Вадим Дунаев

bhv

HTML, скрипты и стили

3-е издание

- HTML 4, HTML 5, XHTML
- CSS 2, CSS 3
- JavaScript , PHP
- SVG

Наиболее полное руководство

в подлиннике®

Вадим Дунаев

HTML, СКРИПТЫ И СТИЛИ

3-е издание

Санкт-Петербург

«БХВ-Петербург»

2011

УДК 681.3.06
ББК 32.973.26-018.2
Д83

Дунаев В. В.

Д83 HTML, скрипты и стили. —3-е изд., переб. и доп. — СПб.: БХВ-Петербург, 2011. — 816 с.: ил. — (В подлиннике)

ISBN 978-5-9775-0502-4

Рассмотрены средства создания Web-сайтов — языки разметки гипертекста (XHTML, HTML 4 и HTML 5), каскадные таблицы стилей (CSS 2 и CSS 3), а также языки сценариев JavaScript и PHP. Изложены краткие теоретические сведения и приведены многочисленные примеры типичных задач разработки сайтов. Предлагаемые решения инвариантны относительно наиболее популярных браузеров (Microsoft Internet Explorer, Mozilla Firefox, Opera, Apple Safari и Google Chrome). Приложения книги содержат краткие сведения о тегах HTML и свойствах CSS. В третьем издании существенно расширены главы, посвященные языкам разметки гипертекста HTML 4, HTML 5 и XHTML и каскадным таблицам стилей CSS 2 и CSS 3. Добавлен материал по масштабируемой векторной графике (SVG) и динамической графике, управляемой скриптами.

Для Web-дизайнеров

УДК 681.3.06
ББК 32.973.26-018.2

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Игорь Шишигин*
Зав. редакцией	*Григорий Добин*
Редактор	*Леонид Кочин*
Компьютерная верстка	*Натальи Караваевой*
Корректор	*Наталия Першакова*
Дизайн серии	*Инны Тачиной*
Оформление обложки	*Елены Беляевой*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 24.01.11.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 65,79.
Тираж 1600 экз. Заказ №
"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.60.953.Д.005770.05.09 от 26.05.2009 г. выдано Федеральной службой
по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12.

ISBN 978-5-9775-0502-4

Оглавление

Предисловие к третьему изданию

Третье издание настоящей книги кардинально отличается от предыдущих как по структуре, так и по содержанию. Существенно расширены главы, посвященные языкам разметки гипертекста HTML 4/5, XHTML и каскадным таблицам стилей CSS 2/3, которые теперь рассматриваются параллельно. Добавлен материал по масштабируемой векторной графике (SVG) и динамической графике, управляемой скриптами (элемент `<canvas>`), а главы по ASP и языку VBScript в данное издание не вошли.

Предыдущие издания книги в основном были ориентированы на господствовавший на рынке браузер Microsoft Internet Explorer версий от 5.0 до 6.0. Тогда коды Web-страниц создавали в основном под Internet Explorer, а их адаптацию к другим браузерам выполняли лишь для охвата весьма малочисленной (2—5%) аудитории, что было весьма хлопотно. Теперь программные коды пишут в соответствии со стандартами, а хлопоты пока доставляет Internet Explorer версий 6, 7 и 8, а не другие современные браузеры.

В данном издании, в отличие от предыдущих, большое внимание уделено соблюдению стандартов и межбраузерной совместимости кодов. По крайней мере, приведенные в книге примеры проверены на работоспособность во всех современных ведущих браузерах, таких как Microsoft Internet Explorer 8.0, Mozilla Firefox 3.6, Opera 11.0, Apple Safari 5.0 и Google Chrome 8.0.

Я старался раскрыть возможности (X)HTML, CSS, JavaScript и PHP для разработки Web-сайтов, не претендуя на полноту изложения. Это не справочник, а скорее расширенное практическое руководство по самостоятельному изучению разнообразных средств и приемов Web-программирования, доступное даже начинающим.

Введение

Как известно, Web-сайты создают с помощью языка разметки гипертекста (X)HTML, каскадных таблиц стилей (CSS) и скриптов (сценариев на языках JavaScript, PHP и др.). (X)HTML применяется для создания (разметки) структуры документа или Web-страницы, таблицы стилей используются для внешнего оформления (стилизации, декорирования) и позиционирования элементов документа, а скрипты — для обеспечения интерактивности и динамичности Web-страниц, а также обработки данных, которыми обмениваются браузер и Web-сервер. Всем этим средствам и посвящена данная книга. Для освоения изложенного материала никаких предварительных знаний Web-программирования не требуется. Наряду с краткими теоретическими сведениями читатель найдет здесь многочисленные примеры решения типичных задач разработки сайтов.

