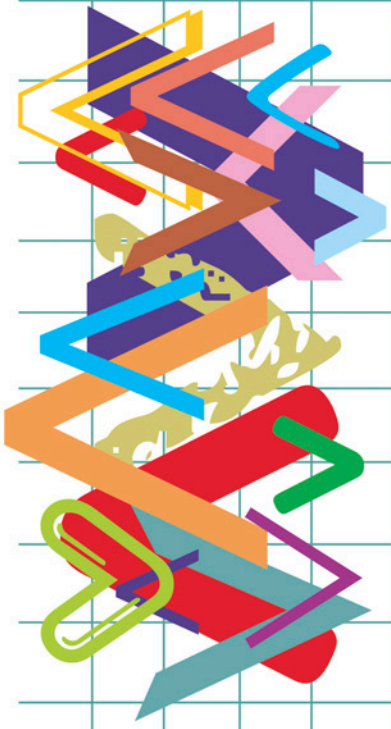
bhv
www.bhv.ru
www.bhv.kiev.ua

Игорь Шапошников

HTML 4

САМОУЧИТЕЛЬ



Стилевое оформление страниц средствами CSS

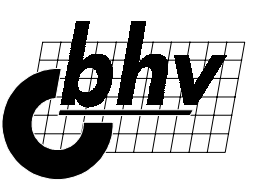
Создание сценариев с помощью JavaScript

Использование DHTML для управления Web-страницами

От Web-страницы к полноценному сайту

Игорь Шапошников

САМОУЧИТЕЛЬ HTML 4



Дюссельдорф ◆ Киев ◆ Москва ◆ Санкт-Петербург

УДК 681.3.06

Книга знакомит читателей с современными технологиями разработки Web-страниц и Web-сайтов. Подробно описан язык HTML версии 4.1. Рассматриваются способы стилевого оформления Web-страниц с помощью каскадных стилевых таблиц. Обсуждаются приемы создания динамических элементов в HTML-документах средствами технологии DHTML и языка JavaScript. Большое количество примеров, листингов и иллюстраций делают книгу особенно полезной для начинающих Web-дизайнеров. Профессионалы также смогут найти интересную информацию.

Для широкого круга пользователей

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зав. редакцией	*Наталья Таркова*
Редактор	*Татьяна Коротяева*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Дизайн обложки	*Игоря Цырульникова*
Зав. производством	*Николай Тверских*

Шапошников И. В.
Самоучитель HTML 4. — СПб.: БХВ-Петербург, 2001. — 288 с.

ISBN 5-94157-123-2

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 21.09.01.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 23,22.
Тираж 4000 экз. Заказ
"БХВ-Петербург", 198005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар, № 77.99.1.953.П.950.3.99
от 01.03.1999 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в Академической типографии "Наука" РАН.
199034, Санкт-Петербург, 9-я линия, 12.

Содержание

Благодарности

Любая книга — это плод деятельности многих людей, которые часто остаются читателю неизвестны. Мне хотелось бы поблагодарить всех, кто помог в работе над этой книгой.

Прежде всего, весь редакторский состав издательства "БХВ-Петербург" и всю группу подготовки издания.

Спасибо моей Ольге — за все. Компании "Петерлинк" — за все, что эти парни сделали для нас. Особая благодарность Сергею Торопу и Юрию Степанову. Моим друзьям в Сети. Финисту, Буке, Лайке, Панку, Умке и многим-многим другим. Мой почтовый ящик всегда открыт для вас.

Огромное спасибо всем читателям этой книги. Ваши вопросы и отзывы показывают, что работа была проделана не зря.

Ольге

Твоя улыбка стоит целого мира.

Введение

Самая распространенная и известная часть Интернета — так называемая "паутина", World Wide Web. Она состоит из Web-сайтов и отдельных Web-страниц, а Web-страницы создаются с помощью языка HTML. Этот язык и является предметом нашего интереса. Мы рассмотрим новейший стандарт языка HTML, его четвертую версию. Если быть совсем точным, то версию под номером 4.1.

Язык HTML довольно прост, и пользоваться его возможностями может каждый, для этого совсем необязательно быть программистом. Достаточно иметь некоторый опыт работы в Интернете и быть обычным пользователем компьютера. Никаких специальных навыков не нужно. Это *действительно* просто. Мы вместе в этом убедимся. С помощью языка HTML каждый в состоянии создать собственную Web-страничку или целый Web-сайт и представить себя таким образом во всемирном информационном пространстве. Впрочем, для чего создавать свой Web-сайт — каждый решает самостоятельно. Задача этой книги — лишь дать читателю подходящий инструмент.

Книга разбита на три главы. В первой главе мы тщательно и подробно рассмотрим полный стандарт языка HTML версии 4.1. с примерами и иллюстрациями. Вторая глава посвящена вопросам единого стилевого оформления Web-страниц с помощью так называемых каскадных стилевых таблиц. Эта технология носит название CSS (Cascading Style Sheets). В заключительной главе мы коснемся вопросов использования динамических элементов. Технология, которую мы будем для этого применять, так и называется — динамический HTML (DHTML). Таким образом, мы научимся создавать полноценные информационные Web-сайты.

Конечно, только знания языка HTML недостаточно для того, чтобы сделать интерактивный торговый сайт. Для таких серьезных целей требуется хоро-

шее знание программирования и различных технологий Интернета. Но в том-то и преимущество языка HTML, что, пользуясь этим несложным инструментом, можно создавать полноценные информационные Web-сайты. Средств HTML с лихвой хватит для нужд большинства посетителей паутины, за исключением пяти или десяти процентов. Если же возникнет надобность использовать на своем сайте какую-либо специализированную технологию, то придется изучить что-то еще, благо литературы сейчас достаточно. Но перед тем как обращаться к новейшим средствам сетевого программирования, следует научиться использовать обширные возможности HTML. Из этой книги мы узнаем о них практически все.

Несмотря на то, что язык HTML достаточно прост по сравнению с прочими современными технологиями сетевого программирования, Всемирная паутина — самая обширная и популярная часть Интернета — целиком базируется на огромной массе частных и корпоративных сайтов, которые созданы только с помощью этого языка. Технология Интернета, а вместе с ней и язык HTML, относятся к тем немногим достижениям компьютерной индустрии, которые повлияли на распространение и развитие гражданских свобод. Любой человек, имеющий доступ в Интернет, пусть и нерегулярный, может получать и сообщать информацию, не прошедшую государственный или корпоративный контроль. Таким образом, свобода слова и легко реализуемое право на получение правдивой информации стали самыми важными достижениями общества, обеспечиваемыми технологиями Интернета. Если прибавить к этому легко реализуемую анонимность или, другими словами, приватность работы в Сети, то становится ясно, что с помощью Интернета человек может выйти из-под информационного контроля, который в той или иной форме обязательно проводится любым современным государством. Плату же за предоставление доступа в Интернет можно считать налогом на свободу, который следует выплачивать сполна. Интернет — это, прежде всего, свобода и право на самовыражение, но для того, чтобы воспользоваться этими преимуществами, следует освоить соответствующие технологии. Мы уже говорили, что для Всемирной паутины база и основа — язык HTML.

