Для создания веб-страниц часто используется язык гипертекстовой разметки HTML. Конечно, каждый сайт индивидуален, но существуют общие правила построения HTML-страниц. Им обязательно нужно следовать — только в этом случае ваш HTML-код будет верно распознаваться и отображаться браузером.

Язык HTML имеет долгую историю развития, и в настоящее время наиболее широко используется версия HTML 4.01, на смену которой приходят версии XHTML и HTML 5. Во всех этих версиях применяется одинаковый принцип гипертекстовой разметки документов, заключающийся в использовании специальных конструкций языка, называемых *тегами*, и задании их параметров, называемых *атрибутами*.

Версии HTML 4, XHTML и HTML 5 различаются набором используемых тегов и атрибутов, а также некоторыми требованиями к синтаксическим конструкциям на их основе. Например, язык HTML 4 допускает использование строчных и прописных букв в названиях тегов и атрибутов — вы можете писать их, не обращая внимания на регистр букв. Однако в языках XHTML и HTML 5 это не так — теги и атрибуты должны быть введены в нижнем регистре. Есть и другие различия принципиального характера, и мы обсудим их далее по мере знакомства с версиями языка HTML.

Сейчас же подробно рассмотрим структуру простого документа HTML 4, чтобы вы могли понять общие принципы форматирования HTML-документов, а далее обсудим различия между версиями языка HTML и что означает на практике наличие нескольких версий HTML.

ПРИМЕЧАНИЕ

Последнее очень важно, поскольку в Интернете существует огромное количество сайтов, подготовленных с помощью HTML разных версий, которые посещают люди, использующие устаревшие браузеры. И если вы хотите обеспечить своему сайту широкую аудиторию, следует с самого начала озаботиться вопросами его совместимости и корректного воспроизведения как можно большим количеством пользователей Интернета.

1.1. Общие понятия HTML

Итак, как только что было сказано, независимо от версии языка HTML, используемой при создании сайта, существуют общие правила форматирования HTML-документов и структурирования их содержимого. Их мы и рассмотрим в данном разделе.

Элемент

Основой языка HTML является *элемент*. Он несет в себе определенную информацию, может описывать документ в целом или способ форматирования текста. Эле-

менты можно назвать основой построения сайта. Все, что вы захотите создать на вашей странице, будет сделано с помощью элементов.

Название элемента помещается в угловые скобки: , полученное выражение называется тегом. В данном случае это открывающий тег. Иногда необходимо задать парный закрывающий тег, который записывается так: . В основном парные теги используются при форматировании текста, они задают начало и конец блока форматирования. В одних случаях закрывающий тег вообще не требуется, а в других его можно пропустить, однако для корректной обработки документа рекомендую всегда использовать закрывающий тег.

Кратко функции открывающего и закрывающего тегов можно описать так: открывающий тег включает форматирование, а закрывающий выключает. При этом основным различием в записи тегов, кроме постановки символа / в закрывающем теге, является отсутствие у последнего атрибутов.

Примером необходимости использования закрывающего тега является работа с элементом P, который обозначает абзац:

```
<p>Текст абзаца</p>
```

Однако и в данном случае закрывающий тег является необязательным, но желательным. Элемент IMG, который добавляет картинку на сайт, наоборот, не требует наличия закрывающего тега. По назначению элемента зачастую можно догадаться, требуется ли ему закрывающий тег.

Элементы используются для того, чтобы сказать браузеру, какой блок вы хотите видеть в определенном месте страницы и какую информацию этот блок должен содержать. Кроме того, браузеру нужно сообщить, как отображать данную информацию. Для этого используют атрибуты элементов.

Атрибут

Атрибуты позволяют указывать различные способы отображения информации, добавляемой с помощью одинаковых элементов, а в некоторых случаях использование элемента без атрибутов не дает результатов.

