Терри Фельке-Моррис

Большая книга веб-дизайна

- *JavaScript* • *CSS3* • *HTML5*
- *Поисковая оптимизация*
- *Веб-разработка для мобильных устройств*

УДК 004.42
ББК 32.973.26
Ф 39

Фельке-Моррис Т.

Ф 39 Большая книга веб-дизайна / Терри Фельке-Моррис ; пер. с англ. Н. А. Райтмана. – М. : Эксмо, 2012. – 608 с. + 1 CD. – (Мировой компьютерный бестселлер).

ISBN 978-5-699-55404-1

За HTML5 и CSS3 будущее Всемирной паутины, и «Большая книга веб-дизайна» расскажет вам обо всем, что необходимо знать для овладения этими инновационными стандартами.

Читая книгу, вы изучите современный мир Всемирной паутины, рассмотрите приемы подготовки успешных и доступных веб-сайтов, научитесь верстке на языке HTML5 с использованием всех новых элементов стандарта. Вы также освоите каскадные таблицы стилей (CSS), с помощью которых сможете создать самые невероятные веб-сайты. Познакомившись с языком JavaScript, вы научитесь создавать простые сценарии, превращающие ваши страницы в интерактивные многофункциональные ресурсы. А изучив главы, посвященные электронной коммерции и поисковой оптимизации, научитесь зарабатывать на своем сайте деньги.

УДК 004.42
ББК 32.973.26

ISBN 978-5-699-55404-1

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Книга «Большая книга веб-дизайна» рекомендована для использования в качестве начального курса изучения веб-разработки. Курс охватывает основы, необходимые веб-разработчикам для развития необходимых навыков:

- концепции Интернета;
- создание веб-страниц с помощью HTML5;
- изменение текста, цвета и макета страницы с помощью каскадных таблиц стилей (CSS);
- использование эффективных методов веб-дизайна;
- процесс веб-разработки;
- использование мультимедийных и интерактивных средств на веб-страницах;
- знание свойств CSS3;
- продвижение веб-сайта и поисковая оптимизация;
- электронная коммерция и Всемирная паутина;
- основы JavaScript.

В качестве дополнения к данной книге вы найдете «Справочник веб-разработчика», который представляет собой набор приложений, включающих: справочник HTML5, список специальных HTML-символов, справочник CSS-свойств и руководство по приемам передачи файлов по протоколу FTP.

Это издание включает темы, касающиеся HTML5 и CSS3, такие как форматирование текста, цвета и макета страницы, фокусируясь на вопросах дизайна и веб-стандартов.

Книга освещает современные техники разработки веб-страниц, которые будут интересны всем веб-дизайнерам, желающим освоить язык разметки HTML5. Синтаксис XHTML также упоминается, но акцент сделан именно на HTML5.

Текст книги, а также примеры, в максимальной мере адаптированы для российской аудитории.

Файлы примеров доступны на диске, прилагающемся к книге. Эти файлы включают решения практических заданий.

Организация текста

Книга разработана таким образом, что может быть легко адаптирована для различных потребностей пользователей. Глава 1 содержит вступительный материал, который может быть пропущен или изучен, в зависимости от уровня подготовки пользователей. Главы 2–4 знакомят с приемами верстки кода HTML и CSS. Глава 5 описывает эффективные методы веб-дизайна и может быть изучена сразу после главы 3 (или даже одновременно с главой 3). Главы 6–9 продолжают изучение HTML и CSS.

Любая из перечисленных ниже глав может быть пропущена или использована для отдельного изучения, в зависимости от ограниченности времени и потребностей пользователей: глава 7 (Дополнительные сведения о ссылках, макетах и мобильных устройствах), глава 10 (Разработка веб-сайта), глава 11 (Мультимедийные и интерактивные элементы на веб-страницах), глава 12 (Электронная коммерция), глава 13 (Продвижение сайта) и глава 14 (Краткий обзор JavaScript). Взаимосвязь между главами показана на рис. В1.

Краткий обзор каждой главы

Глава 1: Интернет и Всемирная паутина. Это краткое вступление охватывает темы и концепции, связанные с Интернетом и Всемирной паутиной, которые необходимо знать веб-разработчикам. Глава 1 предоставляет базовые знания, на которых построена вся остальная книга.

Глава 2: Основы разметки веб-страниц. Пользователи знакомятся с языком разметки HTML, примеры и упражнения помогают им создавать образцы страниц и получать полезный опыт. Необходимые страницы для выполнения практических заданий этой и последующих глав доступны на диске, прилагающемся к книге.

Глава 3: Изменение цвета и текста с помощью CSS. Знакомство с техникой использования каскадных таблиц стилей (Cascading Style Sheets) для создания страниц. Пользователи создают страницы по мере прочтения книги.

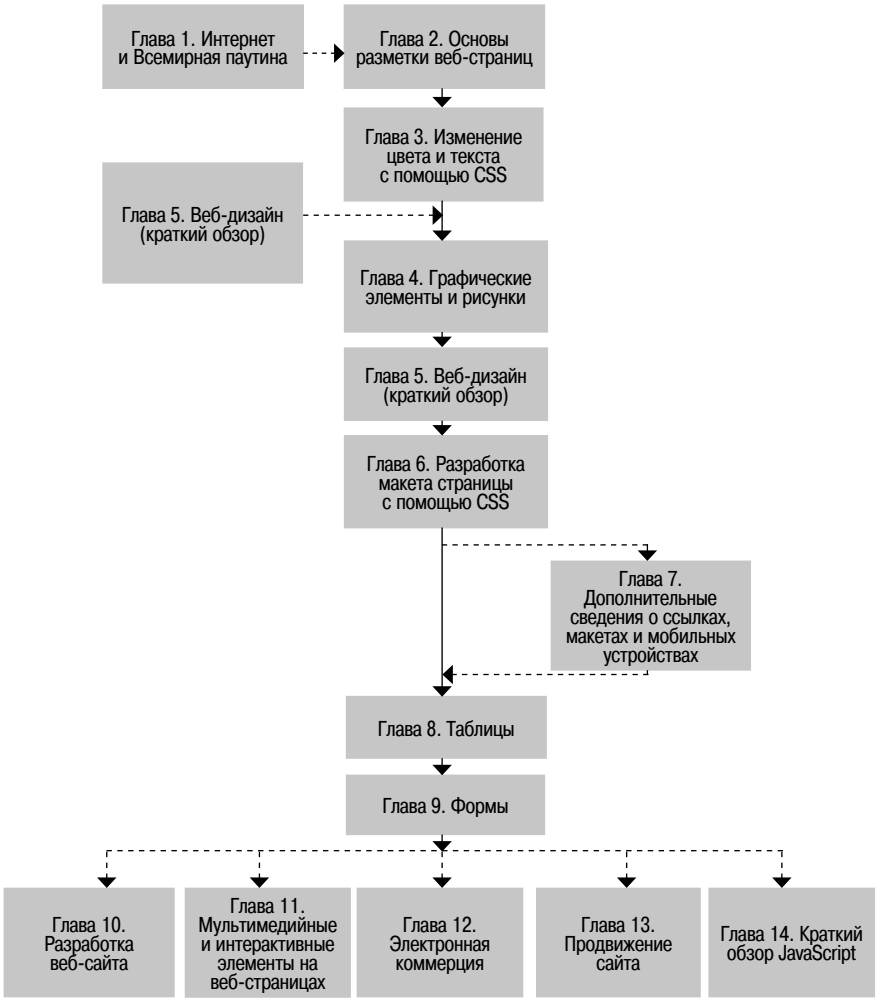


Рис. В.1. Книга может быть адаптирована к индивидуальным потребностям пользователя

Глава 4: Графические элементы и рисунки. Эта глава рассматривает приемы использования визуальных эффектов и графических изображений на веб-страницах. Пользователи выполняют задания по созданию страниц по мере прочтения книги.

Глава 5: Веб-дизайн. Эта глава сфокусирована на рекомендуемых методах дизайна веб-страниц. Частично это закрепление навыков, потому что советы по рекомендуемым методам дизайна веб-сайтов включены в другие главы.

Глава 6: Разработка макета страницы. Эта глава продолжает процесс изучения CSS, начатый ранее, и знакомит с методами позиционирования и обтекаемыми элементами веб-страницы, в том числе двухколоночным макетом CSS. Новые семантические элементы HTML5 также представлены.

Глава 7: Дополнительные сведения о ссылках, макетах и мобильных устройствах. Эта глава возвращается к рассмотренным ранее темам и знакомит с более продвинутыми методами, связанными с гиперссылками, спрайтами CSS, трехколоночным макетом страницы CSS, предпечатной настройкой страниц посредством CSS и оптимизацией веб-сайтов для мобильных устройств. Пользователи создают страницы по мере прочтения книги.

Глава 8: Таблицы. Эта глава сосредоточена на HTML-элементах, используемых для создания таблиц. Рассматриваются методы конфигурации таблиц с помощью CSS. Пользователи создают страницы по мере прочтения книги.

Глава 9: Формы. Эта глава сосредоточена на HTML-элементах, используемых для создания форм. Рассматриваются методы конфигурации форм с помощью CSS. Элементы HTML5 и значения их атрибутов для управления формами также рассматриваются. Пользователи создают страницы по мере прочтения книги.

Глава 10: Разработка веб-сайта. Изучение процесса разработки веб-сайта включает список специализаций, необходимых для создания масштабного проекта, процесс веб-разработки и веб-хостинга. В эту главу включен контрольный список веб-хостинга.

Глава 11: Мультимедийные и интерактивные элементы на веб-страницах. Эта глава освещает темы, связанные с добавлением мультимедийных и интерактивных элементов на веб-страницы. Рассматриваются новые возможности по внедрению видео- и аудиообъектов посредством HTML5, технология Flash®, апплеты Java™, графические галереи CSS, новые свойства CSS для построения трансформаций и переходов, JavaScript и Ajax. Пользователи создают страницы по мере прочтения книги.

Глава 12: Электронная коммерция. Эта глава знакомит с приемами электронной коммерции, обеспечения безопасности и обработки заказов во Всемирной паутине.

Глава 13: Продвижение сайта. Эта глава рассматривает продвижение сайта с точки зрения веб-разработчика и основы поисковой оптимизации.

Глава 14: Краткий обзор JavaScript. Эта глава знакомит с написанием клиентских скриптов на языке JavaScript.

Приложения *«Справочник веб-разработчика»*: справочник HTML5, список специальных символов HTML, справочник свойств CSS и руководство по приемам передачи файлов по протоколу FTP.

Особенности книги

Всеобъемлющий выбор тем. Эта книга включает как уроки верстки HTML5, CSS и JavaScript (главы 2–4, 6–9 и 14), так и полезные сведения по веб-дизайну (глава 5), продвижению веб-сайта (глава 13) и электронной коммерции (глава 12). Эта всесторонняя структура поможет пользователям в построении карьеры веб-профессионалов.

Практические задания

Веб-разработка — это навык, а навыки лучше всего приобретаются с помощью практических заданий. В этой книге практические задания представлены упражнениями, включенными в главы.

ЧаВо

Часто задаваемые вопросы. На курсах веб-разработки пользователи часто задают верстальщику веб-страниц похожие вопросы. Они включены в эту книгу и обозначены опознавательным логотипом **ЧаВо**.

Справочные материалы. Приложения «Справочника веб-разработчика» включают справочные материалы по языку HTML5, список специальных символов, справочник свойств CSS и руководство по приемам передачи файлов по протоколу FTP.

Благодарности

Особая признательность людям в компании Pearson: Майклу Хиршу (Michael Hirsh), Мэтту Гольдштейну (Matt Goldstein), Челси Харакозо-

вой (Chelsea Kharakozova), Эмме Снайдер (Emma Snider) и Джеффри Холькомбу (Jeffrey Holcomb).

Спасибо нижеперечисленным людям за их рецензии, комментарии и предложения для этой книги:

Джеймсу Беллу (James Bell) — *Центральный колледж штата Вирджиния;*

Кэролин З. Джиллай (Carolyn Z. Gillay) — *Колледж Седлбек;*

Джейсону Хеберту (Jason Hebert) — *Колледж Перл Ривер;*

Джин Кент (Jean Kent) — *Колледж Сиэтла;*

Бобу Макферсону (Bob McPherson) — *Колледж Сарри;*

Терезе Никерсон (Teresa Nickerson) — *Университет Дубьюка;*

Аните Филипп (Anita Philipp) — *Колледж Оклахома-сити;*

Отдельное спасибо Джин Кент (Jean Kent), колледж Сиэтла, и Терезе Никерсон (Teresa Nickerson), университет Дубьюка, обеспечившим связь со студентами, комментировавшими эту книгу.

Спасибо коллегам из колледжа Харпера за их поддержку, в частности Кену Перкинсу (Ken Perkins), Саре Старк (Sarah Stark), Энрике Д'Эмико (Enrique D'Amico) и Дэйву Брауншвейгу (Dave Braunschweig).

Больше всех я хотела бы поблагодарить свою семью за их терпение и поддержку. Мой чудесный муж, Грег Моррис, был постоянным источником любви, понимания, поддержки и одобрения. Спасибо тебе, Грег! Большая признательность моим детям, Джеймсу и Карен Фелке, которые росли, думая, что у всех мам есть веб-страницы. Спасибо вам обоим за ваше понимание, терпение и своевременные предложения!

Об авторе

Терри Фелке-Моррис — адъюнкт-профессор в колледже Уильяма Рейни Харпера в Палатине, штат Иллинойс. Она — обладатель степени магистра наук по информационным системам и множества сертификатов, включая Adobe Certified Dreamweaver 8 Developer, WOW Certified Associate Webmaster, Microsoft Certified Professional, Master CIW Designer и CIW Certified Instructor.

Миссис Фелке-Моррис была награждена памятной наградой Гленн А. Рейха колледжа Харпера за учебные технологии в знак признания за ее работу в разработке программ и курсов колледжа по CIS веб-разработке. В 2006 году она получила награду Blackboard Greenhouse Exemplary Online Course за использование веб-технологий в учебной обстановке. Миссис Фелке-Моррис получила две международные награды в 2008: Instructional Technology Council's Outstanding e-Learning Faculty Award за выдающееся мастерство и награду MERLOT за образцовые сетевые обучающие ресурсы — MERLOT Business Classics.

Имея за плечами более двадцати лет опыта в информационных технологиях в бизнесе и производстве, миссис Фелке-Моррис опубликовала свой первый веб-сайт в 1996 году и с тех пор работала с проектами для Всемирной паутины. Давний поборник веб-стандартов, она является членом оперативной группы по проектному изучению веб-стандартов. Миссис Фелке-Моррис помогла разработать сертификат веб-разработчика и научные программы в колледже Уильяма Рейни Харпера и ныне является старшим преподавателем в этой области. За более подробной информацией о миссис Фелке-Моррис заходите на сайт **terrymorris.net**.

Глава 1
ИНТЕРНЕТ И ВСЕМИРНАЯ ПАУТИНА

Цели главы

В этой главе вы узнаете следующее:

- описание эволюции Интернета и Всемирной паутины;
- предназначение веб-стандартов;
- описание стандартов веб-дизайна;
- определение достоверных источников информации во Всемирной паутине;
- предназначение веб-браузеров и веб-серверов;
- обзор сетевых протоколов;
- определение URI и доменных имен;
- описание HTML, XHTML и HTML5;
- описание популярных тенденций во Всемирной паутине.

Интернет и Всемирная паутина — часть нашей повседневной жизни. Как все начиналось? Какие сетевые протоколы и языки программирования используются «за кадром», чтобы веб-страница отобразилась? Данная глава знакомит с некоторыми из этих тем и дает основную информацию, которую необходимо знать веб-разработчику. Вы познакомитесь с языком разметки гипертекста, Hypertext Markup Language (HTML), используемым для создания веб-страниц; расширяемым языком разметки гипертекста, eXtensible Hypertext Markup Language (XHTML), более стандартизированной версией языка HTML; и HTML5 — новейшей черновой версией языка HTML.

1.1. Интернет и Всемирная паутина

Интернет, объединение компьютерных сетей, в наши дни встречается повсюду. Он стал частью нашей жизни. Вы можете смотреть телевизор или слушать радио и тут же услышать предложение посетить веб-сайт. Даже газеты и журналы нашли свое место в Интернете.

Интернет

Интернет начинался как сеть, связывающая компьютеры в исследовательских центрах и университетах. Сообщения в такой сети приходили в пункт назначения множеством путей и маршрутов. Это позволяло сети функционировать, даже если она была частично нарушена или уничтожена. Сообщение перенаправлялось через рабочую часть сети, путешествуя до пункта назначения. Эта сеть была предложена *Агентству по перспективным исследовательским проектам* (ARPA, Advanced Research Project Agency) — так родился *ARPAnet*. Четыре компьютера (расположенные в Калифорнийском университете в Лос-Анджелесе, Стэнфордском исследовательском институте, Университете Калифорнии в Санта-Барбаре и Университете Юты) были объединены в конце 1969 года.

Развитие Интернета

Время шло, и другие сети, такие как NSFnet Национального научного фонда, были созданы и объединены с ARPAnet. Использование объединенной сети, или Интернета, изначально ограничивалось правительственными, исследовательскими и образовательными целями. Запрет на коммерческое использование был снят в 1991 году. Рост Интернета продолжается — Internet World Stats сообщает о более чем 2 млрд пользователей в Интернете в 2011 году.

Рисунок 1.1 демонстрирует рост числа пользователей Интернета между 2000 и 2011 годами.

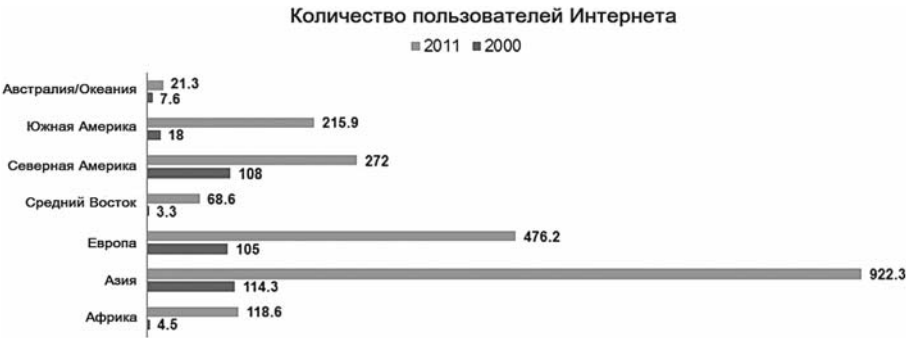


Рис. 1.1. Рост числа пользователей Интернета[1]

[1] Источник статистических данных: **www.internetworldstats.com**

Снятие запрета на коммерческое использование Интернета создало предпосылки для появления в будущем электронной коммерции: теперь компании стали в Интернете желанными гостями. Однако в Интернете по-прежнему отображался только текст, и Интернет не так просто было использовать. Несколько последующих разработок решили эту проблему.

Зарождение Всемирной паутины

Работая в ЦЕРН, исследовательской организации в Швейцарии, **Тим Бернерс-Ли** предвидел значение обмена информацией для ученых, где они могли «перейти» на страницу другой научной статьи или обсуждения и мгновенно отобразить ее. Бернерс-Ли создал ***Всемирную паутину***, которая могла выполнять эту функцию, и в 1991 году опубликовал код в группе новостей. Эта версия Всемирной паутины использовала ***протокол передачи гипертекста*** (HTTP, Hypertext Transfer Protocol) для установки связи между компьютером клиента и веб-сервером, и ***язык разметки гипертекста*** (HTML, Hypertext Markup Language), чтобы форматировать документы, и была текстовой.

Первый графический браузер

В 1993 году стал доступен первый графический веб-браузер — Mosaic (рис. 1.2).

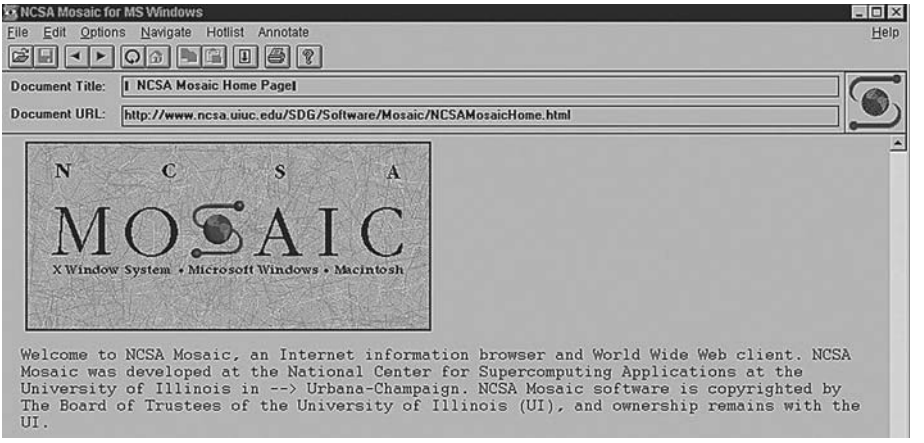


Рис. 1.2. Mosaic — первый графический веб-браузер

Марк Андриссен и аспиранты, работавшие в Национальном центре по приложениям для суперкомпьютеров (NCSA, National Center for Supercomputing Applications) при университете Урбана Шампейна штата Иллинойс, разработали приложение Mosaic. Некоторые участники этой группы позже создали другой хорошо известный веб-браузер — Netscape Navigator — прародителя нынешнего браузера Mozilla Firefox.

Слияние технологий

К началу 1990-х годов стали доступны персональные компьютеры, оснащенные простыми в использовании графическими операционными системами (такими как Windows корпорации Microsoft, OS/2 корпорации IBM и Macintosh OS корпорации Apple). Интернет-провайдеры, к примеру, компании CompuServe, AOL и Prodigy предложили недорогие варианты подключения к Интернету.

Наличие доступных компьютеров, простых в использовании операционных систем, дешевого подключения к Интернету, протокола HTTP, языка HTML и графического браузера значительно облегчило доступ к информации в Интернете. Появилась Всемирная паутина — графический пользовательский интерфейс, ведущий к информации, хранящейся на подключенных к Интернету веб-серверах!

Кто управляет Интернетом?

Вы, возможно, знаете, что не существует такого человека или группы людей, которые управляют всем Интернетом. Каждая сеть управляется отдельно. Однако существуют группы, которые разрабатывают стандарты и рекомендации, в том числе ***Рабочая группа инженеров Интернет*** (IETF, Internet Engineering Task Force) и ***Архитектурный совет Интернета*** (IAB, Internet Architecture Board). IETF — это рычаг управления проектированием и разработкой спецификаций новых интернет-протоколов. Это основные принципы, внедренные в разработку новых спецификаций интернет-стандартов. IETF — открытое международное сообщество сетевых разработчиков, операторов, производителей и исследователей, заинтересованных в развитии архитектуры Интернета и бесперебойной работы Интернета. Настоящая техническая работа IETF выполняется ее рабочими группами. Эти рабочие группы объединены по областям работы, например, безопасность и маршрутизация.

IAB является комиссией IETF и осуществляет инструктаж и общее руководство IETF. Как исполнитель этой цели IAB отвечает за публикацию ***запросов на комментарии*** (RFC, Request for Comments). RFC[1] — это формальный документ IETF, который создается комиссией и последовательно изучается заинтересованными лицами. Некоторые RFC носят информативный характер, в то время как другие превращаются в интернет-стандарты. В последнем случае окончательная версия RFC становится новым стандартом. Будущие изменения стандарта должны производиться с помощью последующих RFC.

Организация по присвоению имен и адресов в Интернет (ICANN, Internet Corporation for Assigned Numbers and Names)[2] была создана в 1998 году и является некоммерческой организацией. Ее основной функцией является координация присвоения доменных имен в Интернете, номеров IP-адресов, параметров протоколов и номеров портов. До 1998 года ***Полномочный орган по цифровым адресам в Интернете*** (IANA, Internet Assigned Numbers Authority)[3] координировал эти функции. IANA до сих пор выполняет некоторые функции под руководством ICANN.