ГЛАВА 1

Язык HTML

Основы

Интернет — это огромное объединение компьютерных сетей в планетарном масштабе. Если учесть, что и обитатели международной космической станции пользуются услугами электронной почты, то становится ясно, что Интернет уже шагнул за пределы планеты. Очень часто Интернет ошибочно отождествляют с самой популярной и масштабной его частью — Всемирной паутиной, которая в английском языке получила наименование WWW (World Wide Web). По сути дела, паутина — это огромное количество взаимосвязанных документов. Ключевое слово — *взаимосвязанных*. В текст Web-страницы органично вставляются гиперссылки, которые обеспечивают соединения с другими Web-страницами. Описать несколькими словами механизм гиперссылок трудно, но тот, кто хоть раз посетил какой-либо Web-сайт, сразу поймет их значение.

Именно гиперссылки, позволяющие связывать друг с другом самые различные документы из Сети, создали ту удивительную общность, которая составляет теперь главную отличительную черту Всемирной паутины. Гиперссылки используют для поиска документа его уникальный адрес во Всемирной паутине, который также называется URL (Universal Resource Locator).

Как мы знаем, основное назначение Web-страниц — *отображать* информацию и делать ее доступной для пользователя. При этом существует ряд функциональных ограничений: заранее неизвестно, какой именно компьютер у пользователя, просматривающего Web-страницу, какое разрешение у его монитора, или какие размеры окна просмотра он установил. Мы даже не можем заранее знать, какая используется операционная система или платформа. Web-страницы должны практически одинаково отображаться и на Intel-машинах, и на Макинтошах, и на телевизионных Web-приставках. Неизвестно, какие шрифты установлены и используются в операционной системе пользователя, или какая глубина цвета поддерживается его видеокар-

той. Отсутствие или недостаток информации должны были бы стать непреодолимым барьером на пути создания общего языка, но этого не случилось.

Дело в том, что еще в 1986 году Международной организацией по стандартизации ISO (International Organization for Standardization) был создан язык разметки документов SGML (Standard Generalized Markup Language), который предусматривал практически все допустимые варианты отображения информации как на бумаге, так и на мониторах. Чтобы учесть все возможные случаи, была разработана мощная система. Казалось бы, для Web-страниц — это идеальный вариант. Но только описание правил этого языка занимает сотни страниц. На разработку программ, которые могли бы отображать страницы, созданные на основе такого языка, ушло бы очень много времени, поэтому для нужд Интернета было выбрано некоторое подмножество языка SGML, которое получило самостоятельное наименование — HTML (HyperText Markup Language), т. е. язык разметки гипертекстовых документов.

В файле описания Web-страницы на языке HTML основная информация чередуется с инструкциями по ее отображению. По сути, это обычный текстовый файл, но читать его без применения соответствующих специализированных программ-браузеров трудно, т. к. инструкции по отображению информации затрудняют чтение текста. А графику тем более нельзя увидеть, потому что в самом HTML-файле вместо графики стоит тэг, указывающий браузеру, что именно в это место надо вставить некое изображение.

Web-страницы, написанные на HTML, просматриваются с помощью специализированных программ, которые обычно называют *браузерами*. Это калька англоязычного термина. Прямой перевод на русский язык, программы-обозреватели, почему-то не прижился. Что ж, будем называть их устоявшимся термином. Основная задача браузера — по запросу пользователя найти требуемый документ в Интернете и без искажений отобразить его. Сначала браузер анализирует инструкции, написанные на языке HTML, а затем, пользуясь этими инструкциями, отображает информацию, находящуюся на Web-странице. Отсюда вывод — если мы хотим создавать собственные Web-страницы, то жизненно необходимо знать язык HTML.

Конечно, можно писать код Web-страницы вручную, пользуясь каким-нибудь простеньким текстовым редактором, таким, например, как тривиальный "Блокнот", но это не самое приятное времяпрепровождение. Сейчас существует огромное множество визуальных редакторов Web-страниц, которые позволяют простым и естественным образом размещать информацию, не заботясь о ее переводе в HTML. Казалось бы, если все так замечательно, то зачем изучать HTML самостоятельно? Оказывается, при создании HTML-кода эти редакторы в некоторых случаях пишут неверный, частично неверный или избыточный код. Иногда просто не удается добиться именно того результата, который необходим, и надо знать язык HTML, чтобы понять, как преодолеть барьеры, встроенные в эту технологию.

Также следует упомянуть о последствиях так называемой "браузерной войны". Дело в том, что в самом начале развития WWW лидерство на рынке захватил браузер Netscape Navigator от Netscape. Фирма Microsoft изначально не смогла правильно оценить будущего потенциала WWW, и браузер Netscape Navigator благополучно завоевал почти весь рынок. Но когда менеджмент Microsoft понял, что они упустили, в ход пошла вся тяжелая артиллерия. Срочно был создан браузер Internet Explorer, и началась браузерная война. В целях продвижения собственного браузера каждая из конкурирующих фирм немного "улучшала" стандарт HTML, т. е. добавляла в него конструкции, которые мог правильно обрабатывать только собственный браузер. Конечно, WWW Consortium, организация, курирующая развитие интернет-технологий, некоторые из этих новшеств включала в последующие версии стандартов, но конкуренты не останавливались на достигнутом. Более того, в рамках все той же браузерной войны Microsoft включила свой браузер в операционную систему Windows 9x. Закончилось вся эта эпопея, как мы помним, судебным разбирательством, в котором ставший в одночасье известным всему миру судья Джонсон обязал Microsoft вывести браузер Internet Explorer из состава операционных систем.

К тому времени задача-минимум уже была решена. Браузер Internet Explorer занял около половины объема рынка браузеров. Так закончилась "браузерная война". Или, если быть точным, перешла из "горячей" фазы в "холодную". Последствия этого противостояния до сих пор ощущаются разработчиками Web-страниц, т. к. им приходится проверять, как выглядит разрабатываемая страница в каждом браузере. Более того, некоторые визуальные HTML-редакторы прямо ориентированы на тот или иной браузер, и, следовательно, используя их, разработчик тоже будет ориентироваться только на один браузер или ограничивать функциональность и наполнение Web-страниц, упрощая их структуру до предела.

Но выход есть. Код, генерируемый визуальным редактором, практически всегда необходимо исправлять вручную, а для этого, повторюсь, стоит знать язык HTML. Без знания HTML просто нельзя создавать Web-страницы хорошего качества. В любой технологии есть свои подводные камни, и если мы не знаем основы этой технологии, мы обязательно на них наткнемся. Для того чтобы добиться максимального соответствия создаваемой Web-страницы первоначальному замыслу, действительно необходимо изучать язык HTML. Чему, собственно, и посвящена эта книга — замечание для тех, кто еще не догадался...

Структура HTML-документа

Конструкции HTML называются *тэгами*. Для того чтобы браузер мог отличить их от обычного текста, они заключаются в угловые скобки. Тэг обозначает начало действия какой-либо инструкции отображения. Если эта инст-

рукция применяется ко всему документу, то такой тэг не имеет своего закрывающего двойника. Большинство тэгов все-таки парные, и второй тэг прекращает действие первого. Например, каждая Web-страница должна начинаться с тэга <html>, а заканчиваться его закрывающим двойником </html>. Обратите внимание, закрывающий тэг отличается от открывающего лишь наличием косой черты после первой угловой скобки.