Например, в одном абзаце нужно выровнять текст по левому краю, а в другом — по правому. Чтобы задать выравнивание абзаца, используем атрибут align элемента P:

```
<p align="left">Выравнивание по левому краю</p>

<p align="right">Выравнивание по правому краю</p>
```

Значения атрибутов задаются после знака равенства в кавычках и могут быть разными. Некоторым атрибутам присущ набор фиксированных значений, для других же количество значений не ограничено.

Элементы и их атрибуты являются основой языка HTML, но для правильного отображения страницы в браузерах еще важно верно создать структуру документа. Для этого существуют строгие правила. Есть элементы, без которых HTML-документ не может обойтись, потому что именно они определяют его структуру.

1.2. Структура HTML-документа

Для создания структуры документа и хранения служебной информации в нем предусмотрено много элементов, которые охватывают все необходимые пункты построения документа. Приведу пример документа HTML 4, чтобы вы могли увидеть эти элементы воочию (листинг 1.1). Документы XHTML и HTML 5 будут иметь некоторые отличия от приведенного примера, но мы пока отложим их рассмотрение — сейчас важно понять общий принцип построения HTML-документов.

Листинг 1.1. Описание документа HTML 4 в элементе DOCTYPE

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Это листинг структуры документа HTML</title>
...Элементы заголовка...
</head>
<body>
...Тело документа...
</body>
</html>
```

Документ HTML 4 из листинга 1.1 состоит из следующих компонентов:

- строки объявления типа документа;
- декларативного заголовка;
- тела документа.

Объявление типа документа HTML 4

В начале каждого документа HTML 4.01 следует помещать строку объявления такого рода:

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

Эта строка предоставляет браузеру общую информацию об HTML-документе. Несмотря на то что вам вряд ли придется заполнять строку как-то иначе, рассмотрим ее по частям и выясним, какую информацию о документе несут данные из элемента `DOCTYPE`.

- `HTML` — показывает, что для создания документа используется язык HTML.
- `PUBLIC` — указывает на то, что при создании документа использована общепринятая версия HTML.
- `"-//W3C//DTD HTML 4.01//EN"` — задает публичное имя спецификации языка, используемого для разметки документа. Если браузер по этому имени сможет распознать, где находятся правила обработки данного документа, он воспользуется ими, иначе сможет загрузить их по ссылке в следующем атрибуте. В данном случае это язык HTML версии 4.01, общепринятой на момент написания книги.
- `"http://www.w3.org/TR/html4/strict.dtd"` — URL-адрес документа, содержащего *наборы определений типа документа*, подготовленного в соответствии с языком HTML 4.01.

Что такое набор определения типа документа (Document Type Definition, DTD), мы обсудим в главе 2, когда приступим к рассмотрению языка XHTML и его отличий от HTML 4. Сейчас же просто запомните, что это сведения, которые необходимы браузеру или другой программе, предназначенной для работы с данным документом HTML, для его правильной обработки. Для документов HTML 4.01 организация W3C (World Wide Web Consortium) определила три набора таких правил DTD.

- Набор *строгих правил DTD*, которые требуют, чтобы данный HTML-документ точно соответствовал всем требованиям спецификации HTML 4.01. Документы с этим набором правил содержат такое объявление:

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

- Набор *переходных (transitional) правил DTD*, которые допускают использование устаревших, не поддерживаемых в версии HTML 4.01 элементов и атрибутов. Документы с этим набором правил содержат такое объявление:

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

- Набор *правил DTD для документов HTML 4.01 с фреймами*. Что такое фреймы, вы, возможно, знаете: если веб-страница в окне браузера выглядит как набор нескольких окон со своими полосами прокрутки и прочими средствами просмотра, то это

и есть фреймовый HTML-документ. Документы такого типа должны содержать следующее объявление:

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

В объявлении `<!DOCTYPE` содержится общая служебная информация о данном документе HTML 4.01. Если вы, создавая документ HTML, пропустите его, браузер, скорее всего, сам сможет догадаться, как ему отобразить соответствующую страницу. Но организация W3C настоятельно рекомендует включать объявления `<!DOCTYPE` во все разрабатываемые документы.

Могут возникнуть вопросы: а зачем существуют строгие и переходные правила DTD и какие из них следует использовать? Ответ связан с историей развития Интернета в целом и языка HTML в частности. Отказ от устаревших средств языка и использование строгих DTD может привести к тому, что ваш документ HTML 4.01 не будет корректно воспроизводиться отдельными устаревшими программами, поэтому в настоящее «переходное» время безопаснее использовать переходный набор DTD.

После того как вы ввели общую информацию о странице, необходимо разобраться с ее структурой.

Элемент HTML

Корневым элементом HTML-документа является `HTML`. Это значит, что все остальные элементы содержатся внутри тегов `<html>` и `</html>`. Тем не менее в документах HTML 4.01 данный элемент не является обязательным, хотя W3C рекомендует снабжать им свои документы. В документах же XHTML и HTML 5 наличие тега `<html>` обязательно.

В элементе `HTML` могут применяться следующие атрибуты:

- `lang` — определяет язык документа;
- `dir` — задает направление чтения на языке документа: `RTL` — справа налево, `LTR` — слева направо;
- `version` — определяет версию стандарта HTML, использованного при составлении документа (это устаревший атрибут, и его применение не рекомендовано);
- `title` — определяет всплывающую подсказку для страницы.

В листинге 1.2 представлен пример использования элемента `HTML` вместе с атрибутами, указывающими на русский язык и направление чтения слева направо.

Листинг 1.2. Элемент HTML с атрибутами

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html lang="ru" dir="LTR" title="Моя страничка">
  <!--Содержимое документа-->
</html>
```