Интранет и Экстранет

Вспомните, что Интернет — это объединенные вместе компьютерные сети, доступ к которым можно получить из любой точки земного шара. Если организации необходимы коммуникационные ресурсы Интернета, но при этом она не желает открывать всем подряд доступ к собственной информации, подойдет Интранет или Экстранет.

Интранет — это частная сеть, функционирующая внутри организации или компании. Ее цель — способствовать обмену внутренней информацией между сотрудниками. При соединении с Интернетом от несанкционированного доступа Интранет обычно защищает шлюз или брандмауэр.

Экстранет — частная сеть, в которой часть информации о компании или ее действиях без риска предоставляется внешним партнерам (поставщикам, продавцам и покупателям). Сети типа Экстранет можно использовать для обмена данными, передачи информации от одного

[1] RFC доступны для просмотра на веб-сайте **www.ietf.org/rfc.html**.
[2] www.icann.org
[3] www.iana.org

бизнес-партнера другому и совместной работы с другими организациями. При пользовании экстранетом важно решить вопросы секретности и безопасности. К технологиям, применяемым для обеспечения секретности и безопасности в Экстранете относятся цифровые сертификаты, кодирование сообщений и виртуальные частные сети. Об использовании цифровых сертификатов и кодирования в электронной коммерции мы поговорим в главе 12.

1.2. Стандарты Всемирной паутины

Что касается Интернета в целом, нет такого человека или группы людей, которые бы управляли Всемирной паутиной. Однако ***Консорциум Всемирной паутины*** (W3C, World Wide Web Consortium)[1] играет проактивную роль в разработке рекомендаций и прототипов технологий, относящихся к сети. Темы, которых касается Консорциум W3C, — веб-архитектура, стандарты веб-дизайна и обеспечение доступности веб-сайтов для инвалидов (по зрению и пр.). В попытке стандартизировать веб-технологии W3C (логотип показан на рис. 1.3) создает спецификации, которые называются рекомендациями.

Рис. 1.3. Логотип Консорциума W3C

Рекомендации Консорциума W3C

Рекомендации W3C создаются рабочими группами при участии множества крупных корпораций по разработке веб-технологий. Эти рекомендации не являются правилами; это инструкции. Большая часть компаний, производящих веб-браузеры, такие как Microsoft, не всегда следуют Рекомендациям W3C. Это усложняет жизнь веб-разработчикам, потому что не все браузеры могут отображать веб-страницы одинаково.

[1] www.w3.org

Хорошая новость в том, что новые версии большинства браузеров все тщательнее придерживаются Рекомендаций W3C. Существуют даже организованные группы, такие как Проект веб-стандартов (The Web Standards Project)[1], чья задача — внедрять Рекомендации W3C (их также часто называют веб-стандартами) не только среди создателей браузеров, но также среди разработчиков и дизайнеров. Вы будете следовать рекомендациям Консорциума W3C, когда будете выполнять примеры из этой книги — верстать веб-страницы. Следование рекомендациям W3C — первый шаг на пути осуществления доступности своего сайта.

Веб-стандарты и доступность

Инициативная группа по веб-доступности (WAI)[2] является главной областью работы W3C. С тех пор как Всемирная паутина стала неотъемной частью повседневной жизни, появилась необходимость сделать ее доступной для всех.

Всемирная паутина может создавать преграды людям со зрительными, слуховыми, физическими или неврологическими ограничениями. Инициативная группа по веб-доступности разработала рекомендации для разработчиков веб-содержимого, программного обеспечения, веб-браузеров и разработчиков других агентов пользователя, для облегчения использования сети людям с особыми потребностями[3].

Требования по доступности веб-сайтов в Российской Федерации

Стандарт «Интернет-ресурсы. Требования доступности для инвалидов по зрению» (ГОСТ Р 52872-2007) была внесен в 2009 году, чтобы обязать предоставлять людям с ограниченными возможностями доступ к информационным технологиям, сравнимый с доступностью, предоставляемой остальным людям. Стандарт обязует разработчиков, создающих веб-страницы, обеспечивать их ***доступность***. Текст стандарта вы найдете на диске, прилагающемся к книге.

---

[1] webstandards.org

[2] www.w3.org/WAI/

[3] См. документы с указаниями по обеспечению доступности веб-контента (WCAG, Web Content Accessibility Guidelines) на сайте **www.w3.org/WAI/WCAG20/glance/WCAG2-at-a-Glance.pdf**.

1.3 Информация во Всемирной паутине

Сегодня любой может опубликовать во Всемирной паутине что угодно. В этом разделе мы разберемся, как узнать, достоверна ли найденная во Всемирной паутине информация и каким образом можно ее использовать.

Достоверность источников информации во Всемирной паутине

Веб-сайтов великое множество, но какие из этих источников информации надежны? Заходя на сайт в поисках сведений о чем-либо, важно не принимать на веру все, что там написано. Задайте себе вопросы, перечисленные ниже:

- **Внушает ли доверие организация?**

  Любой пользователь может опубликовать во Всемирной паутине что угодно! Поступайте мудро при выборе источника информации. Сначала оцените, насколько надежен сам веб-сайт. У него есть собственное доменное имя, к примеру, **http://mywebsite.com**? Или это бесплатный веб-сайт — папка с файлами, хранящаяся где-то на бесплатном веб-сервере? В URL-адресе веб-сайта, размещенного на бесплатном веб-сервере, обычно содержится часть названия сервера, и он начинается примерно так: **http://mysite.tripod.com** или **http://www.angelfire.com/foldername/mysite**. Информация, полученная с веб-сайта с собственным доменным именем, обычно (но не всегда) более достоверна, чем выкладываемая на бесплатном сайте.

- **Изучите тип доменного имени.**

  Что это: некоммерческая организация (.org), компания (.com или .biz) или образовательное учреждение (.edu)? Компании часто искажают информацию в свою пользу, так что будьте осторожны. Некоммерческие организации и учебные заведения иногда более объективны при изучении проблемы.

- **Актуальна ли размещенная информация?**

  Необходимо также узнать дату создания или последнего обновления веб-страницы. И хотя есть информация, не меняющаяся с течением времени, очень часто веб-страница, не обновлявшаяся несколько лет, устаревает и оказывается не лучшим информационным источником.

- **Даются ли ссылки на дополнительные ресурсы?**

  Гиперссылки указывают на веб-сайты, содержащие вспомогательные или дополнительные сведения, которые могут оказаться полезны для более глубокого изучения темы. Ищите подобные гиперссылки, они помогут вам в исследовании.

- **Это Википедия?**

  Википедия вполне подходит для начала исследования, однако не принимайте на веру все, что в ней прочитаете, и старайтесь не пользоваться ресурсами Википедии при выполнении учебных заданий. Почему?

  В статьи Википедии (кроме нескольких защищенных тем) любой пользователь может добавить какую угодно информацию!

  Обычно рано или поздно ее сортируют, но помните, что текст, который вы читаете, возможно, недостоверный.

  Не бойтесь пользоваться Википедией, когда только начинаете разбираться в проблеме, однако затем перейдите в конец страницы и найдите раздел «Ссылки». Изучите указанные в нем веб-сайты и любые другие, которые вам попадутся.

  При сборе информации с этих сайтов учитывайте и другие критерии: надежность, доменное имя, свежесть и ссылки на дополнительные ресурсы.

Этичное использование информации во Всемирной паутине

Эта прекрасная технология, названная Всемирной паутиной, предоставляет доступ к информации, изображениям и музыке — все фактически бесплатно (конечно, после того как заплатите своему интернет-провайдеру). Давайте рассмотрим следующие вопросы, связанные с этичным использованием этой информации:

- Приемлемо ли копировать чьи-либо графические изображения для использования на своем веб-сайте?
- Приемлемо ли копировать чей-либо дизайн веб-сайта для использования на своем сайте или на сайте клиента?
- Приемлемо ли копировать статью, которая размещена на веб-странице, и использовать ее или ее части как собственное сочинение?

- Приемлемо ли оскорблять кого-либо на вашем веб-сайте или ссылаться на его сайт в унизительной манере?

Ответ на все три вопроса — нет. Использование чьей-либо графики без разрешения то же самое, что воровство. Копирование дизайна веб-сайта другого человека или компании также является формой воровства. Веб-сайт **pirated-sites.com** представляет несколько необычный взгляд в этом смысле. Любой текст или изображения на веб-сайте автоматически получают авторские права в Соединенных Штатах, появляется или нет на сайте знак авторских прав. Оскорбление человека или компании на вашем веб-сайте или ссылка на них в унизительной манере может считаться формой клеветы.

Подобные вопросы, относящиеся к интеллектуальной собственности, авторскому праву и свободы слова, регулярно обсуждаются и решаются в судах. Правила хорошего тона во Всемирной паутине требуют спрашивать разрешения, прежде чем использовать чужую работу, отдавать должное за то, что вы используете, и упражняться в свободе слова, не причиняя вреда другим. ***Всемирная организация по защите интеллектуальной собственности*** (WIPO, World Intellectual Property Organization), **wipo.int** занимается вопросами международной защиты прав интеллектуальной собственности.

Что если вы захотите сохранить право собственности, но позволить другим легко использовать и применять вашу работу? Creative Commons, **creativecommons.org** — некоммерческая организация, которая предоставляет бесплатные услуги, позволяющие авторам и создателям регистрировать типовые авторские договоры, называемые лицензиями Creative Commons. Существует несколько видов лицензий, в зависимости от прав, которые вы хотите предоставить. Лицензия Creative Commons информирует других, что именно они *могут* и *не могут* делать с плодами вашего творчества.

1.4. Краткий обзор сети

Сеть состоит из двух или более компьютеров, объединенных с целью обмена информацией и общего доступа к ресурсам. Общие компоненты сети показаны на рис. 1.4 и включают следующее:

- серверный (-е) компьютер (-ы);
- клиентский (-е) компьютер (-ы) (рабочие станции);

- устройства общего доступа, например принтеры;
- сетевые устройства (маршрутизатор и коммутатор) и носители, которые их объединяют.

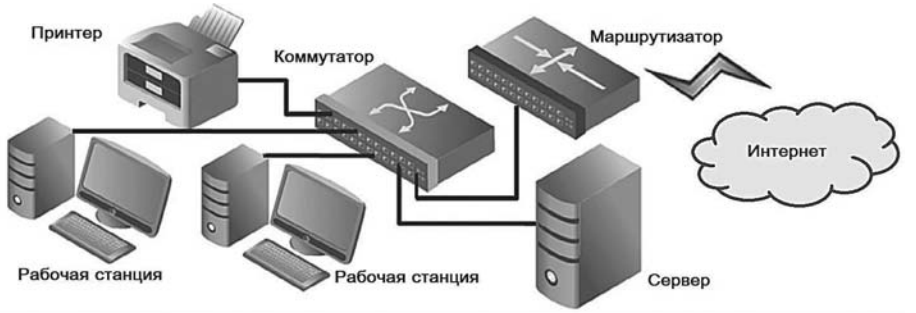


Рис. 1.4. Общие компоненты сети

Клиент — это рабочая станция, используемая отдельным пользователем, например, персональный компьютер на столе. ***Сервер*** получает запросы от компьютеров клиентов на ресурсы, такие как файлы. Компьютеры, используемые как серверы, обычно содержатся в защищенном, охраняемом месте, и доступ к ним имеют только администраторы сети. Сетевые устройства, такие как хабы и коммутаторы, обеспечивают связь между компьютерами, а маршрутизаторы направляют информацию из одной сети в другую. ***Носители***, объединяющие клиенты, серверы, периферийные и сетевые устройства, могут состоять из кабелей типа «витая пара», оптико-волоконных кабелей или беспроводных технологий.

Сети различаются по размерам. ***Локальная вычислительная сеть*** (LAN, Local Area Network) обычно ограничена одним помещением или группой взаимосвязанных помещений. Ваш школьный компьютер может использовать LAN. Если вы работаете в офисе, вы, скорее всего, используете компьютер, подключенный к LAN. Не так давно множество людей начали настраивать LAN дома, чтобы обмениваться ресурсами между компьютерами. ***Городская сеть*** (MAN, Metropolitan Area Network) объединяет пользователей с компьютерными ресурсами в пределах географического региона. Она также может использоваться для объединения двух и более LAN. ***Глобальная вычислительная сеть*** (WAN, Wide Area Network) географически рассредоточена и обычно использует определенную форму публичных или коммерческих сетевых средств коммуникации. Например, организация с офисами на восточном и западном побережьях Соединенных Штатов, скорее всего, ис-

пользует глобальную сеть, чтобы обеспечить связь между локальными сетями каждого офиса.

Магистральный канал передачи данных[1] — это высокопропускной канал связи, который передает данные, собранные из меньших каналов связи, объединенных с ним. В Интернете магистральный канал — это набор маршрутов, которые объединяются локальными или региональными сетями (MAN) для более удаленной связи. Интернет — это группа взаимосвязанных сетей с очень высокоскоростной связью, обеспеченной магистральными каналами.

1.5. Модель клиент-сервер

Появление термина «***клиент-сервер***» датируется прошлым тысячелетием (1980-ми годами) и обозначает персональные компьютеры, объединенные в сеть. Понятие «клиент-сервер» также описывает отношения между двумя компьютерными программами — клиентской и серверной. Клиент запрашивает некую услугу (например, доступ к файлу или базе данных) у сервера. Сервер выполняет запрос и передает результаты клиенту через сеть. Хотя обе программы, клиентская и серверная, могут находиться на одном компьютере, обычно они запускаются на разных компьютерах (рис. 1.5). Обычно сервер обрабатывает запросы от множества клиентов.

Рис. 1.5. Веб-клиент и веб-сервер

Интернет — отличный пример клиент-серверной архитектуры сети. Рассмотрим следующий сценарий: пользователь, работающий за компью-

[1] Иногда называемый «бэкбон» (от англ. *backbone*). — *Прим. ред.*

тером, использует веб-серверную программу, запущенную на компьютере, с IP-адресом, который соответствует **http://www.yahoo.com**. Программа получает запрос, находит веб-страницу и связанные с ней ресурсы, на которые поступил запрос, и отвечает, отправляя их пользователю.

Ниже в общих словах приведены различия между веб-клиентами и веб-серверами:

Веб-клиент:

- подключен к Интернету, когда необходимо;
- обычно запускает веб-браузер (клиент), например, Internet Explorer или Firefox;
- использует протокол HTTP;
- запрашивает веб-страницы у сервера;
- получает веб-страницы и файлы от сервера.

Веб-сервер:

- постоянно подключен к Интернету;
- запускает веб-серверные программы (такие, как Apache или Internet Information Server);
- использует протокол HTTP;
- получает запрос на веб-страницу;
- отвечает на запрос и передает код состояний, веб-страницу и связанные с ней файлы.

Когда клиенты и серверы обмениваются файлами, им часто необходимо указывать тип передаваемого файла; это делается с помощью типа ***MIME***. ***Многоцелевые расширения почты в Интернете*** (Multi-Purpose Internet Mail Extensions) — это правила, которые позволяют обмениваться мультимедийными документами среди множества различных компьютерных систем. Сначала MIME предназначались для расширения первоначального интернет-протокола электронной почты, но они также используются HTTP. MIME дают возможность обмена семью разными мультимедийными типами данных в Интернете: аудио, видео, изображениями, приложениями, сообщениями, составными и текстом. MIME также использует подтипы, чтобы подробнее описать данные. Тип MIME для веб-страницы — это text/HTML. MIME-типы изображений gif и jpeg, соответственно, image/gif и image/jpeg.

Веб-сервер определяет тип MIME-файла до передачи его веб-браузеру. Тип MIME высылается вместе с документом. Веб-браузер использует тип MIME, чтобы определить, как отображать документ.

Как информация передается из веб-сервера в веб-браузер? Клиенты (такие как веб-браузер) и серверы (такие как веб-сервер) обмениваются информацией, используя протоколы связи, такие как HTTP, TCP и IP — рассматриваемые в следующем разделе.

1.6. Интернет-протоколы

Протоколы — это правила, которые описывают, как клиенты и серверы общаются через сеть. Не существует единого протокола, который заставляет работать Интернет и Всемирную паутину — необходимо множество протоколов с особыми функциями.

Протокол передачи файлов (FTP)

Протокол передачи файлов (FTP, File Transfer Protocol) — это набор правил, которые позволяют обмениваться файлами между компьютерами через Интернет. В отличие от HTTP, который используется браузерами для запроса веб-страниц и связанных с ними файлов для отображения веб-страниц, ***FTP*** используется для простого перемещения файлов из одного компьютера на другой. Веб-разработчики обычно используют протокол FTP, чтобы передавать файлы веб-страницы со своих компьютеров на веб-серверы. FTP также обычно используется для загрузки программ и файлов из других серверов на индивидуальные компьютеры.

Протоколы электронной почты

Большинство из нас воспринимают электронную почту как нечто само собой разумеющееся, но в ее равномерной работе участвуют два сервера — сервер входящей почты и сервер исходящей почты. Когда вы отправляете письма другим, используется ***простой протокол передачи почты*** (SMTP, Simple Mail Transfer Protocol). Когда вы получаете почту, могут использоваться ***протокол почтового отделения*** (POP, Post Office Protocol, сейчас используется POP3) и ***протокол доступа к электронной почте Интернета*** (IMAP, Internet Message Access Protocol).

Протокол передачи гипертекста (HTTP)

HTTP — это набор правил для обмена файлами, например, текстом, графическими изображениями, звуком, видео и другими мультимедийными файлами во Всемирной паутине. Веб-браузеры и веб-серверы, как правило, используют этот протокол. Когда пользователь веб-браузера запрашивает файл, набирая адрес веб-сайта или щелкая мышью по гиперссылке, браузер строит HTTP-запрос и посылает его серверу. Веб-сервер на компьютере назначения получает запрос, делает необходимые процедуры и отвечает запрошенным файлом и любыми связанными с ним мультимедийными файлами.

Протокол управления передачей/межсетевой протокол (TCP/IP)

Протокол управления передачей/межсетевой протокол (TCP/IP, Transmission Control Protocol/Internet Protocol) был принят как официальный протокол связи в Интернете. TCP и IP имеют разные функции, которые работают вместе для обеспечения надежной связи в Интернете.

Протокол TCP

Цель TCP — обеспечить целостность сетевой связи. Сначала TCP разбивает файлы и сообщения на отдельные части, называемые ***пакетами***. Эти пакеты (рис. 1.6) содержат такую информацию, как пункт назначения, источник, порядковый номер и значения контрольной суммы, которые используются для подтверждения целостности данных.

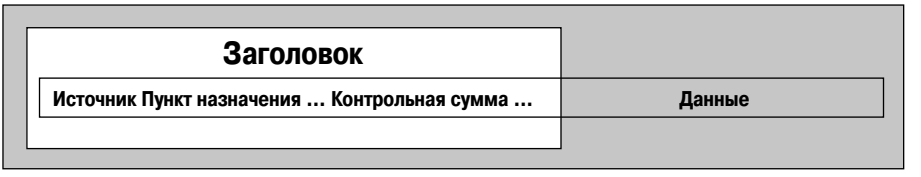


Рис. 1.6. Пакет TCP

TCP используется вместе с IP для эффективной передачи файлов через Интернет. IP вступает после того, как TCP создает пакеты, используя IP-адреса для передачи каждого пакета через Интернет, используя наилучший маршрут в каждый конкретный момент времени.

Когда адрес назначения достигнут, TCP проверяет целостность каждого пакета, используя контрольную сумму, запрашивает повторную отправку, если пакет поврежден, и собирает файл или сообщение из множества пакетов.

Протокол IP

Работающий в согласованности с TCP, IP представляет собой набор правил, контролирующих, каким образом данные пересылаются между компьютерами в Интернете. IP направляет пакет по правильному адресу. Как только пакет отправлен, он направляется к ближайшему маршрутизатору (аппаратное устройство, созданное для управления сетевым трафиком), пока не прибывает в пункт назначения.

Каждое устройство, подключенное к Интернету, имеет свой уникальный ***IP-адрес***. Эти адреса состоят из набора четырех групп цифр, которые называются октетами. Используемая версия IP, **IPv4**, использует 32-битную (двоичное число) адресацию. В результате получается десятичное число в формате *xxx.xxx.xxx.xxx*, где каждое значение *xxx* является числом от 0 до 255. Теоретически доступно 4 миллиарда возможных IP-адресов (хотя множество потенциальных адресов зарезервированы для особых нужд). С быстрым ростом мобильных устройств даже такого количества адресов может стать недостаточно в самые ближайшие годы.

IPv6, Internet Protocol Version 6, — следующая версия интернет-протокола. IPv6 был разработан как эволюционный набор улучшений нынешней IPv4 и имеет с ней обратную совместимость. Провайдеры услуг и пользователи Интернета могут обновлять протокол до версии IPv6 независимо, без необходимости координироваться друг с другом.

IPv6 предоставляет больше интернет-адресов, потому что значения увеличены с 32-битных до 128-битных. Это означает, что существует 2^{128} потенциальных IP-адресов, или 340 282 366 920 938 463 463 347 607 431 768 211 456. (Теперь будет достаточно IP-адресов для всех персональных компьютеров, ноутбуков, мобильных телефонов, пейджеров, карманных компьютеров, автомобилей, тостеров и т.д!)

IP-адрес может соответствовать доменному имени. ***Система доменных имен*** (DNS, Domain Name System) ассоциирует эти IP-адреса с текстовыми URL-адресами и доменными именами, которые вы набираете в поле адреса веб-браузера (более подробно об этом дальше). Напри-

мер, в то время, когда была написан этот текст, IP-адресом Google был 74.125.95.104. Вы можете ввести эти числа в поле адреса веб-браузера (как показано на рис. 1.7), нажать клавишу **Enter**, после чего отобразится главная страница Google. Конечно, намного проще напечатать «google.com», именно поэтому и были созданы доменные имена. Поскольку людям трудно запомнить длинные последовательности чисел, была введена система доменных имен, связывающая текстовые имена с числовыми IP-адресами.

Рис. 1.7. Ввод IP-адреса в веб-браузере

1.7. Унифицированные идентификаторы ресурса и доменные имена

URI и URL

Унифицированный идентификатор ресурса (URI, Uniform Resource Identifier) — идентифицирует ресурс в Интернете. ***Унифицированный указатель ресурса*** (URL, Uniform Resource Locator) — это тип URI, который представляет местонахождение в сети ресурса, например,

веб-страницы, графического или MP3-файла. URL-адрес состоит из протокола, доменного имени и иерархического расположения файла на веб-сервере.

URL-адрес **http://www.webdevfoundations.net**, показанный на рис. 1.8, означает использование протокола HTTP и веб-сервера www с доменным именем webfoundations.net. В данном случае будет показан корневой файл (обычно *index.html* или *index.htm*) в папке *chapter1*.

Рис. 1.8. URL-адрес файла внутри папки

Доменные имена

Доменное имя определяет местонахождение организации или другого субъекта в Интернете. Цель системы доменных имен (DNS) в том, чтобы разделить Интернет на логические группы и понятные имена, устанавливая точный адрес и тип организации. DNS связывает текстовые доменные имена с уникальным числовым IP-адресом, присвоенным устройству.

Давайте рассмотрим доменное имя **www.yahoo.com**. Часть .com — это доменное имя верхнего уровня. Часть yahoo.com — это доменное имя, зарегистрированное Yahoo!, оно считается доменным именем второго уровня. Часть www — это имя веб-сервера (иногда его называют ***основной веб-сервер***) домена yahoo.com. Все вместе www.yahoo.com называется ***полностью определенным именем домена*** (FQDN, Fully-Qualified Domain Name).

Доменные имена верхнего уровня

Домен верхнего уровня (TLD**, Top-Level Domain Names**) определяет самую правую часть доменного имени. TLD — либо общий домен верхнего уровня, такой как com для коммерческого, либо домен верхнего уровня с кодом страны, такой как fr для Франции. ICANN управляет общими доменами верхнего уровня, показанными в табл. 1.1.

Доменные имена .com, .net и .org используются без ограничений, это означает, что владелец обувного магазина (не связанный с сетью) может зарегистрировать сайт shoes.net.

