НОВАЯ

БОЛЬШАЯ КНИГА CSS

Дэвид Макфарланд

O'REILLY®

ПИТЕР®

CSS

David Sawyer McFarland

O'REILLY®
Beijing | Cambridge | Farnham | Köln | Sebastopol | Tokyo

НОВАЯ

БОЛЬШАЯ КНИГА

CSS

Дэвид Макфарланд

Санкт-Петербург • Москва • Екатеринбург • Воронеж
Нижний Новгород • Ростов-на-Дону
Самара • Минск

2016

ББК 32.988.02-018
УДК 004.738.8
М17

Макфарланд Д.

М17 Новая большая книга CSS. — СПб.: Питер, 2016. — 720 с.: ил. — (Серия «Бестселлеры O'Reilly»).

ISBN 978-5-496-02080-0

Технология CSS3 позволяет создавать профессионально оформленные сайты, но тонкости этого языка могут оказаться довольно сложными даже для опытных веб-разработчиков. Полностью переработанное четвертое издание этой книги поможет вам поднять навыки работы с HTML и CSS на новый уровень; оно содержит множество ценных советов, описаний приемов, а также инструкции, написанные в стиле справочного руководства. Веб-дизайнеры, как начинающие, так и опытные, при помощи этой книги быстро научатся создавать красивые веб-страницы, которые молниеносно загружаются как на ПК, так и на мобильные устройства.

12+ (В соответствии с Федеральным законом от 29 декабря 2010 г. № 436-ФЗ.)

ББК 32.988.02-018
УДК 004.738.8

ISBN 978-1491918050 англ.
ISBN 978-5-496-02080-0

Краткое содержание

Часть III. Верстка страниц с помощью CSS

Часть IV. Профессиональные приемы CSS-верстки

Часть V. Приложения

Оглавление

Часть I. Основы CSS

Часть II. Применение CSS

Часть IV. Профессиональные приемы CSS-верстки

Часть V. Приложения

Об авторе

Дэвид Макфарланд (David McFarland) — веб-разработчик, преподаватель и автор. Создает сайты с 1995 года: именно тогда он разработал свой первый проект — онлайн-журнал для специалистов в области коммуникаций.

Дэвид преподавал веб-дизайн в Высшей школе журналистики в Беркли, Центре электронного искусства (Electronic Art) и Государственном университете Портленда. В настоящий момент он главный преподаватель на сайте онлайн-образования Treehouse (teamtreehouse.com).

О творческой команде

Нэн Барбер (Nan Barber) — редактор серии книг Missing Manuals («Исчерпывающее руководство»). Живет в Массачусетсе вместе со своим мужем. Ее электронный адрес: nanbarber@oreilly.com.

Мелани Ярброух (Melanie Yarbrough) — литературный редактор и композитор. Проживает и трудится в Кембридже, штат Массачусетс. Увлекается выпечкой и любит совершать прогулки на велосипеде вокруг города. Ее электронный адрес: myarbrough@oreilly.com.

Молли Ивс Бровер (Molly Ives Brower) — внештатный редактор и корректор. Любит Интернет с тех пор, как в 1990 году получила адрес BITNET. В наше время ее можно найти на сайте mjibrower.com или в социальной сети Twitter, где она известна под ником @vintagereader. Ее электронный адрес: molly@mjibrower.com.

Рон Штраус (Ron Strauss) — составитель алфавитного указателя. Специализируется на составлении указателей для различных книг, посвященных информационным технологиям. Рон также талантливый скрипач. Он живет в Северной Калифорнии со своей женой и по совместительству коллегой Энни и карликовым пинчером Кенгой. Адрес электронной почты: rstrauss@mchsi.com.

Рич Костер (Rich Koster) — бета-ридер. Купил свой первый компьютер Mac (17-дюймовый MacBook Pro) в 2009 году и больше никогда не переходил на сторону пользователей Windows. В третьем издании книги «iPhone. Исчерпывающее руководство» Рич выступил техническим редактором. Он женат, воспитывает детей и в свободное время поддерживает собственноручно созданный сайт Disney Echo (DisneyEcho.emuck.com).

Благодарности

Большое спасибо всем, кто помогал в работе над этой книгой, включая моих студентов, которые всегда оценивали технические издания с позиции новичков. Благодарю своих технических редакторов, Дэниел Куинн (Daniel Quinn) и Дженнифер Дэвис (Jennifer Davis), которые предостерегли меня от досадных ошибок. Кроме того, мы все в долгу перед веб-дизайнерами, которые стали новаторами, используя CSS с творческим подходом, и поделились своими наработками в сообществе, посвященном веб-дизайну.

Спасибо Дэвиду Поугу (David Pogue), который много лет назад помог мне начать это длинное приключение. Спасибо Нэн Барбер за приведение в порядок моих рукописей и прочую помощь в написании этой книги.

Дэвид Сойер Макфарланд

Введение

Каскадные таблицы стилей, или Cascading Style Sheets (CSS), обеспечивают творческую свободу в разметке и дизайне веб-страниц. Пользуясь CSS, вы сможете украсить текст страниц привлекательными заголовками, буквицами и рамками, как в красочных глянцевых журналах. Можно точно разместить и позиционировать изображения, сделать колонки и создать баннеры, выделить ссылки динамическими эффектами. Можно также добиться постепенного появления и исчезновения элементов, перемещения объектов по странице или медленного изменения цвета кнопки при прохождении над ней указателя мыши.

Вы думаете, что все это довольно сложно? *Напротив!* Каскадные таблицы стилей как раз и предназначены для упрощения процесса оформления веб-страниц. Следующие несколько страниц будут посвящены изучению основ CSS.

Что такое CSS?

CSS — это язык стилей. Он используется для того, чтобы придать страницам на HTML — фундаментальном языке Всемирной паутины — совершенный вид. Надеюсь, вы будете использовать каскадные таблицы стилей, чтобы сделать свои страницы идеальными. После прочтения этой книги вы сможете создавать красивые, функциональные и простые в использовании сайты.

Прежде чем перейти к изучению каскадных таблиц стилей, необходимо понять, что такое язык HTML.

Что необходимо знать

Эта книга предполагает, что вы уже знакомы с языком HTML. Подразумевается, что вы создали пару сайтов (или по крайней мере несколько веб-страниц) и знакомы с основными элементами, такими как `html`, `p`, `h1`, `table`, составляющими основу языка гипертекстовой разметки документов. CSS бесполезен без HTML, поэтому вы должны знать, как создать простейшую веб-страницу с использованием основных HTML-элементов.

Если вы раньше создавали веб-страницы на HTML, но чувствуете, что знания требуется освежить, вам поможет следующий раздел книги.

ПРИМЕЧАНИЕ

Если вы только знакомитесь с HTML и возможностями его применения на практике, то посетите следующие сайты: HTML Dog (tinyurl.com/oujmfqw) и W3Schools (w3schools.com/html). Если вам

больше нравится читать книги, обратитесь к руководствам по созданию сайтов: «HTML5. Недостающее руководство» Мэтью Макдональда (издательство «БХВ-Петербург») или «Изучаем HTML, XHTML и CSS» Элизабет Фримен и Эрика Фримена (издательство «Эксмо»).

HTML: структура языка

В языке гипертекстовой разметки HTML используются простые команды, именуемые тегами, для определения различных частей — фрагментов. Ниже приведен HTML-код простой веб-страницы:

```
<!doctype html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Это заголовок веб-страницы</title>
  </head>
<body>
  <p>А это абзац этой веб-страницы</p>
</body>
</html>
```

Конечно, пример очень простой, но демонстрирует все основные элементы, необходимые обычной веб-странице. В нем вы заметите то, что называется объявлением типа документа — `doctype`, за ним следует *открывающий* тег `<html>` (со скобками), потом элемент `head` (голова, *раздел заголовка*), следом `body` (*тело*, раздел тела), а в нем непосредственно содержимое веб-страницы. Все это завершается *закрывающим* тегом `</html>`. Открывающий и закрывающий теги образуют *HTML-элемент*.

Типы документов

Все веб-страницы начинаются с объявления типа документа — строки кода, определяющей разновидность HTML, которой вы пользовались при написании страницы. В течение многих лет использовались два типа документов — HTML 4.01 и XHTML 1.0, и каждый из них имеет два стиля: *строгий* и *переходный*. Например, объявление переходного типа документа HTML 4.01 имеет следующий вид (другие объявления типа документа для HTML 4.01 и XHTML 1.0 выглядят примерно так же):

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

ПРИМЕЧАНИЕ

Примеры всех разновидностей объявлений типа документа можно найти на сайте tinyurl.com/onw4nq.

Если посмотреть на код примера HTML-страницы, показанный в этом разделе, то вы увидите, что в нем используется краткая форма объявления типа документа:

```
<!doctype html>
```

Объявление типа документа появилось в языке HTML5. По сравнению с предшественниками, в HTML5 заложена простота и рациональность использования. В этой книге применяется объявление типа документа из HTML5, поддерживаемое любым популярным браузером (даже старым Internet Explorer 6). Применять другие объявления, отличные от простого `doctype` из HTML5, не имеет смысла.

ПРИМЕЧАНИЕ

Несмотря на то что объявление типа документа работает в старых браузерах, это не значит, что они поддерживают все элементы или особенности HTML5. Например, Internet Explorer 8 или более ранние версии новые HTML5-элементы не распознают. Чтобы в этих версиях внести в элементы стиль с помощью CSS, нужно будет воспользоваться кодом JavaScript. Как инструктировать старые браузеры на поддержку современных веб-страниц, будет показано далее.

Независимо от предпочитаемого типа документа, объявляемого с помощью `doctype`, важно, чтобы использовалось объявление хотя бы одного из них. Без этого ваши страницы будут выглядеть по-разному в зависимости от браузера вашего посетителя, поскольку браузеры, не имеющие в качестве руководства объявления типа документа, по-разному отображают информацию, форматированную с помощью CSS.

Каждое объявление типа документа требует от вас написания HTML-кода определенным образом. Например, элемент для разбиения строк имеет в HTML 4.01 следующий вид:

```
<br>
```

Но в XHTML у этого элемента такой вид:

```
<br />
```

И здесь проявляется еще одно преимущество HTML5: он допускает применение любого из этих вариантов.

Как работают HTML-элементы

HTML-элементы состоят из тегов. В приведенном выше примере, как и в HTML-коде любой веб-страницы, большинство элементов используют пары тегов, начиная и завершая какой-то фрагмент — блок текста или другие команды. Будучи заключенными в скобки, эти теги представляют собой команды, которые говорят браузеру, каким образом отображать веб-страницу. Теги являются «разметочной» (markup) частью гипертекстового языка разметки — Hypertext Markup Language.

Открывающий тег каждого элемента показывает браузеру, где команда начинается, а заканчивающий — где заканчивается. Закрывающий тег всегда предваряется прямым слешем (/) после первого символа скобки (<).

На любой веб-странице обычно имеются как минимум следующие четыре HTML-элемента.

- Самая первая строка примера содержит **объявление типа документа** — элемент `doctype`, рассмотренный в предыдущем разделе.

- Тег <html> требуется в начале веб-страницы и (с добавленным слешем) в конце: </html>. Элемент html сообщает браузеру, что документ является программным кодом на языке HTML. Все содержимое страницы, включая остальные элементы, находится между открывающим и закрывающим тегами элемента html.

  Если представить веб-страницу в виде дерева, то элемент html будет его стволом. Две основные части любой веб-страницы — *раздел заголовка* и *тело* — представляют собой ветви.

- Раздел заголовка веб-страницы (head) содержит ее название. Здесь также может содержаться другая информация, не отображаемая при просмотре веб-страницы, например описание страницы, которая предназначена для браузеров и поисковых машин. Раздел заголовка заключается в открывающий и закрывающий теги элемента head.

  Кроме того, раздел заголовка может содержать информацию, используемую браузером для оформления HTML, имеющегося на странице, и для придания странице интерактивности. Вы увидите, что раздел заголовка может содержать код CSS (вроде того, который вы будете учиться создавать) или ссылку на другой файл, содержащий таблицу стилей.

- *Тело* веб-страницы, следующее непосредственно за разделом заголовка и заключенное в теги элемента body, содержит все, что должно появиться в окне браузера: заголовки, текст, изображения и т. д.

  Внутри тела страницы, как правило, можно найти следующие элементы:

- элемент p — открывающий тег начинает абзац, а закрывающий завершает;

- элемент strong — выделяет текст как важный; например, код Внимание! сообщит браузеру о том, что слово «Внимание!» должно быть выделено;

- элемент a, или элемент привязки, — создает гиперссылку, при щелчке на которой можно переместиться в другую позицию веб-страницы или на другую страницу (нужно указать браузеру эту ссылку путем размещения ее внутри открывающего тега <a>, например Щелкните здесь!).

Браузер знает, что при щелчке кнопкой мыши на ссылке со словами Щелкните здесь! посетитель вашей страницы должен перейти на сайт с адресом http://www.piter.com. Часть тега a — слово href — называют атрибутом, а URL (*унифицированный указатель ресурса*, или URL-адрес) является его значением. В этом примере http://www.piter.com — значение атрибута href.

HTML5: больше элементов

HTML5, актуальная версия языка HTML, существует уже несколько лет. Иногда вы будете слышать названия, не относящиеся к HTML-элементам, например локальные хранилища (способ хранения данных с сайта на компьютере посетителя), геолокация (способ определения координат компьютера посетителя) и рисование на веб-странице с помощью библиотеки WebGL. Строго говоря, эти технологии не

являются частью HTML, но они относятся к новым возможностям браузера, появившимся вместе с HTML5.

В этой книге термин *HTML5* всегда относится к типу документа HTML5, а также к новым элементам, являющимся частью нового стандарта HTML5. HTML5 не несет каких-либо кардинальных отличий от своих предшественников — его целью является забота о том, чтобы Всемирная паутина продолжала работать так же, как и прежде, поддерживая новые потребности дизайнеров. В версии HTML5 были добавлены новые элементы. Например, элемент `header` может включать в себя содержимое, которое обычно встречается в верхней части страницы, — логотип и общие для всего сайта навигационные ссылки. Новый элемент `nav` заключает в себе набор ссылок, использующихся для навигации по сайту, а элемент `footer` размещает в себе все, что обычно помещается в нижней части страницы, например юридическую информацию, контакты по электронной почте и т. д.

