на отдельном сервере.

Для решения этих проблем используется дескриптор `<base>`. С его помощью устанавливается база документа, заданная по умолчанию, т.е. место его размещения. В предыдущем примере документ в корневом каталоге сервера `example.com` имеет дескриптор `<base>`, как показано ниже:

```
<base href="http://www.example.com/document.html">
```

Любые абсолютные ссылки в документе (ссылки, включающие полный протокол и путь) будут продолжать указывать на абсолютные цели. Однако любые относительные ссылки (без определения полного протокола и пути) будут указывать на путь, определенный дескриптором `<base>`.

Создание автоматических обновлений и перенаправлений

Метадескрипторы могут также использоваться для обновления содержимого документа или для перенаправления клиентского браузера на другую страницу. Обновление документа целесообразно в том случае, если он содержит такие периодически обновляемые динамические данные, как, например, биржевые курсы. Перенаправление очень удобно при перемещении документа, когда с помощью этой функции можно автоматически перенаправить посетителя к новому документу.

Для обновления или перенаправления документа используется атрибут `http-equiv "refresh"` в дескрипторе `<meta>`. Этот атрибут представлен в следующей форме:

```
<meta http-equi v="refresh" content="время_ожидания_в_секундах;
url-ссылка">
```

Допустим, например, что страница на сайте (`example.com`) была перемещена в другое место. В корневом каталоге сервера данная страница называлась `bio.html`, однако теперь эта страница представлена в каталоге `bio` под именем `index.html` (`/bio/index.html`). Вам, конечно же, хочется, чтобы посетители, которые прежде делали закладку на старой странице, могли зайти и на новую страницу. При размещении следующего документа в корневом каталоге сервера (`bio.html`), посетители спустя 3 секунды автоматически перенаправляются на новую страницу:

```
<!D0CTYPE  HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
   <title>Мою биографию вы можете найти на другой странице!</title>
   <meta http-equiv="pragma" content="no-cache">
   <meta http-equiv="refresh" content="$3$;
      URL= http://www.example.com/bio/index.html">
</head>
<body>
<p>Мою биографию вы можете найти на другой странице. Если не осуще-
ствится автоматический переход через 3 секунды, щелкните на следую-
щей ссылке.</p>
<a href="http://www.example.com/bio/index.html ">Моя новая
биография.</a>
</body>
</html>
```

Для обновления текущей страницы необходимо просто разместить ее абсолютную ссылку, указав ее в качестве значения атрибута `refresh`.

Неплохо использовать метадескриптор `pragma no-cache` вместе с атрибутом `refresh`. Благодаря этому предотвращается кэширование документа браузером и отображение кэшированной копии документа вместо обновленной версии документа. В силу того, что различные браузеры по-разному работают с дескриптором `no cache pragma`, было бы также неплохо добавлять метадескриптор `expires`, как показано ниже:

```
<meta http-equiv="expires" content="0">
```

В этом случае документ сразу же удаляется из кэш-памяти, а следовательно, вообще не кэшируется.

Фоновый цвет страницы и фоновые изображения

Одним из наиболее простых изменений, которым может подвергаться Web-страница, является модификация фонового цвета документа. В большинстве браузеров используется белый цвет, поэтому при настройке другого фонового цвета или изображения ваш документ будет отличаться от всех остальных.

Определение фонового цвета документа

Если HTML-код создается в соответствии с переходным форматом языка HTML, то можно использовать атрибут `bgcolor` в дескрипторе `<body>`. Тем не менее, использование этого атрибута не рекомендуется по следующим причинам:

- атрибут недействителен для строгой версии HTML, а значит, может нарушить процесс проверки подлинности документа;
- если требуется изменить фоновый цвет документов, следует изменить каждый индивидуальный дескриптор `<body>` в каждом документе.

Наиболее оптимальным решением здесь будет использование соответствующих стилей, обычных во внешних таблицах стилей.

Фоновый цвет документа устанавливается с помощью свойства `background-color`. Например, чтобы установить голубой фоновый цвет, следует использовать следующее определение стиля:

```
<style>
   body { background-color: blue; }
</style>
```

Более подробная информация о стилях представлена в главах 15 и 16.

Определение фонового изображения

Помимо настройки основного фонового цвета документа, в качестве фона можно использовать и изображение. Как и в случае с атрибутом фонового цвета в дескрипторе `<body>`, здесь также есть атрибут фонового изображения (`background`). Однако этот атрибут все же не рекомендуется использовать.

Вместо него лучше использовать свойство `background-image`, как показано ниже:

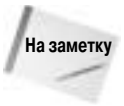
```
<style>
body { background-image: url(путь_к_изображению); }
</style>
```

Например, в результате применения следующего стиля получается изображение, заданное в файле `grid.jpg`, которое размещается на странице в качестве фона документа:

```
<style>
   body { background-image: url(grid.jpg); }
</style>
```

Результат выполнения кода представлен на рис. 3.2.

Рис. 3.2. Решетка, образующая фон документа, сформирована на основе файла `grid.jpg`



При изменении фонового цвета на темный или при использовании темного фонового изображения, необходимо также изменять цвет текста, чтобы он мог контрастировать с фоном. Например, в следующем примере стиль определяет черный фон страницы и белый цвет текста:

```
<style>
   body { background-color: black; color: white; }
</style>
```

Резюме

В этой главе представлено описание основных элементов, наличие которых обязательно для любого HTML-документа. Читатель узнал о некоторых основных правилах кодирования с помощью HTML, а также о способах добавления в документ информации относительно заголовка, например названия, и метаинформации, используемых поисковыми машинами. Кроме того, в главе речь идет об определении основного пути документа, о способах перенаправления пользователя на другую страницу, а также методах изменения фоновых цветов документа.

В последующих главах представлены более подробные сведения о различных HTML-дескрипторах.