На рис. 1.1 представлен результат использования атрибутов элемента `HTML`.

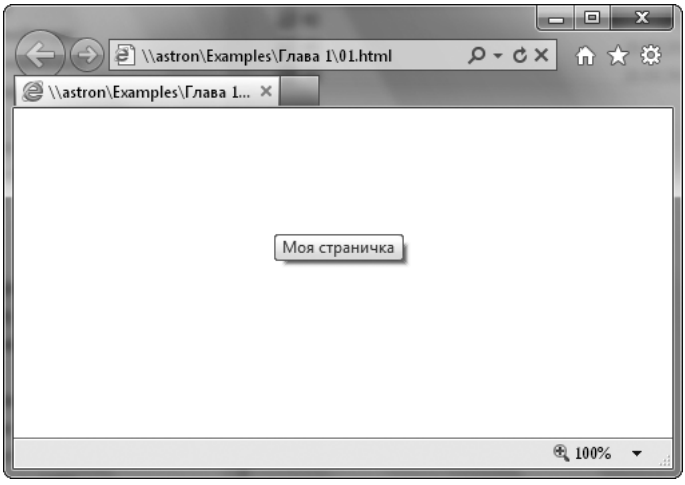

Рис. 1.1. Значение атрибута title отображено в виде экранной подсказки

Следует отметить, что атрибут `dir` в данном случае не обеспечен достаточной поддержкой известных браузеров даже последних версий, некоторые устаревшие браузеры просто игнорируют этот атрибут. Так что к его применению нужно относиться с осторожностью, если вы намерены использовать текст с альтернативным направлением чтения.

Элемент HEAD и декларативный заголовок

После того как для документа создан корневой тег `<html>`, внутри него нужно создать заглавную область документа, для чего используется элемент `HEAD`. Информация, вводимая в этот элемент, не отображается в окне браузера, а помогает ему в обработке страницы.

В заголовке должны присутствовать как открывающий, так и закрывающий теги `<head>` и `</head>`, между которыми и располагаются другие элементы, несущие служебную информацию о странице. Элементы, находящиеся внутри элемента

HEAD, играют очень важную роль: содержащиеся в них данные помогают браузеру в обработке страницы, а поисковым системам — в индексации документа.

