Популярный самоучитель

Александр Чиртик

HTML

2-е издание

Эта книга поможет вам:

- узнать, как устроены веб-страницы
- разобраться в HTML — языке, на котором написано почти все содержимое Интернета
- создавать полнофункциональные и привлекательные веб-страницы
- использовать современные технологии CSS и DHTML

ПИТЕР

Александр Анатольевич Чиртик

HTML: Популярный самоучитель

Текст предоставлен издательством
http://www.litres.ru/pages/biblio_book/?art=183576
HTML: Популярный самоучитель: Питер; Санкт-Петербург; 2008
ISBN 978-5-91180-937-9

Аннотация

В книге кратко и просто описывается язык HTML. Прочитав ее, вы научитесь создавать собственные веб-страницы, причем не только простые, но и содержащие таблицы, видео и звук. Более гибко оформить веб-страницы вам поможет рассмотренная в книге технология CSS. А при желании вы сможете сделать веб-страницы динамичными с помощью сценариев JavaScript: описание этого языка вместе с кратким описанием DOM (объектной модели документа) также приведено в этой книге. В последних главах рассматривается пример создания небольшого сайта с использованием всех рассмотренных в книге технологий, а также освещаются основные вопросы публикации сайта в сети Интернет. Приведенные в книге коды можно найти на сайте www.piter.com.

Содержание

Александр Анатольевич Чиртик HTML: Популярный самоучитель

Введение

Путешествуя по просторам Всемирной паутины, можно обратить внимание, как различаются между собой оформление и возможности веб-страниц. Однако не каждый пользователь знает, что почти все это разнообразие реализовано на основе одного средства разметки текста – HTML.

Язык разметки гипертекста HTML – стандартное средство представления информации в среде World Wide Web (WWW) в виде веб-страниц. Поскольку HTML является стандартизированным языком разметки, документы, написанные с его использованием, можно легко просматривать и редактировать на компьютерах с различным программным и аппаратным обеспечением.

Существует большое количество графических сред, предназначенных для быстрой и качественной разработки веб-сайтов. Тем не менее одной из замечательных особенностей

HTML является то, что полноценные веб-страницы можно создавать, имея под рукой лишь простейший текстовый редактор. Создание веб-страниц с использованием HTML и CSS или JavaScript – уже не простое форматирование текста, а интересное занятие, очень похоже на программирование.

Большая часть представленной вашему вниманию книги посвящается именно HTML. В девяти первых главах последовательно изложен материал, в котором рассмотрены практически все возможности «чистого» языка HTML. После изучения этого материала вы научитесь создавать не только простые страницы, но и достаточно сложные сайты, а также познакомитесь со средствами HTML, позволяющими без привлечения дополнительных средств и технологий (например, CSS, DHTML) создавать привлекательный и функциональный интерфейс своих веб-страниц. Последние главы книги посвящены двум более новым технологиям, дополняющим HTML и позволяющим разнообразить и «оживить» статичные по своей природе HTML-документы, – CSS (каскадные таблицы стилей) и DHTML (динамический HTML) с основами DOM (объектной модели документа). В книге также приведен небольшой пример создания сайта целиком и освещены основные вопросы публикации созданного сайта в Интернете.

Большинство примеров, приведенных в книге, достаточно просты, чтобы можно было легко понять, какие именно их части относятся к рассматриваемой теме. При желании тек-

сты примеров можно дополнить самостоятельно, тем более что экспериментирование – это наилучший способ узнать и запомнить как можно больше полезной информации.

Тексты программ на сайте издательства

Тексты программ (листинги), приведенные в книге, вы можете скачать с сайта издательства по адресу http://www.piter.com/download.

От издательства

Ваши замечания, предложения, вопросы отправляйте по следующему адресу электронной почты: gurski@minsk.piter.com (издательство «Питер», компьютерная редакция).

На сайте издательства http://www.piter.com вы найдете подробную информацию о наших книгах.

Глава 1
Введение в HTML

Итак, почему же HTML так широко распространен и используется для публикации информации в сети Интернет? Своей популярностью HTML во многом обязан удобству навигации между документами и простоте создания самих HTML-документов. Так, нажимая кнопку мыши над участком текста или изображением, можно открывать документы, расположенные в различных частях света на компьютерах с различными операционными системами.

Сеть из связанных между собой гипертекстовых документов является прозрачной для пользователя. Поэтому при работе с ней вас совершенно не заботят операции, которые осуществляются сотнями устройств глобальной компьютерной сети только для того, чтобы доставить на ваш компьютер содержимое HTML-документа.

1.1. Краткая история HTML

Началось все для HTML (а вместе с ним и для WWW) в конце 1980-х годов, когда у ученых из Европейской лаборатории элементарных частиц (CERN) возникла необходимость обмениваться множеством различных документов при помощи быстро развивающейся сети Интернет. Тогда необходимо было придумать способ публикации документов в сети таким образом, чтобы к любому нужному документу можно было легко получить доступ. К тому же нужно было, чтобы документы отображались на всех компьютерах одинаковым образом.

Решение поставленной задачи было найдено сотрудником CERN Бернерс-Ли. В 1989 году Тим Бернерс-Ли создал новый язык форматирования документов. В его основу был положен другой ранее созданный язык – SGML, который предусматривал установку связей между документами с помощью гиперссылок. Новый язык разметки был назван HTML (HyperText Markup Language). Этот же человек реализовал и первую программу для просмотра HTML-документов – браузер.

Затея с гипертекстом очень быстро прижилась. Вскоре в Интернете выросла большая сеть гипертекстовых документов, которую постепенно стали называть World Wide Web. К тому же в 1993 году команда программистов, руководи-

телем которой являлся основатель компании Netscape Марк Андриессен, разработала первый браузер, имеющий полноценный графический интерфейс и позволяющий работать с мышью, – Mosaic. Это не могло не добавить популярности быстро развивающейся Сети.