Кроме того, в HTML5 добавлены новые элементы, позволяющие внедрять на страницу видео- и аудиоконтент, и новые элементы формы, добавляющие такие сложные компоненты, как ползунковые регуляторы, всплывающие панели выбора даты, а также встроенную браузерную поддержку проверки допустимости данных, введенных в форму (которая гарантирует правильное заполнение ваших форм посетителями). На протяжении всей книги, особенно в следующей главе, вы будете встречать примеры использования языка HTML5.

Программное обеспечение, используемое для CSS

Чтобы создавать веб-страницы на языках HTML и CSS, вполне достаточно обычного текстового редактора, такого как Блокнот (Notepad) в операционной системе Windows или Text Edit в OS X. После верстки нескольких сотен строк кода HTML или CSS вы, наверное, захотите пользоваться программой, более подходящей для работы с веб-страницами. В этом разделе перечислены некоторые из них. Одни бесплатные, другие придется приобрести.

ПРИМЕЧАНИЕ

Существуют сотни программ, которые могут помочь вам в создании веб-страниц, поэтому здесь приводится неполный список. Все же это самые популярные программы, которыми пользуются любители CSS на сегодняшний день.

Бесплатное программное обеспечение

На данный момент создано много бесплатных программ для редактирования веб-страниц и таблиц стилей. Если вы все еще пользуетесь обычным текстовым редактором, то имеет смысл попробовать одну из нижеприведенных программ.

- **Brackets** (Windows, OS X, Linux; brackets.io). Бесплатный текстовый редактор с открытым кодом, созданный под руководством компании Adobe, содержит множество инструментов для работы с языком HTML и каскадными таблицами стилей. Он будет особенно полезен веб-дизайнерам и разработчикам.

- **Atom** (Windows, OS X, Linux; atom.io). Еще один бесплатный текстовый редактор с открытым кодом, созданный разработчиками из компании GitHub, чрезвычайно популярный благодаря функциям обмена кодом и совместной разработки сайта. Как и программа Brackets, этот новый текстовый редактор в первую очередь предназначен для веб-разработчиков.
- **jEdit** (Windows, OS X, Linux; jedit.org). Бесплатный текстовый редактор, использующий Java, работает практически на всех компьютерах. В нем вы найдете большинство тех функций, которые доступны в коммерческих программах, включая подсветку синтаксиса для CSS.
- **Notepad++** (Windows; notepad-plus.sourceforge.net). Множество людей просто преклоняются перед этим текстовым редактором. Естественно, в нем есть встроенная функциональность, которая идеально подходит для написания HTML- и CSS-кода, включая подсветку синтаксиса, — HTML-элементы и другие ключевые слова имеют собственные цвета, что значительно облегчает их поиск среди других элементов HTML и CSS.

Платное программное обеспечение

Существует множество коммерческих программ для создания сайтов: от недорогих текстовых редакторов до мощных комплексов для верстки кода.

- **EditPlus** (Windows, editplus.com) — относительно недорогой ($35) текстовый редактор, который поддерживает подсветку синтаксиса, FTP, автозавершение ввода и другие функции.
- **skEdit** (OS X, skedit.com) — редактор веб-страниц ($30), полная поддержка FTP/SFTP, подсказка команд и другие полезные функции.
- **Coda2** (OS X; panic.com/coda) — многофункциональное средство для создания веб-страниц стоимостью $99. Включает в себя текстовый редактор, средство предварительного просмотра страниц, FTP- и SFTP-клиент и графический интерфейс для создания CSS-стилей.
- **Sublime Text** (OS X, Windows, Linux; sublimetext.com) — мощный текстовый редактор ($70), созданный веб-разработчиками. Обычно используется в компаниях, занимающихся веб-дизайном.
- **Dreamweaver** (OS X и Windows, adobe.com/ru/products/dreamweaver.html) — визуальный редактор веб-страниц (подписка на месяц стоит от $19,99). Он позволяет видеть, как ваша страница выглядит в браузере. Программа также включает мощный текстовый редактор и превосходные инструменты создания кода CSS. Для эффективного использования этого приложения см. документацию по программе или печатные издания (например, книгу «Adobe Dreamweaver CC. Официальный учебный курс», Эксмо, 2014).

ПРИМЕЧАНИЕ

Здесь речь идет о программах, которые позволяют редактировать код, написанный на языках HTML и CSS. Для создания веб-страниц вам достаточно изучить всего одну из них.

Об этой книге

Всемирная паутина — действительно очень удобное изобретение. К сожалению, правила, по которым *работает* Всемирная сеть, не так просты для понимания. Программисты и технические специалисты, которые пишут официальную документацию, поясняющую основные понятия ее функционирования, не ориентируются на среднестатистического пользователя. Зайдите на сайт tinyurl.com/ncyzfj8, чтобы осознать, может ли это быть понятно обычному человеку.

Люди, приступающие к изучению CSS, как правило, не знают, с чего начать. А тонкости, имеющиеся в CSS, могут сбить с толку даже маститых веб-разработчиков. Цель этой книги — служить руководством для обучения. В ней вы найдете пошаговые инструкции для создания красивых веб-страниц с использованием языка CSS.

Эта книга написана так, чтобы помочь читателям любого уровня. Для извлечения максимальной пользы из материала вы обязательно должны учиться на приведенных примерах HTML и CSS. Если же вы никогда раньше не создавали веб-страницы, то обратитесь к практикуму в конце главы 2. Материал, содержащийся в этих главах, написан для тех, кто уже немного освоился в данной области и имеет средний уровень знаний. Если же вы плохо знаете принципы создания веб-страниц, то для лучшего понимания освещаемой темы должны ознакомиться с текстом врезок «В курс дела!». С другой стороны, если у вас имеется большой опыт создания веб-страниц, обратите внимание на врезки «Для опытных пользователей». Они содержат подсказки, приемы и методы для опытных программистов.

Основные разделы книги. Книга разделена на пять частей. Первые четыре части содержат по несколько глав, а последняя часть состоит из приложений.

- **Часть I. Основы CSS.** Здесь описано создание каскадных таблиц стилей в целом и дан краткий обзор ключевых понятий, таких как *наследование*, *селекторы* и *каскадность* таблиц стилей. Попутно с изучением CSS вы получите основные навыки написания HTML-кода. Практикумы закрепят вводимые в главах основные понятия и позволят вам почувствовать эффективность использования CSS.
- **Часть II. Применение CSS.** Перенесет вас в реальный мир веб-дизайна. Вы изучите наиболее важные свойства CSS и их использование для форматирования текста, попрактикуетесь в создании полезных инструментов навигации и сможете улучшить внешний вид своих экспериментальных веб-страниц, добавив графику. Вы также узнаете о том, как с помощью CSS создавать простую анимацию. Эта часть также содержит рекомендации о том, как создавать красивые таблицы и формы.
- **Часть III. Верстка страниц с помощью CSS.** Поможет вам разобраться с самой запутанной, но очень полезной функцией CSS: с управлением размещением элементов на вашей странице. Вы познакомитесь со схемами дизайна (размещение контента в две и три колонки) и узнаете, как добавить боковые панели. Будет рассказано о двух основных методах позиционирования элементов на странице:

абсолютном и *относительном*. Вы также научитесь создавать сайты, адаптируемые для лучшего восприятия в браузерах настольных систем, планшетных компьютеров и мобильных устройств, и применять flexbox — новый мощный инструмент для создания макета веб-страниц.

- **Часть IV. Профессиональные приемы CSS-верстки.** Содержит советы от профессионалов по улучшению ваших каскадных таблиц стилей, а также расскажет о Sass — мощном и эффективном способе верстки таблиц стилей.
- **Часть V. Приложения**, включая два справочника. Справочник свойств CSS описывает каждое свойство в отдельности в простой и доступной форме, чтобы вы могли быстро узнать о полезных свойствах CSS, которые раньше вам могли не попадаться, или быстро освежить в памяти уже знакомые свойства. Во втором приложении приводится описание инструментов и средств для создания и применения каскадных таблиц стилей.

Основы

При чтении книги и выполнении примеров на компьютере вы должны быть знакомы с некоторыми терминами и понятиями.

- **Щелчок кнопкой мыши.** Пользуясь мышью или тачпадом (трекпадом) вашего компьютера, вы можете выполнить три действия. *Щелчок кнопкой* мыши означает, что нужно навести указатель мыши на какой-либо объект на экране монитора, а затем, не перемещая указатель, нажать и отпустить левую кнопку мыши (тачпада). *Щелчок правой кнопкой* мыши означает соответственно быстрое нажатие правой кнопки мыши, опять-таки не перемещая указатель. *Двойной щелчок* кнопкой мыши означает соответственно быстрое нажатие левой кнопки мыши дважды, опять-таки не перемещая указатель. *Перетаскивание* мышью означает перемещение указателя при нажатой и удерживаемой кнопке мыши.

Если говорится, что нужно щелкнуть кнопкой мыши, нажав клавишу ⌘ (OS X) или Ctrl (Windows), то следует щелкнуть левой кнопкой мыши, предварительно нажав клавишу ⌘ или Ctrl.

- **Меню.** Это строка с кнопками, находящаяся вверху вашего экрана или окна: Файл (File), Редактирование (Edit) и т. д. Чтобы появился список команд, соответствующих каждому конкретному пункту меню, нужно просто щелкнуть кнопкой мыши на соответствующем слове (пункте меню), в результате раскроется перечень команд меню. Подразумевается, что вы умеете запускать программы, открывать сайты и скачивать файлы. Вы должны уметь пользоваться меню Пуск (Start) в Windows или панелью Dock в OS X, а также окном Панель управления (Control Panel) в Windows или Системные настройки (System Preferences) в OS X.
- **Сочетания клавиш.** Убирая руки с клавиатуры для перемещения указателя мыши, вы теряете время и можете сбиться. Поэтому многие опытные компьютерные

специалисты вместо команд меню везде, где только можно, используют сочетания клавиш. Когда для быстрого вызова той или иной функции предлагается сочетание клавиш, подобное Ctrl+S (⌘+S) (это сочетание клавиш сохраняет изменения, внесенные в текущий документ), это говорит о том, что вам нужно удерживать в нажатом состоянии клавишу Ctrl или клавишу ⌘ и в это время нажать клавишу S, а затем отпустить обе клавиши.

Об ▸ этих ▸ стрелках

В этой книге вам будет попадаться текст такого рода: «Откройте папку Система ▸ Библиотека ▸ Шрифты (System ▸ Library ▸ Fonts)». Это сокращение более пространной инструкции, которая предписывает открыть три последовательно вложенные друг в друга папки и должна звучать следующим образом: «На жестком диске найдите папку под названием Система (System). Откройте ее. В окне системной папки есть папка Библиотека (Library); дважды щелкните на ней, чтобы открыть. В этой папке находится папка Шрифты (Fonts). Дважды щелкните на ней и также откройте».

Точно такие же сокращенные записи команд со стрелками помогут вам открыть нужный пункт меню, как показано на рис. 0.1.

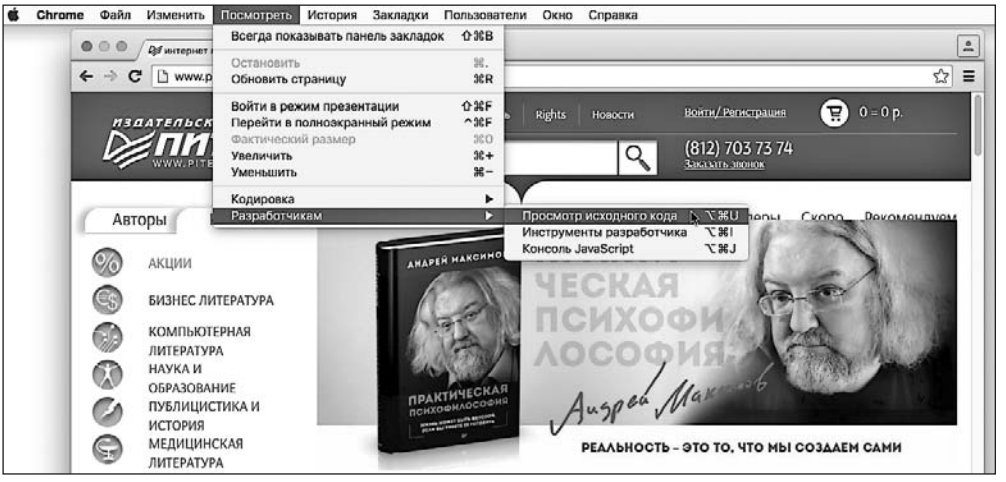

Рис. 0.1. Запись со стрелками позволяет упростить инструкции по использованию меню

Например, запись Просмотреть ▸ Разработчикам ▸ Просмотр исходного кода (View ▸ Developer ▸ View Source) представляет собой краткий способ дать указание, выполняемое на рис. 0.1: «В пункте меню Просмотреть (View) выберите подпункт Разработчикам (Developer), а после этого выберите подпункт Просмотр исходного кода (View Source)».

Соглашения, использованные в данной книге

В текущем подразделе приводится список соглашений, которые используются в данной книге.

Шрифт для названий

Применяется для отображения URL, а также названий папок и выводимой на экран информации.

`Шрифт для команд`

Используется для имен файлов, названий путей, имен переменных и команд. Например, путь будет выглядеть так: `/Developer/Applications`.

`Шрифт с постоянной шириной`

Применяется для отображения примеров исходного кода и содержимого файлов.

`Полужирный шрифт с постоянной шириной`

Используется для выделения кода, добавленного в старый код.

Вам следует обращать особое внимание на специальные заметки, отделенные от основного текста.