Таблица 1.1. Домены верхнего уровня

Общие TLD	Предназначение
.aero	Авиатранспортная индустрия
.asia	Для резидентов Азиатско-Тихоокеанского региона
.biz	Бизнес
.cat	Каталонское лингвистическое и культурное сообщество
.com	Коммерческие субъекты
.coop	Кооперативные организации
.eco	Ресурсы, связанные с экологией
.edu	Образовательные проекты и аккредитованные заведения высшего образования США
.gov	Только государственные структуры США
.info	Неограниченное использование
.int	Международные организации (редко используется)
.jobs	Сообщество управления трудовыми ресурсами
.mil	Только военные структуры США
.mobi	Сайты ориентированы на работу с мобильными телефонами и беспроводными устройствами
.museum	Музеи
.name	Персоны
.net	Объекты, связанные с поддержкой сети Интернет, обычно провайдеры интернет-услуг или телекоммуникационные компании
.org	Некоммерческие организации
.post	Почтовые организации
.pro	Бухгалтеры, медики и адвокаты
.tel	Контактная информация для физических и юридических лиц
.travel	Индустрия туризма
.xxx	Сайты для взрослых

Доменные имена верхнего уровня с кодом страны

Двухбуквенные коды стран также присваиваются как доменные имена верхнего уровня. Изначально они предназначались, чтобы связывать код страны доменного имени с географическим расположением человека или организации, зарегистрировавшей имя. На практике довольно просто получить доменное имя с кодом любой страны. Для примера посетите ресурс **r01.ru** и множество других сайтов регистраторов доменных имен. Таблица 1.2 перечисляет некоторые коды стран, используемые во Всемирной паутине.

Таблица 1.2. Примеры кодов стран

TLD с кодом страны	Страна
.au	Австралия
.de	Германия
.eu	Европейский союз (группа стран, а не единственная страна)
.in	Индия
.jp	Япония
.ru	Российская Федерация
.nl	Нидерланды
.us	Соединенные Штаты
.рф	Российская Федерация (имена доменов пишутся кириллическими буквами)
.中国	Китай
.السعودية	Саудовская Аравия

На веб-сайте IANA[1] представлен полный список кодов стран.

DNS связывает доменные имена с IP-адресами. Каждый раз, когда новый URL-адрес вводится в веб-браузер, происходит следующее:

1. Производится доступ к DNS.
2. Обнаруживается соответствующий IP-адрес и возвращается веб-браузеру.
3. Веб-браузер посылает HTTP-запрос на компьютер назначения с соответствующим IP-адресом.
4. HTTP-запрос получен веб-сервером.
5. Необходимые файлы найдены и отправлены HTTP-ответом в веб-браузер.
6. Веб-браузер обрабатывает полученные данные и выводит веб-страницу и связанные с ней файлы.

В следующий раз, когда вы будете удивляться, почему нужно столько времени, чтобы отобразить веб-страницу, подумайте про весь процесс, происходящий «за кадром».

[1] www.iana.org/cctld/cctld-whois.htm

ЧаВо

СКОРО ЛИ ПОЯВЯТСЯ НОВЫЕ ДОМЕНЫ ВЕРХНЕГО УРОВНЯ?

Корпорация ICANN согласилась увеличить количество функционально-ориентированных доменов верхнего уровня и объявила о приеме заявок на новые пользовательские домены. Представьте, какие возможности для маркетинга — указать название вашей компании, товара или услуги в домене верхнего уровня! Слишком хорошо, чтобы быть правдой? Да, подвох есть. Стоимость заявки 185 тыс. долларов США, а ежегодная плата 25 тыс. долларов, так что вам нужно быть толстосумом. Подробнее об этом читайте на сайте **icann.org**.

1.8. Языки разметки

Языки разметки состоят из набора инструкций, которые «указывают» браузеру (и другим агентам пользователя, например, мобильным телефонам), как отображать веб-документ и управлять им. Эти инструкции обычно называются тегами и выполняют такие функции, как отображение графики, форматирование текста и формирование гиперссылок.

Стандартный обобщенный язык разметки (SGML)

SGML (Standard Generalized Markup Language) — это стандарт для установления языка разметки или набора тегов. SGML, по сути, не язык документации, а описание того, как его установить и создать ***определение типа документа*** (DTD, document type definition). Когда Тим Бернерс-Ли создал HTML, он использовал SGML для создания спецификации.

Язык разметки гипертекста (HTML)

HTML (Hypertext Markup Language) — это набор символов разметки или кода, размещенных в файле, предназначенном для отображения веб-браузером. Веб-браузер обрабатывает код в HTML-файле и отображает документ веб-страницы и связанные с ним файлы. Консорциум W3C устанавливает стандарты для HTML.

Расширяемый язык разметки (XML)

XML (Extensible Markup Language) был разработан консорциумом W3C как гибкий метод создания общих форматов информации и пере-

дачи форматов и информации через Всемирную паутину. Это текстовый синтаксис, разработанный, чтобы описывать, доставлять и передавать структурированную информацию. Он призван не заменить HTML, но расширить возможности HTML, отделяя данные от презентации. Используя XML, разработчики могут создавать любые теги, которые им нужны для описания информации.

Расширяемый язык разметки гипертекста (XHTML)

XHTML был разработан консорциумом W3C, чтобы переформулировать HTML 4.01 как приложение XML. Он объединяет возможности форматирования HTML 4.01 и структуру данных и возможности расширения XML.

Главные преимущества XHTML включают возможность расширять язык, создавая новые теги, а также перспективу повышенной совместимости платформ, поскольку мобильные устройства все чаще используются для доступа к сети.

HTML5 — Следующая версия (X)HTML

В то время когда велась работа над этой книгой, рабочая группа W3C по HTML (HTML WG) была занята созданием черновика рекомендации для **HTML5**, который должен стать следующей версией HTML4 и заменить XHTML. Черновой вариант HTML5 включает свойства как HTML, так и XHTML, добавляет новые элементы, предоставляет новые функции, такие как редактирование веб-форм. Также планируется сделать его совместимым с более старыми версиями языка. И хотя пока существует только черновой вариант, большинство из заявленных свойств стабильно работает и поддерживается современными браузерами. С помощью этой книги вы сможете научиться применять синтаксис HTML5.

1.9. Популярные решения во Всемирной паутине

Электронная коммерция

Электронная коммерция, покупка и продажа товаров через Интернет уже стали важной частью Всемирной паутины. Согласно исследо-

ваниям компании Forester Research к 2014 году на розничную продажу через Интернет будет расходоваться 248 млрд долл. США[1].

Учитывая, что по данным[2] веб-сайта Internet World Stats, Интернетом в мире пользуются более 2 млрд человек, потенциальных покупателей довольно много!

Поскольку беспроводной доступ к Всемирной паутине становится все более распространенным, электронная коммерция и доступ к Интернету будут совершаться не только со стационарных компьютеров, но и с мобильных устройств — карманных компьютеров, нетбуков, смартфонов и устройств, которые еще даже не придуманы.

Доступ с мобильных устройств

Все чаще пользователи выходят в Интернет не с настольных компьютеров, ноутбуков и нетбуков, а с иных устройств. Согласно прогнозам компании Gartner, к 2013 году мобильные телефоны (включая смартфоны и телефоны с браузерами) «оставят позади настольные компьютеры как наиболее распространенные в мире устройства для выхода в Интернет»[3].

Сайт SearchBlog приводит предположение, высказанное корпорацией Google в 2011 году о том, что «через полтора года с мобильных устройств будет поступать от 15 до 30% трафика»[4]. Журнал *Bloomberg Businessweek* пишет, что производители электроники ожидают огромного роста спроса на ***планшетные*** ПК (такие как Apple iPad, Motorola Xoom и Blackberry Playbook) и планируют к 2015 году заработать на продажах 49 млрд. долл США[5]. Веб-дизайнерам придется учитывать, как их страницы будут отображаться и функционировать не только на персональных компьютерах, но и на смартфонах, планшетных ПК и прочих мобильных устройствах.

Блоги

Тенденция ведения сетевого журнала, или блога, была введена отдельными людьми как форма личного самовыражения. ***Блог*** — журнал,

[1] techcrunch.com/2010/03/08/forrester-forecast-online-retail-sales-will-grow-to-250-billion-by-2014/
[2] www.internetworldstats.com/emarketing.htm
[3] www.gartner.com/it/page.jsp?id=1278413
[4] www.mediapost.com/publications/article/147794/
[5] buswk.co/fK2Q9e

доступный через веб, — это часто обновляемая страница с хронологическим списком идей и ссылок. Темы блогов охватывают диапазон от политических журналов до технической информации и личных дневников. Блоги могут охватывать одну тему или ряд разнообразных тем — зависит от человека, называемого блогером, который создает и поддерживает блог. Блогеры обычно обновляют свои блоги ежедневно с помощью простого в использовании программного обеспечения, позволяющего людям со слабой технической подготовкой или вообще без нее обновлять и вести блог. Многие блоги размещены на платформах, таких как **blogspot.com**, **wordpress.com** и **www.xanga.com**. Другие размещены на индивидуальных веб-сайтах, такие как блог эксперта CSS Эрика Мейера на **meyerweb.com**. Предприниматели отметили значение блогов как средства общения и построения отношений с клиентами. Корпорации, такие как IBM[1] и Adobe[2], используют блоги с этой целью.

Вики

Вики — это веб-сайт, который в любой момент может быть обновлен посетителями, использующими простую форму на веб-странице. Некоторые вики предназначены для небольших групп людей, например, членов организации. Самая влиятельная вики — это Википедия (Wikipedia), информационные и ресурсные ссылки — хороший отправной пункт для изучения новой темы. Вики — это действующая социальная программа. Пользователи делятся знаниями, создавая ресурс для свободного использования всеми желающими. И хотя не обходится без розыгрышей или случаев размещения в Википедии недостоверной информации, все эти сведения, а также относящиеся к ним ресурсы, замечательно подходят, чтобы начать изучение проблемы.

Социальные сети

Блоги и вики дают пользователям Всемирной паутины возможности взаимодействовать с веб-сайтами и другими людьми — это называется ***социальная компьютеризация*** или ***социальные сети***. Сегодня модно состоять в социальных сетях, таких как Facebook[3], MySpace[4] или ВКон-

[1] www.ibm.com/developerworks/mydeveloperworks/blogs
[2] feeds.adobe.com/
[3] facebook.com
[4] myspace.com

такте[1]. По данным исследования, проведенного компанией eMarketer, к 2014 году почти две трети пользователей Интернета станут завсегдатаями социальных сетей[2].

Кажется, что в Facebook зарегистрированы почти все ваши друзья? Об этом мы и говорим. В 2011 году социальная сеть Facebook насчитывала уже более 500 млн. активных пользователей[3]. В то время как ресурс LinkedIn создавали, преследуя профессиональные и бизнес-цели, бизнес-компании также обнаружили пользу сайта Facebook для продвижения своих продуктов и услуг.

Twitter[4] — это социальная сеть для ***микроблогинга***, или частого общения с помощью коротких сообщений (140 символов или меньше), называемых ***твитами***. Пользователи Twitter (их называют твиттерянинами) отвечают на обновления друзей и поклонников и делятся собственными ежедневными событиями и наблюдениями. Twitter не ограничивается только личным использованием. Мир бизнеса также открыл маркетинговые возможности, которые дает Twitter. Издание *Information Week* сообщает, что использование Twitter компанией Dell принесло прибыль в 3 млн долларов США[5].

RSS-каналы

Действительно простая синдикация (Really Simple Syndication, RSS) используется для создания лент новостей с блогов или других веб-сайтов. RSS-каналы содержат сводку новых статей, опубликованных на сайте. URL-адрес RSS-канала обычно обозначен буквами XML или RSS, написанными белым текстом на оранжевом прямоугольнике. Чтобы получить доступ к этой информации, нужна ***программа чтения новостей***, или ***RSS-агрегатор***. Некоторые браузеры, такие как Firefox, Safari и Internet Explorer (версия 7 или выше) могут отображать RSS-каналы. Также доступны коммерческие или условно-бесплатные программы чтения новостей. Программы чтения новостей запрашивают RSS-каналы через интервалы времени и отображают новые заголовки при запросе. RSS дает веб-разработчикам возможность предоставлять

[1] vkontakte.ru
[2] www.emarketer.com/Report.aspx?code=emarketer_2000644
[3] www.digitalbuzzblog.com/facebook-statistics-stats-facts-2011/
[4] twitter.com
[5] www.informationweek.com/news/hardware/desktop/217801030

новую информацию заинтересованным лицам и (если повезет) вызвать ответные визиты на сайт.

Подкасты

Подкасты — это аудио- и видеофайлы во Всемирной паутине — они могут иметь формат блога, радиопередачи или интервью. Подкасты обычно присылаются через RSS-каналы, но также могут быть доступны посредством записи в мультимедийные файлы и предоставления ссылок на веб-страницах. Эти файлы вы можете сохранить на компьютере или MP3-проигрывателе (например на iPod) и прослушать позже.

Веб 2.0

Flickr[1] и del.icio.us[2] — это два социальных сайта, которые предоставляют возможности обмена информацией. Flickr, сайт-фотохранилище, позиционируется «лучшим способом хранить, искать, сортировать и показывать свои фотографии». Приобретенный компанией AVOS Systems, ресурс del.icio.us — это коллекция любимых сайтов, позволяющая зарегистрированным пользователям публиковать списки любимых сайтов, делиться списками с другими и открывать для себя новые сайты. Веб-сайты, такие как Wikipedia, Flickr, Twitter и del.icio.us, являются примерами ***Веб 2.0***. Пока консенсус по определению Веб 2.0 еще не достигнут, можете считать его следующим шагом в переходе Всемирной паутины от изолированных статичных веб-сайтов к платформе, которая использует технологию, предоставляющую мощные интерфейсы и возможности социальных сетей. Посетите сайт **www.go2web20.net** и воспользуйтесь поисковой службой, чтобы найти сайты Веб 2.0. Прочтите статью про Веб 2.0 на сайте Википедии[3], чтобы узнать больше информации по этой теме.

Единственная тенденция, которая наверняка сохранится, — это тренд к постоянным изменениям. Технологии, связанные с Интернетом и Всемирной паутиной, постоянно находятся в состоянии разработки и улучшения. Если постоянные изменения и необходимость учить что-то новое вас вдохновляет, веб-разработки будут увлекательным полем деятельности. Знания и навыки, которые вы получите в этой книге, дадут вам прочное основание для дальнейшего обучения.

[1] www.flickr.com/
[2] del.icio.us
[3] ru.wikipedia.org/wiki/Веб_2.0

Глава 2
ОСНОВЫ РАЗМЕТКИ ВЕБ-СТРАНИЦ

Цели главы

В этой главе вы узнаете следующее:

- описание HTML, XHTML и HTML5;
- как определять язык разметки в файле веб-страницы;
- как использовать элементы `HTML`, `head`, `body`, `title` и `meta` при верстке типичной веб-страницы;
- как форматировать тело веб-страницы: заголовки, абзацы, переносы, списки, цитаты и элементы `div`;
- как форматировать текст с помощью строчных элементов;
- специальные символы;
- как использовать элементы привязки для создания ссылок со страницы на страницу;
- как создавать абсолютные, относительные ссылки и ссылки на адрес электронной почты;
- как верстать, сохранять и отображать документ веб-страницы;
- как проверять синтаксис разметки документа веб-страницы.

Изучив эту главу, вы создадите свою первую веб-страницу. Эта глава знакомит с языком разметки гипертекста (HTML) — языком, который используется для создания веб-страниц, с расширяемым языком разметки гипертекста (XHTML), стандартизированной версией HTML и с HTML5 — новейшей черновой версией языка HTML. Глава начинается с введения в синтаксис XHTML и HTML5, знакомит с блочным и строчным форматированием и демонстрирует гиперссылки по мере создания примеров страниц. Вы узнаете больше, если поработаете с примерами страниц. Использование HTML — это навык, а навыки лучше всего закрепляются практикой.

2.1. Обзор HTML

Языки разметки представляют собой наборы указаний для браузеров (и иных пользовательских агентов, таких как ПО мобильных телефонов), как отображать и организовывать веб-документ. Эти указания обычно называют ***тегами***. Они выполняют функции отображения графики, форматирования текста и перехода по гиперссылкам.

Всемирная паутина состоит из файлов, содержащих ***язык разметки гипертекста*** (***HTML***) и другие языки разметки, которые описывают веб-страницы. HTML был разработан с использованием стандартного обобщенного языка разметки (SGML). SGML прописывает стандартный формат для внедрения описательной разметки в документ и для описания структуры документа. Сам по себе SGML не является языком документа, но скорее описанием того, как задать его и создать ***определение типа документа*** (***DTD***).

Консорциум W3C устанавливает стандарты для HTML и связанных с ним языков. HTML (как и сам веб) постоянно изменяется.

HTML

HTML — это набор символов разметки или кода, помещенных в файл, предназначенный для отображения на странице веб-браузера. Эти символы разметки и код определяют структурные элементы, такие как абзацы, заголовки и списки. HTML также позволяет размещать мультимедийные элементы (такие как графика, видео и аудио) на веб-странице и описывать расчетные таблицы. Браузер интерпретирует код разметки и визуализирует страницу. HTML поддерживает независимое от платформы отображение информации в сети. Это значит, что независимо от того, на каком компьютере была создана веб-страница, любой браузер, запущенный на любой операционной системе, сможет отобразить страницу.

Каждый индивидуальный код разметки называется ***элементом*** или ***тегом***. У каждого тега своя цель. Теги заключается в угловые скобки — символы < и >. Большинство элементов указываются парно: открывающий тег и закрывающий тег. Эти теги исполняют роль контейнеров и иногда называются контейнерными. Например, текст, содержащийся между тегами `<title>` и `</title>`, на веб-странице будет отображаться в строке заголовка окна браузера. Некоторые элементы используются поодиночке и не являются частью пары. Например, элемент разрыва

строки, который инициирует перенос текста в соответствующей позиции на веб-странице, `br`, является одиночным или автономным и не имеет закрывающего тега. Вы познакомитесь с элементами лучше, когда начнете их использовать. Большинство элементов можно изменять с помощью ***атрибутов***, чье предназначение будет описано дальше.

XHTML

Стандартизированная версия HTML, используемая сегодня, это ***расширяемый язык разметки гипертекста*** (***XHTML***). XHTML использует элементы и атрибуты HTML4 вместе с синтаксисом XML. HTML изначально был создан, чтобы предоставить доступ к электронным документам через веб-браузер. Веб-браузеры, которые развивались одновременно с HTML, разрабатывались так, чтобы «прощать» ошибки кода, игнорировать ошибки синтаксиса и позволяли использовать «сырой» HTML-код. Веб-браузеры содержат множество программных инструкций, созданных, чтобы игнорировать ошибки, например, пропущенные закрывающие теги, и угадывать, как разработчик хотел отобразить страницу. Это не проблема для персонального компьютера с достаточно большой производительностью. Однако это могло стать проблемой для электронных устройств с меньшими ресурсами, такими как карманный компьютер или мобильный телефон.

Целью XHTML было предоставить основу для веб-доступа, независимого от устройств. XHTML был разработан W3C в качестве переформулировки HTML как приложение XML. XHTML совмещает возможности форматирования HTML и структуру данных и возможности расширения XML. Поскольку XHTML был создан с помощью XML, мы поверхностно рассмотрим XML.

XML **(Расширяемый язык разметки, eXtensible Markup Language)** — это стандартный метод W3C для создания новых языков разметки, которые смогут поддерживать отображение нетрадиционного контента, такого как математические обозначения, а также будут поддерживать новейшие устройства отображения, такие как КПК и мобильные телефоны. XML может удовлетворять этим разнообразным требованиям, потому что это расширяемый язык — он создан, чтобы позволить создание новых тегов или разметки. Синтаксис XML требует, чтобы мобильные устройства не тратили вычислительные силы, угадывая, как именно должен отображаться документ, но были способны эффективно отображать информацию. Документ XML должен быть ***правильно оформлен*** в соответствии с правилами синтаксиса языка.

HTML5

На момент написания этой книги рабочая группа Консорциума Всемирной паутины (W3C), разрабатывающая язык HTML, как раз создавала черновой вариант рекомендаций по применению HTML5. Эта версия языка должна стать преемницей HTML4 и заменить собой XHTML. HTML5 сочетает в себе свойства языков HTML и XHTML наряду с собственными элементами и предоставляет новые возможности, такие как редактирование веб-форм и поддержка видеофайлов. Также планируется сделать его совместимым с более старыми версиями языка.

Начать использовать HTML5 можно уже сейчас! Последние версии популярных браузеров Internet Explorer, Firefox, Safari, Google Chrome и Opera поддерживают некоторые новые свойства языка разметки HTML5. Можно ожидать, что с выходом новых версий браузеров HTML5 будет поддерживаться все лучше. Обучаясь создавать веб-страницы, вы должны не только знать, какие функции работают в современных браузерах, но и быть готовы использовать новые техники верстки на языке HTML5. Чтобы помочь вам в достижении этой амбициозной цели, книга освещает синтаксис языков XHTML и HTML5, а также содержит инструкции по верстке веб-страниц на HTML5 с элементами, совместимыми с предыдущими версиями, которые будут работать в современных браузерах. В ней вы также найдете практические задания с новыми свойствами языка HTML5, работающими только в последних версиях браузеров. Однако язык HTML5 существует пока еще только в черновом варианте и может измениться после того, как эта книга выйдет из печати. Поэтому чтобы увидеть список самых новых элементов HTML5, перейдите на сайт консорциума W3C[1].

ЧаВо

КАКИЕ ПРОГРАММЫ МНЕ НЕОБХОДИМЫ?

Для создания веб-страницы не требуется специального программного обеспечения — только текстовый редактор. В операционной системе Microsoft Windows имеется встроенный текстовый редактор Блокнот (Notepad). Операционная система OS X содержит редактор TextEdit. В качестве альтернативы встроенным редакторам операционных систем можно использовать один из множества бесплатных

[1] www.w3.org/TR/HTML-markup

или условно-бесплатных редакторов, к примеру, Notepad++[1] для Windows и TextWrangler[2] для OS X.

Другой популярный вариант — платные инструменты для создания веб-страниц, такие как Microsoft Expression Web или Adobe Dreamweaver. Какое бы программное обеспечение вы не выбрали, рекомендуется освоить основы языка HTML.

Вам понадобится тестировать веб-страницы в наиболее популярных браузерах, перечисленных ниже (все дистрибутивы вы найдете на диске, прилагающемся к книге):

- Internet Explorer[3];
- Mozilla Firefox[4];
- Apple Safari[5];
- Google Chrome[6];
- Opera[7].

Также пригодится дополнение Web Developer для браузера Firefox[8].

2.2. Определение типа документа

Поскольку существует множество версий HTML и XHTML, консорциум W3C рекомендует указывать тип языка разметки, который используется в веб-странице посредством ***определения типа документа*** (***DTD***). В определении типа документа указывается версия языка HTML, на котором он написан. Браузеры и валидаторы HTML-кода используют информацию, содержащуюся в DTD при обработке страницы. Определение DTD, которое принято указывать как **`DOCTYPE`**, находится в верхней части кода веб-страницы.

На данный момент общепринятая версия языка HTML — XHTML, поддерживаемая большинством популярных браузеров. Новейшая версия, HTML5, все еще находится в разработке. В большинстве примеров данной книги используется синтаксис HTML5. Этот синтаксис поддерживается в популярных браузерах, за исключением некоторых специально отмеченных случаев.