Со временем Интернет стал востребован не только учеными: к нему пришли и рядовые пользователи, причем их число неуклонно возрастало. Для многих было очевидно, что HTML, не предполагающий никакой динамики в создаваемых с его помощью документах, достаточно скучен и невзрачен. Это дало толчок развитию технологий CSS, введению поддержки апплетов Java, а после и сценариев (первым языком был JavaScript).

Нельзя не отметить, что с образованием и ростом Сети создателей HTML не на шутку взволновала «чистота» своего детища. Существовали небезосновательные опасения, что производители браузеров, которые скоро должны были прийти на рынок, будут «баловать» своих пользователей фирменными нововведениями, которые, естественно, будут поддерживаться только фирменными браузерами. В 1994 году была создана организация, взявшая на себя разработку единых стандартов развития WWW – World Wide Web Consortium (W3C). Эта организация и занялась подготовкой стандартов HTML (начиная с HTML 2.0). Правда, несмотря на наличие W3C, нововведения в HTML начинали поддерживаться производителями браузеров гораздо раньше, чем

эта организация их стандартизировала (так, например, обстояло дело с фреймами, с внедрением сценариев в HTML-документы, с объектной моделью документов и т. д.).

Организация W3C существует и сейчас. Она занимается теми же вопросами стандартизации, однако уже давно не HTML (последняя спецификация HTML 4.01 была принята 24 декабря 1999 года). Развитие языка HTML сегодня практически завершено, а основные усилия W3C сконцентрированы на работе над более новыми технологиями, например расширяемым языком разметки XML, языком преобразований XSLT и др. (если интересно, можно заглянуть на www.w3.org – официальный сайт этой организации).

Язык HTML в том виде, в котором он существует сейчас, обладает большим потенциалом представления информации. Причем он рассчитан не только на пользователей персональных компьютеров. Документы HTML можно просматривать на очень большом количестве различных по своим возможностям устройств: от черно-белого экрана мобильного телефона до телетайпа или терминала. Кроме того, в последние версии HTML включен ряд возможностей, облегчающих представление документов неграфическими средствами (например, речевыми синтезаторами), позволяющими пользоваться услугами WWW людям с физическими недостатками.

1.2. Базовые понятия HTML

После маленького исторического экскурса можно наконец-то перейти к практической части. Перед подробным рассмотрением HTML нужно ознакомиться с основными понятиями, используемыми в книге, а также с базовыми элементами языка HTML.

Для начала следует разобраться с тем, что такое HTML-документ. HTML-документ – это обычный текстовый документ, созданный в любом текстовом редакторе, например в Блокноте, и оформленный в соответствии с правилами языка HTML. Для файлов, содержащих HTML-документы, используется расширение HTML или HTM (например, MyWedPage.html или MyWedPage.htm). Расширение HTM использовалось ранее для корректного отображения имен файлов в формате MS-DOS. На самом деле неважно, какое из приведенных расширений использовать.

Далее в книге HTML-документы могут называться просто документами, а также страницами и веб-страницами. Под сайтами в тексте книги подразумевается несколько документов, объединенных единой системой навигации.

Элементы

Элемент – это конструкция языка HTML, содержащая

данные. HTML включает в себя различные типы элементов, которые позволяют задавать абзацы, гипертекстовые ссылки, списки, таблицы, изображения и т. д. Конструкция <P>Привет !</P> представляет собой элемент. Обычно элемент можно разделить на три части. Первая часть – <P> – называется открывающим тегом (англ. tag). Далее идет содержание элемента, которое в данном случае состоит из слова Привет !. И наконец, </P> является закрывающим тегом. Как видно, название элемента (P) присутствует и в открывающем, и в закрывающем теге. Регистр символов в названии элемента не имеет значения. Однако в соответствии с соглашениями, принятыми большинством разработчиков, в примерах данной книги названия элементов записаны в верхнем регистре.

Открывающий и закрывающий теги нужны для указания начала и конца элемента. Теги всегда начинаются символом < и заканчиваются символом >. В закрывающем теге перед его именем помещается символ /. Для некоторых типов элементов допускается отсутствие закрывающего тега (например, элемент P, указывающий начало абзаца). Существуют также элементы, не имеющие закрывающего тега, то есть его не просто можно опустить, а он вообще не существует в языке.

Атрибуты

Элементы могут содержать параметры, называемые атрибутами. Атрибуты могут иметь определенные значения (по умолчанию или устанавливаемые авторами). Пара атрибут/значение указывается в начальном теге элемента перед символом >, например:

<BODY bgcolor = «#FF0000»>

Каждому атрибуту может быть присвоено значение определенного типа. В приведенном примере указание атрибута bgcolor (имеющего тип %Color) в элементе BODY приведет к тому, что цвет фона страницы станет красным. Значения атрибутов заключаются в кавычки, хотя в определенных случаях кавычки необязательны.

В начальном теге элемента может быть указано любое количество допустимых пар атрибут/значение, разделенных пробелами, например:

<BODY bgcolor = «#FF0000» text = «#0000FF»>

В приведенном примере устанавливаются красный цвет фона страницы и синий цвет основного текста. При установке значений нескольких атрибутов порядок их записи не имеет значения. Важно отметить, что регистр, в котором записываются названия атрибутов, также не имеет значения. Однако для повышения читабельности HTML-кода назва-

ния атрибутов обычно записываются в нижнем регистре (как в приведенных выше и ниже примерах).