ПРИМЕЧАНИЕ

Это подсказка, пожелание, заметка общего типа. Содержит полезную прикладную информацию по рассматриваемой теме.

ВНИМАНИЕ

Это предостережение или указание, говорящее о том, что вам необходимо быть внимательными. Оно часто указывает на то, что ваши деньги или ваша частная информация могут оказаться под угрозой.

СОВЕТ

Это совет. Советы содержат полезную информацию по рассматриваемой теме, зачастую выделяя важные концепции или лучшие практические решения.

Интернет-ресурсы

Приобретя данное руководство, вы получаете не только книгу для чтения. Во Всемирной паутине можно найти файлы-примеры для получения практического опыта, а также советы, статьи и, может быть, даже несколько видеосюжетов по конкретной теме.

Вы также можете связаться с издательством «Питер» и сообщить о том, что вам понравилось (или не понравилось) в данной книге. Для этого заходите на сайт piter.com.

Примеры к книге

Эта книга создана для того, чтобы работа над вашими веб-проектами велась быстрее и профессиональнее. Поэтому естественно, что половина всего ценного, что есть в этой книге, доступно во Всемирной паутине.

По ходу чтения вам будут встречаться *практикумы* — пошаговые обучающие уроки, которые вы сможете выполнять, используя исходные данные (графические файлы и незаконченные веб-страницы). Их можно загрузить с сайта github.com/mrightman/css_4e. От простого прочтения этих пошаговых уроков толку будет мало. Но если их прорабатывать на компьютере, можно разобраться, каким образом профессиональные дизайнеры создают веб-страницы.

Среди примеров вы также найдете законченные страницы, которые позволят вам сравнить свою работу с требуемым конечным результатом.

ЧАСТЬ I

Основы CSS

1 HTML и CSS

Каскадные таблицы стилей без языка HTML ничто. Код на языке HTML образует структуру веб-страниц с их содержимым (контентом). Хотя страница, созданная с помощью только языка HTML, не очень красива, без него Всемирной паутины не существовало бы. Поэтому для получения от CSS наибольшей отдачи ваш HTML-код должен предоставить однородную, хорошо скроенную основу. В этой главе вы познакомитесь с основами каскадных таблиц стилей, узнаете, как создавать HTML-код, более приспособленный под нужды CSS.

Важно отметить, что повсеместное использование на сайте CSS существенно облегчает создание HTML-кода. Вам больше не нужно будет применять средства HTML для создания и улучшения дизайна страниц (собственно, HTML для этого никогда и не предназначался). Все, что связано с графическим дизайном страниц, обеспечивается с помощью CSS. Соответственно, работа над основным кодом веб-страниц на языке HTML упрощается, поскольку HTML-страницы, написанные для совместной работы с CSS, имеют менее громоздкий, более понятный и прозрачный код. При этом, естественно, страницы загружаются быстрее, что немаловажно для посетителей вашего сайта.

Посмотрите на рис. 1.1. Дизайн обеих страниц одинаков, однако верхняя создана лишь с использованием CSS, а нижняя — только с помощью HTML. Размер HTML-файла верхней страницы всего 4 Кбайт, в то время как его размер для нижней страницы, полностью написанной на HTML, — почти в четыре раза больше (14 Кбайт). Подход к написанию веб-страниц с применением исключительно HTML требует намного большего объема кода, чтобы достичь практически тех же визуальных эффектов: 213 строк HTML-кода по сравнению с 71 строкой для версии с применением CSS.

HTML: прошлое и настоящее

Язык HTML на сегодняшний день составляет основу для написания любой веб-страницы во Всемирной паутине. При использовании CSS написание кода на языке HTML упрощается. Теперь вам не нужно использовать HTML-элементы (такие как элемент `font`) для управления оформлением веб-страниц. Эту работу выполняют каскадных таблицы стилей. Прежде чем мы начнем знакомство с CSS, давайте поговорим о прошлом и настоящем.

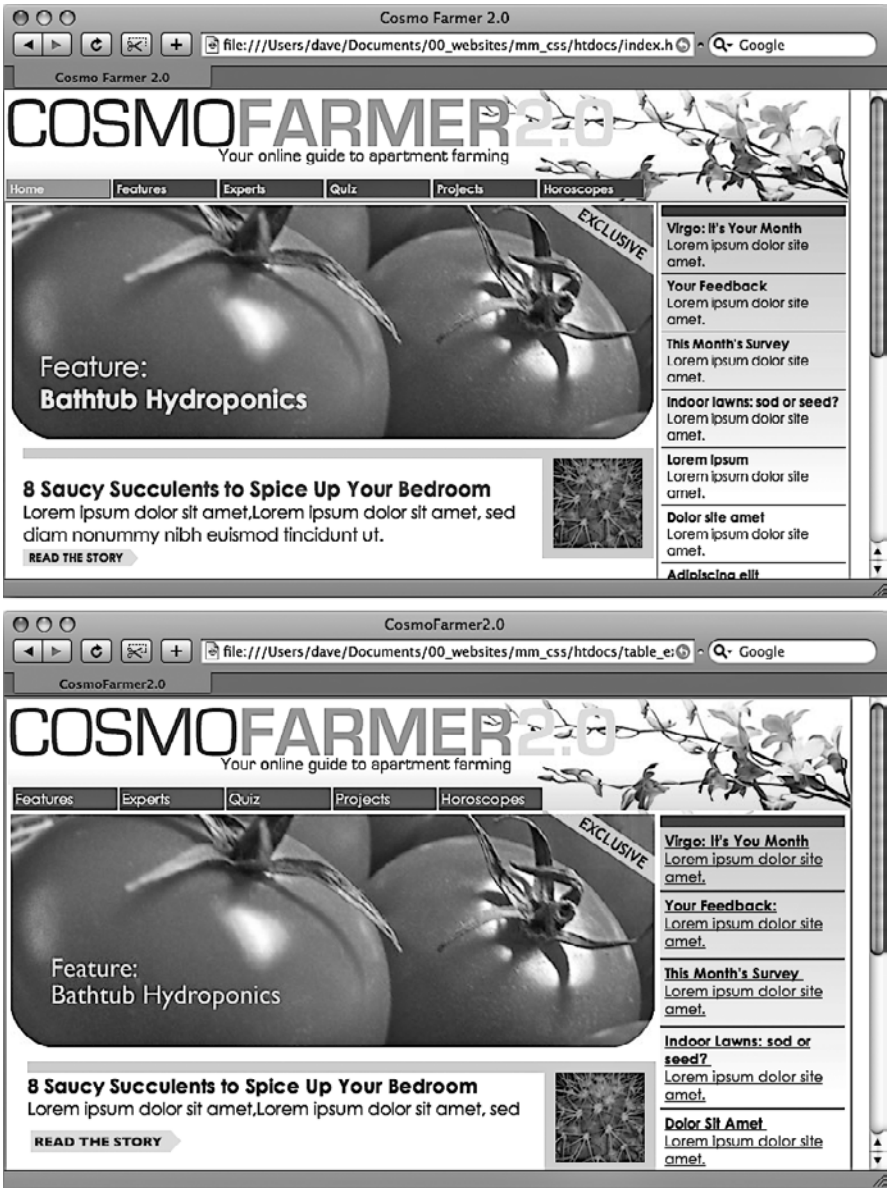

Рис. 1.1. Каскадные таблицы стилей упрощают использование языка HTML

Прошлое HTML: лишь бы все выглядело хорошо

Когда группа ученых создала Всемирную паутину, которая первоначально предназначалась для совместного использования и поиска технический документации, никто к графическим дизайнерам не обращался. Язык HTML нужен был ученым только для структурирования информации с целью ее более простого восприятия.

Возьмем для примера элемент h1. Он выделяет в тексте важный заголовок, а элемент h2 создает подзаголовок с меньшим размером шрифта. Другой широко используемый элемент — ol (ordered list — «упорядоченный список») — создает нумерованный список для перечислений.

Когда язык HTML стали применять не только ученые, но и обычные пользователи, они захотели, чтобы их веб-страницы выглядели красиво. С тех пор дизайнеры веб-страниц начали использовать теги для форматирования страниц в дополнение к их основному назначению — структурированию данных. Например, вы можете применить открывающий и закрывающий теги элемента blockquote (предназначен для цитирования материала из другого источника) к любому тексту, который нужно выделить небольшим отступом вправо. Вы можете пользоваться тегами элементов заголовков, чтобы выделить любой текст полужирным шрифтом большего размера, независимо от того, заголовок это или нет.

Достаточно сложный способ применения элемента table был придуман веб-дизайнерами, чтобы создавать колонки текста, а также точно позиционировать изображения и текст на странице. Поскольку элемент первоначально предназначался для отображения таких таблиц, как результаты исследовательских данных, расписания поездов и т. д., проектировщикам пришлось упражняться в применении элемента table самыми необычными способами, создавая неоднократно вложенные таблицы, чтобы содержимое веб-страниц смотрелось красиво.

Тем временем производители браузеров также прилагали усилия в совершенствовании языка разметки, вводя новые HTML-элементы и атрибуты для улучшения дизайна веб-страниц. Например, элемент font позволяет определять цвет, начертания и один из семи кеглей шрифта (это приблизительно в 100 раз меньше типоразмеров шрифта, чем предлагает, скажем, редактор Microsoft Word).

Наконец, когда проектировщики не могли добиться нужных результатов, они часто применяли графику. Например, использовали очень большой рисунок в качестве фона для веб-страницы или *нарезали* его на маленькие графические файлы и собирали их воедино в таблицах, чтобы воссоздать оригинальное изображение.

По мере использования всех этих премудростей и применения атрибутов элементов, широкого употребления изображений и т. д. для изменения дизайна страниц HTML-код разрастался до неузнаваемости. Но сложный код замедляет визуализацию страницы в браузере, а также снижает скорость ее загрузки.

Настоящее HTML: подготовка рабочего материала для CSS

Независимо от того, что представляет собой веб-страница — календарь промыслового сезона, схему проезда к ближайшему супермаркету или фотоальбом прошлого дня рождения вашего ребенка, — именно ее дизайн создает имидж, заставляя сайт выглядеть профессионально. Хороший дизайн страниц сайта помогает донести послание его посетителям, которые должны легко найти там именно то, что ищут.

Таким образом, чтобы веб-страницы на языке HTML выглядели привлекательно, дизайнерам приходилось усиленно трудиться. Дизайн с помощью CSS позволяет языку HTML вернуться к исполнению своей прямой обязанности — созданию

структуры документа. Использование HTML для дизайна веб-страниц на сегодняшний день является признаком дурного тона. Поэтому не волнуйтесь по поводу того, что у элемента `h1` слишком большой размер шрифта или отступы упорядоченного списка очень велики. Вы сможете изменить их с помощью каскадных таблиц стилей. Во время написания кода думайте о HTML как о средстве структурирования. Используйте его, чтобы упорядочить содержимое страницы, а CSS — чтобы сделать это содержимое привлекательным.

Верстка HTML-кода вместе с CSS

Для тех, кто не очень хорошо знаком с веб-дизайном, возможно, будут полезны некоторые подсказки по языку HTML. Если раньше вы уже создавали веб-страницы, то сможете избавиться от стиля написания HTML-кода, который лучше быстрее забыть. Сейчас речь пойдет о способе написания HTML-кода для его совместного использования с кодом CSS.

Думайте о структуре

HTML придает особый вид тексту путем деления его на логические блоки и их определения на веб-странице: например, главное значение элемента `h1` — создать заголовок, предшествующий основному содержимому страницы. Заголовки второго, третьего уровней и т. д. — подзаголовки — позволяют делить содержимое страниц на менее важные, но связанные разделы. У веб-страницы, как и у книги, которую вы держите в руках, должна быть логическая структура. У каждой главы этой книги есть заголовок (отформатированный, например, элементом `h1`), а также несколько разделов и, соответственно, подзаголовков (например, с элементом `h2`), которые, в свою очередь, содержат подразделы с заголовками более низкого уровня. Представьте, насколько сложнее было бы читать эту книгу, если бы весь текст состоял из одного длинного абзаца, без деления на разделы, подразделы, пункты, без выделения примечаний, гиперссылок и т. д.

ПРИМЕЧАНИЕ

Руководство по языку HTML можно найти на сайте tinyurl.com/4mwq8. Краткий список всех HTML-элементов есть в справочнике Mozilla Developer по адресу tinyurl.com/p8rexlq.

Помимо заголовков, в HTML есть множество других элементов для разметки содержимого веб-страницы, а также для определения назначения ее каждого логического фрагмента. Наиболее часто применяют следующие элементы: `p` — для создания абзацев текста, `ul` — для создания маркированных (ненумерованных) списков. Далее по степени применения идут элементы, отображающие специфичное содержимое, например `abbr` — сокращения, аббревиатуры, `code` — программный код.

При написании HTML-кода для CSS используйте элементы, размещая их рядом друг с другом, насколько это возможно. Ориентируйтесь на роль, которую играет фрагмент текста на веб-странице, а не на внешний вид, который текст приобретает благодаря этому элементу (рис. 1.2). Например, перечень ссылок

на панели навигации — это и не заголовок, и не абзац текста. Больше всего это похоже на маркированный список. Таким образом, выбираем элемент `ul`. Вы можете сказать, что элементы маркированного списка расположены вертикально один за другим, а нам требуется горизонтальная панель навигации, в которой все ссылки располагаются горизонтально. Об этом можно не беспокоиться. Средствами CSS очень просто преобразовать вертикальный список ссылок в элегантную горизонтальную панель навигации, о чем вы сможете прочитать в главе 9.

Рис. 1.2. Пример изменения оформления текста на странице с помощью CSS

Разнообразие HTML-элементов

Пестрого ассортимента HTML-элементов не всегда достаточно для оформления широкого диапазона контента, который вы хотели бы опубликовать на своей вебстранице. Конечно, элемент `code` прекрасно подходит для разметки программного кода, но большинство людей считают элемент `recipe` удобнее. К счастью, HTML обеспечивает несколько «структурных» элементов, которые позволяют лучше идентифицировать и группировать контент и в процессе работы обеспечивают «маркеры», помогающие назначить те или иные стили CSS определенным элементам на странице. К их числу относятся элементы `div` и `span`, существующие практически с момента появления языка HTML. Язык HTML5 предоставил широкий спектр элементов, которые позволяют группировать контент. Например, элемент `footer` можно использовать для отображения колонтитулов — такой информации, как сведения об авторских правах, контактных данных, или списка источников.