Данная книга не является справочником по (X)HTML, CSS и скриптовым языкам. Ее задача состоит в том, чтобы ввести начинающего разработчика в предметную область создания Web-сайтов, показать некоторые приемы обращения с теми или иными технологическими средствами. Каждая из тем ((X)HTML, CSS и языки сценариев) в действительности настолько обширна, что ее полному изложению можно посвятить отдельную книгу примерно такого же объема, что и данная. Поэтому мне потребовалось тщательно отбирать материал, необходимый на первых порах при решении задач разработки домашних страниц и сайтов. Разумеется, многие интересные вопросы пришлось оставить без ответа. Недостающие сведения можно получить из соответствующих справочников, которые нетрудно найти в Интернете. Вместе с тем хочу заметить, что я не навязываю читателю каких-либо технических приемов в качестве "правильных" и единственно возможных для достижения успеха. Практически любую цель можно достичь несколькими путями. И начинающие, и опытные разработчики должны самостоятельно выбирать средства, наиболее подходящие для решения стоящей перед ними задачи. В этом и состоит, на мой взгляд, главное в индивидуальном творчестве.

В первой части книги (*главы 1—14*) рассматриваются средства (X)HTML и CSS на примерах решения основных задач создания сайтов. При этом я старался не использовать тех тегов и атрибутов, главное назначение которых состоит в задании внешнего вида элементов, предпочитая им инструменты CSS. В *главе 11* рассматривается вставка в (X)HTML-документ содержимого из внешних файлов: графики, Flash-анимации, видео и аудио. *Глава 13* посвящена языку SVG для создания

масштабируемой векторной графики. Хотя SVG — особый язык, он устроен подобно языкам разметки гипертекста и описание на нем графики легко может быть встроено в (X)HTML-документ.

Вторая часть (*главы 15—19*) посвящена языку JavaScript программирования клиентских сценариев. Здесь, помимо прочего, рассматриваются AJAX и работа с элементом `<canvas>` для создания графики, управляемой скриптами.

В третьей части (*главы 20—22*) рассматриваются основы языка PHP программирования серверных сценариев.

В *приложениях 1* и *2* приведены перечни тегов HTML 5 и свойств CSS соответственно.

Все приведенные в книге примеры, за исключением некоторых особо оговоренных случаев, работают во всех современных браузерах.

Ваши отзывы и замечания я буду рад принять в гостевой книге на своем сайте по адресу **http://dunaevv1.narod.ru**. По этому же адресу можно найти множество примеров, в том числе и рассмотренных в данной книге.

Благодарности

Я благодарен многочисленным читателям, приславшим свои замечания и отзывы, которые я попытался учесть при подготовке данного издания.

Особую благодарность я выражаю своей жене Валентине за поддержку в нелегкий для меня период работы над этой книгой.

ЧАСТЬ I

HTML и стили

Глава 1

Что такое HTML и таблицы стилей CSS

В данной главе мы рассмотрим язык разметки и таблицы стилей. Здесь важно, не вникая в тонкости синтаксиса, понять их назначение, взаимосвязь и основные направления применения. Далее мы все проясним и поставим на свои места.

1.1. Языки разметки документа

Допустим, вам требуется создать документ (письмо другу, резюме для работодателя, объявление о продаже какого-то имущества, прайс-лист, литературное произведение и т. п.), который вы намерены опубликовать — распечатать на принтере, а затем отправить в издательство или разместить на Web-сайте. Ваш документ будет содержать обычный текст и, возможно, картинки. Это — максимум из того, что вы можете позволить себе при публикации документа на бумажных носителях. Разумеется, все то же самое можно опубликовать и в Интернете, стремясь максимально расширить аудиторию читателей. Однако даже в этом простейшем случае нередко возникают порой нетривиальные задачи форматирования или, другими словами, оформления документа. Текст должен быть структурирован, т. е. разбит на части (разделы, подразделы, абзацы), иногда снабженные заголовками. Даже обычное письмо объемом всего в одну страницу, как правило, разбивают на несколько абзацев, не говоря уже о статьях и более крупных произведениях, таких как повести и романы. Заголовки и абзацы выделяются, по крайней мере, своим местоположением относительно друг друга. Кроме того, заголовки разделов и даже некоторые словосочетания обычно выделяют из общего текстового потока шрифтом — гарнитурой, размером, цветом и другими характеристиками. Если в тексте предусмотрены графические иллюстрации, то задача форматирования документа еще более усложняется: необходимо определить, каким образом текст будет "обтекать" вставленные картинки.

При публикации документа в Интернете вы можете сделать значительно больше, чем при печати: вставить в документ звуковые и видеоклипы, формы, заполняемые читателем и отправляемые на сервер, элементы интерфейса (ссылки на другие документы, кнопки, поля ввода, переключатели и т. п.), с помощью которых можно управлять отображением содержимого и решать другие задачи.