Для элемента HEAD определены те же атрибуты, что и для HTML, плюс атрибут profile. Этот атрибут позволяет задать адрес файла с настройками, которые определяются элементами META внутри заголовка. С его помощью можно будет избежать многократной записи одних и тех же элементов META в свои документы. Однако описываемый атрибут пока не введен в действие и рассчитан на будущее развитие языка HTML.

Элементы, которые можно использовать внутри элемента HEAD, представлены в табл. 1.1.

Таблица 1.1. Элементы, используемые внутри элемента HEAD

Элемент	Описание
TITLE	Задает заголовок окна, отображающего документ, требует закрывающего тега. В заголовке он может быть только один, и его наличие обязательно
META	Определяет различную служебную информацию
SCRIPT	Позволяет добавлять сценарии
LINK	Задает ссылку на таблицы стилей (см. главы 8–10)
STYLE	Позволяет добавить стили для страницы (см. главы 8–10)
BASE	Задает базовый адрес документа
OBJECT	Определяет методы обработки нестандартного содержимого (см. главу 5)

Как видно из краткого описания, приведенного в таблице, элементы содержат данные, единые для всей страницы. Здесь элементы описаны кратко, дальше мы рассмотрим некоторые из них подробнее. Начнем с элемента, определяющего заголовок страницы.

Элемент TITLE

Данный элемент задает название веб-страницы, которое нужно придумывать, логически исходя из ее содержимого. Согласно спецификации HTML 4.01 в содержимом элемента HEAD обязательно наличие элемента TITLE, причем в единственном экземпляре.

Элемент требует наличия закрывающего тега </title>. Текст, содержащийся между открывающим и закрывающим тегами, будет отображаться в строке заголовка браузера.

Помимо основной функции — рассказать посетителю, о чем страница, — элемент выполняет ряд косвенных задач. Текст, помещенный в элемент TITLE, имеет огромное значение для решения задач продвижения сайта — именно по нему по-

исковые машины пытаются определить соответствие сайта поисковому запросу. Вам следует самым тщательным образом продумать это название, включив в него наиболее важные слова и фразы, по которым пользователи Интернета будут искать ресурсы, соответствующие вашему сайту. Но не стоит и раздувать содержимое тега `<title>`. Во-первых, текст не поместится в заголовке окна браузера, а во-вторых, поисковики сокращают объем просматриваемого текста названия (как правило, это 50–60 знаков).

По тексту заголовка пользователь получает дополнительную информацию о том, что это за сайт и как называется текущая страница. Не стоит думать, что достаточно в документе указать логотип сайта и проигнорировать заголовок. Посетитель может сворачивать окна, и тогда заголовки будут отображаться на кнопках **Панели задач** — по ним можно будет легко сориентироваться, с каким сайтом работать.

Большинство браузеров поддерживает возможность сохранения веб-страницы на компьютер. В этом случае имя сохраненного файла совпадает с названием заголовка документа. Если в тексте заголовка содержатся символы, недопустимые в имени файла (`\ / : * ? " < > |`), то они будут проигнорированы или заменены разрешенными.

При сохранении в разделе браузера **Избранное** в качестве названия ссылки будет использоваться текст, записанный в элементе `TITLE`. В этом случае адрес текущей страницы с ее заголовком помещается в список ссылок. Поскольку этот список, как правило, хранится в виде отдельных файлов, к их именам также применяется вышеописанное правило.

В листинге 1.3 показан пример использования элемента `TITLE`.