До появления CSS для достижения определенных визуальных эффектов дизайнеры пользовались элементом `font` и другими средствами HTML:

```
<p>
  <strong>
    <font color="#0066FF" size="5" face="Verdana,
      Arial, Helvetica, sans-serif">Urban Agrarian
      Lifestyle</font></strong>
    <br />
    <font color="#FF3300" size="4" face="Georgia,
      Times New Roman, Times, serif">
    <em>
  <strong>A Revolution in Indoor Agriculture
    <br /></strong></em></font>
      Lorem ipsum dolor sit amet...
</p>
```

Используя CSS, можно добиться тех же результатов (и даже лучше) с гораздо меньшим объемом HTML-кода (см. рис. 1.2):

```
<h1>The Urban Agrarian Lifestyle</h1>
<h2>A Revolution in Indoor Agriculture</h2>
<p>Lorem ipsum dolor sit amet...</p>
```

Кроме того, применяя CSS для дизайна веб-страницы, вы используете HTML по его прямому назначению, то есть именно для разметки кода веб-страницы на логические фрагменты, не заботясь о форматировании и внешнем виде страницы.

Использование элементов div и span

Элементы `div` и `span` применяются на протяжении всего существования Всемирной паутины. Обычно они используются для организации и группирования контента, с чем не всегда справляются другие HTML-элементы. Они похожи на пустые сосуды, которые вы сами и заполняете. Элемент `div` (предназначен для деления на фрагменты) определяет любой отдельный блок содержимого, как, например, абзац или заголовок, поэтому называется *блочным*. Поскольку у элементов `div` и `span` нет никаких свойств для визуализации, вы можете применять к ним CSS-стили, чтобы фрагменты внутри этих тегов выглядели так, как вам хочется. Однако вы также можете логически объединить любой набор таких элементов, как заголовок, несколько абзацев, маркированный список и т. д., в единственном блоке `div`.

Элемент `div` — замечательное средство разбивки веб-страницы на такие логические фрагменты, как баннер, колонтитул, боковая панель и т. д. Впоследствии, используя CSS, вы сможете позиционировать любой из этих фрагментов в выбранное место веб-страницы, создавая сложную схему разметки (см. часть III).

В КУРС ДЕЛА!

Простой HTML — помощь поисковым системам

Научившись писать HTML-код так, чтобы использовать его исключительно для структурирования содержимого документов, и применяя CSS как инструмент для дизайна, форматирования, придания страницам законченного внешнего вида, вы обнаружите дополнительные преимущества простого и понятного HTML. С одной стороны, вы сможете поднять позиции своих веб-страниц в поисковых системах, заняв верхние строки в списках таких поисковых систем, как Google, Yahoo! и Bing. Когда поисковые системы сканируют просторы Всемирной паутины в поисках новых сайтов и новой информации, они систематизируют данные, просматривая *весь* HTML-код веб-страниц на предмет актуального содержимого. Старый способ написания HTML-кода с применением элементов форматирования (например, `font`) и множества таблиц для верстки дизайна страницы влияет на работу поискового сервера. Некоторые поисковые системы прекращают чтение HTML-кода страницы после просмотра определенного количества символов. Если у вас на страницах имеется большой объем HTML-кода для создания дизайна, то поисковый сервер может упустить важное содержимое страниц или отдельные страницы вообще не будут систематизированы.

Простой и структурированный HTML, напротив, будет легко просматриваться и систематизироваться поисковыми системами. Использование элемента `h1` для выделения важных тем страниц (в противоположность форматированию текста большим полужирным шрифтом) — грамотный подход, поскольку поисковые серверы придают большое значение содержимому этого элемента во время сканирования страниц.

Рекомендации Google по созданию веб-страниц для правильного восприятия их поисковыми системами смотрите по адресу tinyurl.com/olx2lyk.

Элемент span применим к *строчным* элементам страницы, то есть к словам, фразам, которые находятся в пределах абзаца текста или заголовка. Его можно использовать точно так же, как и другие строчные HTML-теги, к примеру, как a (элемент привязки, позволяющий добавить ссылку к фрагменту текста) или strong (позволяющий отформатировать фрагмент текста в абзаце полужирным начертанием). Можно применить элемент span, например, к названию компании и затем использовать CSS, чтобы выделить это название другим шрифтом, цветом и т. д.

Рассмотрим пример использования этих элементов в работе. К ним добавлены атрибуты id и class, предназначенные в том числе для применения стилей к фрагментам страницы.

```
<div id="footer">
  <p>Собственность 2015, <span class="bizName">SuperCo.com</span></p>
  <p>Звоните в службу поддержки потребителей по номеру 555-555-5501
     для подробной информации.</p>
</div>
```

Разговор об этих элементах не ограничится кратким введением. Они часто встречаются на тех веб-страницах, где широко применяется CSS, и эта книга поможет вам научиться использовать их в комбинации с CSS для получения творческого контроля над веб-страницами.

Элементы в HTML5

Элемент div имеет общий характер — это блочный элемент, используемый для разбиения страницы на разделы. Одна из целей HTML5 — предоставление в распоряжение разработчиков широкого выбора из других, *семантически* более осмысленных элементов. Придание HTML-коду большей семантики просто означает использование элементов, в точности описывающих свое содержимое. Как уже упоминалось в текущем разделе, вы можете воспользоваться элементом h1 (заголовок первого уровня), помещая в него текст, описывающий основное содержимое страницы. По аналогии с этим элемент code четко дает понять, информация какого сорта в него помещена (программный код).

HTML5 включает в себя множество различных элементов, имена которых отражают тип их содержимого. Они могут использоваться вместо элемента div. Элемент article (статья), к примеру, применяется для обозначения раздела страницы, содержащего завершенную, независимую публикацию, статью: запись блога, описание товара в интернет-магазине или просто основной текст страницы. Точно так же элемент header (верхний колонтитул) является признаком *верхнего колонтитула*, или *баннера* в верхней части страницы, который обычно содержит логотип, навигацию, относящуюся ко всему сайту, название страницы с рекламным слоганом и т. д.

ПРИМЕЧАНИЕ

Дополнительные сведения об HTML-элементах можно найти на сайтах tinyurl.com/d4gsdrr и tinyurl.com/pc3ddbc, а также в книге Дженнифер Роббинс «HTML5. Карманный справочник», Вильямс, 2015.

Многие HTML5-элементы предназначены для расширения возможностей обычного элемента div. Для структурирования содержимого страницы часто используются и другие HTML5-элементы.

- `section` (раздел) — группирует взаимосвязанное содержимое, например главу книги. К примеру, вы можете разбить содержимое главной страницы на три раздела: вводную информацию о сайте, контактную информацию и самые свежие новости.
- `aside` (отступление) — предназначен для обозначения содержимого, связанного с неким контентом. Например, пометки на полях в печатном журнале.
- `footer` (нижний колонтитул) — содержит информацию, которая обычно помещается в нижнем колонтитуле страницы, например сведения об авторских правах, другая правовая информация, ссылки для навигации по сайту и т. д. Но на количество элементов `footer` на одной странице ограничений не накладывается, вы можете, скажем, поместить нижний колонтитул внутри элемента `article`, чтобы хранить в нем информацию, относящуюся к публикации, например сноски, ссылки или выписки.
- `nav` (навигация) — используется для обозначения содержимого в виде основных навигационных ссылок.
- `figure` (рисунок) — применяется для иллюстраций. Вы можете поместить в него элемент `img`, а также `figcaption`, предназначенный для отображения подрисуночной подписи — пояснения к фотографии или иллюстрации, находящейся внутри элемента `figure`.

СОВЕТ

Разобраться в том, какой из HTML5-элементов лучше использовать, то есть должен ли ваш текст быть статьей — article или разделом — section, бывает порой непросто. Удобная блок-схема, которая поможет разобраться в предназначении новых элементов разбиения информации, имеющихся в HTML5, представлена по адресу tinyurl.com/o298cs6.

Существуют и другие HTML5-элементы, и многие из них просто предоставляют более описательную альтернативу элементу `div`. В этой книге используются как элементы `div`, так и новые HTML5-элементы, придающие веб-странице более выраженную организацию содержимого. Недостаток HTML5 заключается в том, что браузер Internet Explorer 8 и его более ранние версии не распознают новые элементы без посторонней помощи (см. врезку «Обходной прием» далее в этой главе).

Следует также заметить, что, кроме чувства сопричастности к самым последним течениям в веб-дизайне, от использования некоторых из этих HTML5-элементов на самом деле нет никакой ощутимой пользы. Например, использование элемента `article` просто для того, чтобы содержать в нем сводку новостей веб-страницы, не улучшит ее внешний вид. Если хотите, можете и впредь применять элемент `div`, избегая элементов разбиения страницы на разделы, предлагаемых HTML5.

Кроме того, даже используя HTML5-элементы, вы все же иногда будете вынуждены прибегать к помощи `div` для простой группировки других HTML-элементов. Вы будете делать это в тех случаях, когда необходимо переместить группу элементов в другое место на странице, чтобы назначить им последовательный фоновый цвет или нарисовать контур и добавить тень.

Представление о макете страницы

Когда для обозначения основной темы страницы используется элемент `h1`, а для добавления текстового абзаца — элемент `p`, вы в конечном счете хотите увидеть привлекательную страницу. Когда после прочтения части III книги вы научитесь использовать CSS для компоновки макета страницы, вам уже не нужно будет следить за дизайном еще на этапе написания HTML-кода.

Макет страницы можно представить себе в виде искусно расположенных прямоугольных элементов (пример такого расположения показан на рис. 1.3). В конечном счете дизайн страницы, состоящей из двух вертикальных колонок текста, фактически представляет собой лишь два смежных прямоугольника. Верхний колонтитул — логотип, лозунг, поля поиска и элементов навигации по сайту — представляет собой не что иное, как широкий блок (прямоугольник), занимающий всю верхнюю часть окна браузера. Иными словами, если вы представите себе группировку и макет содержимого страницы, то должны увидеть блоки, установленные друг на друга, следующие друг за другом и находящиеся друг под другом.

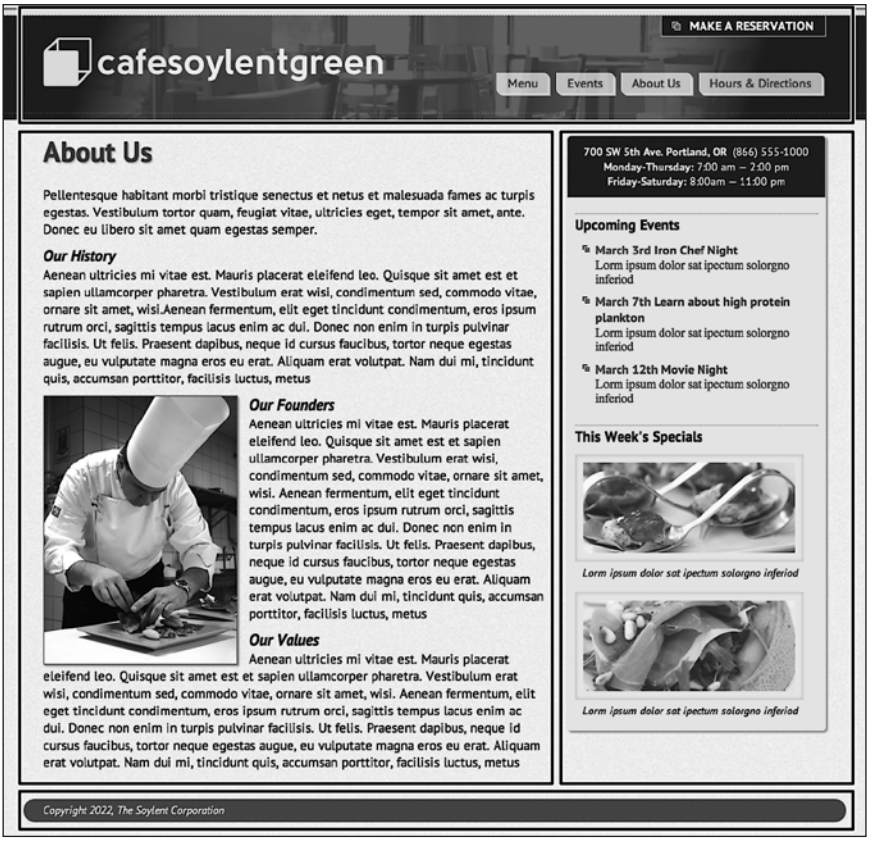

Рис. 1.3. Стандартный двухколоночный макет страницы, включающий такие основные структурные блоки, как баннер (в верхней части), колонка основного содержимого (слева по центру), боковая панель (справа по центру) и нижний колонтитул (в нижней части)

ОБХОДНОЙ ПРИЕМ

Как заставить браузер Internet Explorer 8 поддерживать HTML5-элементы

Язык HTML5 предоставляет в ваше распоряжение много новых элементов. От описывающих характер своего содержимого, например элемента `nav`, и до предоставляющих дополнительные возможности, таких как элемент `video`, который предназначен для вставки видеоконтента, и элемент `audio`, предназначенный для вставки аудиофайла. По мере изучения языка HTML5 вы, возможно, начнете применять эти новые элементы на своих веб-страницах.