Некоторые тэги обладают параметрами, которые уточняют правила отображения содержимого. Немного позже мы на примере увидим, как применяются эти параметры, а сейчас лишь отметим, что параметры могут указываться только в открывающем тэге.

Наименования тэгов и их параметров могут быть написаны в любом регистре как заглавными символами, так и строчными. Анализаторы HTML, встроенные в каждый браузер, не обращают внимания на регистр символов, которыми набраны все служебные конструкции HTML-документов.

В HTML, как и в любом компьютерном языке, нельзя обойтись без комментариев, содержимое которых не обрабатывается браузером и не отображается. Они служат лишь для удобства разработчика, для внутреннего документирования структуры текста. Комментарии располагаются между фрагментами <-- и -->. Вот пример создания комментария:

```
<-- Это комментарий -->
```

Любая Web-страница структурно разбивается на две части: заголовок и тело. В заголовке указывается служебная информация обо всей Web-странице, а в теле Web-страницы мы описываем ее содержимое вместе с правилами его отображения. При этом заголовок Web-страницы ограничивается тэгами <head> и </head>, а тело документа обозначается тэгами <body> и </body>. По правилам хорошего стиля программирования перед заголовком ставится идентификатор применяемого стандарта HTML. Структура любой Web-страницы выглядит следующим образом:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
↳"http://www.w3.org/TR/HTML4/strict.dtd">
<html>
  <head>
  Заголовок документа
  </head>
  <body>
  Тело документа
  </body>
</html>
```

Первый тэг <!DOCTYPE> со всеми его параметрами — это идентификатор, который сообщает браузеру, какая именно версия HTML была использована для создания данной Web-страницы. Эта достаточно громоздкая и непонятная конструкция на самом деле является пришельцем из языка более высо-

кого уровня, чем HTML. Подразумевается, что в будущем браузеры смогут работать одновременно как с обычными Web-страницами, написанными на языке HTML, так и с HTML-документами. С расчетом на это светлое будущее и используется данный тэг-идентификатор. Точная дата наступления этого светлого будущего, как обычно, неизвестна, поэтому очень часто этим идентификатором пренебрегают без каких-либо последствий. Но предусмотрительность, как известно, лучше, чем недальновидность, поэтому идентификатор лучше все-таки использовать.

Теперь рассмотрим заголовок. В него могут входить тэг, отображающий наименование Web-страницы, тэг стилевого оформления Web-страницы, тэг выполняемого сценария и так называемые метаданные. Стилевое оформление Web-страниц будет рассматриваться во второй главе, а выполняемые сценарии — в третьей. О метаданных мы поговорим чуть позже, а сейчас узнаем, как использовать наименование Web-страницы.

Вы наверняка замечали, что при загрузке Web-страницы в самой верхней строке браузера появлялось краткое наименование загружаемого документа. Для создания такого заголовка используется тэг `<TITLE>` с соответствующей закрывающей конструкцией. Начальный блок Web-страницы с обозначением подобного заголовка может выглядеть следующим образом:

```
<head>
<title> Заголовок Web-страницы</title>
</head>
```

Заголовком Web-страницы никогда не следует пренебрегать, т. к. это самое первое, что видит посетитель Web-сайта. Заголовок отображается еще до того, как произойдет окончательная загрузка содержимого страницы, поэтому выбирать его следует тщательно.

С первой частью структуры Web-страницы мы разобрались и теперь можем переходить к рассмотрению тела HTML-документа, его основной части. Как мы уже знаем, содержимое Web-страницы располагается между тэгами `<body>` и `</body>`. В простейшем случае это может быть обыкновенный текст. Браузер правильно интерпретирует его и отобразит. Попробуем увидеть это на примере.

Для создания нашей первой Web-страницы нам потребуется обычный текстовый редактор. Стандартный "Блокнот" вполне подойдет. Нам достаточно будет создать текстовый файл, содержимое которого приведено в листинге 1.1.

Листинг 1.1

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
↳"http://www.w3.org/TR/HTML4/strict.dtd">
<html>
```

```
  <head>
  <title> Моя первая Web-страница</title>
  </head>
  <body>
  Доброго времени суток всем посетившим мой скромный сайт.
  </body>
</html>
```

Не забудьте, сохраняя файл, установить для него расширение htm или html. Если после этого один или два раза (в зависимости от настроек операционной системы) щелкнуть кнопкой мыши по имени файла в Проводнике Windows, то автоматически будет запущен браузер, установленный по умолчанию, и в него уже будет загружен выбранный HTML-документ. Как выглядит наша первая Web-страница в браузере Internet Explorer, видно на рис. 1.1.

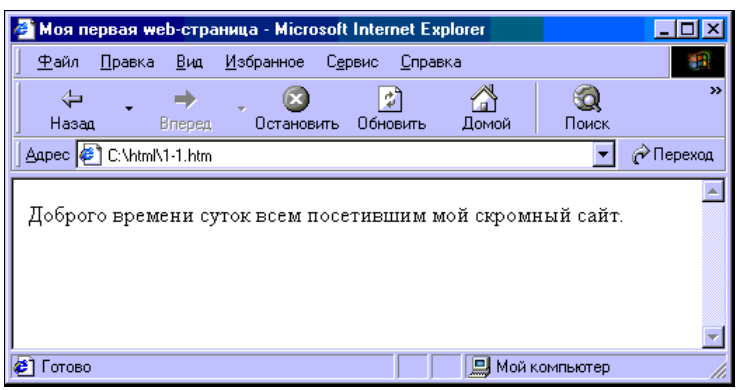


Рис. 1.1. Результат просмотра в браузере файла, приведенного в листинге 1.1

Следует отметить, что тэг `<body>` может содержать дополнительные параметры. Мы уже говорили немного ранее, что параметры включаются в состав стартового тэга конструкции. Теперь пришло время увидеть, как это происходит на самом деле.

Чаще всего параметр представляет собой пару "наименование-значение". Рассмотрим простой пример. Параметр `bgcolor` позволяет задавать цвет фона, на котором будет отображаться содержимое Web-страницы. Например, если мы хотим использовать зеленый фон, то мы должны применить следующую конструкцию:

```
<body bgcolor="green">
```

Все текстовые значения параметров обычно заключаются в кавычки. О том, как задаются цвета, мы узнаем в следующих разделах этой главы, а сейчас вернемся к параметрам тэга `<body>`.

О параметре `bgcolor`, который позволяет устанавливать цвет фона Web-страницы, мы уже знаем. Рассмотрим остальные параметры.