[1] notepad-plus-plus.org/download
[2] www.barebones.com/products/textwrangler/download.html
[3] windows.microsoft.com/ru-ru/internet-explorer/products/ie/home
[4] www.mozilla.org/ru/firefox/fx/
[5] www.apple.com/ru/safari/download/
[6] www.google.com/chrome
[7] ru.opera.com/
[8] addons.mozilla.org/ru/firefox/addon/web-developer/

2.3. Пример веб-страницы XHTML

В этом примере используется определение DTD для переходной версии XHTML 1.0. Это менее строгий вариант XHTML 1.0. DTD для переходной версии XHTML 1.0 выглядит следующим образом:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
" http://www.w3.org/TR/XHTML1/DTD/XHTML1-
transitional.dtd ">
```

Остальная часть кода веб-страницы содержит элементы разметки HTML и текст. После DTD каждая строка начинается с открывающего тега <HTML> и завершается закрывающим тегом </HTML>. Эти теги указывают, что находящийся между ними текст отформатирован для отображения в браузере с помощью языка HTML. Все веб-страницы, создаваемые вами на языке XHTML, будут содержать определение DTD, за которым последуют элементы `HTML`, `head`, `title`, `meta` и `body`. Базовый шаблон веб-страницы на языке XHTML (см. файл *Примеры\Глава_02\templatex.html* на диске, прилагающемся к книге) выглядит так:

```
<!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 "
lang="en" xml:lang="en">
<head>
<title>Заголовок страницы размещается здесь</title>
<meta http-equiv="Content-Type" content="text/HTML;
charset=utf-8" />
</head>
<body>
...текст страницы и прочие XHTML-теги размещаются здесь...
</body>
</HTML>
```

Обратите внимание, что теги XHTML указываются с маленькой буквы. Это согласуется с синтаксисом XML. Также обратите внимание, что описание DTD не соответствует этому синтаксису. Описание DTD

указывает, какой язык разметки используется и имеет свой собственный формат — смешанный вариант. За исключением заголовка, индивидуального для каждой страницы, первые восемь строк на всех веб-страницах, созданных на языке XHTML, обычно одинаковы.

В XHTML элемент `HTML` также необходим для описания ***пространства имен XML*** (***xmlns***), местонахождения документации для используемых элементов. Эта дополнительная информация добавляется в элемент `HTML` в виде ***атрибута***, изменяющего или более подробно описывающего функцию элемента. Атрибут `xmlns` указывает на URL-адрес пространства имен XHTML, используемого в документе, как правило, **`http://www.w3.org/1999/XHTML`**. Необязательные атрибуты **`lang`** и **`xml:lang`** определяют язык документа. Например, `lang="en"  xml:lang="en"` обозначает английский язык. Поисковые машины и программы экранного доступа могут считывать эти атрибуты.

Поначалу может казаться, что концепция определения DTD и URL-адреса xmln весьма сложна. Но не волнуйтесь, эти строки повторяются вновь и вновь на каждой странице. Создав один шаблон веб-страницы, вы получите готовые выражения, которые сможете использовать на каждой следующей странице.

2.4. Пример веб-страницы HTML5

Итак, мы изучили пример веб-страницы, созданной на языке XHTML, а теперь давайте сосредоточимся на HTML5. Его синтаксис замечательно проработан и прост в использовании. Мы применим тот же стиль верстки: будем набирать текст в нижнем регистре и поместим значения атрибутов в кавычки. Выражения определения типа документа на HTML5 следующее:

```
<!DOCTYPE HTML>
```

Как и в XHTML, это первая строка документа. Далее код веб-страницы начинается с открывающего тега `<HTML>` и завершается закрывающим тегом `</HTML>`. Эти теги указывают, что находящийся между ними текст отформатирован на языке HTML для отображения в браузере. Все веб-страницы, создаваемые вами на языке HTML5, будут содержать определение DTD, за которым последуют элементы `HTML`, `head`, `title`, `meta` и `body`. Базовый шаблон веб-страницы на языке

HTML5 (см. файл *Примеры\Глава_02\template.html* на диске, прилагающемся к книге) выглядит так:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Заголовок страницы размещается здесь</title>
<meta charset="utf-8">
</head>
<body>
. . .текст страницы и прочие HTML5-теги размещаются здесь. . .
</body>
</HTML>
```

В следующем разделе мы поговорим о назначении элементов `head`, `title`, `meta` и `body`.

2.5. Элементы `head`, `title`, `meta` и `body`

Веб-страница состоит из двух разделов: ***головы*** (`head`) и ***тела*** (`body`). Раздел головы, чаще называемый ***разделом заголовка*** (header), содержит информацию, которая описывает веб-страницу. Тело страницы содержит элементы, которые отображаются непосредственно на веб-странице в окне браузера: теги, текст, изображения и прочие объекты.

Раздел заголовка страницы

Элементы, расположенные в разделе заголовка, включают заголовок веб-страницы (`title`), метатеги, описывающие документ (такие как используемая кодировка символов и информация, к которой имеют доступ поисковые службы), и ссылки на скрипты и стили. Многие из этих данных не отображаются на веб-странице. Элемент `head`, содержащий раздел заголовка, начинается тегом `<head>` и заканчивается тегом `</head>`. Вы всегда вводите еще как минимум два элемента в этом разделе: `title` и `meta`.

Первый элемент в разделе заголовка, `title`, содержит текст, который появится в строке заголовка окна браузера. Этот текст, разме-

щаемый между тегами <title> и </title>, называется заголовком (названием) веб-страницы и используется, когда веб-страница сохраняется в Избранное или выводится на печать. Заголовок должен быть описательным. Если веб-страница предназначается для компании или организации, заголовок должен включать название компании или организации.

Элемент **meta** (метатег) используется для описания характеристик веб-страницы, таких как кодировка символов. ***Кодировка символов*** — это международное представление букв, цифр и символов в веб-странице или любом другом файле, который хранится на компьютере и может быть передан через Интернет. Существует множество наборов кодировок символов. Однако установившейся практикой является использование кодировок, которые широко поддерживаются, таких как utf-8, которая является вариантом Unicode. Метатег не использует пару из открывающего и закрывающего тегов. Он считается ***одиночным*** или ***автономным*** тегом. В XHTML и HTML5 метатег указывается по-разному. В XHTML метатег содержит больше детализированных атрибутов и пишется с /> в конце. Метатег в HTML5 сокращен и содержит только атрибут charset, определяющий кодировку символов.

Метатег в XHTML

```
<meta http-equiv="Content-Type" content="text/HTML;
charset=utf-8" />
```

Метатег в HTML5

```
<meta charset="utf-8">
```

Раздел тела страницы

Раздел тела страницы включает в себя текст и элементы, отображающиеся на странице в окне браузера. Его задача — конфигурировать содержимое веб-страницы.

Элемент **body** содержит раздел тела страницы, начинающийся с тега <body> и завершающийся тегом </body>. Вы проведете бóльшую часть времени за написанием кода в теле веб-страницы. Если вы пишете текст в теле, он появится непосредственно на веб-странице.

2.6. Ваша первая веб-страница

Практическое задание 2.1

Итак, вы ознакомились с основными элементами, присутствующими в каждом веб-документе, и теперь пора приступить к созданию первой веб-страницы.

Создание папки

При разработке веб-страниц и создании собственных сайтов рекомендуем организовывать файлы в отдельные папки. Создайте на жестком или съемном диске новую папку под названием *mychapter2*.

Для создания новой папки в операционной системе OS X:

1. Запустите файловый менеджер Finder и выберите директорию, в которой хотите создать папку.
2. Выберите команду меню **Файл** ⇒ **Новая папка** (File ⇒ New Folder) и создайте безымянную папку.
3. Для переименования выделите папку и щелкните мышью по ее названию. Введите желаемое имя папки и нажмите клавишу **Return**.

Для создания новой папки в операционной системе Windows:

1. Запустите программу Проводник (Windows Explorer) (выберите команду меню **Пуск** ⇒ **Все программы** ⇒ **Стандартные** ⇒ **Проводник** (Start ⇒ All programs ⇒ Accessories ⇒ Windows Explorer)) и перейдите в каталог, где хотите создать папку, например, в **Документы** (Documents) на диске **C**.
2. Нажмите кнопку **Новая папка** (New Folder).
3. Чтобы переименовать папку, щелкните по ней правой кнопкой мыши и выберите в контекстном меню пункт **Переименовать** (Rename). Введите новое имя папки и нажмите клавишу **Enter**.

Создание веб-страницы

Теперь вы готовы создать свою первую веб-страницу на языке HTML5. Запустите программу Блокнот (Notepad) или другой текстовый редактор и напечатайте следующий код:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Моя первая HTML5-веб-страница</title>
<meta charset="utf-8">
</head>
<body>
```

```
Привет, мир!
</body>
</HTML>
```

Обратите внимание, что первые строки файла содержат определение DTD. HTML-код начинается с открывающего тега `<HTML>` и заканчивается закрывающим тегом `</HTML>`. Цель этих тегов — обозначить, что содержимое между ними является веб-страницей. Раздел заголовка ограничен тегами `<head>` и `</head>` и содержит пару тегов заголовка со словами «Моя первая HTML5-веб-страница» между ними вместе с элементом `meta`, предназначенным для указания кодировки. Тело страницы ограничивается тегами `<body>` и `</body>`.

Фраза «Привет, мир!» написана в строке между элементами тела. На рис. 2.1 изображен снимок кода, как он выглядит в программе Блокнот (Notepad). Вы только что создали исходный код веб-документа.

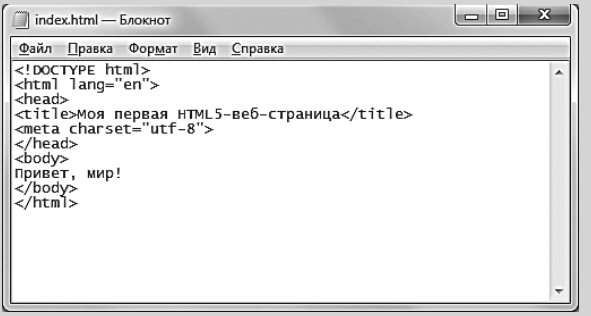


Рис. 2.1. Исходный код файла

ЧаВо

ОБЯЗАТЕЛЬНО ЛИ НАЧИНАТЬ КАЖДЫЙ ТЕГ С НОВОЙ СТРОКИ?

Нет. Браузер может отобразить страницу, даже если все теги написаны один за другим на одной строке без пробелов. Людям, однако, легче писать и читать код веб-страниц, если используются переносы строк.

Сохранение файла

Вы сохраните файл с именем *index.html*. Для файла главной (домашней) веб-страницы обычно используется имя *index.htm*, либо *index.html*. Для всех примеров в этой книге в именах файлов веб-страниц используется расширение *.html*. Выберите пункт меню **Файл** ⇒ **Сохранить как** (File ⇒ Save as). Откроется диалоговое окно **Сохранить**

как (Save as). Используя рис. 2.2 как пример, введите имя файла, укажите тип файла — **Все файлы** (All files) и выберите кодировку **UTF-8**, чтобы кириллические символы на странице отображались правильно. Нажмите кнопку **Сохранить** (Save). Если хотите, можете сравнить свою работу с файлом на диске (*Примеры\Глава_02\index.html*) прежде, чем тестировать свою страницу.

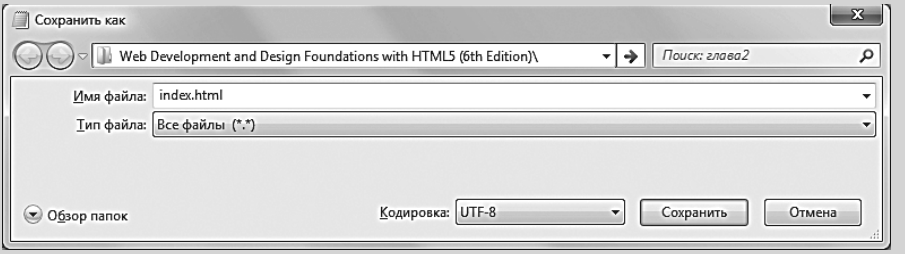


Рис. 2.2. Диалоговое окно **Сохранить как**

ЧаВо

ПОЧЕМУ МОЙ ФАЙЛ ИМЕЕТ РАСШИРЕНИЕ *.TXT*?

В некоторых старых версиях Windows программа Блокнот (Notepad) автоматически добавляет расширение *.txt*. Если это произойдет, напишите имя файла в кавычках **"index.html"** и сохраните файл еще раз.

Тестирование страницы

Есть два способа протестировать вашу страницу:

1. **Запустите программу Проводник (Windows Explorer).** Найдите файл *index.html*. Дважды щелкните мышью по файлу *index.html*. Браузер, используемый по умолчанию, запустит и отобразит вашу страницу *index.html*. Ваша страница должна быть похожа на показанную на рис. 2.3.

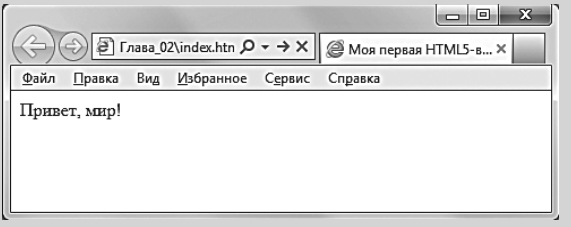


Рис. 2.3. Веб-страница, отображенная в браузере Internet Explorer

2. **Запустите браузер.** Перетащите файл *index.html в окно браузера*. Ваша страница должна быть похожей на показанную на рис. 2.3. Вид страницы в браузере Opera показан на рис. 2.4.

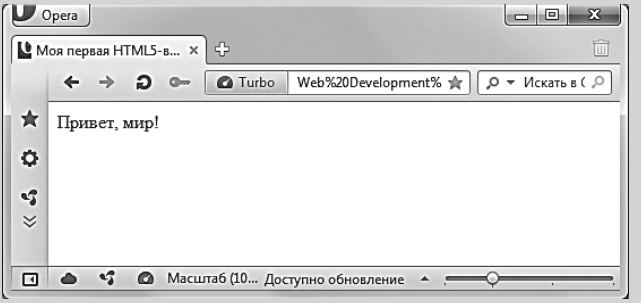


Рис. 2.4. Страница, отображенная браузером Opera

Изучите свою страницу. Внимательно осмотрите окно браузера. Обратите внимание, что заголовок окна браузера содержит текст «Моя первая HTML5-веб-страница». Некоторым поисковым системам необходим текст, окруженный тегами `<title>` и `</title>`, чтобы определить релевантность поиска по ключевым словам, так что убедитесь, что ваша страница имеет содержательный заголовок. Элемент `title` также используется, когда посетители сохраняют вашу страницу как закладку или добавляют в Избранное. Привлекательное и описательное название страницы может соблазнить посетителя посетить вашу страницу снова. Если ваша веб-страница предназначена для компании или организации, хорошая идея включить имя компании или организации в заголовок.

2.7. Элемент заголовка

Элементы заголовка организованы по уровням от `h1` до `h6`. Текст, содержащийся в элементе заголовка, воспринимается браузером как «блок» (блочное отображение). Кроме того, сверху и снизу он отделяется пустым пространством (которое иногда называют «пробельный символ»). Размер текста наибольший для `h1` (заголовок первого уровня) и наименьший для `h6` (заголовок шестого уровня). В зависимости от используемого шрифта (более подробно о шрифтах в главе 3), текст, содержащийся в элементах `h4`, `h5` и `h6`, может выглядеть меньшим, чем установленный по умолчанию. Весь текст, окруженный тегами заголовка, отображается полужирным шрифтом. На рис. 2.5 показан вебдокумент с шестью уровнями заголовков.

ЧаВо

ПОЧЕМУ ЭЛЕМЕНТ ЗАГОЛОВКА РАСПОЛОЖЕН НЕ В РАЗДЕЛЕ ЗАГОЛОВКА?

Начинающие веб-дизайнеры все время пытаются добавить элемент заголовка в одноименный раздел веб-документа. Однако в этом случае браузеры не всегда правильно отображают веб-страницу. Несмотря на то что понятия «элемент заголовка» и «раздел заголовка» похожи, элементы заголовка всегда необходимо размещать в разделе тела веб-страницы.

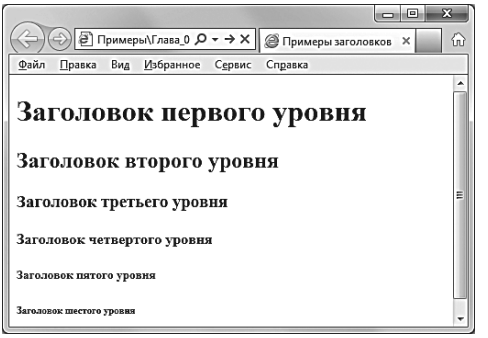


Рис. 2.5. Примеры заголовков

Практическое задание 2.2

Для создания веб-страницы, показанной на рис. 2.5, запустите программу Блокнот (Notepad) или любой другой текстовый редактор. Чтобы отредактировать файл *Примеры\Глава_02\template.html,* выберите команду меню **Файл** ⇒ **Открыть** (File ⇒ Open). Измените код страницы, как показано ниже (выделено полужирным):

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Примеры заголовков</title>
<meta charset="utf-8">
</head>
<body>
<h1>Заголовок первого уровня</h1>
<h2>Заголовок второго уровня</h2>
<h3>Заголовок третьего уровня</h3>
<h4>Заголовок четвертого уровня</h4>
<h5>Заголовок пятого уровня</h5>
```

```
<h6>Заголовок шестого уровня</h6>
</body>
</HTML>
```

Сохраните файл с именем *heading.html*. Запустите браузер, например Internet Explorer или Firefox, чтобы протестировать страницу. Она должна выглядеть так, как показано на рис. 2.5. Вы можете сравнить свою работу с файлом на диске в формате HTML5 (*Примеры\Глава_02\heading.html*) и XHTML (*Примеры\Глава_02\headingXHTML.html*). Обратите внимание, что в этом примере код в обоих случаях одинаковый, за исключением определения типа документа, элемента `HTML` и метатегов. То же самое вы увидите в большинстве примеров главы. Файлы всех практических заданий на языке HTML5 вы найдете на диске, прилагающемся к книге. XHTML-варианты предоставляются для сравнения, когда это необходимо.

Дополнительные параметры заголовков в синтаксисе HTML5

Вы, возможно, слышали о новых элементах группировки заголовков HTML5. Они предлагают дополнительные параметры для создания заголовков, но поддерживаются только в последних версиях браузеров. Мы рассмотрим их в главе 6.

2.8. Элемент абзаца

Элементы абзаца применяются для группировки предложений и разделов текста. Текст, окруженный тегами `<p>` и `</p>`, отображается одним «блоком» (так называемое блочное отображение). Сверху и снизу от него остается пустое пространство. На рис. 2.6 показана веб-страница, содержащая абзац после первого заголовка.

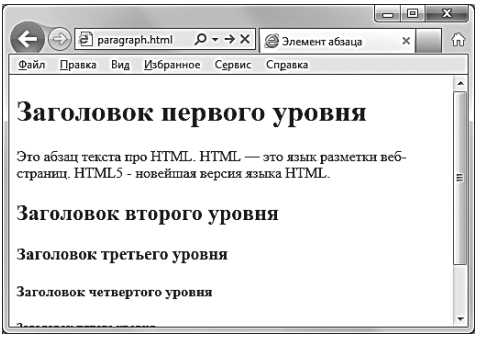


Рис. 2.6. На веб-странице используются заголовки и абзац текста

Практическое задание 2.3

Откройте файл *heading.html* в текстовом редакторе. Как показано в приведенном ниже коде, измените заголовок страницы, добавьте абзац текста между строк с элементами `h1` и `h2`.

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Элемент абзаца</title>
<meta charset="utf-8">
</head>
<body>
<h1>Заголовок первого уровня</h1>
<p>Это абзац текста про HTML. HTML — это язык разметки
веб-страниц. HTML5 - новейшая версия языка HTML.
</p>
<h2>Заголовок второго уровня</h2>
<h3>Заголовок третьего уровня</h3>
<h4>Заголовок четвертого уровня</h4>
<h5>Заголовок пятого уровня</h5>
<h6>Заголовок шестого уровня</h6>
</body>
</HTML>
```

Сохраните страницу с именем *paragraph.html*. Запустите браузер, чтобы протестировать страницу (рис. 2.6). Сравните ее с файлом примера на диске (*Примеры\Глава_02\paragraph.html*).

Обратите внимание, что текст переносится автоматически при изменении размеров окна браузера.

Выравнивание текста

Во время тестирования веб-страниц вы могли заметить, что заголовки и текст начинаются от левой границы. Это называется ***выравниванием по левому краю*** и является выравниванием по умолчанию для веб-страниц. Иногда вы захотите выровнять абзац текста или заголовок по центру или по правому краю (по ширине). Для этого можно использовать атрибут `align`. Цель ***атрибута*** — в изменении свойств HTML-элементов. В данном случае атрибут **`align`** меняет горизон-

тальное выравнивание элемента (левое, по центру или правое) на веб-странице. Чтобы выровнять элемент по центру, используйте атрибут со значением `align="center"`.

Чтобы выровнять элемент по правому краю веб-страницы, используйте `align="right"`. По умолчанию выравнивание установлено по левому краю. Атрибут `align` может использоваться для множества блочных элементов, в том числе абзаца (`p`) и заголовка (от `h1` до `h6`). Атрибут выравнивания не применяется в версии HTML5. Это значит, что он использовался в языке XHTML, но был удален консорциумом W3C из чернового варианта HTML5. В главе 6 вы научитесь выравнивать объекты, используя более современный подход — каскадные таблицы стилей (CSS).

2.9. Элемент разрыва строки

Элемент разрыва строки используется, чтобы принудительно перенести текст на новую строку перед отображением следующего элемента или фрагмента текста на веб-странице в браузере. Элемент разрыва строки не используется в виде пары открывающего и закрывающего тегов. Это ***одиночный*** или ***автономный*** тег. В синтаксисе HTML5 элемент разрыва строки выглядит как **`<br>`**. Если использовать XHTML (который следует синтаксису XML), элемент разрыва строки указывается как `<br>` (закрывающие символы `/>` обозначают автономный тег). На рис. 2.7 показана веб-страница с разрывом строки после первого предложения абзаца.

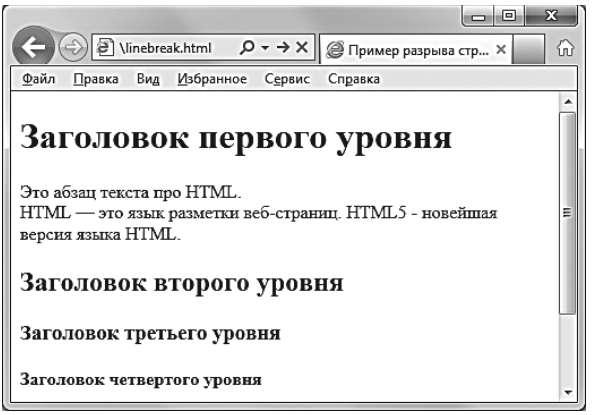


Рис. 2.7. Элемент `br` создает разрыв строки после первого предложения

Практическое задание 2.4

Откройте файл *paragraph.html* в программе Блокнот (Notepad). Измените заголовок страницы на «Пример разрыва строки». Установите курсор после первого предложения в абзаце (после слов «Это абзац текста про HTML»). Нажмите клавишу **Enter**. Сохраните страницу. Протестируйте страницу в браузере и обратите внимание, что хотя в исходном коде предложение «Это абзац текста про HTML» расположено на отдельной строке, браузер не так его обрабатывает. Чтобы изменить поведение браузера, необходим элемент `br`. Откройте файл *paragraph.html* в программе Блокнот (Notepad) и добавьте элемент `br` после первого предложения абзаца. Ваш исходный код (показан фрагмент) должен выглядеть подобным образом:

```
<body>
<h1>Заголовок первого уровня</h1>
<p>Это абзац текста про HTML. <br> HTML — это язык разметки веб-страниц. HTML5 - новейшая версия языка HTML.
</p>
<h2>Заголовок второго уровня</h2>
<h3>Заголовок третьего уровня</h3>
<h4>Заголовок четвертого уровня</h4>
<h5>Заголовок пятого уровня</h5>
<h6>Заголовок шестого уровня</h6>
</body>
```

Сохраните файл с именем *linebreak.html*. Запустите браузер, чтобы протестировать страницу. Она должна выглядеть похожей на изображенную на рис. 2.7. Вы можете сравнить свою работу с файлом на диске (*Примеры\Глава_02\linebreak.html*). В случае использования синтаксиса XHTML, требуемый файл сохранен как *Примеры\Глава_02\linebreakXHTML.html*.

ЧаВо

ПОЧЕМУ МОЯ СТРАНИЦА НЕ ИЗМЕНИЛАСЬ?

Часто пользователи вносят изменения в веб-страницы, но бывают сбиты с толку, потому что их браузеры показывают старую версию страницы. Перечисленные ниже советы помогут, если вы знаете, что изменяли свою веб-страницу, но изменения не появляются в браузере:

1. Убедитесь, что сохранили страницу после внесения изменений.
2. Проверьте адрес, по которому вы сохранили страницу — жесткий диск, определенная папка.

3. Проверьте адрес, по которому браузер запрашивает страницу — жесткий диск, определенная папка.
4. Убедитесь, что нажали кнопку браузера **Обновить** (Refresh) или **Перезагрузить** (Reload).

2.10. Элемент цитирования

Кроме организации текста в абзацы и заголовки, иногда вам понадобится добавить на веб-страницу цитату. **Элемент цитирования** предназначен для отображения блока текста цитаты в особой форме — с отступом от левой и правой границ. Блок текста с отступом начинается с тега `<blockquote>` и заканчивается тегом `</blockquote>`. На рис. 2.8 показана веб-страница, содержащая заголовок, абзац и цитату.

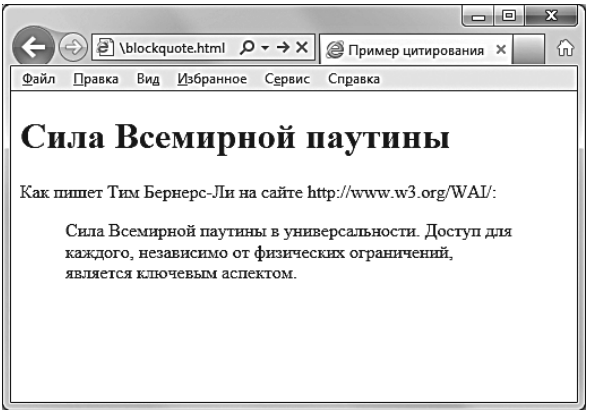


Рис. 2.8. Страница с цитатой

Практическое задание 2.5

Откройте файл *Примеры\Глава_02\template.html* в программе Блокнот (Notepad). Измените заголовок страницы на «Пример цитирования». Внесите изменения в код согласно листингу ниже:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Пример цитирования</title>
<meta charset="utf-8">
</head>
<body>
```

```
<h1>Сила Всемирной паутины</h1>
<p>Как пишет Тим Бернерс-Ли на сайте http://www.w3.org/
WAI/:</p>
<blockquote>
Сила Всемирной паутины в универсальности. Доступ для
каждого, независимо от физических ограничений, является
ключевым аспектом.
</blockquote>
</body>
</HTML>
```

Сохраните файл с именем *blockquote.html*. Запустите браузер, чтобы протестировать свой файл. Ваша страница должна быть похожей на страницу, показанную на рис. 2.8; сравните ее с примером на диске, прилагающемся к книге (*Примеры\Глава_02\blockquote.html*).