К сожалению, Internet Explorer 8 и его более ранние версии не распознают эти новые элементы и не будут реагировать на применяемый к ним код CSS. Действительно, если использовать HTML5 и просматривать веб-страницы в Internet Explorer 8, эта книга не принесет никакой пользы. Но… это не совсем так. Есть один способ включения устаревших версий Internet Explorer в список поддержки, заставляющий их понимать весь CSS-код, применяемый к HTML5-элементам. Нужно просто поместить перед закрывающим тегом `</head>`, находящимся в верхней части кода вашего HTML-файла, следующий фрагмент:

```
<!--[if lt IE 9]> <script src="//html5shiv.
googlecode.com/ svn/trunk/html5.js"></script>
<![endif]-->
```

Этот несколько необычный фрагмент кода представляет собой то, что называется условным комментарием Internet Explorer для вставки кода на языке JavaScript, видимого только версиями Internet Explorer (IE), предшествующими 9-й. Иными словами, на этот код реагируют только IE 6, 7 и 8, а все остальные браузеры (включая более новые версии Internet Explorer) его попросту игнорируют. Этот код инструктирует более ранние версии IE загружать небольшой JavaScript-сценарий, позволяющий браузеру распознавать HTML5-элементы и применять к ним код CSS, предназначенный для этих элементов.

Код влияет только на то, как браузер выводит HTML5-элементы на экране или при печати, и не заставляет браузер «понимать» те HTML5-элементы, которые на самом деле совершают какие-либо действия. Например, IE 8 и более ранние версии браузера не поддерживают элемент `video` и не могут проигрывать HTML5-видеоролики (даже с добавленным JavaScript-сценарием).

Если вы не знаете, нужно ли учитывать поддержку Internet Explorer 8 в вашем проекте, прочитайте врезку «ЧаВо» на следующей странице.

В HTML-коде эти блоки, или структурные единицы, создаются с помощью элемента `div` или одного из структурных HTML5-элементов, например `footer`, `header`, `article` и `aside`. Заключите HTML-элементы, к примеру образующие область баннера, в один `div`-контейнер, HTML-код, формирующий колонки текста, — в другой `div`-контейнер и т. д. Если вы уже разбираетесь в HTML5, то можете создать дизайн, показанный на рис. 1.3, с помощью элемента `header` для верхнего баннера, элемента `article` — для основного текста, элемента `aside` или `section` — для боковой панели и элемента `footer` — для нижнего колонтитула страницы. Иными словами, если вы собираетесь сгруппировать HTML-элементы в какой-либо позиции страницы, их нужно заключить в такие элементы разбиения на разделы, как `div`, `article`, `section` или `aside`.

Как вы узнаете из части III, каскадные таблицы стилей содержат мощные инструменты для работы с макетом страницы. Вы можете буквально поместить HTML-код в любой позиции окна браузера. Новые объекты, например `flexbox` (см. главу 17), дают вам свободу при написании HTML-кода. Однако я все же рекомендую группировать связанный контент в элементах-контейнерах (`div`) или структурном элементе HTML5.

ЧАВО

Стоит ли волноваться насчет Internet Explorer 8?

Я предполагаю, что Internet Explorer 6 уже отслужил свое и о нем больше не стоит беспокоиться. Но так ли это? А как насчет других версий Internet Explorer?

Если вы веб-дизайнер, то, наверное, на вашем компьютере установлены самые последние версии браузеров Internet Explorer, Firefox, Safari, Chrome или Opera. Однако все зависит от вашей аудитории. Возможно, им следует обновить браузер или их компьютеры настолько стары, что не в состоянии работать с новыми версиями программного обеспечения.

К счастью, браузеры Internet Explorer 6 и 7 стремительно уходят в историю, хотя все еще используются в таких странах, как Китай, Индия и Венесуэла (tinyurl.com/q8htudu). Возможно, указанные выше версии установлены на экспонатах в музеях компьютерной истории.

А вот с Internet Explorer 8 по-прежнему нужно считаться. Он не очень популярен, однако, в зависимости от источников исследования, с ним еще работают от 2 до 19 % всех пользователей в мире. Данные об использовании браузеров можно найти на сайтах NetMarketShare (tinyurl.com/8nhrqh3) и GlobalStats StatCounter (gs.statcounter.com).

Но даже статистика, включающая географический регион, в котором находится аудитория вашего сайта, не дает достоверной картины относительно пользователей, предпочитающих этот браузер. Если вы создаете сайт, нацеленный на прогрессивных веб-дизайнеров, вполне вероятно, что он практически никогда не будет просматриваться в программе Internet Explorer 8. Но если ваш сайт предназначен для жителей отдаленных регионов России, то вам, возможно, придется иметь дело с IE 8 (а возможно, даже с IE 6 и 7). Лучше всего определить ту часть трафика, которая приходится на запросы от различных браузеров, путем просмотра журналов веб-серверов или оформить подписку на сервисе Google Analytics (google.com/analytics/), чтобы можно было отслеживать браузеры ваших посетителей (а также многое другое).

Корпорация Microsoft объявила, что прекратит поддержку браузера Internet Explorer 8 в январе 2016 года. В это время пользователи операционной системы Windows должны будут обновиться до более поздних версий программы или перейти к другому браузеру, например Chrome или Firefox. *Основная* проблема браузера Internet Explorer 8 заключается в том, что он не может работать с HTML5-элементами, а это значит, что вам необходимо формировать их непосредственно с помощью каскадных таблиц стилей. Если вы действительно хотите, чтобы ваш сайт правильно отображался в браузере IE 8, то для определения структуры страниц вместо HTML5-элементов, описанных в подразделе «Использование элементов div и span» выше, используйте элемент `div` или JavaScript-сценарий, описанный во врезке «Как заставить браузер Internet Explorer 8 поддерживать HTML5-элементы».

О каких HTML-элементах рекомендуется забыть

Каскадные таблицы стилей позволяют использовать более простой и понятный HTML-код. Вам больше не нужно работать с устаревшими элементами и атрибутами языка HTML для создания и улучшения дизайна веб-страниц. Элемент `font` — наглядный тому пример. Единственная его цель — изменить цвет, размер и начертание шрифта текста страницы. Он не выполняет никакого структурирования и не делает страницу логически более понятной.

Далее приведен список элементов и атрибутов языка HTML, которые вы можете легко, без ущерба для внешнего вида страниц заменить CSS-стилями.

- **Избавьтесь от элемента font для управления форматированием текста.** Каскадные таблицы стилей выполнят это гораздо лучше (о форматировании текста читайте в главе 6).
- **Не используйте элементы b и i для изменения начертания шрифта.** Если вы хотите действительно выделить текст, пользуйтесь элементом `strong` (обычно браузеры отображают текст, выделенный этим элементом, с полужирным начертанием). Если вы хотите придать тексту чуть меньший акцент, то пользуйтесь для его выделения элементом `em` (браузеры выделяют содержимое этого элемента курсивом). Кроме того, с помощью каскадных таблиц стилей можно выделить текст курсивом, сделать его полужирным или и то и другое одновременно.

  В то время как в версии HTML 4 пытались отказаться от использования элементов `b` и `i`, HTML5 вернул их к жизни. В языке HTML5 элемент `b` позволяет сменить начертание шрифта текста на полужирное без семантического выделения. Аналогично элемент `i` используется для смены начертания шрифта текста на курсивное, без подчеркивания его значимости.

СОВЕТ

Чтобы выделить название публикации курсивом, пользуйтесь элементом cite — таким образом вы убьете сразу двух зайцев. Он одновременно выделяет название курсивом и помечает его как цитату для поисковых систем. Конечно, каскадные таблицы стилей позволяют делать с элементами все что угодно, поэтому, если вы хотите сослаться на публикацию, но не выделять ее курсивом, можете также воспользоваться элементом cite.

- **Не пользуйтесь элементом table для компоновки макета страницы.** Применяйте его только с целью отображения табличной информации (например, электронных таблиц, списков, диаграмм). Из части III этой книги вы узнаете, что каскадные таблицы стилей позволяют компоновать макеты веб-страниц гораздо быстрее и с меньшим объемом кода, нежели при использовании элемента `table`.
- **Не злоупотребляйте элементом
.** Если вы привыкли пользоваться им для вставки разрывов строк, не создавая новый абзац, то можете попасть в затруднительное положение (иногда браузеры автоматически добавляют промежуток между абзацами, а также между заголовками и абзацами; раньше разработчики шли сложными обходными путями, чтобы избежать появления этого интервала между абзацами, заменяя единственный элемент `p` несколькими элементами разрыва строки и элементом `font`, чтобы первая строка абзаца была *похожа* на заголовок). Использование свойства `margin` в CSS дает свободу определения таких параметров, и вы с легкостью можете установить интервал между абзацами, заголовками и другими блочными элементами (см. раздел «Управление размерами полей и отступов» главы 7).

ПРИМЕЧАНИЕ

Из главы 5 вы узнаете о технике, называемой сбросом стилей. Она устраняет проблему лишних разрывов строк, вставляемых между абзацами и другими элементами.

В целом добавление в элементы атрибутов, управляющих цветом, границами, фоновыми изображениями, выравниванием текста, форматированием таблиц, — устаревший стиль написания HTML-кода. Сюда также входят атрибуты позиционирования изображений, текста в абзацах и ячейках таблицы. Вместо этого обратитесь к средствам CSS, которые обеспечат выравнивание текста (см. раздел «Форматирование абзацев» главы 6), установку границ (см. раздел «Добавление границ» главы 7), фоновые параметры (см. раздел «Установка цвета фона» главы 7) и позиционирование изображений (см. раздел «Добавление фоновых изображений» главы 8).

В КУРС ДЕЛА!

Проверяйте правильность кода веб-страниц

В языке HTML существуют определенные правила: например, элемент `html` охватывает все содержимое веб-страницы, а элемент `title` должен находиться внутри `head`. Если забыть об этих правилах или просто сделать при наборе опечатку, то *некорректный* HTML-код станет причиной некоторых проблем (например, веб-страница в различных браузерах отобразится по-разному). Более того, даже правильный CSS-код может вместе с ошибочным HTML-кодом работать не так, как ожидалось. К счастью, существуют программные средства для проверки правильности — *валидности* — HTML-кода.

Самый легкий способ проверить HTML-код — выполнить синтаксический контроль (валидацию) на сайте W3C по адресу validator.w3.org (рис. 1.4). Консорциум Всемирной паутины — World-Wide Web Consortium (W3C) — организация, ответственная за определение стандартов веб-технологий и языков программирования, включая HTML и CSS. При нахождении ошибок на ваших веб-страницах W3C-валидатор сообщит о них. Можно либо указать адрес существующей страницы во Всемирной паутине, либо загрузить файл с HTML-кодом на сайт валидатора, либо вставить HTML-код веб-страницы в окно формы на сайте и нажать кнопку запуска проверки.

Расширение Web Developer для браузеров Chrome, Firefox и Opera (tinyurl.com/2353zt) позволит быстро проверить код страницы.

Рекомендации для веб-дизайнеров

Всегда неплохо иметь подробное пошаговое руководство к действию. Если вы все еще не уверены в том, как именно пользоваться языком HTML для создания хорошо структурированных веб-страниц, просмотрите представленные ниже несколько советов для начинающих.

- Используйте заголовки, чтобы указать относительную важность текста. Если два заголовка имеют одинаковую степень важности в теме вашей страницы, то применяйте элемент заголовка одного уровня. Если один из заголовков имеет меньшую значимость, то есть является подтемой другого, применяйте заголовок на уровень ниже. Например, от заголовка `h2` переходите к подзаголовку `h3` (рис. 1.5). Лучше использовать заголовки по порядку и стараться не пропускать их номера, например, никогда не переходите от элемента `h2` сразу к `h5`.
- Используйте элемент `p` для абзацев текста.
- Применяйте маркированные списки (`ul`), если у вас есть перечень связанных элементов, таких как ссылки навигации, содержания, подсказки и др.

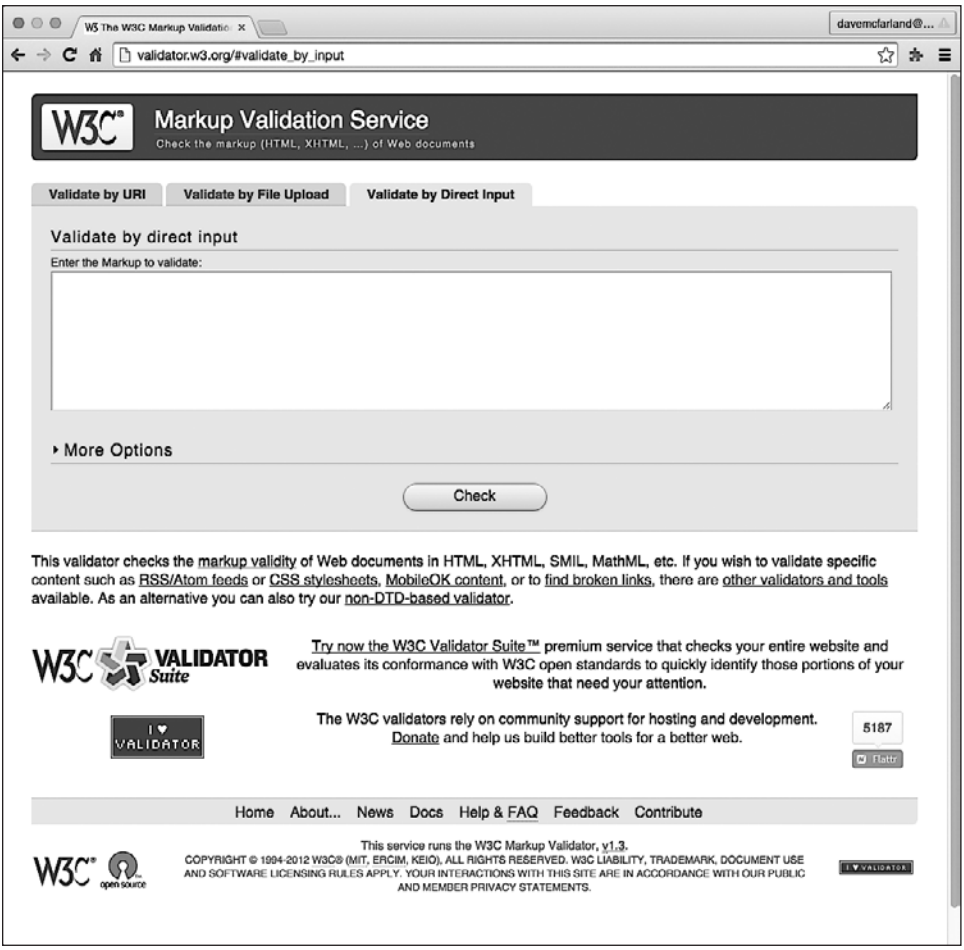

Рис. 1.4. HTML-валидатор Консорциума Всемирной паутины (W3C) быстро проверит правильность кода веб-страницы