- Параметр `background` дает возможность использовать в качестве фона какое-либо графическое изображение. Значением параметра является адрес этого изображения, т. е. его URL (Universal Resource Locator).
- Параметр `text` задает цвет шрифта, которым будет отображаться текстовое содержимое Web-страницы.
- Параметр `link` позволяет устанавливать цвет, которым будут отображаться в окне просмотра браузера текстовые гиперссылки, внедренные в содержимое Web-страницы.
- Параметр `vlink` задает цвет гиперссылок, которые пользователь уже просматривал в текущем сеансе работы.
- Параметр `alink` определяет, какой цвет будет использоваться для отображения гиперссылок, выделенных пользователем.
- Параметр `lang` указывает, на каком языке написано текстовое содержимое Web-страницы. В качестве значения используются кодовые двухбуквенные обозначения языков, приведенные в документе RFS 1766. Все обозначения нам знать не нужно. В подавляющем большинстве случаев мы будем использовать русский или английский язык. Их коды — "`ru`" и "`en`", соответственно.

Помимо вышеперечисленных параметров тэг `<body>` может обладать двумя идентифицирующими параметрами `id` и `class`, но на практике они в этом тэге почти никогда не задаются.

Как видно, все просто и незатейливо. Теперь самое время узнать, что же такое метаданные. Метаданные можно определить как неотображаемую информацию о документе. Она используется для идентификации документа и указания режима отображения Web-страницы. Для внедрения метаданных в Web-страницу применяется тэг `<meta>`. Чаще всего он имеет следующий вид:

```
<meta name="имя переменной" content="значение переменной">
```

Таким образом, если мы хотим указать авторство какой-либо Web-страницы, достаточно вставить в блок ее заголовка следующую конструкцию:

```
<meta name="Author" content="It's me!!!">
```

Установка собственных переменных в метаданных необходима только в том случае, если Web-страницы обрабатываются с помощью какого-либо специализированного интернет-приложения. Метаданные могут понадобиться и для просмотра Web-страницы браузером.

Как мы все знаем, недостаточно просто поместить сайт во Всемирную паутину, надо еще сделать так, чтобы он попал в списки поисковых машин. Мы не будем сейчас рассматривать полностью процедуру регистрации сайта

на поисковых машинах, тем более что для каждой такой машины процедура регистрации своя. Нас интересует кое-что другое.

Откуда поисковые машины берут информацию о содержимом той или иной Web-страницы? Как раз из метапеременных. Чаще всего используются метапеременные с именами `keywords` и `description`. Переменная `keywords` содержит список ключевых слов Web-страницы. А переменная `description` предназначена для хранения краткой аннотации Web-страницы. Приведем пример использования подобных метаданных. Предположим, что наша Web-страница посвящена сложному и деликатному вопросу правильного кормления хомячков, тогда ее структура должна выглядеть приблизительно следующим образом:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
↳"http://www.w3.org/TR/HTML4/strict.dtd">
<html>
  <head>
  <title> Все о кормлении хомячков</title>
  <meta name="keywords" lang="ru" content="хомяк, грызун, кормление,
↳животное, уход">
  <meta name="description" content="Web-страница о кормлении мелких
↳грызунов, в частности хомяков, в условиях домашнего содержания">
  </head>
  <body>
  О, эти маленькие симпатичные животные — хомячки...
  </body>
</html>
```

Зная, что идентификация содержимого Web-страниц поисковыми машинами осуществляется с помощью ключевых слов, указываемых разработчиком, вы можете ввести в их состав слова, не отражающие суть документа, но часто запрашиваемые посетителями поисковых машин. Должен заметить, что этот фокус, вероятно, не получится. Дело в том, что поисковые машины чаще всего проверяют еще и текст самой Web-страницы, и если какое-либо ключевое слово не встречается в нем, то оно просто не учитывается.

Следует также обратить внимание на то, что при указании ключевых слов мы добавили в тэг `<meta>` дополнительный параметр `lang`. Этот параметр предназначен для указания языка, на котором написан тот или иной текст. Мы можем задать наборы ключевых слов на разных языках, используя для этого несколько тэгов `<meta>`. В нашем примере мы указали, что перечисленные ключевые слова написаны на русском языке.

Кроме того, метаданные позволяют описывать так называемые заголовки HTTP. Здесь необходимо сделать маленькое техническое отступление. Все HTML-документы передаются с помощью специализированных программ, называемых Web-серверами, по определенным правилам. Набор правил

приема и передачи информации в компьютерной индустрии называется протоколом. Для передачи Web-страниц и данных от удаленных пользователей применяют протокол HTTP (HyperText Transfer Protocol). Он обладает набором директив и переменных, которые часто называют заголовками HTTP-протокола.

Мы не ставим перед собой задачу изучить все переменные протокола HTTP, нам достаточно будет узнать о наиболее часто применяемых его заголовках. Прежде всего, стоит упомянуть о переменной `Expires`, которая позволяет устанавливать так называемый "срок годности" Web-страницы. Дело в том, что браузеры и некоторые другие коммуникационные программы сохраняют посещенные пользователем Web-страницы в кэше, а затем, когда пользователь запрашивает их снова, предлагают ему эти копии, экономя, таким образом, время получения. Web-страницы все-таки достаточно часто обновляют, поэтому пользователь может получить устаревшую копию страницы. Конечно, существуют способы настройки правил работы с кэшем, но далеко не все их используют. Лучше подстраховаться и указать "срок годности" Web-страницы. Если он прошел, то браузер вместо использования копии из кэша запросит документ из Сети.

Тэг `<meta>`, приспособленный для указания срока годности Web-страницы, выглядит приблизительно следующим образом:

```
<META http-equiv="Expires" content="Tue, 20 Aug 1996 14:25:27 GMT">
```

Из примера видно, что для указания наименования стандартной переменной HTTP-протокола используется параметр `http-equiv`, а для установки значения этой переменной — уже знакомый нам параметр `content`. Легко заметить, что установка срока последнего использования документа производится с помощью переменной `Expires`, ее значение должно быть записано в определенном текстовом формате с указанием времени по гринвичскому меридиану.

Информация на страничке может меняться настолько быстро, что ее необходимо несколько раз перезагружать в процессе одного сеанса работы. Это достаточно частое явление встречается в чатах и на Web-страницах с информацией, обновляемой в реальном времени, например, при отображении изменений котировок ценных бумаг во время операционного дня на фондовой бирже. В этом случае необходимо использовать переменную с наименованием `Refresh`. Значение этой переменной указывается в секундах. Рассматриваемый нами тэг `<meta>` приобретет следующий вид.