Существует любопытная разновидность атрибутов – булевы атрибуты. Для них возможны только два значения: ИСТИНА и ЛОЖЬ. По умолчанию эти атрибуты имеют значение ЛОЖЬ. Чтобы присвоить им истинное значение, достаточно просто указать имя этого атрибута, не присваивая ему никакого значения. Ниже приведен пример, в котором устанавливаются истинные значения двух атрибутов HTML-элемента INPUT:

```
<INPUT readonly disabled>
```

Вложенные элементы

Важным моментом HTML является возможность использования вложенных элементов. Элемент, находящийся внутри другого элемента, называется вложенным. Пример использования вложенных элементов для задания начертания шрифта:

```
<I>Курсив<B>-Полужирный курсив-</B>Курсив</
I>
```

При обработке приведенного HTML-кода получится страница, показанная на рис. 1.1 (как и чем обрабатываются HTML-документы, будет рассказано далее).

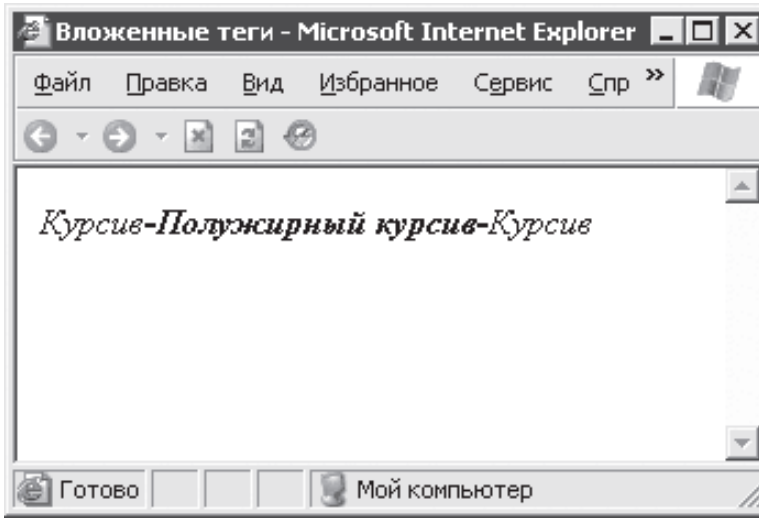

Рис. 1.1. Применение вложенных элементов

На приведенном рисунке видно, как действие внешнего (или родительского) элемента I (задание курсива) дополняет действие внутреннего элемента B (задание полужирного начертания шрифта).

При использовании вложенности следует помнить, что вложенные элементы должны закрываться до того, как будут закрыты внешние элементы. Так, следующий пример является неверным:

```
<I>Неправильное закрытие<B> внутреннего элемента</I>
до закрытия внешнего</B>
```

Блочные и встроенные элементы

Различают также блочные и встроенные элементы (иногда их называют элементами уровня блока и элементами уровня текста). Основным отличием блочных элементов является форматирование их браузером как обособленной части документа. Блочные элементы задаются парными тегами и могут содержать вложенные блочные или встроенные элементы и, естественно, текст.

Встроенные элементы обычно находятся прямо в тексте и могут иметь содержимое или не иметь его. Примерами встроенных элементов могут служить приведенные ранее элементы B и I, а также элементы перевода строки, изображения и т. д. В отличие от блочных элементов, встроенные элементы могут содержать только текст или вложенные встроенные элементы.

1.3. Просмотр HTML-документа

Сам по себе HTML-документ практически нечитабелен для обычного пользователя. Для чего же тогда применяется форматирование документа с использованием HTML? Для просмотра HTML-документов используются специальные программы – браузеры. При открытии HTML-документа браузер распознает теги и учитывает их при отображении текста. Если по каким-то причинам (например, при ошибке в записи тега) тег не распознается браузером, то он игнорируется.

Существует большое количество программ, позволяющих просматривать HTML-документы. Это такие распространенные приложения, как Internet Explorer, Opera, Firefox, Netscape Navigator.

Ниже приведен пример простого HTML-документа (назначение используемых элементов будет рассмотрено далее в книге) (пример 1.1).

Пример 1.1. HTML-документ

```
<HTML>
<TITLE>Простой HTML-документ</TITLE>
<BODY>
<H1>Заголовок</H1>
```

```
Текст страницы
</BODY>
</HTML>
```

Этот HTML-документ отображается браузером Internet Explorer так, как показано на рис. 1.2.

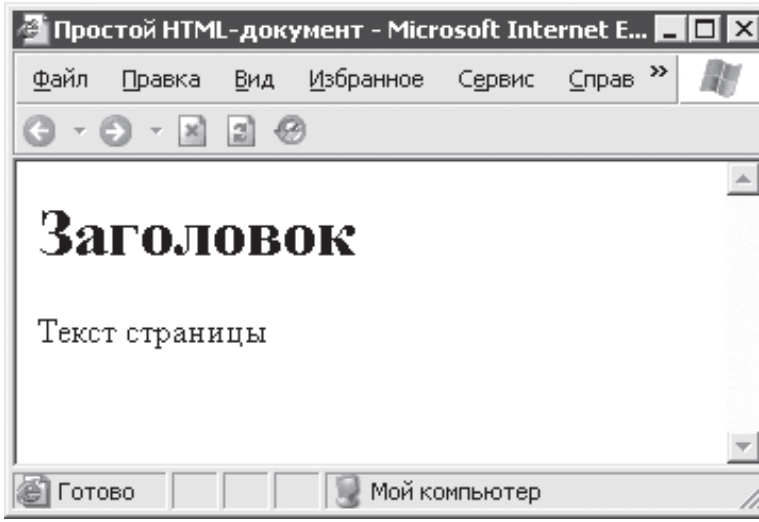

Рис. 1.2. Отображение HTML-документа

1.4. Универсальный идентификатор ресурса URI

Чтобы полностью понимать, как происходит взаимодействие HTML-документов, переход между страницами и откуда вообще компьютер пользователя получает данные при работе с сетью, нужно рассмотреть, как и к чему осуществляется доступ при помощи Глобальной сети.