Даже в обычных бумажных документах нередко встречаются ссылки на разделы этого же или другого документа (например, "см. разд. 2.5", "в книге [7] можно найти…", "в приложении 1 приведен список …" и т. п.). Встретив такую ссылку в книге,

читатель при желании должен либо перелистывать ее страницы, либо искать другую книгу, чтобы затем в поисках нужной информации "порыться" в ней. В Web-документах, просматриваемых в браузерах, щелчок левой кнопкой мыши на ссылке приводит к автоматическому поиску в Сети и отображению в окне браузера соответствующего (релевантного) документа. Такие ссылки в связи с их мощной и удобной функциональностью называют гиперссылками, а тексты, их содержащие, — гипертекстами (HyperText). Гиперссылки могут указывать на документы различного характера — обычные и гипертексты, графические изображения, звуковые и видеоклипы и др. Активизация гиперссылки означает запрос к Web-серверу указанного в ней документа и, если он найден в Сети, пересылку и отображение его в Web-браузере. Благодаря гиперссылкам множество различных документов можно связать в некую единую систему, образовав сайт или, другими словами, Web-узел. Так, вы можете создать, по крайней мере, один документ, содержащий гиперссылки на другие, чтобы не только представить самого себя в Интернете, но и послужить проводником к другим ресурсам Всемирной сети. Простая в своей основе идея и адекватный ей инструмент гиперссылки позволил создать и далее развивать технологии, лежащие в фундаменте Всемирной паутины (World Wide Web), преобразившей в последнее время все информационное пространство и коммуникации в нем.

Итак, при подготовке любого документа к публикации его требуется отформатировать или, иначе говоря, разметить — с помощью специальных символов явно указать, что и каким образом следует представить пользователю (читателю) этого документа. Набор специальных символов (меток, дескрипторов, команд или тегов), а также правила их употребления составляют то, что называют языком разметки (Markup Language). В настоящее время существуют несколько таких языков, среди которых наиболее популярны HTML (HyperText Markup Language — язык разметки гипертекста), XHTML (eXtensible HyperText Markup Language — расширяемый язык разметки гипертекста) и XML (eXtensible Markup Language — расширяемый язык разметки). HTML и XHTML применяются для разметки Web-документов, а XML — как средство структуризации информации для обмена между компьютерными программами. Между этими языками очень много общего, хотя имеются и различия, о которых будет рассказано позже.

Разметка документа, как уже упоминалось, осуществляется посредством специальных дескрипторов, называемых тегами (tag — метка, признак, ярлык). В языках разметки в большинстве случаев они имеют вид: *<имя_тега>* и *</имя_тега>*. Например, `<p>` и `</p>`, где `p` — имя тега. Эти две формы одного и того же тега играют роль подобно открывающей и закрывающей скобкам при написании математических формул. Между ними можно разместить некоторое содержимое — обычный текст и/или тег, содержащий, возможно, другие теги. Тег, имеющий открывающую и замыкающую части, называют еще контейнерным. Вот его синтаксис:

```
< имя_тега > содержимое контейнерного тега </ имя_тега >
```

Существуют и неконтейнерные теги (без замыкающей части), например `<img>` — для вставки графического изображения и `<br>` — для указания перехода на новую строку.

Теги не отображаются в окне браузера, а лишь указывают ему, какие элементы входят в документ и как они между собой соотносятся. Так, текстовый документ состоит из абзацев, которые могут объединяться в разделы. Разделы, в свою очередь, могут входить в более крупные структурные единицы, например главы или части и т. д. Язык HTML обеспечивает поддержку иерархической декомпозиции документа довольно простыми средствами. Например, чтобы указать, что данный фрагмент текста относится к одному и тому же абзацу, достаточно просто перед этим фрагментом написать тег `<p>`, а после него — `</p>`. Другими словами, текст абзаца следует заключить в контейнер `<p>`, например,

```
<p>Здравствуйте, дорогой Иван Иванович.</p>.
```

Чтобы несколько элементов документа (например, абзацев) объединить в один раздел, их следует заключить в какой-нибудь подходящий для этой цели контейнер, например `<div>` (листинг 1.1).

Листинг 1.1. Пример использования контейнера `<div>`

```
<div>
       <p>
           Здравствуйте, дорогой Иван Иванович.
       </p>
       <p>
           В первых строках своего письма передаю поклон Марье Ивановне.
       </p>
</div>
```

Здесь контейнер `<div>` содержит два контейнера `<p>`, каждый из которых заключает в себя некий текст.

Самый крупный контейнер для содержательной части документа — тег `<body>` `(тело документа)`, он содержит другие контейнеры. Например, формируя письмо, мы могли бы написать фрагмент кода на языке разметки, приведенный в листинге 1.2.

Листинг 1.2. Фрагмент кода на языке разметки

```
<body>
   <div>
       <p>
          Здравствуйте, дорогой Иван Иванович.
       </p>
       <p>
          В первых строках своего письма передаю поклон Марье Ивановне.
       </p>
   </div>
</body>
```

Итак, мы уже упомянули несколько тегов: <p>,
, <body>. В HTML имеется много других тегов для группировки элементов документа в логические блоки, образующие в совокупности структуру документа.