```
<META http-equiv="Refresh" content=10>
```

Страница с подобной конструкцией в блоке заголовка будет автоматически перезагружаться каждые десять секунд.

На этом заканчивается рассмотрение структуры заголовка HTML-документа. Мы переходим к изучению структуры основного раздела Web-страницы.

Как мы помним, вся отображаемая в окне просмотра браузера информация размещается между тэгами <body> и </body>. О том, какие возможности отображения содержимого Web-страницы нам предоставляет язык HTML, мы узнаем в следующих разделах этой главы. Здесь мы рассматриваем лишь общую структуру HTML-документа.

HTML позволяет для каждого применяемого тэга задать уникальный идентификатор. Скажем, если наш текст разбит на абзацы, то каждому абзацу мы можем присвоить специфичное наименование, а затем, с помощью некоторых дополнительных средств языка HTML, управлять отображением этих абзацев. Мы можем делать некоторые из них невидимыми, менять цвет шрифта, т. е. изменять правила их отображения. Это относится не только к абзацам, а ко всем частям содержимого Web-страницы, которые заключены в те или иные тэги.

Для идентификации какого-либо тэга применяется параметр id. Вернемся к примеру с абзацами текста. Забегая немного вперед, можно сказать, что абзацы указываются с помощью пары тэгов и . Таким образом, описание абзацев, которые мы сможем потом отличать, выглядит так:

```
<p id="p1">Первый абзац</p>
<p id="p2">Второй абзац</p>
```

Значения всех параметров id в HTML-документе должны быть уникальными. Если встречается пара идентификаторов, имеющих одинаковые значения, то эти идентификаторы просто игнорируются. Естественно, применение параметра id не является обязательным. Имеет смысл использовать его только в тех случаях, когда конструкция с идентифицируемым тэгом будет подвергнута стилевой обработке (о которой мы поговорим во второй главе), или этот тэг будет закладкой в документе, на которую указывает какая-либо гиперссылка, либо предполагается динамическая обработка тэга с помощью инструкций DHTML, о которых мы узнаем в третьей главе. Идентификаторы применяются и в тех случаях, когда HTML-документ обрабатывается специализированными приложениями, но это уже для совсем серьезных программистов. Нам это пока не нужно.

Если параметр id позволяет отличать тэг от всех остальных, то с помощью параметра class мы можем относить тэг к той или иной группе. Этот параметр используется только для стилевого оформления. Мы разбиваем некоторые элементы Web-страницы на классы, а затем достаточно в одном месте изменить описание правил отображения класса, и это изменение автоматически распространится на все тэги, которые вошли в искомый класс.

Нам доступны методы объединения соседних элементов Web-страницы в единые блоки. Все элементы оформления HTML-документов разделяются на два типа: Inline-элементы, которые чаще всего являются просто элементами текста, и блочные элементы. Inline-элементы могут быть частью стро-

ки, а блочные элементы всегда занимают обособленное место на Web-странице и обязаны начинаться постоянно с новой строки. Блочные элементы могут включать в себя другие блочные элементы и inline-элементы. По вполне понятным причинам inline-элементы не могут содержать блочные элементы.

Объединение элементов Web-страницы в блоки позволяет применять к ним единое оформление, осуществлять некое подобие верстки. Можно изменить расположение блока, переопределив один объединяющий тэг. Естественно, это удобнее, чем менять расположение каждого элемента Web-страницы по отдельности.

Для объединения элементов блочного типа используется тэг `<div>` с его закрывающим двойником`</div>`. Inline-элементы объединяются парой тэгов `<span>` и `</span>`. Учитывая вышесказанное, ясно, что блок с тэгом `<div>` не может располагаться внутри блока с тэгом `<span>`, т. к. блочные элементы не могут входить в состав inline-элементов.

Также следует отметить, что браузеры обрамляют div-блоки разрывами строки. Проще всего это показать на примере.

Листинг 1.2

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
↳"http://www.w3.org/TR/HTML4/strict.dtd">
<html>
  <head>
  <title> Отображение div-блоков</title>
  </head>
  <body>
  <div>
  Доброго времени <div>суток всем посетившим </div>мой скромный сайт.
  </div>
  </body>
</html>
```

Результат отображения этого HTML-файла браузером Internet Explorer показан на рис. 1.2.

Тэги `<span>` и `<div>` могут также иметь дополнительные параметры. Помимо уже знакомых нам идентифицирующих параметров `id` и `class`, используются параметры `style` и `align`. Параметр `style` применяется для установки стиля отображения содержимого блока, а параметр `align` позволяет устанавливать выравнивание данного блока относительно других элементов содержимого Web-страницы. Более подробно применение этих параметров мы рассмотрим в следующих разделах этой главы.

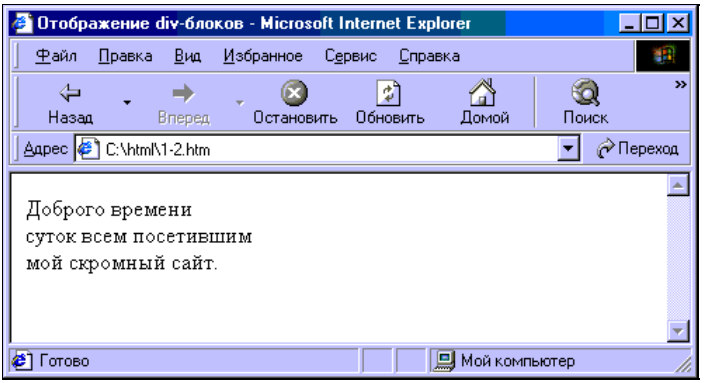


Рис. 1.2. Окно браузера с результатом отображения файла, приведенного в листинге 1.2

К структурным элементам HTML-документа можно отнести различные заголовки в тексте. Для заголовков в HTML существуют собственные тэги. Всего в HTML-документах применяется шесть уровней текстовых заголовков. Самый старший уровень — первый. Для каждого заголовка есть свой тэг и свои правила отображения.

Тэги для обозначения заголовков чрезвычайно просты. Для заголовка первого уровня применяется тэг `<h1>` с его закрывающим двойником `</h1>`, заголовок второго уровня описывается парой `<h2>` — `</h2>`, и т. д., вплоть до заголовка шестого уровня с тэгом `<h6>`. Ниже, в листинге 1.3 приведен пример использования заголовков в HTML-документе.