Многие виды ресурсов, размещенных в Интернете, независимо от того, являются ли они HTML-документами, рисунками или файлами архива, чаще всего представляют собой файлы на жестком диске компьютера (сервера), подключенного к сети. С каждым ресурсом сопоставляется значение, по которому можно однозначно определить его расположение, – универсальный идентификатор ресурса или URI (Universal Resource Identifier). URI широко используются как при самостоятельном доступе пользователя к ресурсу (когда, например, пользователь сам вводит URI в адресной строке браузера), так и при переходе между веб-страницами. URI также используются в HTML-документе для указания браузеру, где искать ресурсы (например, рисунки), используемые в самом документе.

Примечание

В литературе также часто применяется обозначение URL. Следует отметить, что URI является более общим

понятием, включающим в себя URL: любой URL является универсальным идентификатором ресурса и подчиняется тем же правилам, что и URI.

Идентификатор ресурса URI состоит из трех частей: из наименования механизма доступа к ресурсу, доменного имени компьютера и пути файла ресурса. Для пояснения сказанного можно рассмотреть пример:

http://www.somesite.com/info/examples/ex_1.html

Здесь можно увидеть, что для доступа к ресурсу, которым в данном случае является HTML-документ, используется протокол HTTP (Hyper Text Transfer Protocol). Ресурс хранится на компьютере, имеющем доменное имя somesite.com в файле ex_1.html, расположенном в папке /info/examples.

При помощи URI можно также ссылаться на части HTML-документов, например:

http://www.somesite.com/info/examples/ex_1.html#description

При использовании этого URI можно получить доступ к части HTML-документа, имеющей имя description (то, как создавать имена для фрагментов HTML-документов, будет рассмотрено в гл. 5).

URI также позволяют ссылаться на ресурсы в пределах одного компьютера. При этом указывается относительный путь ресурса. Например, чтобы из HTML-документа, расположенного в папке /info/examples, сослаться на файл /info/files/

file1.jpg, достаточно задать URI /files/file1.jpg. В HTML-документах при помощи подобных ссылок указываются пути рисунков и других объектов, используемых в документах, но непосредственно не хранимых в них.

В общем случае URI считаются нечувствительными к регистру символов. Однако для полной уверенности в правильности интерпретации URI все же обращайте внимание на регистр символов в URI гиперссылок, рисунков и т. д. Это полезно для устранения таких ситуаций, когда, например, при работе сайта на компьютере под Windows все гиперссылки работают, а при помещении сайта на UNIX-сервер работать отказываются (в UNIX имена файлов чувствительны к регистру).

Глава 2
Структура HTML-документа

В идеальном случае HTML-документ состоит из трех частей, в которых описывается следующая информация:

- данные о версии используемого HTML;
- заголовок документа;
- тело документа.

Выражение «в идеальном случае» означает то, что один или несколько элементов могут пропускаться: если HTML-документ содержит хоть какой-то текст, пусть без информации о версии, без заголовка и без явного указания тела документа, то браузер все равно отобразит информацию, содержащуюся в этом документе, при этом применяя к тексту еще и форматирование. Правда, в этом случае заведомо неизвестно, насколько исказится содержимое документа.

Итак, полноценный (полный, стандартный) HTML-документ должен содержать все три указанные элемента структуры или хотя бы два последних элемента. Далее приводится пример простейшего HTML-документа, содержащего все указанные структурные элементы (пример 2.1).

Пример 2.1. Задание структуры HTML-документа

```
<!DOCTYPE HTML PUBLIC «-//W3C//DTD HTML 4.01//EN»>
<HTML>
<HEAD>
<TITLE>HTML-документ</TITLE>
</HEAD>
<BODY>
<H1>Заголовок</H1>
<P>Первый абзац
<P>Второй абзац
</BODY>
</HTML>
```

2.1. Информация о версии HTML

Первая строка HTML-документа содержит информацию об используемой версии языка HTML: 2.0, 3.0, 3.2, 4.0 и 4.01 (используется в данной книге). Здесь же задается, какое определение типа документа (DTD) должен использовать браузер при интерпретации содержимого документа. Для указания версии при использовании HTML 4.01 можно использовать одно из следующих определений типа документа:

• <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"> – использовать строгое определение HTML версии 4.01, в которое не включаются нежелательные для версии 4.01 элементы и атрибуты, а также фреймы;

• <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional //EN"> – применять переходное определение HTML версии 4.01, в которое включаются нежелательные для версии 4.01 элементы и атрибуты;

• <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"> – использовать определение HTML версии 4.01, в которое включаются нежелательные для версии 4.01 элементы и атрибуты, а также фреймы.

В информации о версии языка также можно указать URI, откуда браузер может загрузить последнюю версию файла с DTD-определением используемой версии HTML. Для трех

указанных выше вариантов определений URI следующие (в том же порядке):

- http://www.w3.org/TR/html4/strict.dtd;
- http://www.w3.org/TR/html4/loose.dtd;
- http://www.w3.org/TR/html4/frameset.dtd.

Пример использования одного из приведенных URI:

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

Информация о версии HTML, безусловно, должна использоваться при профессиональной разработке HTML-документов. Далее будет рассказано еще об одном способе задания версии HTML, который является нежелательным с точки зрения использования чистого языка HTML 4.01, но все же имеет место. В большинстве примеров данной книги версия HTML вообще не указывается для того, чтобы не загромождать HTML-код строками, не относящимися к рассматриваемым примерам.

Однако при создании документов, предназначенных для публикации в Интернете, следует обязательно позаботиться о включении определения версии HTML в документ, особенно если в нем используются сценарии (будут рассмотрены в гл. 12 и 13).