На рис. 1.1 в окне обычного текстового редактора показан пример текста, размеченного тегами абзаца <p> и перевода строки
, а также его вид в окне браузера. Хотя данный документ в целом оформлен не по всем правилам и даже совсем не по правилам HTML, основные Web-браузеры все-таки отображают его, причем одинаково. Дело в том, что файл text.htm, содержащий теги HTML, был открыт в Web-браузере. Все современные браузеры при открытии в них файлов с расширениями htm или html пытаются интерпретировать последние как HTML-коды, т. е. как поток тегов языка разметки. Однако это еще не означает, что формировать Web-документ можно "как попало", вспоминая и используя те или иные теги. Правила создания Web-документов существуют, и соблюдать их настоятельно рекомендуется во избежание несовместимости с различными браузерами и других проблем.

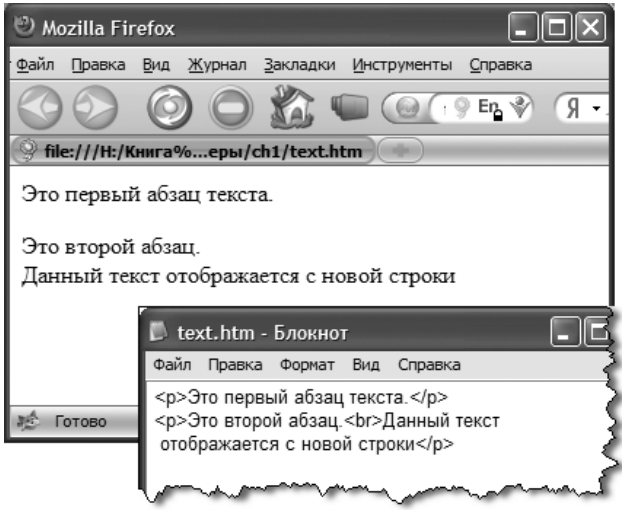

Рис. 1.1. Текст, размеченный тегами абзаца <p> и перевода строки
, в текстовом редакторе и в браузере

Обратите внимание, что в рассматриваемом примере расположение текста в окнах редактора и браузера различается. В текстовом редакторе символы могут располагаться в одной или нескольких строках, что еще не определяет однозначно их расстановку в окне браузера. При наборе символов в текстовом редакторе нажатие на клавиши <Enter>, <Tab> и т. п. приводит к форматированию текста, визуально контролируемому с помощью дисплея. Текстовый редактор отвечает на нажатие указанных клавиш расстановкой в вводимом тексте специальных управляющих символов, которые не видны в общем потоке печатаемого текста. Для управления расположением текста в окне браузера требуется явное задание управляющих символов в виде специальных тегов языка разметки. В рассматриваемом примере тег абзаца <p> указывает браузеру отобразить заключенный в него текст с новой строки

и с отступом в одну строку. В обычном текстовом редакторе этого же эффекта можно добиться, дважды нажав на клавишу <Enter> в конце введенного текста. Далее, переход на новую строку внутри второго абзаца задан с помощью тега `<br>`. Без него этот переход либо не произошел бы, либо мог произойти лишь при достаточно малой ширине окна браузера. Результат зависит от используемого браузера. Дело в том, что браузеры пытаются, так или иначе, перенести поток символов на новую строку, если ширина окна не позволяет расположить текст вдоль одной горизонтальной линии.

Чтобы отображение текста и других элементов документа было инвариантно относительно различных браузеров, необходимо принудительно указывать переносы, отступы и другие управляющие символы с помощью тегов языка разметки и специальных средств (атрибутов тегов и параметров CSS, о чем будет рассказано далее).

О правилах оформления HTML-документов мы поговорим позже, а сейчас остановимся на одной важной особенности HTML. Не все, но многие теги определяют не только структуру документа, но и внешний вид его элементов в окне браузера, т. е. задают еще и способ отображения соответствующей информации в браузере. Например, тег абзаца `<p>` определяет, что все следующее за ним будет отображено с новой строки и с пропуском одной строки; текст, заключенный между тегами `<h1>` и `</h1>`, будет выведен с новой строки максимальным по размеру шрифтом, а последующая информация будет размещена с новой строки (иначе говоря, контейнер `<h1>` задает заголовок первого уровня); текст в контейнере `<i>` выводится курсивом.