- Пользуйтесь нумерованными списками (`ol`), чтобы определить ряд последовательно выполняемых операций или порядок набора элементов. В практикумах этой книги (см. раздел «Практикум: форматирование текста» главы 6) есть наглядный пример такого списка.
- Чтобы создать словарь терминов и их определений, пользуйтесь элементом `dl` (список определений терминов) в сочетании с элементами `dt` (название термина) и `dd` (определение термина). Просмотреть пример использования этой конструкции можно по адресу tinyurl.com/ne888za.
- Если вы хотите включить цитату в виде отрывка текста с другого сайта, чьего-либо высказывания и др., используйте элемент `blockquote` для длинного контекста (высказываний) и элемент `q` — для вставки краткой цитаты в более длинный абзац, например:

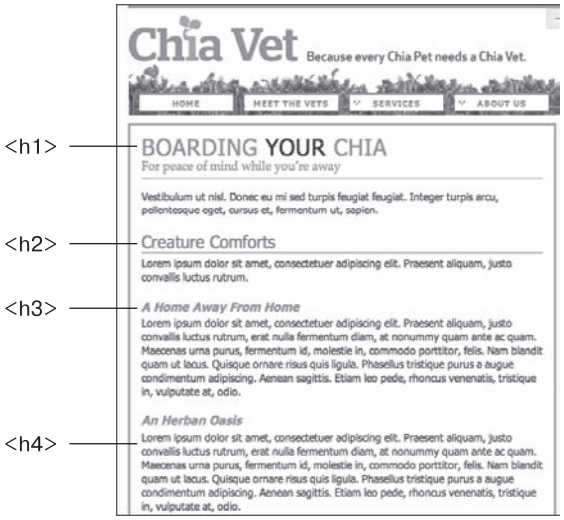

Рис. 1.5. Используйте элементы заголовков (h1, h2 и т. д.) так, как если бы писали доклад: расположите их в тексте в порядке важности. Для самого первого заголовка используйте элемент h1, который должен обозначать «Слушайте! Этот заголовок отражает содержимое всей страницы»

```
<p>Говорят, что Марк Твен как-то раз написал:
  <q>Самой холодной из всех проведенных мною зим было лето
     в Сан-Франциско</q>.
  К сожалению, на самом деле он никогда ничего похожего
  на эту знаменитую цитату не писал.
</p>.
```

- Применяйте такие малоизвестные элементы, как `cite`, чтобы сослаться на книжный заголовок, газетную статью или сайт, а `address` — для обозначения контактной информации автора страницы (удобно применять для указания авторских прав).
- Не используйте элементы или их атрибуты для изменения внешнего вида текста, изображений. Все это выполнят каскадные таблицы стилей.
- Если нет HTML-элемента, соответствующего контенту, который вы хотите выделить на странице для придания ему определенного внешнего вида, то пользуйтесь элементами `div` и `span`. О них я расскажу в следующих главах.
- Не злоупотребляйте элементом `div`. Некоторые дизайнеры полагают, что `div` — это все, что им нужно, и при этом игнорируют элементы, которые могут быть более уместны. Возьмем пример создания навигационной панели. Можно добавить блок `div` на страницу и заполнить его кучей ссылок. Но ведь гораздо лучше использовать маркированный список (элемент `ul`): в конце концов, навигационная панель сама по себе является списком ссылок. Как уже ранее упоминалось, HTML5 предоставляет несколько новых элементов, которыми можно заменить `div`, например `article`, `section` и `footer`. Для панели навигации можно воспользоваться HTML5-элементом `nav`.

- Никогда не забывайте указывать закрывающие теги HTML-элементов. Открывающий тег требует соответствующего ему закрывающего тега , как и любые другие элементы, за исключением одиночных, например
 и <img/>.
- Проверяйте синтаксис своих веб-страниц с помощью W3C-валидатора (см. рис. 1.4 и врезку «В курс дела!» ранее в этой главе). Плохо написанный HTML-код, как и код с опечатками, вызовет множество непредсказуемых ошибок браузера.

Важность объявления типа документа

Язык HTML следует некоторым правилам. Вы сами должны сообщить браузеру, какой версией языка HTML пользуетесь, включив то, что называется *объявлением типа документа*, в начало веб-страницы. Это объявление типа документа указывается в первой строке HTML-файла и определяет используемую вами версию HTML (например, HTML5 или HTML 4.01 Transitional).

Если указать объявление типа документа с ошибкой или опустить его, то браузер переключится в другое состояние, называемое *режимом совместимости*. Режим совместимости — это попытка производителей браузеров заставить свое программное обеспечение вести себя подобно устаревшим браузерам, выпускавшимся ранее 1999 года (Netscape 4, Internet Explorer 5). Если современный браузер сталкивается со страницей, в которой отсутствует правильное объявление типа документа, то он «думает», что эта страница была написана в текстовом редакторе HTML давным-давно, и отображает ее так, как это сделал бы старый браузер. Именно поэтому без правильного объявления типа документа ваши отформатированные с помощью каскадных таблиц стилей веб-страницы, возможно, не будут смотреться так, как они должны выглядеть в соответствии с текущими стандартами. Если, проверяя веб-страницу в браузере, вы невольно просматриваете ее в режиме совместимости, то можете не ломать себе голову, пытаясь исправить проблемы отображения. Они связаны с неправильным объявлением типа документа, а не с ошибками в коде HTML или CSS.

ПРИМЕЧАНИЕ

Для более подробного ознакомления с техническими особенностями работы браузеров в режиме совместимости посетите сайты tinyurl.com/nr9dehz и tinyurl.com/pm8k7ng.

Указать правильное объявление типа документа достаточно просто. Нужно только знать используемую версию HTML. Если используется HTML5, то все существенно упрощается. Объявление типа документа приобретает простой вид:

```
<!doctype html>
```

Поместите этот код в верхнюю часть своего HTML-файла, и все будет в порядке. Если вы по-прежнему пользуетесь старыми версиями HTML или XHTML, например HTML 4.01 Transitional и XHTML 1.0 Transitional, то объявление типа документа будет иметь более запутанный вид.

Если используется версия HTML 4.01 Transitional, то добавляйте приведенное ниже объявление типа документа в самом начале веб-страницы:

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

Объявление типа документа для XHTML 1.0 Transitional имеет похожий вид. Кроме того, необходимо добавить кое-что еще в открывающий тег <html>. Эта строка определяет тип используемого языка XML, в нашем случае это XHTML.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/
TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

Если все это навевает на вас тоску, упростите свою жизнь и воспользуйтесь объявлением типа документа, предлагаемым версией HTML5. Оно имеет краткую форму, легко запоминается, работает во всех браузерах и используется практически во всех новых веб-страницах. Это объявление типа документа можно указать, даже если не применяются никакие новые HTML5-элементы.

ПРИМЕЧАНИЕ

Большинство визуальных средств верстки веб-страниц, в число которых входит программа Dreamweaver, при создании новой веб-страницы автоматически добавляют объявление типа документа, а многие текстовые редакторы, поддерживающие код на языке HTML, предлагают использовать для добавления объявления типа документа специальные сочетания клавиш.

Как работают каскадные таблицы стилей

Каскадные таблицы стилей работают с HTML-кодом, но не имеют никакого отношения к языку HTML. Это совершенно другой язык. HTML структурирует документ, упорядочивая информацию в заголовки, абзацы, маркированные списки и т. д., в то время как CSS тесно взаимодействует с браузером, чтобы оформление HTML-документа имело совершенный *вид*.

Например, вы могли бы использовать HTML, чтобы превратить фразу в заголовок, отделяя его от содержимого страницы, но лучше использовать CSS для форматирования заголовка, скажем, крупным полужирным красным шрифтом с позиционированием на 50 пикселов от левого края окна. В CSS это форматирование текста включает в себя *стиль* — правило, описывающее внешний вид конкретной части веб-страницы. А *таблица стилей* является набором таких стилей.

Можно также создавать стили специально для работы с изображениями. Например, указать формат размещения изображений на странице. С помощью стилей можно выровнять изображение по правому краю веб-страницы, поместить его в цветную рамку, отделить от окружающего текста полем шириной 50 пикселов.

Браузер применяет созданные вами стили для текста, изображений, заголовков и других элементов страницы. Например, вы можете создать стиль и применить его к одному абзацу на странице, чтобы мгновенно изменить размер, цвет и семейство шрифта в этом абзаце. Вы можете создавать стили, которые будут применяться к конкретным HTML-элементам. Так, например, браузер будет одинаково отображать на сайте все заголовки первого уровня (h1) независимо от того, где они расположены. Вы даже можете создавать стили, которые будут применяться только к определенным элементам, отмеченным вами особым образом в HTML-коде.

Создав стиль один раз, можно применять его к текстовым фрагментам, изображениям, заголовкам и любым другим элементам страницы сколько угодно раз. На-

пример, вы можете выбрать абзац текста и применить к нему стиль, тут же изменяющий размер, цвет и шрифт текста. Можно также создать стили для определенных HTML-элементов так, чтобы, например, все заголовки первого уровня (элементы `h1`) на вашем сайте были отображены в одинаковом стиле независимо от того, где они размещены.

В КУРС ДЕЛА!

Версии CSS

Как производители операционных систем и смартфонов iPhone, так и создатели CSS постоянно выпускают новые версии своих программных продуктов. Версия CSS 1, появившаяся в 1996 году, стала основой для дальнейшего совершенствования языка каскадных таблиц стилей. В первой версии языка CSS были введены основные понятия и команды: базовая структура таблиц стилей, концепция селекторов (см. главу 3) и большая часть свойств CSS.

В версии CSS 2 были добавлены новые возможности, включая совместимость с различными принтерами, мониторами и другими устройствами. Здесь также были добавлены новые селекторы и возможность точно позиционировать элементы на веб-страницах.

Следующая версия, CSS 2.1, на сегодняшний день является стандартом языка каскадных таблиц стилей. Она унаследовала все возможности и особенности CSS 1. Сюда добавлены некоторые новые свойства, а также исправлены неточности предыдущих версий. Она не содержит радикальных изменений по сравнению с CSS 2, и большинство браузеров давно адаптированы под эту версию.

Несколько лет назад была выпущена версия CSS3 (с частичной поддержкой браузерами). В отличие от предыдущих версий, CSS3 не является единым стандартом. По мере возрастания сложности CSS Консорциум W3C разбил CSS на несколько отдельных модулей — модуль Selectors, модуль Values and Units, модуль Box Alignment и т. д. Поскольку каждый модуль может разрабатываться независимо от других, какого-то единого стандарта под названием CSS3 не существует. Фактически работа над уровнем 3 модуля Selectors уже завершена и начата работа над уровнем 4.

Иными словами, то, что известно как CSS3, на самом деле является разобщенной коллекцией различных модулей на разных стадиях завершения. Поддержка некоторых новейших модулей уже включена разработчиками браузеров в их продукты, а другие модули в любом отдельно взятом браузере могут не иметь никакой поддержки. В будущем какой-либо стандарт CSS4 не появится, будут только новые версии различных модулей, каждый из которых будет находиться на своем уровне разработки.

Поэтому в данной книге рассматривается ядро CSS 2.1 (которое просто было перенесено в различные модули CSS3), а также наиболее впечатляющие, популярные и широко поддерживаемые новые свойства CSS.

2 Создание стилей и таблиц стилей

Даже самые сложные и красивые сайты (в том числе и тот, который представлен на рис. 2.1) когда-то начинались с определения единственного CSS-стиля. Постепенно, по мере добавления все новых таблиц стилей, можно разработать полностью законченный дизайн сайта. Независимо от того, являетесь вы новичком в изучении CSS или профессиональным веб-дизайнером, всегда следует помнить несколько основных правил создания таблиц стилей. В этой главе мы начнем изучение CSS с основных понятий, которыми нужно руководствоваться в процессе создания и использования таблиц стилей.

СОВЕТ

Многие люди лучше воспринимают материал, обучаясь сразу на конкретных примерах, предпочитая их чтению книг и руководств. Если вы сначала хотите попробовать свои силы в создании таблиц стилей, а затем вернуться к этому теоретическому материалу, чтобы прочитать о том, что только что сами сделали, перейдите к практикуму в конце текущей главы.

Анатомия стиля

Определение стиля в CSS, устанавливающего внешний вид какого-либо элемента (фрагмента) веб-страницы, — это всего лишь правило, которое сообщает браузеру, что и каким образом форматировать: изменить цвет шрифта заголовка на синий, выделить фото красной рамкой, создать меню шириной 150 пикселов для списка гиперссылок. Если бы стиль мог говорить, он сказал бы: «Браузер, сделай, чтобы вот *это* выглядело *так-то*».

Фактически определение стиля состоит из двух основных элементов: самого элемента веб-страницы, который непосредственно подлежит форматированию браузером, — селектора, а также команд форматирования — блока объявления. Селекторами могут быть заголовок, абзац текста, изображение и т. д. Блоки объявления могут, например, окрасить текст в синий цвет, добавить красную рамку (границу) вокруг абзаца, установить фотографию в центре страницы — возможности форматирования бесконечны.

Например, тело сайта, показанного на рис. 2.1, включает простой стиль:

```
body {
  font-family: "Franklin-Book", Helvetica, Arial, sans-serif;
  color: #222;
}
```

Рис. 2.1. Любая отформатированная с помощью CSS веб-страница состоит из отдельных определений стилей

ПРИМЕЧАНИЕ

Как сотрудники Консорциума W3C, так и веб-дизайнеры часто называют CSS-стили правилами. В этой книге оба термина взаимозаменяемы.