2.2. Элемент HTML

Корневым элементом структуры HTML-документа является одноименный элемент HTML. Его использование позволяет явно указать браузеру, что им обрабатывается HTML-код. Элемент HTML содержит в себе все остальные структурные части HTML-документа, например HEAD и BODY, и задается при помощи парных тегов <HTML> и </HTML>. Все, что находится между этими тегами, – есть HTML-документ.

Теги <HTML> и </HTML> являются необязательными. В открывающем теге можно задать используемую версию HTML при помощи атрибута version. Например, при использовании HTML 4.01 можно написать так:

<HTML version = «4.01»>

Это и есть второй вариант задания версии HTML. Правда, при использовании объявления типа документа задание версии в элементе HTML является излишним. На текущем этапе изучения HTML не стоит особо волноваться о том, правильно ли вы указываете версию HTML, ведь HTML-документы нормально отображаются большинством современных браузеров без всякого указания версии HTML, также как и без указания атрибутов, которые все-таки необходимо рассмотреть.

В элементе HTML можно также задать основной язык документа (атрибут lang) и направления текста (атрибут dir). Атрибут dir может принимать одно из двух значений: RTL или LTR (задают направление текста справа налево или слева направо соответственно). Для указания языка в атрибуте lang используется сокращенное стандартное обозначение языка, например: "ru", "en", "de" и т. д. Ниже приведены примеры задания языка и направления текста:

```
<HTML lang = «ru» dir = LTR>
<HTML lang = "en">
```

Задавать атрибуты lang и dir совершенно не обязательно. Они доступны для большинства HTML-элементов (не поддерживаются только для элементов APPLET, BASE, BASEFONT, BR, FRAME, FRAMESET, IFRAME, PARAM, SCRIPT).

Теперь, наконец, можно рассмотреть, как задаются наиболее важные элементы HTML-документа – заголовок и тело документа.

2.3. Заголовок

В заголовке (его еще называют «шапкой») HTML-документа содержатся сведения о документе: название (тема документа), ключевые слова (используются поисковыми системами), а также ряд других данных, которые не являются содержимым документа.

Элемент HEAD

Заголовок HTML-документа содержится в элементе HEAD. Для задания этого элемента используются парные теги <HEAD> и </HEAD>. Все, что находится между двумя указанными тегами, относится к заголовку HTML-документа. Теги <HEAD> и </HEAD> не являются обязательными, но все же лучше их указывать, чтобы можно было легко определить, что именно относится к заголовку HTML-документа (браузер определит это автоматически, независимо от наличия тегов <HEAD> и </HEAD>). В простых примерах книги элемент HEAD опускается, если заголовок документа содержит только элемент TITLE (информация об этом элементе приведена далее).

Рассматриваемые далее элементы TITLE и META являются информативными элементами заголовка HTML-документа. Они определяются внутри элемента HEAD (между

его тегами).

```
<HEAD>
<TITLE>…</TITLE>
<META …>
…
<META …>
</HEAD>
```

Элемент TITLE

В самом простом случае в заголовке документа содержится только один элемент – TITLE. Он используется для задания названия HTML-документа. Этот элемент задается при помощи парных тегов <TITLE> и </TITLE>, причем только один раз. Текст, находящийся между приведенными тегами, воспринимается и отображается браузерами (и поисковыми системами) как название документа. К тексту названия нельзя (бессмысленно) применять форматирование.

Название документа должно быть кратким, но информативным и должно адекватно отражать содержание документа, например:

```
<TITLE>Программирование на Java. Введение</
TITLE>
<TITLE>Операторы языка C++</TITLE>
```

Использование тегов <TITLE> и </TITLE> является обязательным. Правда, браузер (например, Internet Explorer) об-

работает документ и без этих тегов. Но ведь при публикации документов в Интернете документы надо как-то обозначить, чтобы пользователь знал, имеет ли документ отношение к нужной ему теме.

Элемент BASE

Элемент BASE, появляющийся в заголовке HTML-документа, позволяет задать базовый URI, относительно которого разрешаются все относительные URI в документе.

Элемент BASE задается одиночным тегом <BASE>. Его атрибуту href присваивается URI, который будет считаться базовым для HTML-документа, например:

```
<BASE href = «D:\test\test.html»>
…
<IMG src = "image.gif">
```

При таком использовании элемента BASE, где бы ни был расположен открытый HTML-документ, рисунок, задаваемый элементом IMG, будет загружаться из файла D:\test \image.gif.

Метаданные

В заголовке документа помещаются также некоторые важные данные, используемые браузерами и поисковыми системами, но в большинстве случаев не отображаемые, – мета-

данные. Задание метаданных представлено как задание значений переменным (иногда говорят про задание значений свойствам), имеющим определенные имена, осуществляемое с помощью HTML-элемента META (элемент задается одиночным тегом <META>). При использовании элемента META обычно задают значения следующих атрибутов:

• name – имя переменной (значения чувствительны к регистру символов, по крайней мере, официально);

• content – значение переменной;

• lang – код языка, для которого действительно значение переменной; может быть задано несколько значений одной переменной для разных языков;

• http-equiv – применяется для указания браузеру дополнительных параметров по обработке HTML-документа, значением атрибута является название параметра (в отличие от name, значения не чувствительны к регистру символов).

Перечисленные атрибуты обычно используются парами: name и content или http-equiv и content. Атрибут lang применяется, если дополнительно нужно указать значения атрибутов для различных языков.

Сначала будет рассмотрено использование пары name и content. Эта пара атрибутов позволяет задавать значения переменным, которые часто, но не всегда используются поисковыми системами и другими сервисами Интернета: описание и авторство документа, ключевые слова и еще много информации. В табл. 2.1 приведены основные значения атри-

бута name, а также дана расшифровка возможных значений атрибута content в каждом случае.