Таким образом, разметка тегами HTML в большинстве случаев предопределяет не только структурную декомпозицию документа, но и его внешний вид в окне браузера. Разумеется, структурная декомпозиция, в конечном счете, должна быть как-то выражена внешним образом. Например, листая книгу, мы видим ее структуру в виде иерархии глав, разделов, подразделов и абзацев. Вместе с тем нетрудно понять различие между собственно структурной декомпозицией и внешним представлением ее элементов. В самом деле, главы, разделы и подразделы можно отобразить как линейно следующие друг за другом, так и расположенные на страницах в двух и более колонках; заголовкам можно придать тот или иной вид, манипулируя такими параметрами шрифта, как цвет, размер и начертание. При этом логическая структура документа остается постоянной, а изменяется только ее представление на бумаге или в окне браузера. Однако идея четко разделить структурный и представительский аспекты языка разметки не реализована в полной мере в HTML. В HTML средства структурной декомпозиции документа и его внешнего представления оказались изначально сильно связанными между собой. Именно это обстоятельство и сыграло, на мой взгляд, решающую роль в чрезвычайно широкой популярности данного языка разметки: разработчики Web-страниц хотели сразу видеть результат разметки документа в браузере и получили это (см. рис. 1.1).

Для определения элементов, вставляемых в документ, одних только тегов бывает недостаточно. Так, для вставки графического изображения служит тег `<img>`. Однако требуется еще указать источник изображения, т. е. адрес или, точнее, URL (Uniform Resource Locator — унифицированный указатель ресурса) файла. В подобных случаях используются параметры тегов, называемые еще атрибутами. Один или несколько атрибутов записывают в открывающей части тега в произвольном

порядке в следующем виде: *имя_атрибута="значение"*. Атрибуты отделяют друг от друга пробелами. Вот пример тега для вставки графического изображения:

```
<img src="images/mypicture.jpg">
```

Тег `<img>` не имеет заключительной части вида `</img>`.

Чтобы указать размеры графического изображения на странице (в пикселах), можно записать следующее выражение:

```
<img src="images/mypicture.jpg" width="200" height="100">
```

Здесь `src`, `width` и `height` — имена атрибутов тега `<img>`, за которыми следует знак равенства и, далее, значения, указанные в кавычках, двойных или одинарных.

Атрибуты позволяют указать, например, цвет фона, размеры, начертание и цвет шрифта, характеристики выравнивания элементов документа и т. п. Так, в контейнерном теге `<body>`, задающем основную часть документа, с помощью атрибутов `text` и `bgcolor` можно определить цвет текста и фона:

```
<body text="#ff0000" bgcolor="#00ffee">.
```

Таким образом, развитие HTML пошло в сторону добавления средств, определяющих внешнее представление документов. Стали появляться как новые теги, так и новые их атрибуты. На первых порах производители браузеров (Microsoft, Netscape, Sun Microsystems и др.) создавали свои варианты HTML, отличающиеся не только наборами тегов, но и способами их интерпретации. В сложившихся условиях один и тот же документ мог выглядеть по-разному в зависимости от браузера, что явно противоречило цели всемирного распространения информации. Очевидно, стандарты стали насущной необходимостью.

Разработкой стандарта для HTML занялась международная организация World Wide Web Consortium (W3C, Консорциум Всемирной паутины), в состав которой вошли крупнейшие производители программного обеспечения для работы в Интернете (в том числе Microsoft, Sun Microsystems, Netscape и др.). Сведения о стандартах (спецификациях) можно найти на официальном сайте W3C по адресу **www.w3.org**. Схема подготовки спецификаций консорциумом примерно такая: сначала выпускается проект или черновик (draft) спецификации, в результате обсуждения которого появляется ее рабочий вариант. Последний предлагается к обсуждению в течение некоторого времени, по окончанию которого рабочий вариант может стать рекомендацией — официальным вариантом спецификации. Браузеры должны интерпретировать язык разметки соответственно стандарту, хотя на практике это происходит не в полной мере. Одно дело идея, другое — ее реализация.

В июле 1997 г. консорциум W3C выпустил проект спецификации HTML 4.0, который уже в декабре стал официальной рекомендацией для производителей Web-браузеров. С декабря 1999 г. по настоящее время существует версия 4.01. Тем не менее, различия в интерпретации основными браузерами некоторых тегов и атрибутов официальной спецификации HTML все же остались. Для обеспечения дополнительных возможностей визуального представления, программирования поведения, динамичности и интерактивности документов появились каскадные таблицы стилей (CSS — Cascading Style Sheets) и скрипты (сценарии), написанные на специальных языках, наиболее популярный из которых — JavaScript.

Параллельно с HTML развивался другой язык разметки — XML (eXtensible Murkup Language — расширяемый язык разметки). Он был создан как средство структуризации информации для обмена между программами. Web-браузер — одна из них, но далеко не единственная. При этом на способ внешнего представления структурированных данных никаких ограничений не накладывалось. Главная задача, которая ставилась при разработке этого языка, состояла в обеспечении совместимости между различными системами обработки структурированных данных. Передающей стороне не важно, как будет отображен XML-документ получателем, требуется лишь, чтобы он однозначно разобрался в структуре принятого документа.