Листинг 1.3

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
↳"http://www.w3.org/TR/HTML4/strict.dtd">
<html>
  <head>
  <title> Отображение заголовков</title>
  </head>
  <body>
  <h1> Заголовок первого уровня </h1>
  <h2> Заголовок второго уровня </h2>
  <h3> Заголовок третьего уровня </h3>
  <h4> Заголовок четвертого уровня </h4>
  <h5> Заголовок пятого уровня </h5>
  <h6> Заголовок шестого уровня </h6>
  <p>Обычный текст</p>
  </body>
</html>
```

А как это все выглядит, хорошо видно на рис. 1.3.

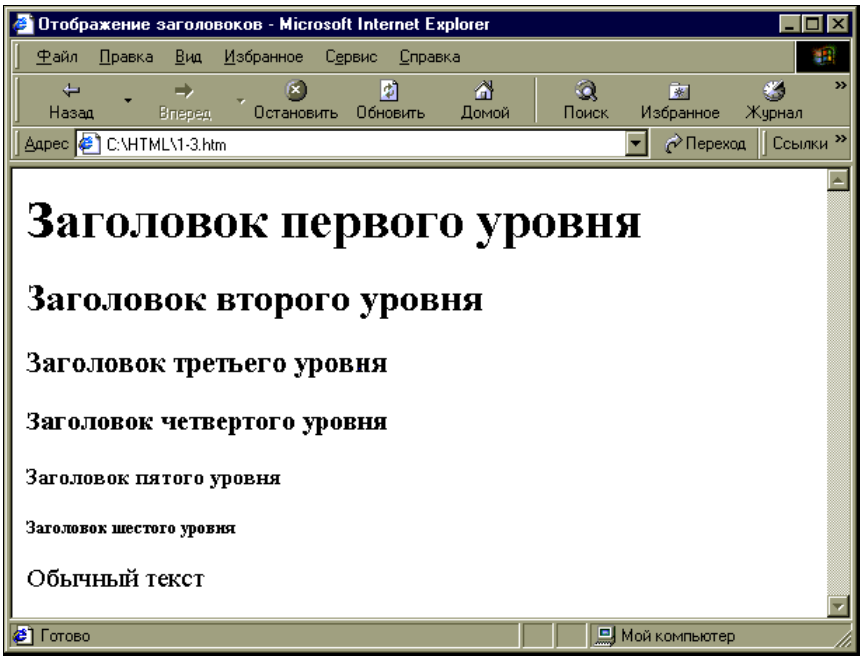


Рис. 1.3. Окно браузера с результатом отображения файла, приведенного в листинге 1.3

Тэги заголовков обладают тем же набором параметров, что и недавно рассмотренные тэги `<span>` и `<div>`, а именно идентификационные `id` и `class`, параметр общего оформления `style` и параметр выравнивания `align`.

На этом заканчивается рассмотрение структуры типичного HTML-документа. Как видно, никаких особых сложностей в этом нет. Все стройно и логично.

Используемые символы

Теперь обратимся к правилам использования символов в HTML. В компьютерах каждый символ — это некое число. Операционная система при отображении текста выводит символ, соответствующий какому-либо числу. Таблица соответствия чисел и символов называется *кодировкой*. Сейчас существует не менее пяти кодировок только для русскоязычных символов. В каждый браузер встроена функция смены кодировки отображаемой Web-страницы. Если браузер не распознает, какая кодировка использована при создании Web-страницы, то вместо текста пользователь увидит мешанину из

непонятных символов. Каждый, наверное, с этим встречался. Язык HTML позволяет указать используемую кодировку, чтобы браузер не пытался распознать ее самостоятельно. Для этого применяется уже знакомый нам тэг <meta>. Среди предопределенных переменных протокола HTTP есть переменная с именем Content-Type. Она задает тип содержимого Web-страницы и дополнительно позволяет указывать наименование применяемой кодировки. Полностью соответствующая конструкция выглядит так:

```
<META http-equiv="Content-Type" content="text/HTML;
↳charset=ISO-8858-5">
```

В приведенном примере значение переменной состоит из двух частей, разделенных знаком точки с запятой. Первая часть говорит о том, что данный документ — это обычный текст с тэгами HTML, а вторая часть указывает используемую кодировку. При этом применение слова charset является обязательным. После знака равенства приводится само название кодировки. В примере использована стандартная кодировка, утвержденная Международной организацией по стандартизации (ISO), с поддержкой кириллицы. Вместо нее можно применять стандартную кириллическую кодировку Windows или КОИ-8.

К сожалению, браузеры не могут обычным способом отображать некоторые символы, встречающиеся в тексте. Если браузер в тексте встретит знак неравенства "меньше", то он интерпретирует его как открывающую скобку для тэга. А так как стандартного тэга за этим знаком не последует, то некоторая часть текста будет просто проигнорирована и не отображена. Более того, некоторые буквы из алфавитов языков на основе латиницы отсутствуют на клавиатуре, и их трудно вставить в текст содержимого Web-страницы. Выход был найден.

Вместо самих символов в текст подставляются последовательности, которые можно правильно интерпретировать. Знак неравенства "меньше", он же открывающая угловая скобка, заменяется последовательностью "<". Вся последовательность заключается в кавычки, начинается она со знака амперсанда, а заканчивается знаком точки с запятой. Подобные последовательности перекочевали в язык следующего поколения — XML (eXtensible Marckup Language), и получили наименование "entities", что в русскоязычной литературе переводят как "сущности". Перевод, конечно, правильный, но, к сожалению, ничуть не разъясняет сути дела. Проще и, наверное, правильней называть эти сущности символьными подстановками. Чаще всего используют четыре символьные подстановки, которые приведены в табл. 1.1.

Таблица 1.1. *Символьные подстановки*

<	Знак неравенства "меньше" "<"
>	Знак неравенства "больше" ">"

Таблица 1.1 *(окончание)*

`&`	Знак амперсанда "&"
`&quot;`	Знак кавычек

Этими четырьмя сочетаниями список символьных подстановок не заканчивается. Оставшаяся часть списка приведена в *приложении 1.*

Подстановки бывают не только символьные. Мы можем с помощью подстановки поместить в текст любой символ из текущей кодировки, если нам известен его числовой код. Для этого применяется конструкция "`&#числовой_код;`". В подобном формате численных подстановок указывают десятичную запись числового кода символа. Если используется шестнадцатеричная запись кода, то подстановка принимает следующий вид: "`&#xчисловой_код;`". После знака решетки добавляется латинский символ "икс".

Цвета и единицы измерения

В коде HTML-документа нам всегда придется указывать размеры тех или иных объектов оформления Web-страницы, а также цветовые свойства этих объектов. В HTML предусмотрены стандартные правила для обозначения цветов и единиц измерения. Начнем мы с цветовых обозначений.

Существует два способа задания цвета. Чаще всего используется способ с указанием RGB-кода требуемого цвета. Как известно, любой цвет разлагается на три основных: красный, зеленый и синий. Браузеры позволяют нам отображать более шестнадцати миллионов цветов. Такое многообразие обеспечивается за счет того, что доля каждого из трех основных цветов может меняться от нуля до двухсот пятидесяти пяти. Любой цвет задается сочетанием трех чисел, каждое из которых определяет долю одного из трех основных цветов. Для удобства обработки в HTML цвет задается в виде группы из шести шестнадцатеричных цифр в следующей форме:

```
color="#FF0000"
```

Перед последовательностью шестнадцатеричных цифр ставится знак решетки. Порядок чисел, указывающих доли основных цветов, должен строго соблюдаться. Сначала красный, затем зеленый, и в самом конце — синий. Легко догадаться, что в примере мы установили красный цвет.

Есть и альтернативный вариант установки цвета. Для шестнадцати наиболее популярных цветов были установлены символьные обозначения, приведенные в табл. 1.2.

Таблица 1.2. *Цветовые обозначения*

Цвет	Шестнадцатеричный код	Буквенное обозначение
Черный	#000000	Black
Серебряный	#C0C0C0	Silver
Серый	#808080	Gray
Белый	#FFFFFF	White
Темно-красный	#800000	Maroon
Красный	#FF0000	Red
Пурпурный	#800080	Purple
Малиновый	#FF00FF	Fuchsia
Зеленый	#008000	Green
Ярко-зеленый	#00FF00	Lime
Оливковый	#808000	Olive
Желтый	#FFFF00	Yellow
Темно-синий	#000080	Navy
Голубой	#0000FF	Blue
Темная морская волна	#008080	Teal
Морская волна	#00FFFF	Aqua

С учетом этих обозначений, наш пример с красным цветом мы могли бы написать таким образом:

```
color="Red"
```

Теперь переходим к рассмотрению единиц измерения длины. Согласно спецификации языка HTML, мы можем указывать размеры каких-либо объектов оформления Web-страниц только двумя способами. Либо задать их размер в пикселах, либо в процентах от размера "родительского" объекта, внутри которого находится описываемый элемент оформления. Если мы на Web-странице размещаем таблицу и указываем, что ее ширина должна составлять пятьдесят процентов, то имеется в виду, что это пятьдесят процентов от ширины окна просмотра браузера. Ширина ячейки таблицы будет рассчитываться в процентах от общей ширины таблицы, в которой находится эта ячейка. Если пользователь изменит размеры окна браузера, соответствующим образом изменится и компоновка содержимого Web-страницы. При создании Web-страницы всегда следует изыскивать такие способы размещения содержимого, чтобы их компоновка не менялась кардинально при изменении размеров окна браузера. Кроме того, следует учитывать, что мони-

торы удаленных пользователей имеют различные разрешения. О том, как определить, какое именно разрешение монитора установлено у посетителя Web-страницы, мы узнаем в третьей главе, а пока вернемся к единицам измерения размеров объектов Web-страниц.

Для того чтобы указать размер некоего элемента в пикселах, достаточно в качестве значения соответствующего параметра задать необходимое число. Если мы захотим установить ширину некоего объекта равной тридцати пикселам, следует использовать следующую конструкцию:

```
width="30"
```

Если ширина должна составлять тридцать процентов от "родительского" объекта, то мы запишем такой код:

```
width="30%"
```

Еще раз обращаю внимание на то, что все значения параметров обрамляются двойными кавычками.

Помимо приведенных двух способов указания размеров, есть и еще один. Это нечто среднее между процентным соотношением и прямым определением размера в пикселах. Мы можем указать размер, кратный некоторому количеству пикселов. Например, есть у нас таблица, состоящая из трех строк. Высота таблицы задана, причем, неважно, каким способом. Если мы хотим, чтобы высота каждой строки была кратна тридцати пикселам, то в каждый тэг, создающий одну из этих строк, мы должны внести следующий фрагмент кода:

```
height="3*"
```

Признаком задания кратного размера служит символ "звездочка" (*). При расчете коэффициента кратности число, стоящее слева от звездочки, увеличивается в десять раз. Браузер же попытается отобразить такие объекты с максимально возможным размером. Если таблица имеет высоту сто восемьдесят пикселов, то высота каждой строки будет шестьдесят пикселов. То же самое случится, если мы создадим таблицу высотой в двести пикселов. Двадцать лишних пикселов просто пропадут. Если необходимо, чтобы наши строки заняли равную высоту, следует использовать следующий вариант задания параметра:

```
height="*"
```

Такой способ задания размера действует по умолчанию. Если для группы одинаковых объектов размеры не указаны, они размещаются максимально равномерно в пространстве "родительского" объекта.

Это все, что мы можем сказать о единицах измерения, принятых в HTML. На самом деле, мы имеем возможность задавать абсолютные размеры элементов оформления Web-страниц не только в пикселах. Но для этого необходимо использовать возможности технологии стилевого оформления CSS,

которые мы рассмотрим в следующей главе. Теперь пора переходить к рассмотрению возможностей отображения текста.

Оформление текста

Как известно, основное наполнение Web-страниц — это текст за редким исключением специальных сайтов — галерей. Неудивительно, что в HTML введено столько различных средств управления отображением текста.

Для того чтобы в окне просмотра браузера отобразить текстовую строку, никаких тэгов применять не требуется. Достаточно написать текст. Но когда нужно разбить его хотя бы на абзацы, тут уже без тэгов не обойтись. В различных компьютерных системах используются разные символы для разбиения текста на абзацы, а HTML-документы должны отображаться по возможности одинаково на любых компьютерных платформах, поэтому и пришлось ввести тэги, обозначающие абзацы.

В начале каждого абзаца ставится тэг `<p>`, а в конце — закрывающий тэг `</p>`. При этом тэг обладает некоторыми параметрами. В их число входят уже известные нам общие параметры идентификации `class` и `id`, параметр стилевого оформления `style`, который нам предстоит рассмотреть во второй главе, и параметр выравнивания `align`. О последнем параметре нам следует поговорить несколько поподробнее.

В HTML термин "выравнивание" означает как горизонтальное, так и вертикальное позиционирование элемента. Когда речь идет об абзацах текста, имеет смысл говорить только о горизонтальном выравнивании, или, как его еще называют, "выключке".

Выключка позволяет прижимать абзац к левому или правому краю окна просмотра браузера, центрировать его или растягивать слова таким образом, чтобы текст равномерно занимал всю ширину отведенного ему места. Для этих целей используются значения `left`, `right`, `center` и `justify`, соответственно. Рассмотрим их использование на примере еще одного HTML-документа.

Листинг 1.4

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
↳"http://www.w3.org/TR/HTML4/strict.dtd">
<html>
  <head>
  <title> Горизонтальное выравнивание абзацев</title>
  </head>
  <body>
  <p align="left">Абзац, прижатый к левому краю</p>
```

```
  <p align="right">Абзац, прижатый к правому краю</p>
  <p align="center">Центрированный абзац</p>
  <p align="justify">Абзац, растянутый по ширине</p>
  </body>
</html>
```

Результат просмотра файла с таким кодом браузером Internet Explorer показан на рис. 1.4. Кстати, для отображения этого файла применялась одна из устаревших версий браузера Internet Explorer, которая не обрабатывала растяжение абзаца по ширине поля просмотра. В подобных случаях, браузеры просто игнорируют неизвестные тэги или значения параметров и используют стандартный вариант отображения.

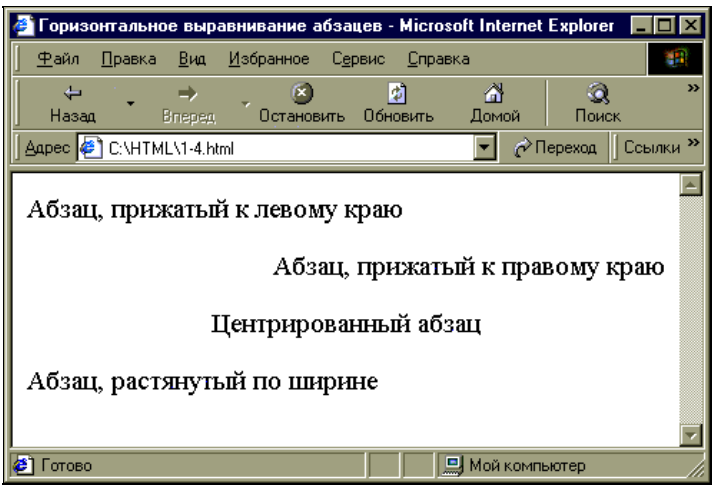


Рис. 1.4. Окно браузера с результатом отображения файла, приведенного в листинге 1.4

Иногда для того, чтобы увеличить расстояние между абзацами, создатели Web-страниц пытаются использовать пустые абзацы, т. е. абзацы, у которых между начальным и конечным тэгом ничего нет. Согласно спецификациям, браузеры должны игнорировать подобные конструкции, поэтому для разделения абзацев или принудительного обрыва строки внутри одного абзаца следует применять тэг <br>. Это директивный тэг. Он обозначает то место, где надо перенести текст на другую строку. Пример использования этого тэга для достижения обеих целей приведен в листинге 1.5.