Таблица 2.1. Значения атрибута name и соответствующие значения атрибута content

Значение name	Описание значения content
Author	Информация об авторе (произвольный текст)
Copyright	Информация об авторских правах на документ (в произвольной форме)
Description	Обычно краткая аннотация содержимого HTML-документа
Keywords	Список ключевых слов. Может использоваться поисковыми системами

На практике полезным оказывается задание переменной, содержащей список ключевых слов. Ведь именно этот список воспринимается поисковыми системами при поиске по запросу пользователя. По этой причине перед публикацией ваших творений в Интернете позаботьтесь о том, чтобы список состоял из слов, наиболее часто используемых (или очевидных) при запросах на тему, соответствующую или близкую теме вашего документа.

При создании списка ключевых слов крайне удобно использование атрибута lang для указания различных наборов ключевых слов на различных языках (и для различных языков поиска). Например, как в следующем случае:

```
<META name = «Keywords» lang = «ru» content = «HTML, веб-дизайн, гипертекст, сайт, сайты …»>
<META name = "Keywords" lang = "en" content = "HTML, web-design, hypertext, site, sites …>
```

Можно, конечно, задать список одним единственным тегом <META> сразу на всех нужных языках. Как удобнее – решать вам.

Теперь пришло время рассмотреть использование второй пары атрибутов: http-equiv и content. Эта пара значений, если говорить упрощенно, позволяет влиять на работу браузера. Кроме того, она может использоваться поисковыми системами. В табл. 2.2 приведены некоторые распространенные значения атрибута http-equiv и описание соответствующих им значений атрибута content.

Таблица 2.2. Значения атрибута http-equiv и соответствующие значения атрибута content

Значение http-equiv	Значение и действие content
Expires	Задает дату и время, по истечении которых документ считается устаревшим (например: Mon, 2005 Jan 24 15:23:00 GMT). Используется для указания браузеру на необходимость обновить документ (загрузить свежую версию копии) по прошествии указанных даты и времени. Имеются в виду браузеры, кэширующие страницы (сохраняющие их на диск после загрузки с сервера, чтобы впоследствии предлагать их пользователю, даже не обращаясь к серверу)
Content-Language	Позволяет указать язык HTML-документа (аналогично атрибуту lang элемента HTML)
Content-Type	Указывает тип документа (обычно text/html). Кроме того, позволяет указать используемую кодировку символов, например: content="text/html; charset=windows-1251". Если у вас установлен Internet Explorer, то список некоторых кодировок можно посмотреть в меню Вид ▸ Кодировка
Refresh	Задает интервал в секундах, по истечении которого браузер должен обновлять содержимое документа. Можно также задать URI, с которого нужно загрузить документ при обновлении (например, content="0; url=index.html"). Последний вариант применяется для перенаправления пользователя на другие документы

Напоследок остается только заметить, что использовать все приведенные значения атрибутов name и http-equiv необязательно. В большинстве случаев достаточно бывает ограничиться заданием ключевых слов и кодировки HTML-

документа.

2.4. Тело HTML-документа

Вся содержательная часть HTML-документа находится в его теле (элемент BODY). Для определения этого элемента используются парные теги <BODY> и </BODY>. Теги <BODY> и </BODY> не являются обязательными, но их наличие, как и в случае тегов <HEAD> и </HEAD>, значительно улучшает наглядность структурной организации HTML-документа и позволяет четко отделить содержимое документа от заголовка.

Все, что помещено между тегами <BODY> и </BODY>, является содержимым документа, показываемым браузером пользователю. В простейшем случае это может быть просто текст без всякого дополнительного оформления. Ниже приведен список наиболее часто используемых атрибутов элемента BODY:

- background – URI, указывающий расположение изображения для фона (обычно берется небольшое изображение, которое размножается для заполнения фона всего документа);
- bgcolor – цвет фона HTML-документа;
- text – цвет шрифта документа;
- link – цвет непосещенных гиперссылок;
- vlink – цвет посещенных гиперссылок;
- alink – цвет гиперссылок при выборе их пользователем

(при нажатии Enter произойдет переход по такой гиперссылке).

Все атрибуты, позволяющие задавать цвет (не только элемента BODY, но прочих элементов, которые будут рассмотрены далее), имеют тип %Color. Значения таких атрибутов могут задаваться шестнадцатеричными числами с символом # в начале каждого числа, например:

bgcolor = «#FF0005»

При задании цвета данным способом следует помнить, что числом задается цвет в RGB-формате. Это значит, что первые два символа задают интенсивность красного цвета от 0 до FF (255 в десятичной системе счисления), третий и четвертый символы – интенсивность зеленого цвета, а два последних – интенсивность синего цвета. В данном примере интенсивности красного, зеленого и синего цветов равны FF, 0 и 5 соответственно.

Кроме того, атрибутам задания цвета можно присваивать предопределенные идентификаторы некоторых наиболее часто употребляемых цветов. Список этих названий и их численные значения приведены в табл. 2.3.

Таблица 2.3. Идентификаторы и значения часто используемых цветов

Название	Значение	Цвет
black	#000000	Черный
silver	#808080	Темно-серый
gray	#C0C0C0	Серый
white	#FFFFFF	Белый
maroon	#800000	Бордовый
red	#FF0000	Красный

Название	Значение	Цвет
purple	#800080	Фиолетовый
fuchsia	#FF00FF	Лиловый
green	#00F000	Зеленый
lime	#00FF00	Ярко-зеленый
olive	#808000	Оливковый
yellow	#FFFF00	Желтый
navy	#000080	Темно-синий
blue	#0000FF	Синий
teal	#008080	Сине-зеленый
aqua	#00FFFF	Бирюзовый

Для закрепления всего, что было описано в этой главе, можно рассмотреть простой пример.