Вы, вероятно, уже обратили внимание, насколько полезен может оказаться элемент `blockquote` для структурирования текста на веб-странице. Думаете, можно ли применять элемент цитирования каждый раз, когда нужно добавить отступ к тексту или же он подходит лишь для длинных цитат? Семантически верно использовать элемент цитирования только для отображения крупных блоков цитируемого текста. Подумайте о будущем Семантической паутины, которая, по мнению журнала *Scientific American*, является «новой формой веб-контента, значимого для компьютеров, [которая] даст толчок революции новых возможностей»[1]. Семантическое, структурированное использование HTML — шаг к созданию Семантической паутины. Так что старайтесь не применять элемент цитирования для добавления отступов к тексту. Далее в книге вы узнаете о современных техниках конфигурирования полей и отступов.

2.11. Элементы логического стиля

Элементы логического стиля, которые также называют ***элементами фразы***, определяют логический стиль, используемый для отображения текста между контейнерными тегами. Браузеры по-разному интерпретируют этот стиль. Элементы логического стиля отображаются в одной строке с текстом (строчное отображение) и могут применяться к разделу текста или даже к одному символу. Например, элемент

[1] www.scientificamerican.com/article.cfm?id=the-semantic-web

strong указывает, что текст, связанный с ним, будет отображен в «акцентированном» виде по отношению к нормальному тексту на странице. В таблице 2.1 представлены часто встречающиеся элементы логического стиля и примеры их использования. Обратите внимание, что некоторые элементы, к примеру, cite и dfn, выполняют ту же функцию, что и элемент em в современных браузерах — начертание текста курсивом. Эти элементы семантически описывают текст как цитату или определение, но отображается он в обоих случаях обычно одинаково — курсивом.

Таблица 2.1. Элементы логического стиля

Элемент	Пример	Использование
abbr	ФБР	Обозначает текст как аббревиатуру; конфигурирует атрибут заголовка, содержащий полное название
b	**Полужирный** текст	Текст, не являющийся исключительно важным, но традиционно выделяемый полужирным шрифтом
cite	*цитированный* текст	Обозначает цитату или ссылку; как правило, отображается курсивом
code	текст **кода**	Обозначает образцы программного кода; как правило, шрифт с фиксированным пробелом
dfn	*определение*	Обозначает определение или термин; как правило, отображается курсивом
em	*выделенный* текст	Выделяет необходимый текст по отношению к остальному тексту; как правило, отображается курсивом
i	*Выделенный* текст	Текст, не являющийся исключительно важным, но традиционно выделяемый курсивом
kbd	**набираемый** текст	Обозначает текст, набираемый с клавиатуры; как правило, шрифт с фиксированным пробелом
mark	выделенный цвет	Выделенный текст, который легко заметить (только в HTML5)
samp	**образец**	Показывает пример выходных данных программы; как правило, шрифт с фиксированным пробелом
small	Маленький текст	Мелким шрифтом отображаются сведения об авторских правах и сноски
strong	**акцентированный** текст	Выделяет необходимый текст из окружающего текста; как правило, отображается полужирным начертанием
sub	Подстрочный текст	Отображает нижний индекс в виде мелкого текста ниже уровня строки
sup	Надстрочный текст	Отображает верхний индекс в виде мелкого текста выше уровня строки
var	*переменная*	Обозначает и отображает переменную или выходные данные программы; как правило, отображается курсивом

Элементы логического стиля — это контейнеры, поэтому необходимо использовать открывающий и закрывающий теги.

В таблице 2.1 указано, что элемент `strong` подчеркивает особую значимость выделенного с его помощью текста. Как правило, в браузере (или другом пользовательском агенте) текст с элементом `strong` выделяется полужирным шрифтом. Телефонный номер в следующей строке отображается полужирным шрифтом:

```
Позвоните нам и узнайте приблизительную стоимость веб-разработки: 888-555-5555
```

Код соответственно:

```
<p>Позвоните нам и узнайте приблизительную стоимость веб-разработки:<strong>888-555-5555</strong></p>
```

Обратите внимание, что открывающий (`<strong>`) и закрывающий (`</strong>`) теги заключены в теги абзаца (`<p>` и `</p>`). В данном коде вложение выполнено верно, значит он считается ***правильно оформленным***. Если вложение выполнено неверно, пары элементов `p` и `strong` перекрывают друг друга, вместо того чтобы быть вложенными один в другой. Код с неверным вложением не пройдет проверку сервисом валидации (см. раздел 2.18 «Проверка HTML-кода») и может привести к проблемам с отображением.

На рис. 2.9 показана веб-страница (см. файл *Примеры\Глава_02\em.html*), на которой элемент `em` используется для выделения курсивом фразы «Доступ для каждого».

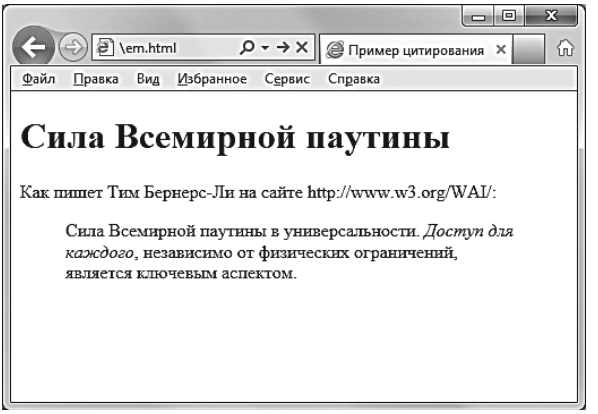


Рис. 2.9. Элемент `em` в действии

Ниже приведен фрагмент кода:

```
<blockquote>
Сила Всемирной паутины в универсальности.
<em>Доступ для каждого</em>,
независимо от физических ограничений, является
ключевым аспектом.
</blockquote>
```

2.12. Неупорядоченные списки

Списки используются на веб-страницах для организации информации. Когда вы верстаете веб-страницы, помните, что заголовки, короткие абзацы и списки упрощают восприятие и чтение контента, размещенного на вашей странице. В языке HTML используются три типа списка: ***списки определений***, ***упорядоченные списки*** и ***неупорядоченные списки***. В этом разделе рассматриваются неупорядоченные списки, также называемые маркированными списками в текстовых редакторах. В **неупорядоченных списках** отображаются маркеры перед каждым пунктом списка. Этот маркер может быть нескольких типов: диск (по умолчанию), квадрат и круг. Пример неупорядоченного списка представлен на рис. 2.10.

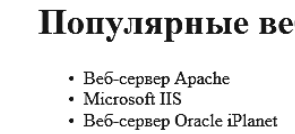


Рис. 2.10. Пример неупорядоченного списка

Неупорядоченные списки начинаются тегом `<ul>` и заканчиваются тегом `</ul>`. Каждый элемент списка начинается тегом `<li>` и заканчивается тегом `</li>`. Фрагмент кода, показанной на рис. 2.10, представлен ниже:

```
<h1>Популярные веб-серверы</h1>
<ul>
<li>Веб-сервер Apache</li>
<li>Microsoft IIS</li>
<li> Веб-сервер Oracle iPlanet</li>
</ul>
```

Атрибут type

Для смены типа маркера используется атрибут type. Например, чтобы создать неупорядоченный список, оформленный квадратными маркерами, используйте код ul **type="square"**. В таблице 2.2 представлен атрибут type и его значения для неупорядоченных списков.

Таблица 2.2. Значения атрибута type для неупорядоченных списков

Значение	Пример
circle	○
disc (по умолчанию)	●
square	▪

HTML5 и неупорядоченные списки

Атрибут type широко применяется в неупорядоченных списках и прекрасно работает в версии языка XHTML. Однако имейте в виду, что атрибут type элемента ul исключен из HTML5, поскольку выполняет исключительно декоративную функцию и не несет в себе никакого смысла. В главе 6 вы узнаете, как создавать маркеры в неупорядоченных списках с помощью CSS.

Практическое задание 2.6

В этом практическом задании вы будете использовать заголовок и неупорядоченный список на одной странице. Запустите программу Блокнот (Notepad) или другой текстовый редактор и откройте файл *Примеры\Глава_02\template.html*. Измените заголовок страницы и внесите остальные изменения в код согласно листингу ниже:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Заголовок и список</title>
<meta charset="utf-8">
</head>
<body>
<h1>Популярные веб-серверы</h1>
<ul>
<li>Веб-сервер Apache</li>
```

```
<li>Microsoft IIS</li>
<li>Веб-сервер Oracle iPlanet</li>
</ul>
</body>
</HTML>
```

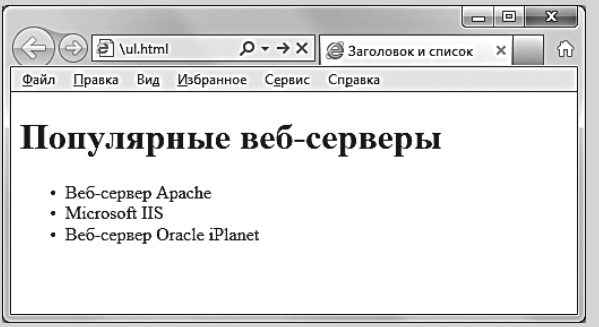


Рис. 2.11. Веб-страница, на которой размещен неупорядоченный список

Сохраните файл с именем *ul.html*. Запустите браузер и протестируйте страницу. Она должна быть похожа на показанную на рис. 2.11. Вы можете сравнить свою работу с файлом *Примеры\Глава_02\ul.html*. Выделите несколько минут, чтобы поэкспериментировать с атрибутом `type`. Установите в неупорядоченном списке квадратные маркеры. Сохраните свою страницу с именем *ulsquare.html*. Протестируйте страницу в браузере. Она должна быть похожа на файл *Примеры\Глава_02\ulsquare.html*.

ЧаВо

МОЖНО ЛИ ИСПОЛЬЗОВАТЬ ИЗОБРАЖЕНИЯ В КАЧЕСТВЕ МАРКЕРОВ НЕУПОРЯДОЧЕННОГО СПИСКА?

Можно. В главе 6 вы узнаете, как с помощью CSS создавать маркеры неупорядоченного списка и использовать для этого графические файлы.

2.13. Упорядоченные списки

Упорядоченные списки используют систему нумерации или букв для организации информации, содержащейся в списке. Упорядоченный список может быть организован с помощью чисел (по умолчанию), заглавных букв, строчных букв, заглавных римских чисел и строчных римских чисел. См. пример упорядоченного списка на рис. 2.12.

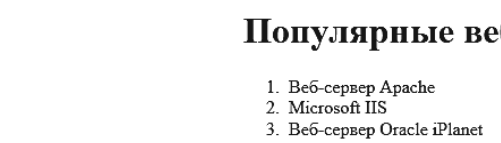


Рис. 2.12. Пример упорядоченного списка

Упорядоченные списки начинаются тегом `<ol>` и заканчиваются тегом `</ol>`. Каждый элемент списка начинается тегом `<li>` и заканчивается тегом `</li>`. HTML-код для создания упорядоченного списка, показанного на рис. 2.12, выглядит следующим образом:

```
<h1>Популярные веб-серверы</h1>
<ol>
<li>Веб-сервер Apache</li>
<li>Microsoft IIS</li>
<li>Веб-сервер Oracle iPlanet</li>
</ol>
```

Атрибут `type`

Атрибут `type` используется для изменения символа нумерации списка. Например, чтобы создать упорядоченный список, организованный заглавными буквами, используйте код `ol type="A"`. В таблице 2.3 представлены значения атрибута `type` для упорядоченных списков.

Таблица 2.3. Значения атрибута `type` для упорядоченных списков

Значение	Символ
`1`	Числа (по умолчанию)
`A`	Заглавные буквы
`a`	Строчные буквы
`I`	Римские числа
`i`	Строчные римские числа

HTML5 и упорядоченные списки

Несмотря на то, что неупорядоченный и упорядоченный списки похожи, HTML5 по-разному воспринимает их атрибут `type`. В HTML5

этот атрибут не применяется для неупорядоченных списков. Однако он действителен для упорядоченных списков, так как последовательность содержит в себе информацию.

***Атрибут* `start`** полезен, если вам нужно, чтобы список начинался с целого числа, но не с 1 (например, `start="10"`). Чтобы маркеры списка отображались в обратном порядке, примените новый ***атрибут* `reversed`** языка HTML5 (`set reversed="reversed"`).

Практическое задание 2.7

В этом практическом задании вы будете использовать заголовок и упорядоченный список на одной странице. Запустите программу Блокнот (Notepad) или другой текстовый редактор и откройте файл *Примеры\Глава_02\template.html*. Измените заголовок страницы и внесите остальные изменения в код согласно листингу ниже:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Заголовок и список</title>
<meta charset="utf-8">
</head>
<body>
<h1>Популярные веб-серверы</h1>
<ol>
<li>Веб-сервер Apache</li>
<li>Microsoft IIS</li>
<li>Веб-сервер Oracle iPlanet</li>
</ol>
</body>
</HTML>
```

Сохраните файл с именем *ol.html*. Запустите браузер и протестируйте страницу. Она должна быть похожа на показанную на рис. 2.13. Вы можете сравнить свою работу с файлом *Примеры\Глава_02\ol.html*.

Выделите несколько минут, чтобы поэкспериментировать с атрибутом `type`. Установите в неупорядоченном списке квадратные маркеры. Сохраните свою страницу с именем *ola.html*. Протестируйте страницу в браузере. Она должна быть похожа на файл *Примеры\Глава_02\ola.html*.

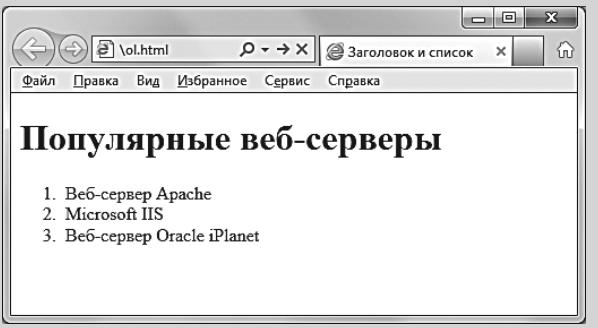


Рис. 2.13. Веб-страница, на которой размещен упорядоченный список

2.14. Списки определений

Списки определений помогают организовать термины и их определения. Термины выделяются, а их определения могут быть любой длины, необходимой, чтобы передать смысл. Каждый определяемый термин начинается на отдельной строке у границы. Каждое определение начинается на отдельной строке и имеет отступ. Списки определений также удобны для организации часто задаваемых вопросов (ЧаВо) и ответов на них. Вопросы и ответы смещаются на отступ. Любой тип информации, которая состоит из списка соответствующих терминов и более длинных определений, хорошо подходит для организации с помощью списка определений. На рис. 2.14 показан пример веб-страницы со списком определений.

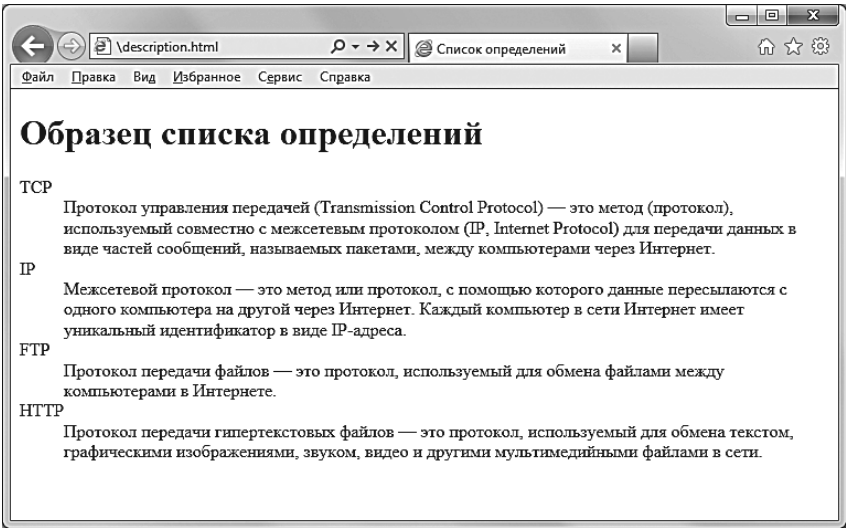


Рис. 2.14. Пример списка определений

Список определений начинается с тега <dl> и заканчивается тегом </dl>. Каждый термин списка начинается с тега <dt> и заканчивается тегом </dt>. Каждое определение термина (определение данных) начинается с тега <dd> и заканчивается тегом </dd>.

Практическое задание 2.8

В этом практическом задании вы будете использовать заголовок и список определений на одной странице. Запустите программу Блокнот (Notepad) или другой текстовый редактор и откройте файл *Примеры\Глава_02\template.html*. Измените заголовок страницы и внесите остальные изменения в код согласно листингу ниже.

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Список определений</title>
<meta charset="utf-8">
</head>
<body>
<h1>Образец списка определений</h1>
<dl>
<dt>TCP</dt>
<dd>Протокол управления передачей (Transmission Control Protocol) — это метод (протокол), используемый совместно с межсетевым протоколом (IP, Internet Protocol) для передачи данных в виде частей сообщений, называемых пакетами, между компьютерами через Интернет.</dd>
<dt>IP</dt>
<dd>Межсетевой протокол — это метод или протокол, с помощью которого данные пересылаются с одного компьютера на другой через Интернет. Каждый компьютер в сети Интернет имеет уникальный идентификатор в виде IP-адреса.</dd>
<dt>FTP</dt>
<dd>Протокол передачи файлов — это протокол, используемый для обмена файлами между компьютерами в Интернете.</dd>
<dt>HTTP</dt>
<dd>Протокол передачи гипертекстовых файлов — это протокол, используемый для обмена текстом, графическими
```

```
изображениями, звуком, видео и другими мультимедийными
файлами в сети.</dd>
</dl>
</body>
</HTML>
```

Сохраните файл с именем *description.html* и протестируйте его в браузере. Ваша страница должна быть похожа на показанную на рис. 2.14. Не волнуйтесь, если перенос слов выглядит немного иначе — главное то, что каждый термин, обозначенный элементом `<dt>`, должен располагаться на новой строке, а соответствующее определение `<dd>` должно располагаться под ним с отступом. Попробуйте изменить размеры окна браузера и обратите внимание, что перенос слов в тексте определений меняется. Вы можете сравнить свою работу с файлом *Примеры\Глава_02\description.html*.

ЧаВо

ПОЧЕМУ HTML-КОД В ПРИМЕРАХ ПРАКТИЧЕСКИХ ЗАДАНИЙ НАПИСАН С ОТСТУПАМИ?

Для браузера не имеет значения, есть ли отступы в HTML-коде, но людям легче читать и редактировать код, когда он логически отформатирован. Просмотрите список определения, созданный в практическом задании 2.8. Обратите внимание, что каждый уровень элементов `dt` и `dd` имеет отступ. Это упростит вам или другому веб-разработчику понимание исходного кода в будущем. Не существует «правил» относительно того, сколько пробелов должно стоять в отступе, хотя у вас или в организации, в которой вы работаете, могут существовать свои стандарты. Согласованные отступы помогают создавать веб-страницы, удобные в обслуживании.

2.15. Специальные символы

Чтобы использовать специальные символы, такие как кавычки, символы больше чем (>), меньше чем (<) и знак авторского права (©) в своем веб-документе, вам нужно использовать ***специальные символы***, иногда называемые ***сущностями***. Например, вы хотите включить строку про авторское право следующим образом:

```
© 2012 Моя Компания. Все права защищены.
```

Вы используете специальный символ **`&copy;`**, чтобы отобразить знак авторских прав. HTML-код будет выглядеть следующим образом:

```
<&copy; 2012 Моя Компания. Все права защищены.>
```

Другой полезный специальный символ, ** **, создает неразрывный пробел. Вы могли заметить, что веб-браузеры воспринимают несколько пробелов как один пробел. Если вам нужно указать некоторое число пробелов в тексте, вы можете использовать символ ` ` несколько раз, чтобы обозначить несколько пустых пробелов. Это приемлемо, если вам нужно всего лишь немного поправить положение элемента. Если вы обнаружите, что на ваших веб-страницах используется много специальных символов ` ` подряд, вам стоит использовать другой метод выравнивания элементов, например каскадные таблицы стилей (см. главы 4 и 6).

В таблице 2.4 и приложении Б представлены описания специальных символов и их кодов.

Таблица 2.4. Распространенные специальные символы

Символ	Имя сущности	Код
"	Кавычки	`&quot;`
‘	Закрывающая одиночная кавычка	`&rsquo;`
©	Знак авторского права	`&copy;`
&	Амперсанд	`&`
Пустой пробел	Неразрывный пробел	` `
—	Длинное тире	`&mdash;`
|	Вертикальная черта	`|`

Практическое задание 2.9

На рис. 2.15 показана веб-страница, которую вы создадите в этом задании. Запустите программу Блокнот (Notepad) и откройте файл *template.html*. Измените заголовок веб-страницы, указав между тегами `<title>` и `</title>` текст «Шаги веб-дизайна». Поскольку страница «Шаги веб-дизайна» будет довольно сильно отличаться от предыдущей созданной страницы, удалите код между тегами `<body>` и `</body>`. Сохраните файл с именем *design.html*.

Страница, показанная на рис. 2.15, содержит заголовок, неупорядоченный список и информацию об авторском праве.

Добавьте текст «Шаги веб-дизайна» в качестве заголовка первого уровня (`h1`) следующим образом:

```
<h1>Шаги веб-дизайна</h1>
```

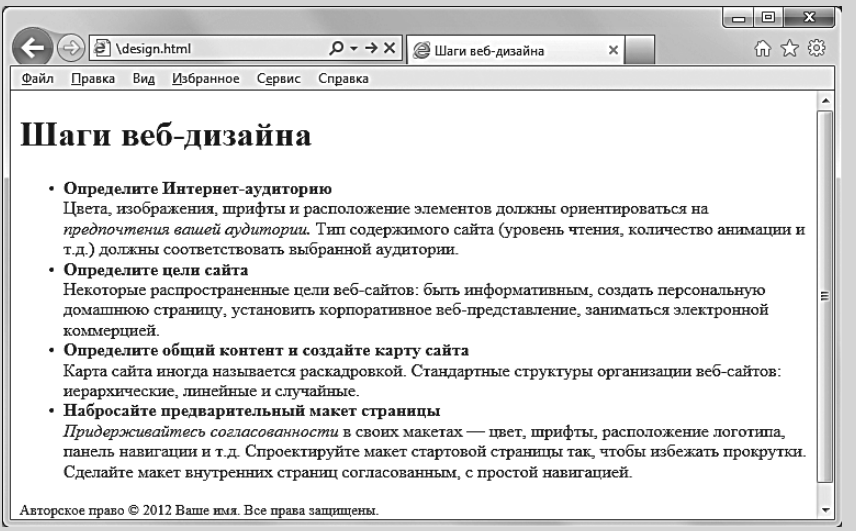


Рис. 2.15. Вид файла *design.html*

Теперь создайте неупорядоченный список. Первая строка каждого маркированного элемента является заголовком шага веб-дизайна. В примере заголовок каждого шага должен быть акцентирован, или выделяться из остального текста. Код начала неупорядоченного списка будет выглядеть следующим образом:

```
<ul>
<li><strong>Определите Интернет-аудиторию</strong><br>
Цвета, изображения, шрифты и расположение элементов должны ориентироваться на <em>предпочтения вашей аудитории.</em> Тип содержимого сайта (уровень чтения, количество анимации и т. д.) должны соответствовать выбранной аудитории.</li>
```

Сверстайте код всего упорядоченного списка. Не забудьте добавить закрывающий тег `</ul>` в конце списка. Не беспокойтесь, что перенос текста будет немного другим — разрешение экрана вашего монитора или размер окна браузера могут отличаться.

Наконец, добавьте информацию об авторском праве. Она должна быть меньше, чем остальной текст. Используйте специальный символ `&copy;` для знака авторского права. Код для строки об авторском праве будет следующим:

```
<p><small>Авторское право &copy; 2012 Ваше имя. Все права защищены.</small></p>
```

Сравните свою работу с файлом на диске (*Примеры\Глава_02\design.html*).

2.16. Элемент `div`

Элемент **`div`** формирует блочную структуру веб-страницы, добавляя пустое пространство сверху и снизу каждого блока. Элемент начинается с тега и завершается тегом . Он используется для форматирования части веб-страницы, к примеру, области логотипа, навигации или нижнего колонтитула. Этот элемент также полезен при создании раздела, содержащего другие блочные элементы: `p`, `ul`, `blockquote` и даже `div`. Далее в книге мы применим каскадные таблицы стилей (CSS) для форматирования цвета, гарнитуры шрифта и разметки элемента `div` и таких структурных элементов, как заголовки, абзацы и списки. В главе 6 описываются новые структурные элементы языка разметки HTML5, поддерживаемые в современных браузерах вместо элемента `div`, и позволяющие конфигурировать стандартные области страницы, включая элементы заголовка, навигации, нижнего колонтитула, боковой панели, статьи и раздела.

Практическое задание 2.10

В этом практическом задании вы потренируетесь применять элемент `div` при редактировании главной страницы сайта Агентства интерактивного дизайна, показанной на рис. 2.16. Запустите текстовый редактор и откройте файл *starter.html* из папки *Примеры\Глава_02.* Сохраните страницу под именем *div.html.*

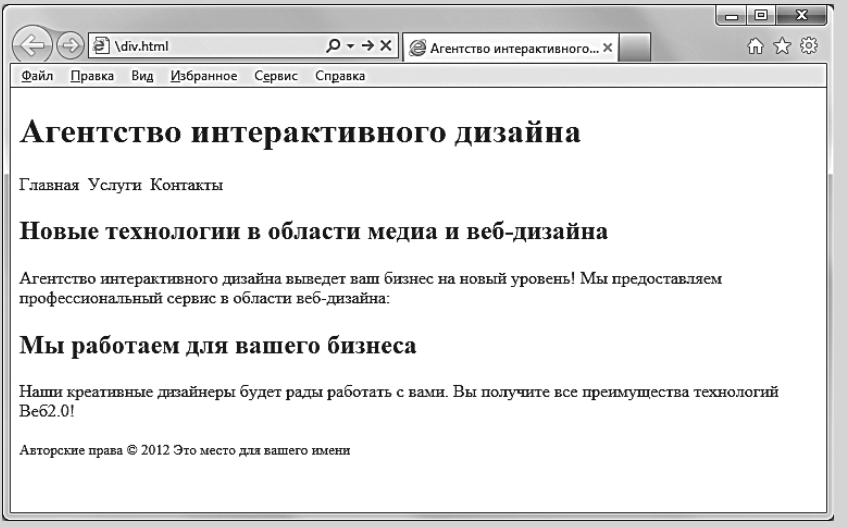


Рис. 2.16. На этой странице используется элемент `div`

Код тела страницы следующий:

```
<body>
<h1>Агентство интерактивного дизайна</h1>
<p>Главная  Услуги  Контакты</p>
<h2>Новые технологии в области медиа и веб-дизайна</h2>
<p>Агентство интерактивного дизайна выведет ваш бизнес на новый уровень! Мы предоставляем профессиональный сервис в области веб-дизайна!</p>
<h2>Мы работаем для вашего бизнеса</h2>
<p>Наши креативные дизайнеры будет рады работать с вами. Вы получите все преимущества технологий Веб2.0!</p>
<p><small>Авторские права &copy; 2012 Это место для вашего имени</small></p>
</body>
```

Просмотрите код и обратите внимание, что строка навигации и информация об авторских правах заключены в теги абзаца, хотя, строго говоря, этот контент абзацами не является. Как и элемент абзаца, `div` конфигурирует текст в виде блока с отступами сверху и снизу. Однако если речь идет не о полноценном абзаце текста, лучше применить элемент `div`. Измените код, заменив теги абзаца, в которых содержатся строка навигации и сведения об авторских правах на теги `<div>` и `</div>` следующим образом:

```
<body>
<h1>Агентство интерактивного дизайна</h1>
<div>Главная  Услуги  Контакты</div>
<h2>Новые технологии в области медиа и веб-дизайна</h2>
<p>Агентство интерактивного дизайна выведет ваш бизнес на новый уровень! Мы предоставляем профессиональный сервис в области веб-дизайна!</p>
<h2>Мы работаем для вашего бизнеса</h2>
<p>Наши креативные дизайнеры будет рады работать с вами. Вы получите все преимущества технологий Веб2.0!</p>
<div><small>Авторские права &copy; 2012 Это место для вашего имени</small></div>
</body>
```

Сохраните файл и просмотрите его в браузере. Внешне он практически не изменился, однако код изменен в лучшую сторону. С точки зрения семантики, элементы `div` лучше, чем элементы абзаца, подходят для конфигурирования областей контента, не являющих-

ся абзацами (таких как строка навигации и сведения об авторских правах). На диске, прилагающемся к книге, представлен готовый пример *Примеры\Глава_02\div.html*. Продолжая разрабатывать веб-страницы, вы поймете, что элементы `div` очень удобно использовать для создания отдельных областей страницы.

ЧаВо

ДОБАВЛЕНЫ ЛИ В HTML5 НОВЫЕ СТРУКТУРНЫЕ ЭЛЕМЕНТЫ, КОНФИГУРИРУЮЩИЕ ОБЛАСТИ ВЕБ-СТРАНИЦ?

Да. Характерная черта языка HTML5 — упор на семантику. Элемент `div`, конечно, полезный, но у него довольно широкое применение. HTML5 предлагает разнообразные структурные элементы специального назначения, такие как раздел, статья, заголовок, навигация, боковая панель и нижний колонтитул. Мы изучим эти элементы в главе 6.

2.17. Элемент привязки

Элемент привязки определяет ***гиперссылку*** (`href`) на целевую веб-страницу. Каждый элемент привязки начинается тегом `<a>` и заканчивается тегом `</a>`. Открывающий и закрывающий теги привязки окружают текст, по которому необходимо выполнить щелчок мышью для перехода по гиперссылке. Чтобы настроить гиперссылку, примените атрибут **`href`**, содержащий имя и местоположение файла, к которому предоставляется доступ. На рис. 2.17 показана веб-страница с элементом привязки, создающим гиперссылку на сайт **http://eksmo.ru/**.

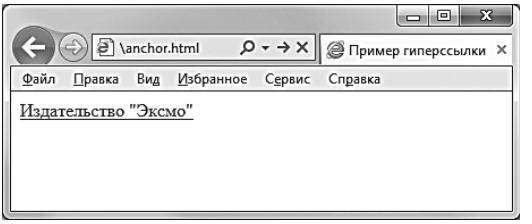


Рис. 2.17. Пример гиперссылки

Код элемента привязки следующий:

```
<a href="http://eksmo.ru/">Издательство "Эксмо"</a>
```

Обратите внимание, что значение атрибута `href` — это URL-адрес веб-сайта. Текст, введенный между двумя тегами `a`, отображается на веб-странице как гиперссылка, и в большинстве случаев подчеркивается браузером. При наведении на гиперссылку указатель мыши превращается в руку с вытянутым указательным пальцем, как показано на рис. 2.17.

Практическое задание 2.11

Для создания веб-страницы, показанной на рис. 2.17, запустите текстовый редактор. Выберите команду меню **Файл** ⇒ **Открыть** (File ⇒ Open) и откройте шаблон, хранящийся на диске, прилагающемся к книге (*Примеры\Глава_02\template.html*). Измените заголовок и добавьте в раздел тела элемент привязки, как показано в коде ниже (выделено полужирным):

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Пример гиперссылки</title>
<meta charset="utf-8">
</head>
<body>
<a href="http://eksmo.ru/">Издательство "Эксмо"</a>
</body>
</HTML>
```

Присвойте документу имя *anchor.html* и сохраните его на жестком или съемном диске.

Запустите браузер и протестируйте страницу. Она должна быть похожа на изображенную на рис. 2.17.

Сравните свою страницу с примером на диске, прилагающемся к книге (*Примеры\Глава_02\anchor.html*).

ЧаВо

МОЖЕТ ЛИ ИЗОБРАЖЕНИЕ БЫТЬ ГИПЕРССЫЛКОЙ?

Да. В этой главе мы уделяем особое внимание текстовым гиперссылкам, однако изображение также можно сделать гиперссылкой. Попрактиковаться в создании графических ссылок вы сможете в главе 4.

Абсолютные ссылки

Абсолютная гиперссылка указывает абсолютное местоположение ресурса в Интернете. Она применяется, когда необходимо сослаться на ресурсы, расположенные на других веб-сайтах. В значении атрибута `href` абсолютной гиперссылки на главную страницу веб-сайта содержится протокол http:// и доменное имя. Вот как выглядит абсолютная гиперссылка на главную страницу сайта издательства этой книги:

```
<a href="http://eksmo.ru/">Издательство "Эксмо"</a>
```

Обратите внимание, чтобы попасть на другие страницы веб-сайта книги, нужно указать название конкретной папки или имя файла. Например, этот элемент привязки создает абсолютную гиперссылку на файл *moscow.php,* хранящийся в папке *purchase\retail\books* на сайте издательства:

```
<a href="http://www.eksmo.ru/purchase/retail/
books/moscow.php">Издательство "Эксмо", магазины
в Москве</a>
```

Относительные ссылки

Когда вам нужна ссылка на веб-страницу в пределах вашего сайта, используйте ***относительную ссылку***. Значение атрибута `href` относительной гиперссылки не начинается с протокола http:// и в нем не указывается доменное имя. Значение `href` содержит только имя файла (иногда вместе с именем папки) веб-страницы, которую нужно отобразить. Гиперссылка указывает местоположение файла относительно страницы, отображаемой в данный момент. Например, если на вашем веб-сайте есть главная страница, которая называется *index.html*, и вы хотите создать ссылку на страницу *email.html*, которая находится в той же папке, что и *index.html*, HTML-код для относительной ссылки будет следующим:

```
<a href="email.html">Контакты</a>
```

Практическое задание 2.12

Лучший способ выучить язык HTML — начать верстать на нем веб-страницы. Давайте поэкспериментируем с элементом привязки и создадим веб-сайт, состоящий из трех страниц.

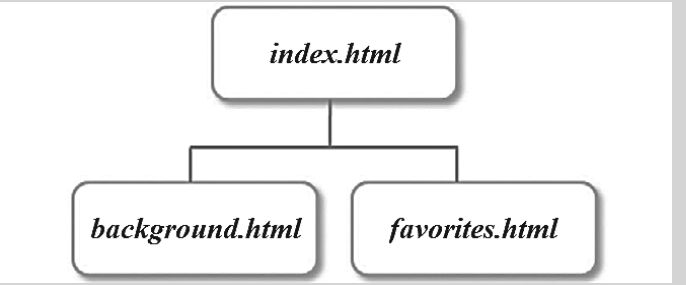


Рис. 2.18. Карта сайта

1. Создайте новую папку и назовите ее *mypractice*. Этот сайт будет примером персонального веб-сайта. Он будет содержать главную страницу *index.html* и две внутренние страницы *background.html* и *favorites.html*. Образец карты сайта (рис. 2.18) демонстрирует архитектуру сайта — главная страница (*index.html*) со ссылками на две страницы (*background.html* и *favorites.html*).

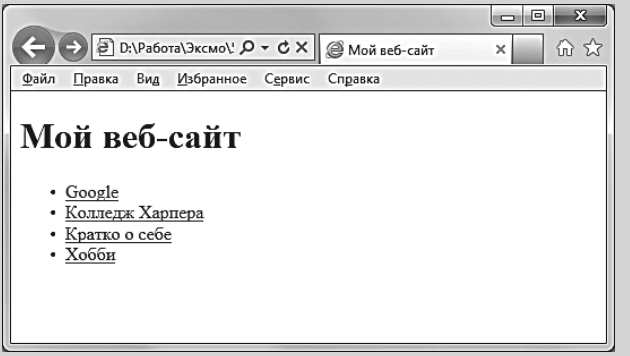


Рис. 2.19. Пример страницы *index.html*, отображенной в браузере Internet Explorer

2. Теперь создайте главную страницу для веб-сайта. Запустите программу Блокнот (Notepad) или другой текстовый редактор и откройте файл *Примеры\Глава_02\template.html*. Измените заголовок страницы и создайте следующие элементы:
   - заголовок «**Мой веб-сайт**» — используйте элементы `h1`;
   - неупорядоченный список, содержащий:
     - — абсолютную ссылку на сайт вашей любимой поисковой системы;
     - — абсолютную ссылку на веб-сайт своей школы;
     - — относительную ссылку на файл *background.html;*
     - — относительную ссылку на файл *favorites.html.*

Раздел тела вашей веб-страницы будет похож на образец кода, приведенный ниже:

```
<body>
<h1>Мой веб-сайт</h1>
<ul>
<li><a href="http://google.com">Google</a></li>
<li><a href="http://harpercollege.edu">Колледж
Харпера</a></li>
<li><a href="background.html">Кратко о себе</a></li>
<li><a href="favorites.html">Хобби</a></li>
</ul>
</body>
```

Сохраните страницу с именем *index.html* в папке *mypractice*. Отобразите страницу в браузере. Она должна быть похожа на страницу, показанную на рис. 2.19. Сравните свою работу с файлом *Примеры\Глава_02\mypractice\index.html*. Протестируйте свою страницу, щелкнув мышью по каждой ссылке. Когда вы щелкаете мышью по абсолютным ссылкам на ресурс поисковой системы и сайт вашей школы, вы должны увидеть соответствующие страницы, если ваш компьютер подключен к Интернету. Относительные ссылки пока не должны работать — далее мы создадим эти страницы.

3. Создайте страницу *background.html*. Запустите программу Блокнот (Notepad) или другой текстовый редактор и откройте файл *index.html*, созданный на предыдущем шаге. Измените заголовок и код в теле страницы, создав следующие элементы:
   - заголовок «**Кратко обо мне»** — используйте элементы `h1`;
   - абзац текста, описывающий вашу работу;
   - навигационную панель, которая содержит относительные ссылки на главную страницу (*index.html*), текущую страницу (*background.html*) и страницу «Хобби» (*favorites.html*). Необходимо добавить пустое пространство между элементами привязки, как показано на рис. 2.20. Раздел тела веб-страницы должен выглядеть, как образец кода, приведенный ниже:

```
<body>
<h1>Мой веб-сайт</h1>
<h2>Кратко о себе</h2>
<p>Как начинающий веб-дизайнер я интересуюсь темами,
связанными с принципами верстки сайтов, HTML и ка-
скадными таблицами стилей.</p>
<div><a href="index.html">Главная</a>
```

```
<a href="background.html">Кратко о себе</a>
<a href="favorites.html">Хобби</a></div>
</body>
```

Сохраните файл. Снова протестируйте страницу *index.html*. В этот раз выполните щелчок мышью по ссылке **Кратко о себе**, после чего должна отобразиться ваша новая страница. Используйте ссылку **Главная** на странице *background.html*, чтобы вернуться на главную страницу. Не волнуйтесь, если гиперссылки не заработают должным образом с первого раза. Если возникнут проблемы, внимательно изучите исходный код страниц и проверьте наличие и местоположение файлов с помощью программы Проводник (Windows Explorer) в операционной системе Windows или Finder в системе OS X.

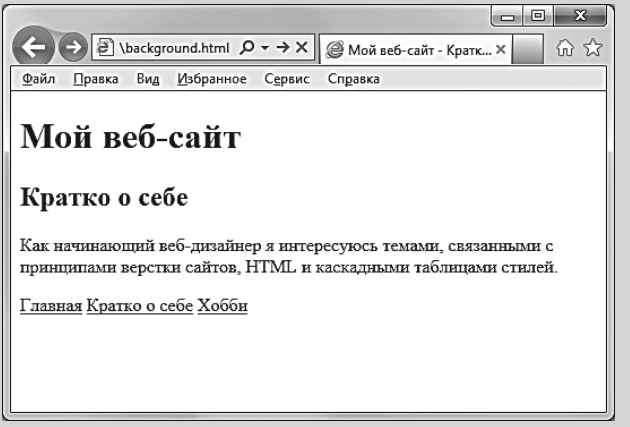


Рис. 2.20. Пример страницы background.html

4. Руководствуясь шагом 3, создайте страницу «Хобби» (*favorites.html*) и разместите на ней неупорядоченный список, содержащий список любимых занятий. См. пример в файле *Примеры\Глава_02\mypractice\favorites.html*.

ЧаВо

ЧТО ДЕЛАТЬ, ЕСЛИ ОТНОСИТЕЛЬНЫЕ ССЫЛКИ НЕ РАБОТАЮТ?

Проверьте следующее:

- Вы сохранили файлы веб-страниц в соответствующую папку?
- Вы сохранили файлы с правильными именами? Используйте программу Проводник (Windows Explorer), чтобы проверить имена файлов, которые вы сохранили.

- Вы правильно написали имена файлов в ссылках `href`? Проверьте код на опечатки.
- Когда вы установите указатель мыши на ссылку, имя файла по относительной ссылке отобразится в строке состояния в нижней части окна браузера. Убедитесь, что отображается правильное имя файла.
- В некоторых операционных системах, таких как UNIX или Linux, использование заглавных и строчных букв в именах файлов имеет значение — убедитесь, что имя файла и ссылка на него имеют одинаковый регистр. Рекомендуется всегда писать имена файлов в сети строчными буквами.

Ссылки на адрес электронной почты

Элемент привязки также можно использовать для создания ссылок на адрес электронной почты. ***Ссылка на адрес электронной почты*** автоматически запустит программу электронной почты, настроенную по умолчанию на компьютере пользователя. Значение атрибута `href` ссылки `e-mail` начинается со значения `mailto:`, а затем указывается действительный адрес электронной почты.

Например, чтобы создать ссылку на адрес электронной почты **help@terrymorris.net**, напишите следующий код:

```
<a href="mailto:help@terrymorris.net">help@
terrymorris.net</a>
```

Рекомендуется писать адрес электронной почты как на веб-странице, так и внутри элемента привязки. Не на всех компьютерах браузер настроен на работу с почтовой программой.

Помещая почтовый адрес в обеих локациях, вы повышаете юзабилити для всех посетителей.

Практическое задание 2.13

В данном практическом задании вам нужно будет модифицировать главную страницу веб-сайта из прошлого задания, добавив в область нижнего колонтитула страницы ссылку на адрес электронной почты. Запустите текстовый редактор и откройте файл *index.html*, хранящийся в папке *mypractice*. В этом примере используется файл *index.html* из папки *Примеры\Глава_02\mypractice*.

Рис. 2.21. Ссылка на адрес электронной почты, добавленная на страницу *index.html*

В нижней части страницы сконфигурируйте элемент `div`, который будет содержать текст «Контакты» и ссылку на адрес электронной почты, как показано на рис. 2.21. В качестве значения атрибута `href` укажите собственный адрес электронной почты. Сохраните страницу и протестируйте ее в браузере. Появившаяся в браузере страница должна быть похожа на изображенную на рис. 2.21. Сравните ее с примером из файла на диске, прилагающемся к книге (*Примеры\Глава_02\mypractice2\index.html*). Таким же образом измените файлы *favorites.html* и *background.html*.

ЧаВо

СТАНУ ЛИ Я ПОЛУЧАТЬ БОЛЬШЕ СПАМА, ЕСЛИ УКАЖУ СВОЙ АДРЕС ЭЛЕКТРОННОЙ ПОЧТЫ?

И да, и нет. Вполне возможно, что недобросовестные спамеры собирают адреса электронной почты на веб-страницах, однако встроенный спам-фильтр вашего почтового клиента скорее всего не позволит рассылке заполнить почтовый ящик. Размещая легкочитаемую гиперссылку на адрес электронной почты, вы повышаете юзабилити сайта для пользователей в случаях, если:

- посетитель вошел с компьютера общего пользования, на котором не установлен почтовый клиент. В этом случае, щелкнув мышью по гиперссылке с адресом электронной почты он, скорее всего, увидит сообщение об ошибке. Пользователь может не суметь связаться с вами таким образом;
- посетитель зашел на сайт с собственного компьютера, но не хочет пользоваться почтовым клиентом (или адресом), предлагаемым браузером по умолчанию. Возможно, кроме него компьютером пользуются другие люди или он стремится сохранить присвоенный по умолчанию адрес электронной почты в тайне.

Если вы четко указали свой почтовый адрес, посетитель в этих случаях все равно сможет связаться с вами (с помощью собственного почтового клиента или онлайн-сервиса электронной почты, такого как Google Gmail). В результате сайт становится более удобным для пользователей.

Привязка блока

Элементы привязки принято использовать для конфигурирования фраз или отдельных слов как гиперссылок. HTML5 вводит новую функцию элемента привязки — ***привязка блока***. Привязка блока позволяет превращать в гиперссылку один или несколько элементов (даже блочных, к примеру `div`, `h1` или абзац). Пример можно найти на диске, прилагающемся к книге (файл *Примеры\Глава_02\block.html)*.

ЧаВо

МОГЛИ БЫ ВЫ ДАТЬ НЕСКОЛЬКО СОВЕТОВ ПО ИСПОЛЬЗОВАНИЮ ССЫЛОК?