Разумеется, CSS-стили не могут быть написаны на обычном языке, как, например, предыдущий абзац. У них есть собственный язык. В частности, чтобы установить красный цвет и размер шрифта 1,5 em для всех абзацев на веб-странице, нужно написать следующее:

```
p { color: red; font-size: 1.5em; }
```

Этот стиль как бы говорит браузеру: «Раскрась текст всех абзацев веб-страницы, заключенных в элемент p, красным цветом и установи размер шрифта равным 1,5 em (*em* — единица измерения размера шрифта текста в браузере, см. главу 6). Любой стиль, даже самый простой, содержит несколько элементов (рис. 2.2).

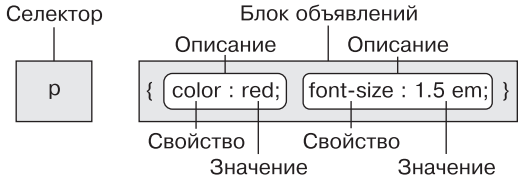

Рис. 2.2. Стиль (или правило) состоит из двух основных частей: селектора и блока объявления

- **Селектор.** Как уже было отмечено, селектор сообщает браузеру, к какому элементу или элементам веб-страницы применяется стиль: к заголовку, абзацу, изображению или гиперссылке. На рис. 2.2 селектор `p` обращается к элементу `p`, передавая браузеру, что все элементы `p` нужно форматировать, используя объявления, указанные в данном стиле. Благодаря большому разнообразию селекторов, предлагаемых языком CSS, и небольшому творческому потенциалу вы сможете мастерски форматировать веб-страницы (в следующей главе селекторы описаны более подробно).
- **Блок объявления.** Код, расположенный сразу за селектором, содержит все команды форматирования, которые можно применить к этому селектору. Блок начинается с открывающей фигурной скобки { и заканчивается закрывающей }.
- **Объявление.** Между открывающей и закрывающей фигурными скобками блока объявления можно добавить одно или несколько *объявлений* — команд форматирования. Каждое объявление имеет две части — *свойство* и *значение*. Двоеточие отделяет имя свойства от его значения, и все объявление заканчивается точкой с запятой.
- **Свойство.** Каскадные таблицы стилей предлагают большой выбор команд форматирования, называемых *свойствами*. Свойство представляет собой слово или несколько написанных через дефис слов, определяющих конкретный стиль. У большинства свойств есть соответствующие простые для понимания имена, такие как `font-size`, `margin-top`, `background-color` и т. д. (в переводе с английского: размер шрифта, верхний отступ, цвет фона). В следующих главах будет описано множество полезных свойств CSS. После имени свойства нужно добавить двоеточие, чтобы отделить его от значения.

СОВЕТ

В приложении 1 вы найдете удобный перечень свойств CSS.

- **Значение.** Наконец настал тот момент, когда вы можете задействовать свой творческий потенциал, присваивая *значения* CSS-свойствам: к примеру, устанавливая фоновый цвет синим, красным, фиолетовым, салатовым и т. д. Как будет описано в следующих главах, различные свойства каскадных таблиц стилей требуют указания определенных типов значений — цвета (`red` или `#FF0000`), размера (`18px`, `200%` или `5em`), URL-адреса (`images/background.gif`), а также определенных ключевых слов (`top`, `center` или `bottom`).

Вам не обязательно описывать стиль на одной строке, как изображено на рис. 2.2. У стилей может быть множество свойств форматирования, и есть возможность упорядочить код таблицы стилей, разбивая объявления на строки. Например, поместите селектор и открывающую скобку на одной строке, каждое объявление — далее на отдельных строках, а закрывающую фигурную скобку — отдельно на последней строке стиля. Это будет выглядеть следующим образом:

```
p {
  color: red;
  font-size: 1.5em;

}
```

Любой браузер игнорирует символы пробела и табуляции, так что вы можете спокойно добавлять их, создавая хорошо читаемые стили CSS. Кроме того, при перечислении свойств полезно сделать отступ табуляцией или несколькими пробелами для явного отделения селектора от блока объявления. К тому же один пробел между двоеточием и значением свойства, конечно, необязателен, но он обеспечивает дополнительную удобочитаемость стилей. Фактически можно добавить любое количество пробелов там, где вам захочется. Например, `color:red`, `color: red` и `color : red` — все варианты будут правильно работать.

ПРИМЕЧАНИЕ

Не забывайте указывать в конце каждой пары «свойство/значение» точку с запятой:

```
color: red;
```

Пропуская эту точку с запятой, вы собьете с толку браузер, в результате чего таблица стилей будет нарушена и веб-страница отобразится некорректно.

Однако не стоит переживать — описанная ошибка достаточно распространена, поэтому используйте валидатор каскадных таблиц стилей, о котором будет рассказано далее в этой главе.

Концепция таблиц стилей

Конечно, один стиль не превратит веб-страницу в произведение искусства. Он может выделить абзацы красным цветом, но, чтобы придать сайту красивый и стильный внешний вид, вам придется указать множество различных стилей. Весь набор определяемых CSS-стилей включается в *таблицу стилей*. Таблицы стилей бывают двух видов — *внутренние* и *внешние* — в зависимости от того, где определена информация о стилях: непосредственно в самой веб-странице или в отдельном файле, связанном с веб-страницей.

Внутренние или внешние таблицы стилей? Еще с момента изобретения CSS внешние таблицы стилей были лучшим способом создания дизайна веб-страниц. Они упрощают создание и поддержку сайтов и ускоряют их обновление. Внешняя таблица стилей сосредотачивает всю информацию о стилях в едином файле, который вы затем присоединяете к странице, написав для этого всего одну строку кода. Вы можете присоединить одну и ту же внешнюю таблицу стилей к каждой странице сайта, создавая таким образом единый дизайн. А обновление внешнего вида всего сайта будет заключаться лишь в редактировании единственного текстового файла — внешней таблицы стилей.

Внешние таблицы стилей позволяют веб-страницам загружаться быстрее. Когда вы используете такие таблицы стилей, веб-страницы содержат только HTML-код, без кода громоздких вложенных таблиц для форматирования, без элементов `font` и подобных, без кода внутренних каскадных таблиц стилей. Кроме того, когда браузер загрузит внешнюю таблицу стилей, он сохранит этот файл на клиентском компьютере посетителя веб-страницы (в специальной системной папке, называемой *кэшем*) для быстрого доступа к нему. Когда посетитель веб-страницы переходит к другим страницам сайта, которые используют ту же внешнюю таблицу стилей, браузеру нет необходимости снова загружать

таблицу стилей. Он попросту загружает запрашиваемый HTML-файл и использует внешнюю таблицу стилей из своего кэша, что дает существенный выигрыш во времени загрузки страниц.

ПРИМЕЧАНИЕ

Когда вы работаете над своим сайтом и пользуетесь предварительным просмотром в браузере, кэш работает без всякой пользы для вас. Подробнее об этом рассказано во врезке «Обходной прием» далее в этой главе.

ОБХОДНОЙ ПРИЕМ

Не попадитесь с кэшированием

Кэш браузера обеспечивает значительное увеличение скорости просмотра веб-страниц.

Всякий раз, когда браузер открывает веб-страницу, он помещает загруженную информацию в кэш. Сюда также попадают такие часто используемые файлы, как внешние каскадные таблицы стилей или изображения, что позволяет сэкономить время загрузки веб-страниц. Вместо того чтобы в следующий раз (при просмотре других страниц этого же сайта или повторного просмотра этих страниц в будущем) повторно загружать те же файлы, браузер может использовать их прямо из кэша, будь то просмотренная ранее веб-страница или рисунок.

Однако не всегда то, что хорошо для посетителей сайта, удобно для вас. Поскольку браузер кэширует файлы внешних таблиц стилей CSS и повторно обращается к ним, это часто сбивает с толку, например во время работы над дизайном сайта. Допустим, работая над форматированием веб-страницы, которая использует внешние таблицы стилей, вы просматриваете ее в браузере, чтобы убедиться, что достигли желаемого результата. Но не все выглядит так, как было задумано, хотя, на ваш взгляд, в CSS-коде ошибок нет. Вы возвращаетесь в редактор HTML-кода и вносите изменения в файл внешних таблиц стилей CSS. Снова вернувшись в браузер и перезагрузив страницу для просмотра результатов только что внесенных изменений, вы видите, что никаких изменений не произошло! Вы только что попали в ловушку, связанную с особенностями кэша. Дело в том, что при перезагрузке веб-страницы браузер не всегда перезагружает данные, уже находящиеся в кэше, в том числе внешние таблицы стилей. Таким образом, невозможно увидеть, как выглядит веб-страница, отформатированная с помощью только что отредактированного CSS-кода из внешней таблицы стилей.

Чтобы обойти эту путаницу, можно выполнить *принудительную перезагрузку* страницы (вместе с перезагрузкой всех связанных файлов), нажав и удерживая клавишу Ctrl (⌘), а затем нажав кнопку Обновить (Reload) в окне браузера. Кроме того, в браузерах Chrome и в Internet Explorer для этой цели предназначено сочетание клавиш Ctrl+F5 (⌘+F5), в Firefox — Ctrl+Shift+R (⌘+Shift+R), а Ctrl+R (⌘+R) работает как в Safari, так и в Chrome.

Внутренние таблицы стилей

Внутренняя таблица стилей — это набор стилей, являющийся частью кода веб-страницы, которая всегда должна находиться между открывающим и закрывающим тегами `<style>` и `</style>` HTML-кода в теле веб-страницы, то есть внутри элемента `head`. Например:

```
<style>
h1 {
  color: #FF7643;
```

```
  font-family: Arial;
}
p {
  color: red;
  font-size: 1.5em;
}
</style>
</head>
<body>

<!-- Далее следует остальная часть вашей веб-страницы... -->
```

ПРИМЕЧАНИЕ

Вы можете поместить элемент style и все его стили после элемента title, но веб-дизайнеры обычно размещают их прямо перед закрывающим тегом </head>, как показано в примере выше. Однако если вы также используете на страницах JavaScript-код, он должен располагаться после таблиц стилей. Часто JavaScript-сценарии полагаются на CSS, поэтому, добавляя таблицы стилей первыми, вы гарантируете, что JavaScript-код будет иметь все необходимые для своего выполнения данные.

Элемент `style` относится к HTML, а не к CSS, но именно он сообщает браузеру, что данные, содержащиеся внутри, являются кодом CSS, а не HTML. Создание внутренней таблицы стилей идентично созданию внешней, с той лишь разницей, что перечень стилей не выносится в отдельный файл, а заключается между тегами элемента `style`.

Внутренние таблицы стилей можно легко добавить в веб-страницу, и так же просто перейти к редактированию HTML-кода этой же веб-страницы. Однако эти таблицы стилей отнюдь не являются самым эффективным способом для проектирования дизайна сайта, состоящего из множества страниц. С одной стороны, вам придется копировать и вставлять код внутренней таблицы стилей в каждую страницу сайта, а это не только трудоемкая, но еще и бессмысленная работа. Этот код делает каждую страницу вашего сайта громоздкой. К тому же такая страница медленно загружается.

Кроме того, внутренние таблицы стилей доставляют много трудностей при обновлении дизайна сайта. Например, нужно изменить представление заголовков первого уровня (заключенных в элемент `h1`), которые первоначально отображались крупным полужирным шрифтом зеленого цвета. Теперь вы хотите, чтобы заголовки были написаны маленьким шрифтом Courier синего цвета. Используя внутренние таблицы стилей, пришлось бы редактировать *каждую* страницу сайта. У кого-нибудь найдется столько времени? К счастью, для устранения данной проблемы нашлось простое решение — использование внешних таблиц стилей.

ПРИМЕЧАНИЕ

Иногда можно прибегнуть к способу добавления информации о стилях непосредственно к каждому конкретному HTML-элементу без применения таблицы стилей. В практикуме этой главы будет показано, как выполнить такой маневр, используя встроенные стили. Мы не рекомендуем применять встроенные стили для разработки веб-страниц, однако многие программисты JavaScript используют их для динамического добавления HTML-контента на страницы. Это яркий пример создания и работы встроенных стилей.

Внешние таблицы стилей

Внешняя таблица стилей — это не что иное, как текстовый файл, содержащий весь набор стилей CSS. Он не должен включать в себя HTML-код, поэтому никогда не добавляйте в файл внешней таблицы стиля элемент `style`. Вдобавок имя этого файла всегда должно заканчиваться расширением CSS. Можно присвоить какое угодно имя этому файлу, но лучше, чтобы оно было информативным. Назовите файл внешней таблицы стилей, например, `global.css`, `site.css` или просто `styles.css`, если это общая таблица стилей, связанная со всеми страницами вашего сайта, или `form.css`, если он содержит код для форматирования веб-форм сайта.

В КУРС ДЕЛА!

Проверяйте правильность CSS-кода

Так же как вы должны были удостовериться в правильности написания HTML-кода веб-страниц (см. врезку «В курс дела!» в главе 1), проверьте CSS-код на отсутствие ошибок. На сайте Консорциума доступен инструмент для проверки синтаксиса кода CSS: tinyurl.com/382hc. Он работает, как и HTML-валидатор: можно указать URL-адрес веб-страницы (или только адрес к внешнему CSS-файлу), загрузить CSS-файл или скопировать и вставить код в форму, а затем нажать кнопку запуска проверки.

При наборе CSS-кода очень просто сделать опечатку или ошибку, которой вполне достаточно для того, чтобы изменить до неузнаваемости весь тщательно продуманный дизайн страниц сайта. Если веб-страница, содержащая CSS-код, выглядит не так, как вы ожидали, то причиной тому может быть небольшая ошибка в коде. CSS-валидатор Консорциума — первое средство поиска проблем с дизайном веб-страниц.