Данный пример базируется на уже полученных знаниях (на теги задания гиперссылок и теги <P> можете пока не обращать внимания, так как важен только цвет гиперссылок и текста) (пример 2.2).

Пример 2.2. Пример задания названия документа и параметров цвета

```
<HTML>
<TITLE>Пример задания цветов в элементе BODY</TITLE>
<BODY
background = "2.2.html-files/back.jpg"
text = "black"
link = "#0080FF"
vlink = "blue"
alink = "navy">
Обычный неформатированный текст должен отображаться черным цветом
<P><A HREF = "ref1">Непосещенная гиперссылка (голубой цвет)</A>
<P><A HREF = "ref2">Посещенная гиперссылка (синий цвет)</A>
<P><A HREF = "ref3">Выделенная гиперссылка (темно-синий цвет)</A>
</BODY>
</HTML>
```

После обработки приведенного примера браузером получится документ, который показан на рис. 2.1.

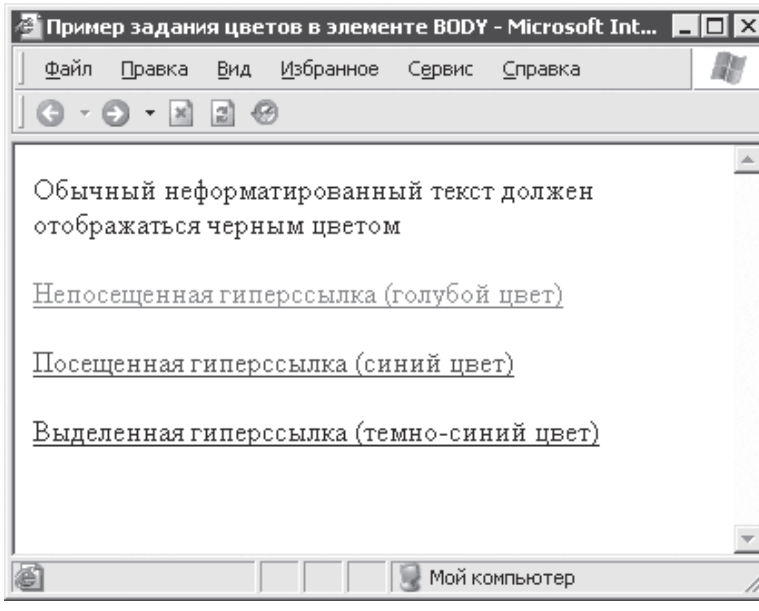

Рис. 2.1. Результат обработки HTML-текста примера

В данном примере использовался неформатированный текст. Но HTML на то и HTML, чтобы всячески способствовать улучшению восприятия содержимого текста и наделять обычный текст возможностями навигации. В последующих главах будут подробно рассмотрены форматирование текста, вставка в него иллюстраций и прочие замечательные возможности HTML.

Глава 3
Текст

Как вы могли заметить, чтобы поместить простой текст на страницу, достаточно ввести его в теле документа. При этом браузер отобразит текст, используя шрифт и цвет, заданные по умолчанию для текста тела страницы.

Чтобы чтение информации, содержащейся в HTML-документе, стало приятным занятием, применяется форматирование и разбиение документа на логически цельные части (структурирование) с визуальным отделением этих частей друг от друга. Далее будут рассмотрены основные возможности HTML, позволяющие сделать содержимое документа легко читаемым и воспринимаемым. Однако перед описанием форматирования и структурирования текста нужно рассмотреть особенности, которые необходимо учитывать при добавлении непечатных и зарезервированных для языка HTML символов в текст документа.

3.1. Особенности ввода текста

При вводе текста в документ часто возникает вопрос: как заставить браузер отобразить зарезервированные символы языка HTML (например, > или &) или символы, которые невозможно ввести с клавиатуры. Для ввода в документ таких символов в HTML предусмотрен механизм ссылок на символы. Таким образом, когда необходимо ввести в документ, например, символ &, то в текст на его место подставляется специальная последовательность – ссылка на данный символ.

Ссылки на символы могут быть представлены любым из указанных ниже способов:

• &#D; – позволяет задать символ, код которого имеет значение D (в десятичной системе счисления);

• &#xH; – позволяет задать символ, код которого имеет значение H (в шестнадцатеричной системе счисления);

• &имя_символа; – позволяет использовать именованную ссылку на символ.

Как можно увидеть, ссылка на символ в тексте HTML-документа начинается символом & и заканчивается точкой с запятой (;). Особо стоит рассмотреть использование именованных ссылок на символы. Дело в том, что использование первых двух вариантов предполагает, что автору известны численные коды символов, которые он собирается добавить

в документ. Однако согласитесь, что постоянно искать в таблицах численные коды нужных символов по меньшей мере неудобно. По этой причине в HTML предусмотрены имена для наиболее часто используемых символов. Эти имена и записываются вместо кода в тексте ссылок на символы. Имена некоторых часто используемых символов приведены в табл. 3.1. Полный список именованных ссылок на символы приведен в приложении 1.

Таблица 3.1. Имена символов

Имя символа	Описание символа
quot	Кавычка
amp	Амперсанд (&)
lt	Меньше (<)
gt	Больше (>)
tilde	Тильда (~)
nbsp	Неразрывный пробел
reg	Знак зарегистрированной торговой марки (®)
copy	Знак авторского права (©)
alpha	Малая греческая буква альфа (α)
pi	Малая греческая буква пи (π)

В качестве примера использования ссылок на символы в HTML-документе можно рассмотреть следующий HTML-код (пример 3.1).