- Делайте имена ссылок содержательными и лаконичными, чтобы свести к минимуму возможную путаницу.
- Избегайте использовать фразы вроде «Щелкните здесь, чтобы» в своих ссылках. Вначале эта фраза была нужна, потому что переходы по ссылкам были непривычны для веб-пользователей. Теперь, когда Всемирная паутина стала частью повседневной жизни, такие фразы выглядят излишними и архаичными.
- Не хороните ссылки в больших блоках текста — используйте неупорядоченные списки или списки определений. Веб-страницы читать труднее, чем печатные страницы.
- Будьте осторожны, когда создаете ссылки на внешние веб-сайты. Всемирная паутина динамична, и возможно, контент внешней страницы изменится или даже она будет удалена. Если это случится, ваша ссылка станет нерабочей.

2.18. Проверка HTML-кода

На сайте W3C есть бесплатный сервис валидации разметки, расположенный по адресу **validator.w3.org/**, который проверит сверстанный вами HTML-код на наличие ошибок синтаксиса. ***Валидация HTML-кода*** дает пользователям возможность быстрой самооценки — вы можете убедить-

ся, что ваш код прошел валидацию. Сервисы валидации HTML-кода являются инструментом гарантии качества. Ошибки в коде могут привести к замедленному отображению страниц браузерами.

Практическое задание 2.14

В этом практическом задании вы воспользуетесь сервисом валидации разметки W3C, чтобы проверить веб-страницу. В примере используется страница *design.html*, созданная в практическом задании 2.8 (файл *Примеры\Глава_02\design.html*). Найдите страницу *design.html* и откройте ее в программе Блокнот (Notepad). Мы добавим ошибку на страницу *design.html*. Удалите первый закрывающий тег `</strong>`. Это изменение должно вызвать несколько сообщений об ошибках. Первое сообщение будет прямым результатом недопустимого синтаксиса.

Далее попробуем проверить файл *design.html*. Запустите браузер и посетите страницу для загрузки файлов на сервис валидации разметки — **validator.w3.org/#validate_by_upload**. Щелкните мышью по кнопке **Обзор** (Browse) и выберите файл *design.html* на своем компьютере. Щелкните мышью по кнопке **Check** (Проверка), чтобы загрузить файл на сайт (рис. 2.22).

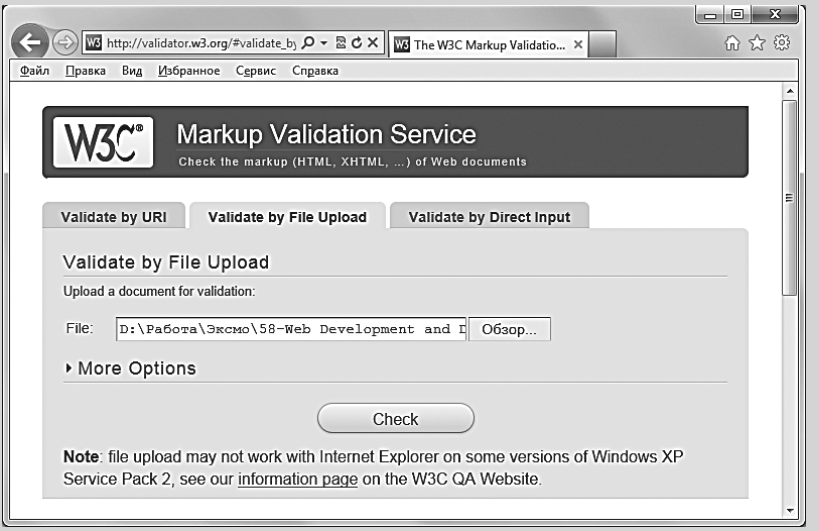


Рис. 2.22. Страница сервиса валидации разметки

Результат проверки должен выглядеть примерно так, как показано на рис. 2.23. Обратите внимание на сообщение «Errors found while

checking this document» («Во время проверки документа обнаружены ошибки»).

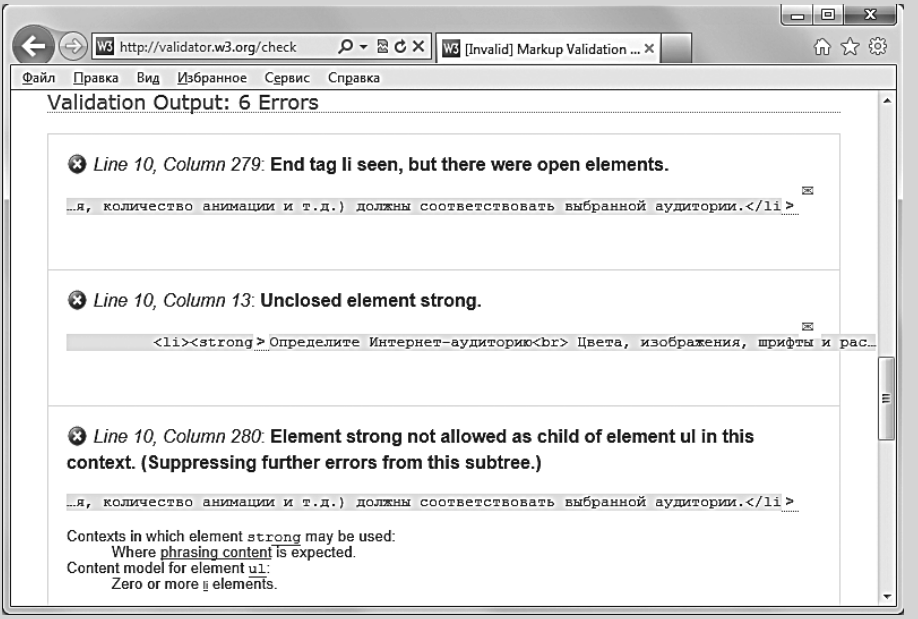


Рис. 2.23. Результаты проверки указывают на ошибки

Вы можете просмотреть ошибки, прокрутив страницу вниз, как показано на рис. 2.24.

Обратите внимание, что сообщение указывает на строку 10 — это первая строка после пропущенного тега `</strong>`. Сообщения об ошибках в разметке HTML часто указывают на строку, которая следует за ошибкой.

Текст сообщения дает понять, что на странице найдены ошибки. Ваша задача выяснить, какие именно. В первую очередь следует проверить контейнерные элементы и убедиться, что они парные. В нашем случае это и есть причина проблемы. Вы можете прокрутить страницу вниз, чтобы просмотреть остальные ошибки. Однако поскольку многие сообщения об одинаковых ошибках обычно отображаются после первой из них, правильнее исправлять их по одной, а затем проводить повторную проверку.

Отредактируйте файл *design.html* в программе Блокнот (Notepad) и добавьте пропущенный тег `</strong>`. Сохраните файл. Запустите браузер и зайдите на сайт **validator.w3.org/#validate_by_upload**. Выберите свой файл, раскройте группу **More Options** (Больше настроек) и убедитесь, что установлены флажки **Show Source** (Пока-

зать исходный код) и **Verbose Output** (Подробный вывод). Щелкните мышью по кнопке **Check** (Проверка), чтобы начать валидацию.

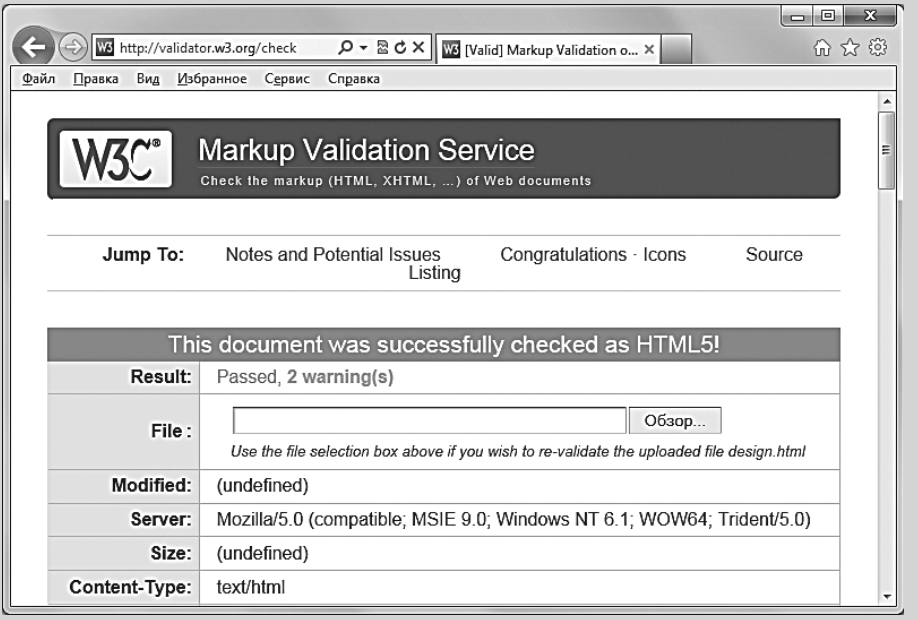


Рис. 2.24. Страница прошла проверку

Результат, показанный на экране, должен выглядеть похожим на рис. 2.24. Обратите внимание на сообщение «This document was successfully checked as HTML5!». Это означает, что ваша страница прошла валидацию на предмет соответствия спецификации HTML5. Поздравляем, ваша страница *design.html* содержит допустимый HTML5-код! Проверять страницы полезно. Однако, проверяя код, опирайтесь на здравый смысл. Поскольку браузеры еще не строго следуют рекомендациям консорциума W3C, возможны ситуации, например, при добавлении мультимедийных элементов на веб-страницу, когда HTML-код, составленный правильно для множества браузеров и платформ, не пройдет проверку.

ЧаВо

КАК ЕЩЕ МОЖНО ПРОВЕРИТЬ HTML-КОД?

Помимо валидатора консорциума W3C существуют другие инструменты, которые можно использовать для проверки синтаксиса кода. Попробуйте применить валидатор HTML5 (**HTML5.validator.nu**) и инструмент **lint.brihten.com/HTML**.

Глава 3
ИЗМЕНЕНИЕ ЦВЕТА И ТЕКСТА С ПОМОЩЬЮ CSS

Цели главы

В этой главе вы узнаете следующее:

- как таблицы стилей эволюционировали от простого средства оформления печатной продукции до инструмента создания веб-страниц;
- какими достоинствами обладает CSS;
- как настраивать цвета фона и текста на веб-страницах;
- как создавать таблицы стилей для конфигурирования цвета и других свойств текста;
- как применять внутренние таблицы стилей;
- как применять глобальные таблицы стилей;
- как применять внешние таблицы стилей;
- как настраивать в CSS селекторы элемента, класса, идентификатора и потомка (контекстные селекторы);
- понятие каскадности стилей CSS;
- как выполнять валидацию CSS.

Сейчас, когда вы уже познакомились с языком HTML, можно приступить к изучению ***каскадных таблиц стилей*** (CSS, Cascading Style Sheets). Разработчики веб-сайтов используют каскадные таблицы стилей для разделения стиля представления страницы и ее содержания. CSS используется для конфигурирования текста, цвета и структуры страницы. CSS не новы — впервые они были предложены в 1996 году в качестве стандарта W3C. В 1998 году вместе с каскадными таблицами стилей уровня 2 (CSS2) в язык были введены дополнительные свойства позиционирования элементов на веб-странице. Спецификация CSS2 существует уже более десяти лет, но статус официально рекомендованной она получила только в 2011 году. Каскадные таблицы стилей продолжают развиваться, и уже существует черновик спецификации CSS третьего уровня (CSS3), поддерживающих такие функции, как внедрение шрифтов, скругленные углы и прозрачность. Эта глава познакомит вас с тем, как использовать CSS для задания цветов и свойств текста.

3.1. Краткий обзор каскадных таблиц стилей

В течение многих лет каскадные таблицы стилей использовались в издательских системах для задания типографских стилей и интервалов в печатных изданиях. CSS предоставляет эти (и многие другие) возможности разработчикам веб-сайтов. В результате разработчики могут применять типографские стили (вид типографского шрифта, размер и пр.) на веб-страницах и изменять их макеты. На сайте CSS Zen Garden[1] приведены примеры, иллюстрирующие мощь и гибкость CSS (рис. 3.1). Посетите этот сайт, чтобы увидеть CSS в действии. Обратите внимание, насколько по-разному выглядит содержимое страницы в зависимости от выбранного дизайна (правил стилей, заданных в CSS). Хотя представленные на этом сайте дизайнерские решения созданы опытными специалистами, в каком-то смысле они так же, как и вы — начинают с основ CSS.

Рис. 3.1. Главная страница веб-сайта CSS Zen Garden

CSS — гибкий, межплатформенный, стандартизованный язык, созданный консорциумом W3C. Описание CSS можно найти на сай-

[1] www.csszengarden.com

те **www.indeep76.com/Style/ru/**. Однако следует знать, что хотя CSS и используется много лет, его все еще относят к новым технологиям и два наиболее используемых браузера до сих пор не поддерживают CSS должным образом. Настоящая глава ориентирована на те функции CSS, которые хорошо поддерживаются всеми распространенными браузерами.

Преимущества каскадных таблиц стилей

При использовании каскадных таблиц стилей вы приобретаете ряд преимуществ:

- **Более простое управление типографическими стилями и структурой страницы.** Настраиваемые параметры: размер шрифта, интервалы между строками и между символами, отступы, поля и положение элементов.
- **Стили отделены от структуры.** Форматы текста и цветов, использующихся на странице, могут конфигурироваться отдельно от основного блока веб-страницы.
- **Стили могут сохраняться.** Вы можете сохранить стили в отдельном документе и связать его с веб-страницей. Если стиль изменен, HTML-код остается неизменным. То есть если ваш клиент пожелает изменить цвет фона с красного на белый, вам нужно будет отредактировать только один файл, вместо того чтобы править каждую страницу веб-документа.
- **Может быть уменьшен размер документа.** Так как форматирование отделено от документа, реальный размер документа уменьшается.
- **Упрощается сопровождение сайта.** Повторим, что если требуется изменить стили, можно провести изменение путем корректировки только одного файла.

Возможно, вам хочется узнать, есть ли у CSS недостатки. Фактически имеется только один существенный недостаток — CSS недостаточно хорошо поддерживается браузерами. В будущем, когда браузеры будут соответствовать стандартам, этот недостаток будет устранен. Спецификация CSS2 поддерживается всеми современными браузерами. Все больше браузеров способны поддерживать и новые функции CSS3, такие как скругленные углы и прозрачность, хотя существуют отличия в синтаксисе и не все браузеры обеспечивают одинаковую визуализацию результа-

тов стилизации. Список функций CSS3, поддерживаемых различными браузерами, можно найти на странице **caniuse.com/#cats=CSS**. В этой книге особое внимание уделяется аспектам каскадных таблиц стилей, стабильно поддерживаемым современными браузерами, и указывается, в каких случаях необходим иной синтаксис.

Типы каскадных таблиц стилей

Существуют четыре способа применения технологии CSS на веб-сайтах, реализованные в виде внутренних, глобальных, внешних и импортированных таблиц (рис. 3.2).

Рис. 3.2. Каскады отдельной таблицы стилей

- **Внутренние стили** содержатся в теле веб-страницы в качестве атрибутов HTML-элементов. Эти стили применяются только к некоторым особым элементам, содержащим их в качестве атрибутов.
- **Глобальные стили** определяются в разделе заголовка веб-страницы и применяются ко всей странице.
- **Внешние стили** содержатся в специальном текстовом файле. Этот текстовый файл связывается с веб-страницей посредством элемента `link`, находящегося в разделе заголовка.
- **Импортированные стили** аналогичны внешним стилям в том, что они могут связывать стили, содержащиеся в отдельном текстовом файле, с веб-документом. Внешние таблицы стилей могут импортироваться в глобальные или другие внешние таблицы стилей с помощью директивы `@import`.

Селекторы и определения CSS

Таблицы стилей представляют собой набор правил, описывающих применяемые стили. Каждое ***правило*** состоит из ***селекторов*** и ***определений***.

- **Селектор правила стиля CSS.** Селектор может быть именем HTML-элемента, именем класса или именем идентификатора (id). В данном разделе сосредоточимся на применении стилей к селектору элемента. С селекторами класса и идентификатора мы поработаем далее этой главе.
- **Определение правила стиля CSS.** Определение указывает на устанавливаемое вами ***свойство*** (например цвет) и ***значение***, которое вы присваиваете этому свойству.

Например, правило CSS, приведенное на рис. 3.3, устанавливает синий цвет для текста, используемого на веб-странице. Селектором является элемент `body`, а определение устанавливает значение свойства `color` (цвет) равным `blue` (синий).

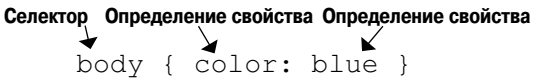


Рис. 3.3. Использование CSS для применения к шрифту синего цвета

Свойство `background-color`

Свойство **`background-color`** конфигурирует цвет фона элемента. Так, желтый фон веб-страницы создается с помощью следующего правила стилей.

Обратите внимание, что определение заключается в скобки, и свойство в нем отделяется от значения двоеточием (:).

```
body { background- color: yellow; }
```

Свойство `color`

Свойство **`color`** конфигурирует цвет текста (переднего плана) элемента. Следующее правило стилей CSS задает синий цвет текста веб-страницы:

```
body { color: blue }
```

Конфигурирование цвета фона и текста

Чтобы задать несколько свойств селектора, разделите определения точкой с запятой (;), как показано ниже:

```
body { color: blue; background-color: yellow; }
```

Пробелы между частями определения оставлять не обязательно. Также не обязателен символ точки с запятой (`;`) в конце, хотя он может оказаться полезен, если в будущем вам понадобится добавить другие правила стилей. Все приведенные ниже примеры верны:

```
body {color:blue;background-color:yellow}
body { color: blue;
background-color: yellow; }
body {
color: blue;
background-color: yellow;
}
```

Возможно, вы спрашивали себя, как узнать, какие свойства и их значения допустимы. Приложение В содержит подробный список свойств CSS. В этой главе вы познакомитесь с некоторыми из наиболее используемых свойств, применяемых CSS для задания цвета и параметров текста (табл. 3.1). В следующих главах мы рассмотрим, как задавать цвета с помощью CSS.

Таблица 3.1. Свойства CSS, используемые в этой главе

Свойство	Описание	Значения
`background-color`	Цвет фона элемента	Любой допустимый цвет
`color`	Основной цвет элемента (текста)	Любой допустимый цвет
`font-family`	Название шрифта или группы шрифтов	Любой допустимый шрифт или группа шрифтов, таких как `serif`, `sans-serif`, `fantasy`, `monospace` или `cursive`
`font-size`	Размер шрифта текста	Изменяемая величина; числовое значение задается в пунктах (`pt`), пикселах (`px`), единицах `em` (которая соответствует ширине заглавной буквы M текущего шрифта); величина в процентах; также возможны значения `xx-small`, `x-small`, `small`, `medium`, `large`, `x-large`, `xx-large`

Свойство	Описание	Значения
font-style	Стиль шрифта	normal, italic, oblique
font-weight	«Жирность» или плотность шрифта	Изменяемая величина, значения переменной normal, bold, bolder и lighter, числовые значения 100, 200, 300, 400, 500, 600, 700, 800 и 900
line-height	Допустимый межстрочный интервал	Для этой величины обычно используются значения в процентах; например, величина 200% соответствует двойному межстрочному интервалу
margin	Упрощенная запись конфигурации полей элемента	Числовое значение (px или em); например, код body { margin: 10px; } установит поля страницы в 10 пикселов. При отключении полей единицы px или em не ставятся, например, body {margin: 0}
margin-left	Задает величину левого поля элемента	Числовые значения (px или em), auto или 0
margin-right	Задает величину правого поля элемента	Числовые значения (px или em), auto или 0
text-align	Выравнивание текста	center, justify, left, right
text-decoration	Определяет, будет ли текст подчеркнут; этот стиль наиболее часто применяется для гиперссылок	При установке значения none гиперссылка не будет подчеркиваться в окне браузера
width	Ширина элемента	Числовые значения (px или em), величина в процентах или auto (величина по умолчанию)

3.2. Использование цвета на веб-страницах

Цвета, отображаемые на мониторе и представленные в виде комбинации красного, зеленого и синего цветов, называются ***цветами по модели RGB***. Значения интенсивности цветов RGB являются числами в диапазоне от 0 до 255. Каждому цвету RGB соответствуют три числа, по одному для каждого из цветовых каналов. Эти цвета обычно перечисляют именно в таком порядке (красный, зеленый, синий) и определяют значения, соответствующие каждому из используемых элементарных цветов. Например, значения указанных величин для красного цвета в системе RGB представляются в виде (255,0,0) — присутствует только красная составляющая, зеленая и синяя отсутствуют. В описании синего цвета (0,0,255) присутствует только синяя составляющая, красная и зеленая отсутствуют. Эти цвета также могут быть описаны с помощью шестнадцатеричных величин.

Шестнадцатеричные значения цветов

Шестнадцатеричная величина — это число, представленное в системе счисления с основанием 16, в которой для записи чисел используются символы 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. При использовании шестнадцатеричных величин для описания цветов RGB указываются пары цифровых величин, принимающие значения от 00 до FF (в системе с основанием 10 — от 0 до 255).

Шестнадцатеричное значение цвета содержит три пары символов, записанных последовательно как одно число. Каждая пара связывается с определенным количеством красного, зеленого и синего цветов. В этой системе красный цвет будет представляться в виде #FF0000, а синий — в виде #0000FF. Символ # обозначает, что данное число — шестнадцатеричное. В записи шестнадцатеричного значения цвета могут использоваться как заглавные, так и строчные буквы — обе записи #FF0000 и #ff0000 описывают красный цвет.

Не беспокойтесь о том, что вам придется проводить какие-либо вычисления при работе с цветами на веб-сайте. Вы можете просмотреть таблицы цветов на диске, прилагающемся к книге, в файле *Примеры\Таблицы_цветов.html*.

Веб-палитра «безопасных» цветов

Ранее, во времена компьютеров с мониторами, поддерживающими глубину цвета не более 8-бит, точное отображение цвета на странице было проблемой.

Для решения задачи была создана ***веб-палитра «безопасных» цветов***, которая представляет собой набор 216 цветов, отображающихся одинаково в операционных системах Windows и OS X. Единственные значения пар, используемые в шестнадцатеричном описании цветов веб-палитры в системе RGB, — это 00, 33, 66, 99, CC и FF. Веб-палитру «безопасных» цветов вы можете просмотреть в файле *Примеры\Таблицы_цветов.html* на диске, прилагающемся к книге. В последнее время применение веб-палитры «безопасных» цветов становится менее обязательным, так как большинство современных мониторов отображают миллионы цветов. А так как веб-палитра цветов весьма ограничена, многие веб-дизайнеры предпочитают творчески подходить к выбору цветов, а не выбирать их из ограниченной веб-палитры.

Синтаксис CSS при определении цвета

Синтаксис CSS позволяет задавать цвета несколькими способами, используя шестнадцатеричные или десятичные значения, а также имена цветов. В таблице 3.2 показан пример синтаксиса CSS для создания абзаца текста, написанного красным шрифтом.

Таблица 3.2. Примеры синтаксиса для конфигурации цвета с помощью CSS

Синтаксис CSS	Значение цвета
`p { color: red; }`	Имя цвета
`p { color: #FF0000; }`	Шестнадцатеричное значение
`p { color: #F00; }`	Укороченное шестнадцатеричное значение (по одному символу для каждой пары)
`p { color: rgb(255,0,0); }`	Десятеричное значение (тройка чисел RGB)

ЧаВо

СУЩЕСТВУЮТ ЛИ ДРУГИЕ СПОСОБЫ ЗАДАНИЯ ЦВЕТОВ С ПОМОЩЬЮ CSS?

Да, модуль CSS3 Color Module[1] (на момент написания книги присвоен статус рекомендации) позволяет веб-разработчикам конфигурировать не только цвет, но и уровень его прозрачности с помощью модели RGBA (красный, зеленый, синий, альфа). Вы изучите эту технику в главе 4.

3.3. Разметка внутренних стилей CSS

Помните, что существуют четыре способа конфигурирования CSS. Они могут быть внутренние, глобальные, внешние и импортированные. В этом разделе мы уделим внимание внутренним стилям CSS с атрибутом `style`.