Листинг 1.3. Использование элемента TITLE

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <title>Пример создания заголовка страницы</title>
    <!--Другое содержимое заголовка-->
  </head>
  <!--Содержимое документа-->
</html>
```

На рис. 1.2 представлен результат работы листинга 1.3. Видно, что текст заголовка отображается на вкладке в окне браузера Internet Explorer 9.

Мы определились с заголовком новой страницы. Теперь посмотрим, какая служебная информация может содержаться внутри элемента `META`.

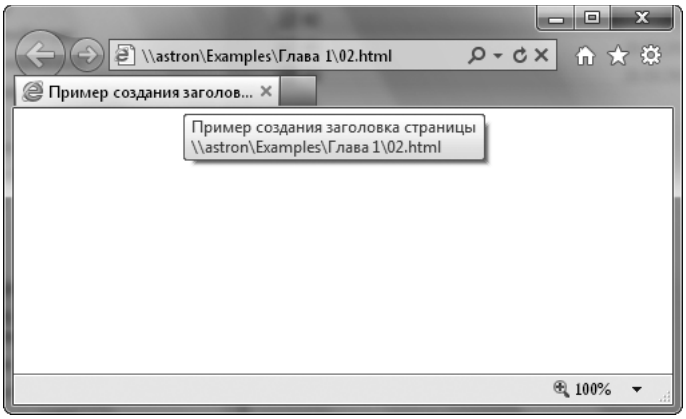

Рис. 1.2. Вид заголовка страницы

Элемент META

Данный элемент используется для хранения дополнительной информации о странице. Эту информацию браузеры применяют для обработки страницы, а поисковые системы — для ее индексации. Например, чтобы указать автора данного HTML-документа, можно использовать элемент `META` следующим образом:

```
<meta name="author" content="Вася Пупкин">
```

Здесь значение атрибута `name` задает имя свойства `author`, которому в атрибуте `content` присваивается имя автора `Вася Пупкин`. В этом и состоит общее правило применения элементов `META`: с их помощью вы задаете в атрибуте `name` имя нового свойства, которому далее в атрибуте `content` присваиваете значение. Вместо `name` может быть использован атрибут `http-equiv`, который служит для обмена служебной информацией браузера с веб-сервером. Например, рассмотрим такой элемент:

```
<meta http-equiv="expires" content="Sun, 1 Nov 2009 16:20:47
GMT">
```

Он просто сообщает браузеру, когда будет исчерпан срок хранения в кэше данной страницы, после чего следует выполнить повторный запрос сервера.

В элементе `HEAD` может быть несколько элементов `META`, потому что в зависимости от используемых атрибутов они могут нести разную информацию. В табл. 1.2 представлены возможные значения атрибута `http-equiv`. Заметьте, спецификация HTML 4.01 не определяет значения этого атрибута, поскольку они определяются протоколом обмена информацией с веб-сервером. Поэтому использовать элементы `META` с такими атрибутами рекомендуется только подготовленным специалистам.

Таблица 1.2. Возможные значения атрибута http-equiv

Значение атрибута	Описание
content-type	Определяет тип содержимого документа. Этот параметр желательно указывать всегда
expires	Задает время действия документа. После даты, указанной для этого свойства, документ будет считаться устаревшим
pragma	Определяет тип кэширования вашего документа, то есть можно запретить браузеру сохранение страницы в кэше
refresh	Позволяет задать параметры автоматической загрузки документа в то же окно браузера, в котором загружен текущий документ
content-language	Задает язык содержимого, аналог атрибута lang элемента HTML
content-script-type	Определяет типы сценариев, используемых на сайте
content-style-type	Задает типы таблиц стилей

Рассмотрим подробнее применение описанных выше атрибутов.

В листинге 1.4 приведен пример того, как с помощью атрибута `http-equiv` задать свойства обработки страницы.

Листинг 1.4. Применение атрибута http-equiv

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html title="Моя страничка" lang="ru">
  <head>
    <title>Использование элемента META</title>
    <meta http-equiv="content-type" content="text/html" charset="windows-1251" />
    <meta http-equiv="refresh" content="10" URL="http://test.ru" />
    <meta http-equiv="pragma" content="no-cache" />
    <meta http-equiv="expires" content="Sun, Oct 2010 10:09:00 GMT+3" />
  </head>
</html>
```

Как видно из примера, значение атрибута `http-equiv` указывает на переменную, значение которой определено с помощью атрибута `content`.