Пример 3.1. Использование ссылок на символы

```
<TITLE>Именованные ссылки на символы</
TITLE>
<BODY text = "yellow" bgcolor = "blue">
&lt;TITLE&gt; Именованные ссылки на символы
&lt;/
TITLE&gt;<BR>
&lt;BODY color = &quot; yellow &quot;
bgcolor = &quot; yellow &quot; &gt; <BR>
Текст HTML-документа <BR>
&lt;/BODY&gt;
</BODY>
```

При обработке данного кода браузером Internet Explorer получится документ, который показан на рис. 3.1.

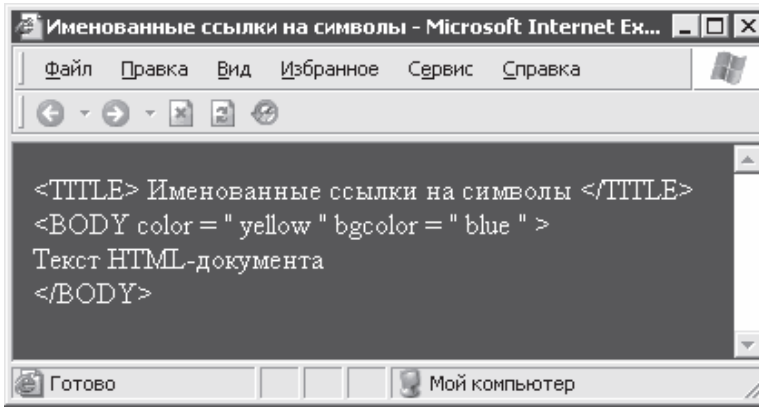

Рис. 3.1. Использование ссылок на символы

Конечно, возможности использования ссылок на символы приведенным примером не ограничены. Ссылки на символы очень часто применяются для записи математических формул. Кроме того, их можно использовать для записи текста на иностранном языке, но с использованием символов другого языка (например, для ввода символа в при использовании символов только английского языка).

3.2. Форматирование текста

Для изменения вида текста, отображаемого браузером, применяется форматирование с использованием специальных HTML-тегов. Форматирование текста HTML-документа сходно с форматированием в любом текстовом редакторе (например, Microsoft Word). Оно позволяет выделить цветом, начертанием, изменением шрифта некоторый текст, подчеркнуть его значимость или просто украсить.

Задание начертания текста

Задание начертания текста является, возможно, самым простым средством форматирования содержимого документа, которое доступно в HTML. Для изменения начертания текста в HTML-код добавляются элементы, приведенные в табл. 3.2.

Таблица 3.2. Элементы задания начертания текста

Элемент	Описание
B	Полужирное начертание текста
I	Курсивное начертание текста
U	Подчеркнутый текст
STRIKE, S	Перечеркнутый текст
BIG	Текст с увеличенным размером шрифта
SMALL	Текст с уменьшенным размером шрифта
SUB	Верхний индекс
SUP	Нижний индекс
TT	Текст, записанный моноширинным шрифтом (все символы имеют одинаковую ширину)
BLINK	Мерцающий текст (редко поддерживается браузерами)

Для наглядности можно рассмотреть пример HTML-документа, в котором используются различные элементы задания начертания текста (пример 3.2).

Пример 3.2. Задание начертания текста

```
<TITLE>Задание начертания текста</TITLE>
<BODY>
<B>Полужирный текст</B><BR>
<I>Курсив</I><BR>
<U>Подчеркнутый текст</U><BR>
<S>Зачеркнутый текст</S><BR>
<BIG>Текст увеличенного размера</BIG><BR>
<SMALL>Текст уменьшенного размера</
SMALL><BR>
```

```
<SUP>Верхний индекс</SUP>Текст<SUB>Нижний индекс</SUB><BR>
<TT>Текст, записанный моноширинным шрифтом</TT><BR>
</BODY>
```

В приведенном коде задействованы все элементы задания начертания текста из табл. 3.2, кроме BLINK.

При обработке HTML-кода браузером получится документ, показанный на рис. 3.2.

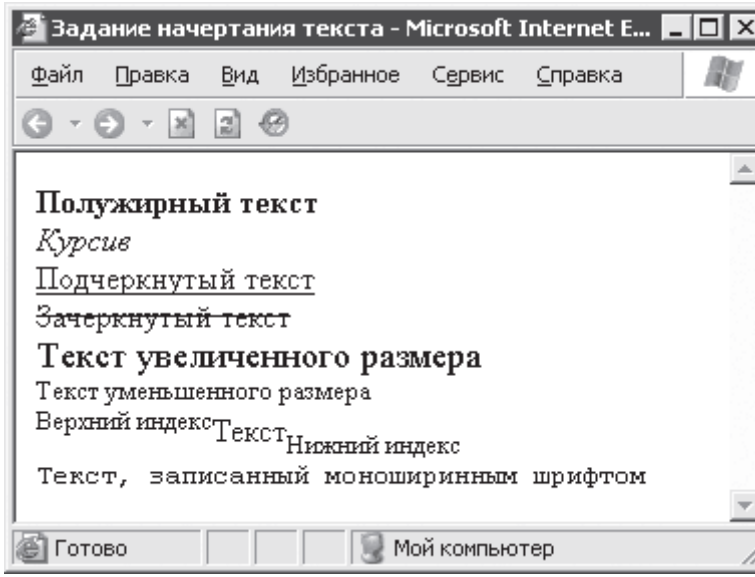

Рис. 3.2. Задание начертания текста

Рассматриваемые HTML-элементы могут быть вложены

друг в друга. При этом на начертание текста влияют все элементы, внутри которых находится текст. Например, чтобы одновременно зачеркнуть и подчеркнуть курсивный полужирный текст, можно использовать следующий код:

Конец ознакомительного фрагмента.