Листинг 1.5

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
↳"http://www.w3.org/TR/HTML4/strict.dtd">
```

```
<html>
  <head>
  <title> Принудительный разрыв строк</title>
  </head>
  <body>
  <p>Абзац, который мы <br>принудительно разорвали</p>
  <p>Следующий абзац</p>
  <br>
  <p>Абзац после принудительного разрыва</p>
  </body>
</html>
```

Как выглядит этот файл при просмотре его с помощью браузера, показано на рис. 1.5.

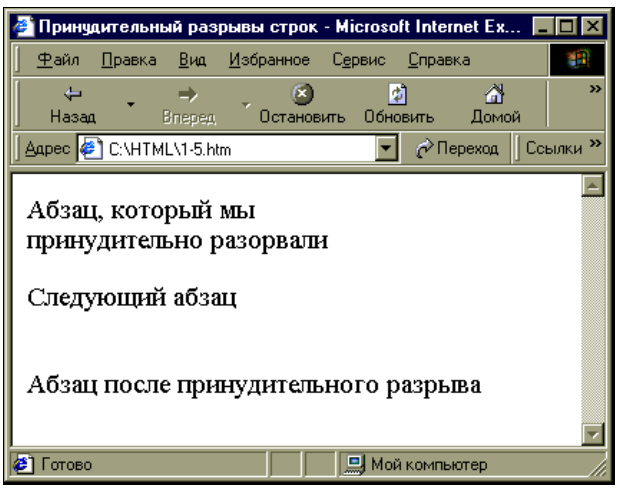


Рис. 1.5. Окно браузера с результатом отображения файла, приведенного в листинге 1.5

Тэг `<br>` помимо обычных параметров идентификации обладает еще параметром `clear`, который применяется для более точного выравнивания текста, когда тот обтекает какой-либо объект, например графическое изображение, внедренное в состав Web-страницы. В качестве значения этого параметра может использоваться одно из четырех предопределенных ключевых слов: `none`, `left`, `right`, `all`. Значение `none` применяется по умолчанию, и указывает, что следующая строка начнется с того места, где она начиналась бы и без использования данного параметра. Значение `left` говорит о том, что следующая строка начнется у левого поля объекта, обтекаемого текстом. Если же необходимо использовать правое поле для этих целей, то следует задать значение `right`. Значение `all` указывает браузеру, что воспользоваться

можно как левым, так и правым полем объекта, лишь бы текст был максимально компактно отображен.

Теперь перейдем к рассмотрению шрифтового оформления текста. В любом месте абзаца мы можем использовать тэг `<font>` с набором параметров, которые и будут определять внешний вид шрифта, применяемого для отображения текста, находящегося после этого тэга. Прекращение действия тэга `<font>` задается тэгом `</font>`.

Тэг `<font>` обладает следующими, присущими именно ему параметрами: `size`, использующийся для указания размера применяемого шрифта, `color` — для установки цвета символов шрифта, и `face`, указывающий, какой именно шрифт будет применяться для отображения текста.

Как мы только что говорили, параметр `size` применяется для указания размера символов используемого шрифта. Значениями этого параметра могут быть числа от единицы до семи. Они обозначают некий относительный размер символов. Дело в том, что в HTML нельзя установить абсолютный размер символов в пунктах, как мы это привыкли делать в обычных офисных приложениях. Пользователь будет просматривать Web-страницу на своем компьютере, и нам заранее неизвестно, какие шрифты могут быть у него установлены и какие их размеры доступны. Мы можем лишь указать относительный размер шрифта, а браузер пользователя сам подберет максимально подходящий размер.

В качестве значения параметра `size` мы можем задать изменение размера шрифта. Например, для того, чтобы увеличить размер шрифта на один уровень, следует использовать такую конструкцию:

```
<font size="+1">
```

Для уменьшения размера символов на два уровня применяется следующий код:

```
<font size="-2">
```

Для использования подобных конструкций необходимо отталкиваться от некоего базового размера. Для установки такого размера применяется тэг `<basefont>` с все тем же параметром `size`. Если же этот тэг не использовать, базовым размером символов будет третий уровень. Приведем пример использования этих тэгов в листинге 1.6.

Листинг 1.6

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
↳"http://www.w3.org/TR/HTML4/strict.dtd">
<html>
  <head>
  <title> Размер символов</title>
  </head>
```

```
  <body>
  <p><font size="7">Седьмой размер</font></p>
  <p><font size="6">Шестой размер</font></p>
  <p><font size="5">Пятый размер</font></p>
  <p><font size="4">Четвертый размер</font></p>
  <p><font size="3">Третий размер</font></p>
  <p><font size="2">Второй размер</font></p>
  <p><font size="1">Первый размер</font></p>
  <p><basefont size=2><font size="+2">Смещение размера</font></p>
  </body>
</html>
```

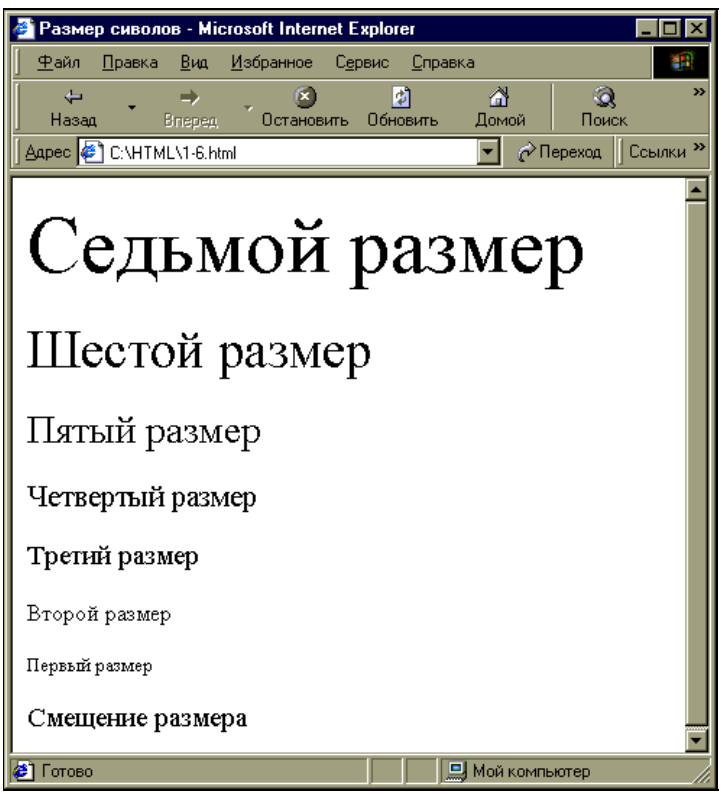


Рис. 1.6. Окно браузера с результатом отображения файла, приведенного в листинге 1.6

Мы рассмотрели применение только одного атрибута тэга `<font>`. На очереди — процедура установки цвета символов применяемого шрифта. Мы уже знаем, что для этого предназначен параметр `color`. О том, как именно записывать обозначение того или иного цвета, говорилось в предыдущем разде-