Значение `content-type`, использованное в примере, будет одинаковым для всех сайтов в кириллической кодировке. Рекомендую всегда указывать его, иначе браузер может некорректно отображать текст на вашей странице.

Использование параметра `refresh` полезно, когда страницу перенесли в другое место или когда у нее много адресов. В таком случае вместо создания множества

сайтов по разным адресам можно просто задать возможность переброски посетителя на основной сайт. В примере из листинга 1.4 через 10 секунд после загрузки страницы отобразится сайт **test.ru**.

СОВЕТ

Не все браузеры поддерживают автоматическую пересылку. На всякий случай на странице, с которой идет переадресация, нужно оставлять текстовую ссылку на страницу переадресации.

Рассмотрим возможные значения атрибута `content` для каждого из представленных свойств (табл. 1.3).

Таблица 1.3. Значения атрибута content для различных значений атрибута http-equiv

Атрибут http-equiv	Значения атрибута content	Описание
content-type	iso-8859-1	Latin-1
	windows-1251	Кириллица (Windows)
	koi8-r	Кириллица (КОИ8-Р)
	cp866	Кириллица (DOS)
	windows-1252	Западная Европа (Windows)
	windows-1250	Центральная Европа (Windows)
expires	Sun, Oct 2010 10:09:00 GMT+3	Дата должна указываться в стандарте [RFC850] на английском языке
pragma	no-cache	Не кэшировать. Этот атрибут можно использовать, если страница получается в результате работы сценария, поскольку тогда нет смысла оставлять ее в кэше
refresh	Время в секундах	Время до автоматической перезагрузки страницы
	Время в секундах, URL	Время до автоматической перезагрузки страницы и адрес страницы, которая по истечении этого срока должна быть загружена в текущее окно
content-language	de	Немецкий
	el	Греческий
	en	Английский
	en-GB	Английский британский
	en-US	Английский американский
	en-cockney	Английский кокни
	es	Испанский
	fr	Французский
	it	Итальянский

Атрибут http-equiv	Значения атрибута content	Описание
	i-navajo	Навахо
	ja	Японский
	he	Иврит
	nl	Голландский
	ru	Русский
	pt	Португальский
	zh	Китайский
content-script-type	text/javascript	JavaScript
	text/perlscript	PerlScript
	text/tcl	TCL
	text/vbscript	VBScript
content-style-type	text/css	Язык таблиц стилей CSS

Большинство значений атрибута `content`, которые вам могут пригодиться, представлены в приведенной таблице. Поначалу вы вообще можете ограничиться использованием свойства `content-type` (оно обязательно), а остальные параметры будете включать при необходимости.

Атрибут `name`, как и `http-equiv`, содержит служебную информацию о документе, однако в нем записывается информация другого плана: например, данные об авторе и содержимом документа. Эти данные не влияют на обработку документа браузером, однако предоставляют информацию для поисковых систем.

В табл. 1.4 представлены возможные значения атрибута `name`.

Таблица 1.4. Возможные значения атрибута name

Значение атрибута	Описание
keyword	Задает ключевые слова, используемые на странице. По этим словам поисковые системы осуществляют поиск
description	Описывает содержимое документа
author	Задает автора документа
document-state	Определяет необходимость переиндексации страницы
resource-type	Задает тип ресурса. Возможные типы мы рассмотрим далее, однако нужно учитывать, что, если тип документа будет отличным от document, поисковые системы не будут его индексировать
revisit	Определяет период переиндексации документа в днях
robots	Содержит указания для роботов и поисковых машин
subject	Задает тематику документа для поисковых машин
URL	Пересылает робота индексации на другую страницу (например, если у нас несколько одинаковых сайтов с разными адресами, а хочется, чтобы был проиндексирован только главный)