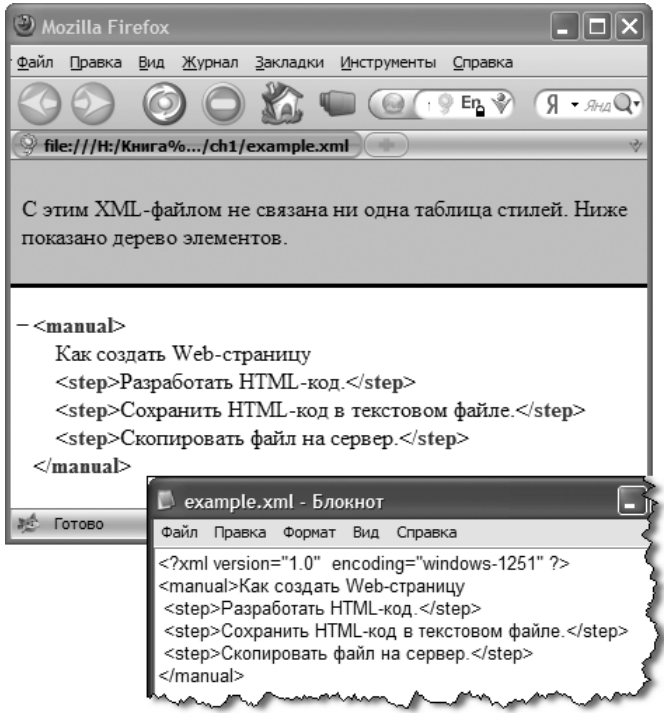

Рис. 1.2. Внешний вид XML-кода, загруженного в браузер

В XML, так же как и в HTML, есть теги и атрибуты. Однако, в отличие от HTML, в языке XML они не предопределены изначально, а могут создаваться автором документа по своему усмотрению. Теги XML задают лишь структуру документа, но не его представление. На рис. 1.2 показан фрагмент XML-кода в текстовом редакторе и результат его обработки Web-браузером. Теги `<manual>` и `<step>` были придуманы автором данной книги для представления структуры инструкции по созданию сайта. Web-браузер не "знает" таких тегов, а потому просто показывает XML-код без какой-либо его интерпретации. Внешний вид элементов XML-документа определяется отнюдь не тегами, а специальными программами-анализаторами, часто называемыми парсерами (parser). Основные Web-браузеры имеют такие встроенные парсеры, но чтобы подключить их к работе и заставить

отобразить собственно информацию (без тегов), необходимо указать, что содержимое документа является XML-кодом и, кроме того, достаточно сослаться на каскадную таблицу стилей (CSS).

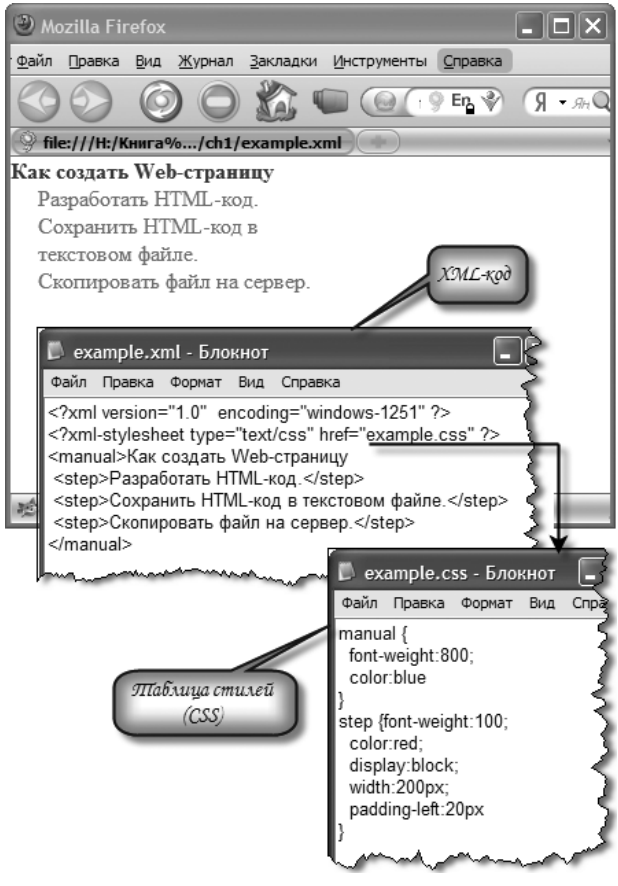

Рис. 1.3. Пример использования таблицы стилей для отображения XML-документа в окне браузера

Таблицы стилей сохраняются в отдельных файлах и содержат параметры визуализации элементов (тегов) XML-документа. На рис. 1.3 показан пример, в котором XML-документ, представляющий инструкцию, в первой строке содержит дескриптор, указывающий тип документа, а во второй строке — ссылку на файл example.css с правилами внешнего представления тегов `<manual>` и `<step>`. В данном случае браузер не отображает теги, а выводит результат их интерпретации. Так, в таблице стилей указаны цвет и "жирность" шрифта, а также требование, согласно которому каждый элемент `<step>` (шаг инструкции) занимает по горизонтали 200 пикселов (`width:200px`) с отступом 20 пикселов слева (`padding-left:20px`) и начинается с новой строки (`display:block`).