СОВЕТ

Если имеется веб-страница с внутренней таблицей стилей, а вы хотите использовать внешнюю таблицу, то вырежьте код стилей, расположенный между тегами элемента style (без самих тегов). Потом создайте новый текстовый файл и вставьте в него CSS-код. Сохраните файл с расширением CSS, например global.css, и свяжите его с вашей веб-страницей, как описано далее.

Создав внешнюю таблицу стилей, вы должны подключить ее к форматируемой веб-странице. Это можно сделать с помощью HTML-элемента `link`:

```
<link rel="stylesheet" href="css/styles.css">
```

Элемент `link` обладает двумя атрибутами:

- `rel="stylesheet"` — указывает тип ссылки; в данном случае это ссылка на таблицу стилей;
- `href` — определяет местонахождение внешнего CSS-файла на сайте. Значение этого атрибута — URL-адрес, который будет отличаться в зависимости от того, где расположен CSS-файл. Он работает так же, как атрибут `src` при добавлении изображения на страницу или атрибут `href` гиперссылки, указывающей на другую веб-страницу.

СОВЕТ

На одну и ту же страницу вы можете добавить несколько элементов link, указывающих на разные каскадные таблицы стилей. Способы организации стилей описаны в главе 18.

Практикум: создание стилей

В этом разделе речь пойдет об основных приемах создания CSS-стилей, в том числе встроенных, а также внутренних и внешних таблиц стилей. По мере прочтения книги на практических примерах вы научитесь создавать различные CSS-стили, от простых элементов дизайна до полноценных CSS-ориентированных макетов веб-страниц. Перед началом урока нужно загрузить файлы с материалом для выполнения заданий практикума, расположенные по адресу github.com/mrightman/css_4e. Перейдите по ссылке и загрузите ZIP-архив с файлами (нажав кнопку Download ZIP в правом нижнем углу страницы). Файлы каждой главы находятся в отдельных папках с названиями 02 (для второй главы), 03 (для третьей) и т. д.

ПРИМЕЧАНИЕ

Кроме папок с заданиями к каждой главе, вы обнаружите другие папки с уже выполненными заданиями. Например, в папке 02_finished находятся файлы готовых заданий к главе 2. Используйте их в качестве подсказки, если запутались и хотите сравнить свой результат с конечным.

Затем следует запустить редактор HTML-кода, которым вы пользуетесь, будь то простой текстовый редактор (Блокнот (Notepad) или TextEdit) или программный комплекс для визуального проектирования (Sublime Text, Notepad++ или Adobe Dreamweaver).

Создание встроенного стиля

Размещая код CSS непосредственно в HTML-коде страницы, вы создаете встроенный стиль. Встроенные стили не экономят ни время загрузки веб-страниц, ни трафик, поэтому нет никаких причин для их использования. В крайнем случае, если обязательно нужно изменить стиль единственного элемента одной веб-страницы, можно прибегнуть к встроенным стилям. (Например, создавая HTML-форматированные электронные письма, лучше использовать встроенные стили. Это, к слову, единственная возможность заставить CSS работать в Gmail.) Если вы все-таки применяете этот метод и хотите, чтобы стиль работал должным образом, то уделите особое внимание размещению команд форматирования внутри элементов, которые следует отформатировать. Рассмотрим пример, наглядно демонстрирующий, как это делается.

1. В редакторе HTML-кода откройте файл `02\index.html`.

   Этот простой и изящно написанный HTML5-файл содержит несколько заголовков, абзац, маркированный список и информацию об авторском праве в элементе `address`. Начнем с создания встроенного стиля для элемента `h1`.

2. В открывающем элементе `<h1>` укажите свойство стиля `style="color: #6A94CC;"`.

   Элемент должен выглядеть следующим образом:

```
<h1 style="color: #6A94CC;">
```

Атрибут `style` относится к HTML, а не к CSS, поэтому после него указывается символ =, а значение атрибута — весь код CSS — заключено в кавычки. Код CSS — это только та часть, что находится в кавычках. В данном случае вы добавили свойство `color`, которое воздействует на цвет текста, и установили его значение равным `#6A94CC`, то есть шестнадцатеричному коду, определяющему синий цвет (об изменении цвета текста читайте в главе 6). Двоеточие отделяет имя свойства от значения, которое вы хотите установить для данного свойства. Далее проверим результат в браузере.

3. Откройте страницу `index.html` в браузере.

   Запустите любой браузер и выберите команду меню **Файл ▸ Открыть** (File ▸ Open) или нажмите сочетание клавиш **Ctrl+O** и выберите файл `index.html` в папке **02** с файлами примеров на своем компьютере (вы можете просто перетащить этот файл из окна папки в открытое окно браузера). Во многих HTML-редакторах доступна функция **Предварительный просмотр в браузере** (Preview in Browser), которая при нажатии определенного сочетания клавиш или выборе пункта меню открывает страницу для просмотра в браузере. Изучите справочные материалы к HTML-редактору: возможно, в нем есть команда, которая сэкономит ваше время. Открыв страницу в браузере, вы увидите, что заголовок стал синим. Встроенные стили могут содержать несколько свойств CSS. Добавим в тег еще одно свойство.

4. Вернитесь в HTML-редактор. После точки с запятой за кодом `#6A94CC` наберите код `font-size: 3em;`.

   Точка с запятой отделяет два различных свойства. Тег `<h1>` должен выглядеть следующим образом:

   ```
   <h1 style="color: #6A94CC; font-size: 3em;">
   ```

5. Посмотрите на страницу в браузере. Нажмите кнопку **Обновить** (Reload), но сначала удостоверьтесь, что сохранили HTML-файл.

   Теперь заголовок стал гораздо больше. Вы почувствовали, как непросто добавлять встроенные стили? Придание соответствующего вида всем заголовкам `h1` веб-страницы требует выполнения тех же действий над каждым из них. На это могут уйти часы и даже целые дни набора и добавления кода в ваш HTML-файл.

6. Вновь перейдите в редактор веб-страниц и удалите из элемента `h1` весь код атрибута форматирования, вернув его к нормальному виду.

Далее мы создадим таблицу стилей внутри веб-страницы (окончательная версия этой части практикума представлена файлом `inline-style.html` в папке **02_finished**).

Создание внутренней таблицы стилей

Вместо встроенных стилей лучше использовать таблицу стилей, содержащую множество стилей CSS, каждый из которых придает внешний вид своему элементу страницы. Прочитав этот раздел, вы научитесь создавать стиль, который изменяет внешний вид всех заголовков первого уровня за один прием. Этот единственный стиль автоматически отформатирует все элементы `h1` веб-страницы.

1. В файле `index.html`, открытом в текстовом редакторе, установите курсор сразу после закрывающего тега `</title>`, нажмите клавишу Enter и наберите код `<style>`.

   Теперь HTML-код должен выглядеть следующим образом (текст, который нужно добавить, выделен полужирным шрифтом):

   ```
   <title>Большая книга CSS -- Глава 2</title>
   <style>
   </head>
   ```

   Открывающий тег `<style>` указывает на начало таблицы стилей. Желательно сразу же, как только вы набираете открывающий тег, закрывать его, поскольку об этом очень легко забыть, переключив внимание на написание CSS. В данном случае вы закроете элемент `style` до того, как станете добавлять стили CSS.

2. Нажмите клавишу Enter дважды и наберите код `</style>`.

   Теперь вы добавите селектор CSS, обозначающий начало вашего первого стиля.

3. Щелкните кнопкой мыши между открывающим и закрывающим тегами элемента `style` и введите `h1 {`.

   Значение `h1` определяет элемент, к которому браузер должен применить последующий стиль.

   Фигурную скобку после `h1` называют *открывающей*. Она обозначает начало определения CSS-свойств для данного стиля. Иначе говоря, за ней начинается самое интересное. Надо отметить, что, как и в случае с закрывающими тегами, желательно указывать *закрывающую скобку* стиля до непосредственного добавления каких-либо свойств этого стиля.

4. Дважды нажмите клавишу Enter и укажите одну закрывающую скобку `}`.

   Ответная закрывающая скобка, соответствующая введенной в предыдущем шаге открывающей, должна сообщить браузеру, что этот CSS-стиль здесь заканчивается.

5. Перейдите к пустой строке между двумя скобками, нажмите клавишу Tab и введите код `color: #6A94CC;`.

   Это текст того же свойства стиля, что и во встроенном варианте, — свойство `color` со значением `#6A94CC`. Точка с запятой обозначает окончание объявления свойства.

ПРИМЕЧАНИЕ

Исходя из синтаксиса языка CSS, не обязательно размещать каждое свойство стиля на отдельной строке, но это в ваших же интересах. Построчный набор свойств облегчает просмотр таблицы стилей и позволяет визуально выделить каждое свойство. Кроме того, есть еще правило оформления, рекомендуемое для лучшего представления структуры CSS-кода, — использование табуляции (вместо нее можно также добавлять несколько пробелов). Такое смещение строк кода обеспечивает быстрый и понятный просмотр таблиц стилей, выстраивая селекторы (в данном примере в качестве селектора выступает h1) в одну линию вдоль левого края страницы и располагая свойства на одинаковом уровне со смещением вправо.

6. Снова нажмите клавишу Enter и добавьте дополнительно два свойства, как показано ниже:

```
font-size: 3em;
margin: 0;
```

Убедитесь в том, что вы не забыли указать точку с запятой в конце каждой строки, иначе код CSS некорректно отобразится в браузере.

Каждое из этих свойств придает заголовку определенный визуальный эффект. Первое свойство назначает размер и шрифт текста, в то время как второе удаляет воздух (пустое пространство) вокруг заголовка. Более подробно об этих свойствах читайте во второй части книги.

Поздравляю — вы только что создали внутреннюю таблицу стилей! Код, который вы добавили на веб-страницу, должен выглядеть так, как выделенный ниже полужирным шрифтом:

```
<title>Большая книга CSS – Глава 2</title>
<style>
h1 {
  color: #6A94CC;
  font-size: 3em;
  margin: 0;
}
</style>
</head>
```

7. Сохраните страницу и просмотрите ее в браузере.

Вы можете сделать это так, как описано в третьем шаге, или, если страница все еще открыта в окне браузера с предыдущего раза, нажмите кнопку Обновить (Reload).

Теперь мы добавим другой стиль.

ПРИМЕЧАНИЕ

Никогда не забывайте добавлять закрывающий тег </style> в конце внутренней таблицы стилей. Если вы не сделаете этого, браузер отобразит на экране код таблицы стилей, за которым последует сама веб-страница без всякого форматирования, а может быть и такое, что браузер вообще не покажет содержимого веб-страницы.

8. Переключитесь обратно в HTML-редактор, установите курсор после закрывающей фигурной скобки стиля h1, который вы только что создали, нажмите клавишу Enter и добавьте следующий стиль:

```
p {
  font-size: 1.25em;
  color: #616161;
  line-height: 150%;
  margin-top: 10px;
  margin-left: 60px;
}
```

Этот стиль форматирует все абзацы веб-страницы. Не переживайте, что пока не знаете, что делает каждое из описываемых свойств CSS. В последующих главах эти свойства будут подробно описаны. А пока просто потренируйтесь правильно набирать код и прочувствуйте, каково это — добавлять CSS на страницу.

9. Просмотрите страницу в браузере.

   Страница выглядит так, как показано на рис. 2.3. Вы видите, как изменился стиль абзаца под первым заголовком? Вы можете посмотреть окончательную версию этой части примера, открыв файл `internal-stylesheet.html` из папки 02_finished.

   Все то, чем вы занимались в практикуме, можно назвать «CSS в двух словах»: начать с HTML-страницы, добавить таблицу стилей, создать прочие CSS-стили, чтобы заставить страницу прилично выглядеть. В следующей части практикума вы увидите, как можно более эффективно работать, используя внешние таблицы стилей.

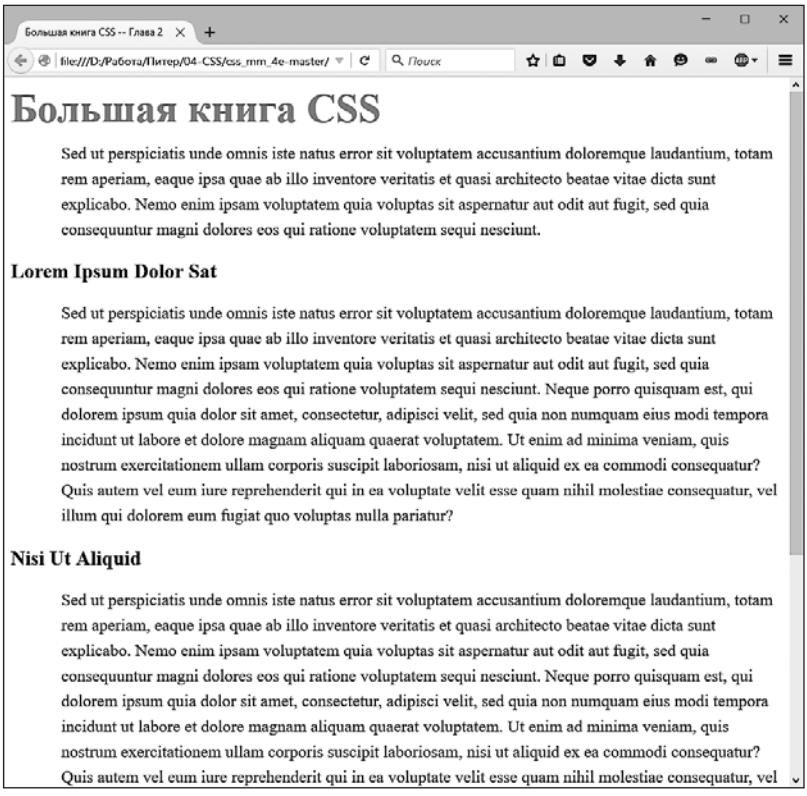

Рис. 2.3. Каскадные таблицы стилей легко справляются с форматированием текста, позволяя изменять начертание, размер, цвет шрифтов текста и даже добавлять декоративные рамки и подчеркивание и многое другое, о чем вы узнаете в главе 6