При желании можно модифицировать параметры отображения элементов документа, оставляя неизменной его логическую структуру.

Итак, в XML воплотилась хорошая идея разделения между структурным и внешним представлением содержания документа. В HTML 4 это не так, но его можно немного модернизировать, чтобы существенно приблизить к XML. На пути к достижению этой цели появился XHTML (Extensible Hypertext Markup Language — расширяемый язык разметки гипертекста).

> **ПРИМЕЧАНИЕ.** XML — подмножество более сложного языка SGML (Standard Generalized Markup Language — стандартный обобщенный язык разметки). На основе XML созданы более специализированные языки разметки, например XHTML, SVG, RSS, MathML и др. Такие языки еще называют словарями данных языка XML. С другой точки зрения XML — текстовый формат описания иерархических баз данных.

Еще до появления XHTML многие разработчики Web-сайтов практиковали следующий технологический прием. С помощью HTML-тегов создавалась лишь структура документа, при этом теги и атрибуты, предназначенные только для задания внешнего вида элементов, практически не использовались (насколько это было возможно), а отображение элементов документа в окне браузера или при печати определялось только в каскадных таблицах стилей. Данный прием можно назвать "применением HTML в стиле XML".

Разметку Web-страницы можно реализовать на основе языка как HTML, так и XHTML. Однако XHTML более строг, ближе к универсальному языку XML и должен поддерживаться одинаковым образом всеми современными браузерами. Поэтому создавать Web-страницы сейчас рекомендуется с помощью языка XHTML. В данной книге все примеры написаны на языке XHTML. Однако во многих случаях я применял следующий шаблон: в приводимых примерах задан тип документа, указывающий на стандарт HTML 4.01, а собственно код написан в соответствии со стандартом XHTML. Хотя дескриптор `<!DOCTYPE …>` типа документа указывает на его соответствие стандарту HTML 4.01, а не XHTML, ничего крамольного в этом нет. Объявив в дескрипторе `<!DOCTYPE …>` свою приверженность HTML 4.01, вы можете писать свои коды разметки в соответствии с официальной спецификацией как HTML 4.01, так и XHTML.

Читатели, знакомые только с HTML 4, увидят не слишком много отличий этого языка от XHTML. Перейти от HTML-кодов к XHTML-кодам довольно легко. Языки HTML и XHTML очень схожи, но различия все же есть. Вот основные (но далеко не все) из них:

- язык HTML регистронезависимый: не имеет значения, строчными или прописными буквами записаны имена тегов и атрибутов. В XHTML для записи имен тегов и атрибутов рекомендуется указывать только строчные буквы. Например, вместо

  ```
  <IMG SRC="mypicture.jpg">
  ```

  следует писать

  ```
  <img src="mypicture.jpg"/>;
  ```

- в HTML большинство тегов контейнерные (например, `<h1>…</h1>`), т. е. имеют открывающий и закрывающий дескрипторы. Вместе с тем есть и неконтейнерные теги: для дескриптора *<тег>* нет закрывающего дескриптора *</тег>*. Например, для вставки графического изображения в HTML применяется тег `<img src="mypicture.jpg">`, а в XHTML — `<img src="mypicture.jpg"/>`. Неконтейнерные теги еще называют

пустыми. В XHTML для таких тегов необходимо указывать слэш (косую черту) перед закрывающей уголковой скобкой: <тег … />, например

 или
;

- в некоторых тегах HTML имеются атрибуты без значений (так называемые булевы или логические атрибуты), например <option selected>. В XHTML для подобных атрибутов следует явно указывать строковое значение, совпадающее с именем соответствующего атрибута, например

<option selected="selected" />;

- в XHTML значения атрибутов тегов нужно заключать в кавычки. В HTML это делать не обязательно. Например, в XHTML следует писать , а не <img src=mypicture.jpg>, как допустимо (хотя тоже не рекомендуется) в HTML. Некоторые атрибуты принимают числовые значения, которые в HTML допустимо (хотя и не рекомендуется) указывать без кавычек, но в XHTML их следует заключать в кавычки, двойные или одинарные, например

;

- в содержательной текстовой части XHTML-документа нельзя непосредственно использовать символы, которые применяются в коде, такие как <, > и &, причем их нельзя записывать даже в значении URL-адреса. Эти специальные символы следует заменять соответствующими буквенными или числовыми эквивалентами <, > и &amp (или <,&#62 и &) соответственно. Об эквивалентах специальных символов рассказано в *разд. 7.3*;
- кодировка файла по умолчанию для HTML-документа — ISO 8859-1, а для XHTML-документа — UTF-8.

Ошибки в HTML-коде браузеры пытаются преодолеть тем или иным образом, так что даже весьма небрежно написанный код все же как-то отображается (во многих случаях вполне удовлетворительно). Более других терпим к ошибкам в HTML-коде и лоялен к разработчику браузер Microsoft Internet Explorer. По рекомендации W3C в случае ошибки в XHTML-коде браузеры должны сообщать об этом и прекращать дальнейшую обработку. Таким образом, XHTML-код подвергается более тщательному анализу прежде, чем начнется его интерпретация — отображение в окне браузера. HTML-код, напротив, не анализируется предварительно на правильность, а отображается в порядке следования тегов. При этом сообщения об ошибках не выводятся, а отображение продолжается, даже если ошибки все же возникли. Очевидно, HTML более дружествен разработчику, чем XHTML. Но более других эту "дружественность" ценят малоопытные и небрежные разработчики своих домашних страниц, а не авторы серьезных Web-проектов. Для последних однозначность интерпретации и дисциплина важнее "угодливости" браузеров, вредность которой проявляется при поиске трудноуловимых смысловых ошибок и при попытках обеспечить межбраузерную инвариантность представления документов.

В XHTML сохранилось большинство тегов HTML 4. Однако некоторые старые HTML-теги, введенные в свое время для обеспечения дополнительных возможностей отображения, теперь применять не рекомендуется.

В настоящее время существуют официальные версии XHTML 1.0 и 1.1. и черновая спецификация XHTML 2.0. Однако работа над последней остановлена в пользу создания нового стандарта — HTML 5.

Начиная с 2004 г. совместными усилиями рабочих групп W3C HTML WG и WHATWG с привлечением таких компаний, как Apple, Mozilla, Opera, Google, Microsoft и др. ведется работа по созданию спецификации HTML 5. К середине 2010 г. существовал лишь черновой ее вариант (**www.w3.org/TR/html5**, **www.whatwg.org/specs/web-apps**). От W3C было заявление, что официальная рекомендация должна появиться к концу 2010 г. по другим источникам — 2012 г. Полная поддержка HTML 5 браузерами осуществится, видимо, еще позднее, но некоторые элементы Firefox, Opera, Safari и Chrome воспринимают уже сейчас. Браузер Internet Explorer приступит к поддержке HTML 5, начиная с версии 9.

HTML 5 призван стать преемником как HTML 4, так и XHTML 1.0, обеспечивая совместимость с ними, по крайней мере, в самом важном. В HTML 5 можно использовать синтаксис как обычного HTML, так и XHTML. Вместе с тем ряд тегов и атрибутов, не рекомендованных к использованию в прежних версиях, в новом стандарте отсутствуют. Например, теги `<applet>` для вставки Java-апплетов, `<center>` для центрирования содержимого по горизонтали и `<frameset>`, `<frame>` для разбиения окна браузера на фреймы в HTML 5 не поддерживаются. С целью обеспечения обратной совместимости в спецификации HTML 5 описывается, как должен поступить браузер в случае использования непредусмотренных тегов. Уже существующие сайты не должны пострадать из-за появления нового языка разметки.

В HTML 5 вводится в оборот ряд новых тегов с отчетливо выраженной семантикой, предназначенных для повышения эффективности разработки Web-приложений, а также для помощи поисковым системам. Так, для создания логической структуры и упрощения типовой верстки документа предлагаются теги `<header>` (заголовок), `<nav>` (навигационная панель), `<article>` (статья), `<section>` (раздел, секция статьи), `<aside>` (боковая колонка), `<footer>` ("подвал"), `<menu>` (меню) и др. В прежних версиях данные элементы обычно создаются тегами `<div>` и сразу понять смысл их применения довольно трудно. Иначе говоря, семантика для в спецификации неоднозначна.

Элементы `<input>` полей ввода данных в HTML 5 приобрели новые атрибуты, заметно облегчающие создание пользовательского интерфейса: `time`, `email`, `url`, `required` и др. Эти атрибуты во многих случаях позволят обойтись без скриптов, проверяющих правильность введенных данных.

Особого внимания заслуживает элемент `<canvas>`, предоставляющий область на странице, в которой с помощью скриптов на языке JavaScript можно рисовать изображения, подобные тем, которые создаются посредством SVG или Flash. Для вставки мультимедиа в HTML 5 предусмотрены теги `<audio>` и `<video>`, обеспечивающие внедрение на страницу соответствующего ресурса и возможность управления его воспроизведением.

HTML 5 предусматривает и другие полезные и удобные возможности. В данной книге будут описаны лишь некоторые из них, которые уже поддерживаются ведущими браузерами.

> **ВНИМАНИЕ!** В данной книге описываются возможности HTML 4 и XHTML, а также применение некоторых элементов HTML 5. Коды примеров написаны в соответствии с синтаксисом XHTML, хотя дескриптор типа документа `<!DOCTYPE...>` в некоторых из них указывает на стандарт HTML 4.01. Такая эклектика вполне допустима. Разумеется,