Атрибут `style`

Внутренние стили размечаются в коде с помощью атрибута `style` HTML-элементов. Значение атрибута `style` — это определение прави-

[1] www.w3.org/TR/css3-color/

ла стиля, используемого для конфигурирования. Определение состоит из свойства и значения.

Свойства отделяются от соответствующих значений двоеточием (`:`). Следующий код установит красный цвет для текста, размеченного элементом `h1`:

```
<h1 style="color:#cc0000">Это будет красный
заголовок</h1>
```

Если имеется более одного свойства, они разделяются точкой с запятой (`;`). Следующий код конфигурирует заголовок с красным текстом на сером фоне:

```
<h1 style="color:#cc0000;background-
color:#cccccc">Это будет красный заголовок на сером
фоне</h1>
```

Практическое задание 3.1

На этом практическом задании вы будете форматировать веб-страницу, используя внутренние стили. Эти стили определяют следующее:

- глобальные стили элемента `body` для создания белого фона и зеленовато-голубого текста. Данные стили наследуются по умолчанию другими элементами, например:

  ```
  <body style="background-
  color:#F5F5F5;color:#008080;">
  ```

- стили элемента `h1` с зеленовато-голубым фоном и белым текстом. Данный стиль подменяет собой глобальные стили, заданные в элементе `body`, например:

  ```
  <h1 style="background-color:#008080;color:#F5F5F5;">
  ```

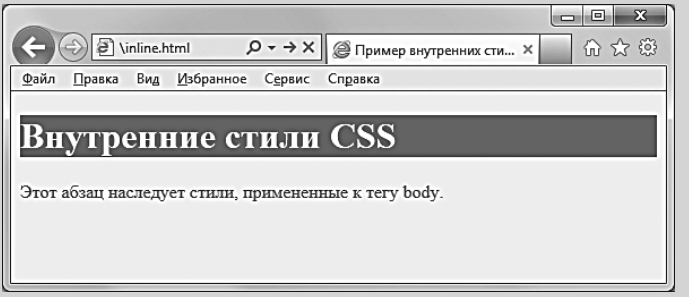


Рис. 3.4. Веб-страница, на которой используются внутренние стили CSS

Пример приведен на рис. 3.4. Откройте файл *Примеры\Глава_02\template.html* в программе Блокнот (Notepad). Внесите изменения в код согласно листингу ниже:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Пример внутренних стилей CSS</title>
<meta charset="utf-8">
</head>
<body style="background-color:#F5F5F5;color:#008080;">
<h1 style="background-color:#008080;color:#F5F5F5;">Внутренние стили CSS</h1>
<p>Этот абзац наследует стили, примененные к тегу body.</p>
</body>
</HTML>
```

Сохраните файл с именем *inline.html*. Откройте вашу страницу в браузере и сравните с рис. 3.4. Обратите внимание, что внутренние стили, применяемые к элементу `body`, наследуются другими элементами на странице (к примеру абзацами), если не установлены более специфичные стили (такие, как прописанные в элементе `h1`). Этот листинг вы найдете в файле *Примеры\Глава_03\inline.html*. Откройте данный файл, чтобы просмотреть полноцветную версию примера.

Продолжим и добавим еще один абзац текста со шрифтом серого цвета:

```
<p style="color:#333333">Этот абзац игнорирует стили, примененные к тегу body.</p>
```

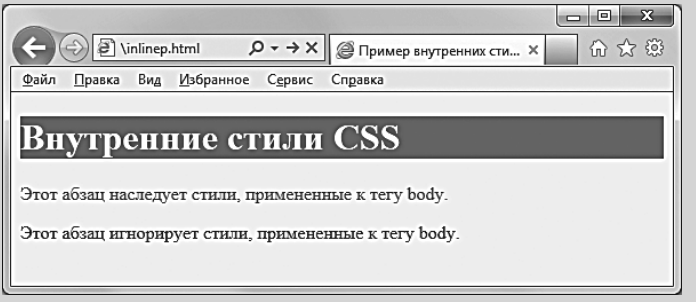


Рис. 3.5. Внутренние стили второго абзаца замещают глобальные стили, заданные в элементе `body`

Сохраните документ под именем *inline2.html*. Он должен быть похож на отображенный на рис. 3.5. Сравните свою работу с файлом *Примеры\Глава_03\inlinep.html*. Обратите внимание, что внутренние стили, примененные ко второму абзацу, замещают глобальные стили, применяемые к разделу тела веб-страницы.

ЧаВо

ИМЕЕТ ЛИ СМЫСЛ ПРИМЕНЯТЬ ВНУТРЕННИЕ СТИЛИ?

Внутренние стили могут быть полезны, однако на практике они применяются редко. Такие стили неэффективны, требуют верстки дополнительного кода и их неудобно обслуживать. Между тем в некоторых случаях внутренние стили весьма удобны, к примеру, когда при публикации статьи в системе управления контентом (CMS) или в блоге вам требуется настроить стили сайта, чтобы красиво расположить текст.

3.4. Разметка глобальных стилей CSS

На первом практическом задании вы добавляли внутренние стили к одному абзацу. Вам требовалось изменить стиль абзаца. Но что делать, если вам необходимо изменить стили не одного, а десяти или двадцати абзацев? Используя внутренние стили, вам пришлось бы проделать одну работу многократно. В отличие от внутренних стилей, применяемых к одному HTML-элементу, глобальные стили распространяются на всю веб-страницу.

Элемент `style`

Глобальные стили помещены внутри ***элемента*** **`style`**, расположенного в разделе заголовка веб-страницы. Открывающий тег `<style>` и закрывающий тег `</style>` содержат список правил глобальных стилей.

В синтаксисе XHTML для указания типа MIME в элемент `style` требуется добавить атрибут `type` со значением `"text/css"`. В синтаксисе HTML5 добавлять атрибут `type` не требуется.

На веб-странице, показанной на рис. 3.6, глобальные стили применяются для назначения цвета текста и фона документа с помощью селектора элемента `body`. См. файл *Примеры\Глава_03\embed.html*.

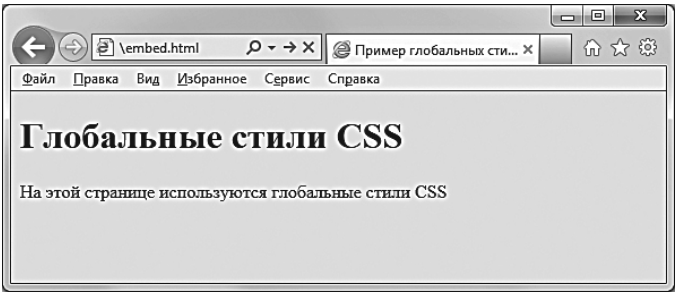


Рис. 3.6. Веб-страница, на которой используются глобальные стили

Код выглядит следующим образом:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Пример глобальных стилей CSS</title>
<meta charset="utf-8">
<style>
body { background-color: #E6E6FA;
color: #191970;
}
</style>
</head>
<body>
<h1>Глобальные стили CSS</h1>
<p>На этой странице используются глобальные стили
CSS</p>
</body>
</HTML>
```

Обратите внимание, что в коде правила стилей расположены по одному на строке. Такое форматирование не обязательно для работы программы, но оно делает код более удобным для чтения и его сопровождение становится легче, чем если бы это была одна длинная строка текста. Эти стили влияют на всю веб-страницу, так как они применяются к элементу `body`, использующему селектор элемента `body`.

Практическое задание 3.2

Запустите редактор Блокнот (Notepad) и откройте файл *starter.html* из папки *Глава_03*. Сохраните файл с именем *embedded.html* и проверьте его в браузере. Ваша страница должна выглядеть так же, как показано на рис. 3.7.

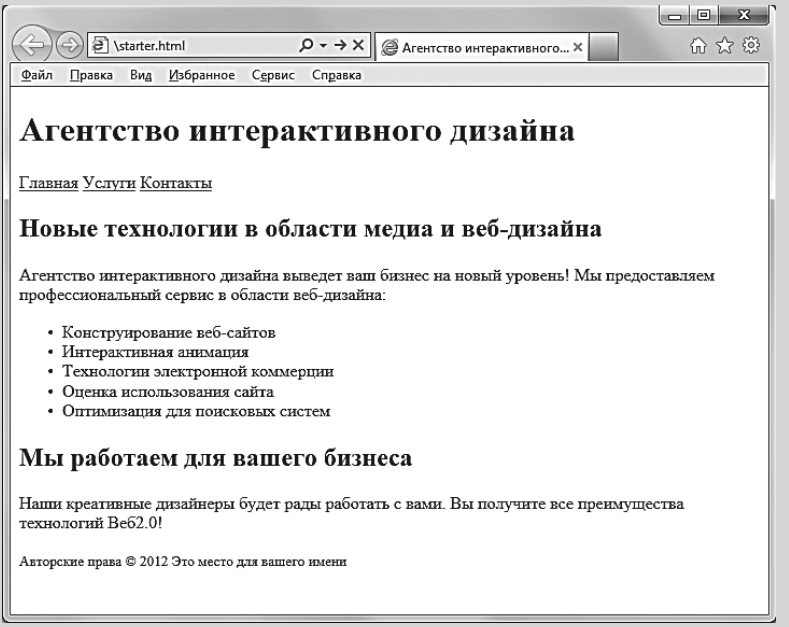


Рис. 3.7. Веб-страница, на которой не используются какие-либо стили

Откройте файл в программе Блокнот (Notepad) и рассмотрите исходный код. Обратите внимание, что в HTML-коде используются элементы `h1`, `h2`, `div`, `p`, `ul` и `li`. На этом практическом задании вы сверстаете код глобальных стилей для задания выбранных цветов фона и текста. Вы задействуете селектор `body` для изменения цвета фона (`#E6E6FA`) и текста (`#191970`) для всей страницы. Вы также будете использовать селекторы элементов `h1` и `h2` для изменения различных цветов фона и текста областей заголовка. Отредактируйте страницу *embedded.html* в редакторе Блокнот (Notepad), добавив приведенный ниже код под элементом `title` в разделе заголовка веб-страницы.

```
<style>
body { background-color: #E6E6FA;
color: #191970;
}
```

```
h1 { background-color: #191970;
color: #E6E6FA;
}
h2 { background-color: #AEAED4;
color: #191970;
}
</style>
```

Сохраните и проверьте ваш файл в браузере. На рис. 3.8 показана получившаяся веб-страница (изменения в цветовом оформлении страницы вы увидите, открыв файл в браузере). Была выбрана монохромная цветовая схема. Обратите внимание, что использование небольшого количества различных оттенков дает интересное цветовое решение и объединяет композицию сайта.

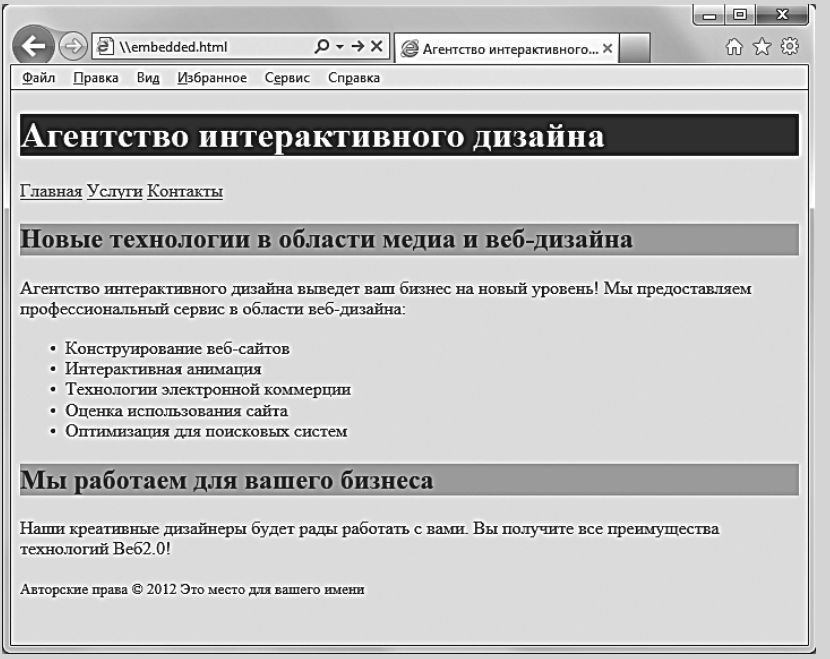


Рис. 3.8. Веб-страница *embedded.html* с использованными стилями

Рассмотрите исходный код в файле *embedded.html* (вы можете открыть файл *Примеры\Глава_03\embedded.html* на диске, прилагающемся к книге) и проанализируйте CSS- и HTML-код. Обратите внимание, что все стили размещены в одной локации на веб-странице. Так как код глобальных стилей размещается в определенной локации, впоследствии их будет проще изменять, чем внутренние стили. Также

заметьте, что вам нужно только один раз определить стиль для селектора элемента `h2` (в разделе заголовка страницы) и к *обоим* HTML-элементам `h2` будет применен этот стиль. Это намного удобнее, чем указывать одинаковые внутренние стили для каждого элемента `h2`.

Однако веб-сайт редко состоит только из одной страницы. Повторять правила CSS в разделе заголовка каждой веб-страницы неэффективно и трудоемко. В следующем разделе вы примените более эффективный подход — конфигурирование внешней таблицы стилей.

ЧаВо

МОЯ ТАБЛИЦА СТИЛЕЙ CSS НЕ РАБОТАЕТ, ЧТО ДЕЛАТЬ?

При верстке кода CSS важны детали. Существует несколько распространенных ошибок, приводящих к тому, что браузер не может правильно применять CSS на веб-странице. Тщательно проверив свой код и следуя нашим советам, вы заставите CSS работать:

- Убедитесь в том, что вы используете символы `:` и `;` в нужных позициях. Их легко перепутать. Символ `:` отделяет свойства от их значений, а `;` отделяет друг от друга комбинации *свойство*`:`*значение*.
- Проверьте, не используете ли вы символ `=` вместо `:` между свойством и его значением.
- Убедитесь, что символы `{` и `}` указаны правильно в начале и в конце правила стиля для каждого селектора.
- Проверьте допустимость синтаксиса селекторов, их использования свойств и значений свойств.
- Если часть кода CSS работает, а другая нет — просмотрите его и найдите первое неработающее правило. Часто ошибка бывает в правиле, находящемся непосредственно *перед* неработающим.
- Используйте программу для проверки CSS-кода. На сайте консорциума W3C расположен бесплатный валидатор кода CSS[1], который поможет вам найти синтаксическую ошибку. В разделе 3.9 предоставлена краткая информация о том, как использовать этот инструмент для валидации вашего кода CSS.

3.5. Изменение текста с помощью таблиц стилей

В главе 2 вы разобрались, как использовать возможности языка HTML для конфигурирования некоторых характеристик текста

[1] jigsaw.w3.org/css-validator

веб-страниц, включая логические элементы стилей (такие как элемент `strong`). Вы также изменяли цвет текста, используя свойство `color`. В этом разделе вы узнаете, как использовать CSS для изменения дополнительных характеристик текста. Форматировать текст с помощью CSS значительно удобнее, чем с использованием HTML-элементов (особенно при использовании внешних таблиц стилей, с которыми вы познакомитесь далее в этой главе), и именно этот способ предпочитают разработчики веб-сайтов.

Свойство `font-family`

Свойство **`font-family`** конфигурирует начертание шрифта. Веб-браузер отображает текст, используя шрифты, установленные на компьютере пользователя. Если задан шрифт, который не установлен на компьютере пользователя, используются шрифты, определенные по умолчанию. В большинстве браузеров по умолчанию используется шрифт Times New Roman. В таблице 3.3 показаны категории шрифтов.

Таблица 3.3. Наиболее используемые шрифты

Категория семейства шрифтов	Описание семейства шрифтов	Имена шрифтов
Serif	У шрифтов с насечками есть небольшие декоративные элементы («насечки») по краям букв. Они часто используются в заголовках	Times New Roman, Georgia, Palatino
Sans-serif	У шрифтов без насечек декоративных элементов нет. Ими обычно форматируется текст веб-страниц	Arial, Tahoma, Helvetica, Verdana
Monospace	Моноширинные шрифты часто используются при оформлении кода программы или разметки	Courier New, Lucida Console
Cursive	Рукописные шрифты; применяйте с осторожностью, так как их может быть нелегко прочитать на веб-странице	Brush Script, Comic Sans, Postino
Fantasy	Вычурные шрифты применяйте с осторожностью, так как их может быть нелегко прочитать на веб-странице	Hobo Std, Impact, MOTO

Не на всех компьютерах установлены одинаковые шрифты[1]. Предусмотрите такую ситуацию, перечислив в значении свойства `font-fami-`

[1] Список веб-безопасных шрифтов см. на странице **www.ampsoft.net/webdesign-l/WindowsMacFonts.html**

ly несколько шрифтов. Браузер применит их в указанном вами порядке. Следующая каскадная таблица стилей конфигурирует селектор элемента p для отображения текста шрифтом Arial (если установлен), Helvetica (если установлен) или заданным по умолчанию шрифтом без насечек:

```
p { font-family: Arial, Helvetica, sans-serif; }
```

ЧаВо

Я СЛЫШАЛ, ЧТО МОЖНО «ВНЕДРЯТЬ» НА СТРАНИЦЕ ОПРЕДЕЛЕННЫЕ ШРИФТЫ — ЗАЧЕМ ЭТО НУЖНО?

В каскадных таблицах стилей CSS3, находящихся пока в разработке, введено свойство `@font-face`, которое можно использовать для «внедрения» шрифтов в веб-страницы (на самом деле вы указываете местоположение шрифта, а браузер его загружает). Например, если вы имеете право бесплатно распространять шрифт MyAwesomeFont, хранящийся в файле *myawesomefont.otf* в той же папке, что и ваша веб-страница, код CSS, показанный ниже, сделает его доступным для посетителей страницы.

```
@font-face { font-family: 'MyAwesomeFont';
src: url('myawesomefont.otf'); }
```

В настоящее время браузеры поддерживают свойство `@font-face`, однако необходимо учитывать размер файла и авторские права. Можно купить шрифт, но необходимо свериться с лицензией и выяснить, имеете ли вы право бесплатно его распространять. Помните, если для области логотипа или изображения на веб-странице вам потребуется нестандартный шрифт, есть простое решение — можно использовать любой шрифт, доступный в графическом редакторе, и создать графический файл.

Дополнительные правила CSS для форматирования шрифтов

Таблицы стилей предоставляют множество вариантов форматирования текста на веб-страницах. В этом разделе вы изучите свойства `font-size`, `font-weight`, `font-style` и `line-height`.

Свойство `font-size`

Свойство **`font-size`** устанавливает размер шрифта. В этих целях доступно большое разнообразие текстовых и числовых значений

(табл. 3.4). Рекомендации по применению приведены в столбце **Примечания** табл. 3.4.

Таблица 3.4. Конфигурирование размера шрифта

Категория	Значения	Примечания
Текстовое значение	`xx-small`, `x-small`, `small`, `medium` (по умолчанию), `large`, `x-large`, `xx-large`	Хорошо масштабируется при изменении размера текста в браузере; ограниченное количество вариантов размера текста
Пикселы (`px`)	Числовое значение с указанием единицы измерения, к примеру `10px`	Отображение шрифтов, настраиваемых значениями в пикселах, зависит от разрешения экрана; может не масштабироваться в некоторых браузерах при изменении размера текста
Пункты (`pt`)	Числовое значение с указанием единицы измерения, к примеру `10pt`	Используйте для конфигурирования печатной версии веб-страницы (см. главу 7); может не масштабироваться в некоторых браузерах при изменении размера текста
Единицы em	Числовое значение с указанием единицы измерения, к примеру `.75em`	Рекомендовано консорциумом W3C; хорошо масштабируется при изменении размера текста в браузере; предоставляет множество вариантов размера текста
Процентное значение	Числовое значение в процентах, например `75%`	Рекомендовано консорциумом W3C; хорошо масштабируется при изменении размера текста в браузере; предоставляет множество вариантов размера текста

Единица em — это относительная единица, происхождение которой уходит своими корнями в индустрию печати. Когда-то принтеры настраивали для печати вручную по размеру символов. Единица em соответствует ширине квадратного блока литеры (обычно заглавной M) для конкретного шрифта и размера. В случае Всемирной паутины единица em соответствует размеру шрифта, установленного по умолчанию в браузере для конкретного (родительского) элемента (обычно это `body`). Таким образом, размер единицы em равен величине шрифта, используемого по умолчанию.

Размеры в процентах подобны единицам em. Например, коды `font-size:100%` и `font-size:1em` производят одинаковое действие. В качестве примера запустите браузер и откройте в нем файл *Примеры\Глава_03\fonts.html*, находящийся на прилагаемом к книге диске.

Свойство `font-weight`

Свойство **`font-weight`** задает «жирность» текста. Правило CSS `font-weight:bold` оказывает то же действие на текст, что и HTML-элементы `strong` и `b`.

Свойство `font-style`

Свойство **`font-style`** обычно используется для придания тексту вида курсива. Значения свойства `font-style` — `normal`, `italic` (то же действие, что и HTML-элементы `i` и `em`) и `oblique`.

Свойство `line-height`

Свойство **`line-height`** регулирует (в процентах) высоту строки текста, задаваемую по умолчанию. Например, код `line-height:200%` задает высоту строки, соответствующую двойному межстрочному интервалу.

Свойство `text-align`

HTML-элементы по умолчанию выровнены по левому краю: они начинаются у левого поля. ***Свойство*** **`text-align`** используется для задания способа выравнивания текста. Значения свойства `text-align`: `left` (по умолчанию), `right` и `center`. Следующий пример кода CSS задает выравнивание элемента `h1` по центру:

```
h1 { text-align: center; }
```

Выравнивание по центру текста, отображаемого в заголовках, — довольно эффективный прием привлечения внимания, однако с осторожностью относитесь к выравниванию по центру текста абзацев. Исследования, проведенные компанией WebAIM[1], показали, что центрированный текст менее удобочитаем, чем выровненный по левому краю.

Свойство `text-indent`

Свойство **`text-indent`** таблицы стилей задает отступ первой строки текста в элементе. Данное значение может быть числовым (измеряться, к примеру, в пикселах (`px`), пунктах (`pt`), единицах `em`) или процентным. Образец кода CSS, приведенный ниже, задает отступ первой строки всех абзацев:

```
p { text-indent: 5em; }
```

[1] www.webaim.org/techniques/textlayout

Свойство `text-decoration`

Предназначение свойства CSS `text-decoration` — изменять отображение текста.

Обычно используются следующие значения данного свойства: `none`, `underline`, `overline` и `line-through`.

Интересовались ли вы когда-нибудь, почему некоторые ссылки не подчеркнуты? Гиперссылки подчеркиваются по умолчанию, однако, применив свойство `text-decoration`, подчеркивание можно удалить. Приведенный ниже пример кода отменяет подчеркивание гиперссылки:

```
a { text-decoration: none; }
```

Практическое задание 3.3

Теперь, когда вы изучили ряд новых свойств CSS, предназначенных для форматирования шрифтов и текста, попробуем применить эти знания для изменения страницы *embedded.html*. Запустите программу Блокнот (Notepad) и откройте файл *embedded.html*. Вы напишите код дополнительных правил, определяющих вид текста на этой странице.

Установите для этой страницы свойства шрифта по умолчанию

Как вы уже видели, правила CSS, примененные к селектору `body`, распространяются на всю страницу. Измените CSS для селектора `body` так, чтобы бóльшая часть текста отображалась шрифтом без засечек. Вся страница будет отображаться этим шрифтом, если только для каких-либо из селекторов (таких как `h1` или `p`) не задан другой шрифт, класс или идентификатор[1] (о классах и идентификаторах речь будет идти дальше).

```
body { background-color: #E6E6FA;
color: #191970;
font-family: Arial, Verdana, sans-serif; }
```

Сохраните файл с именем *embedded1.html* и проверьте его в браузере. Ваша страница должна выглядеть так, как показано на рис. 3.9. Обратите внимание, что даже одна дополнительная строка может изменить внешний вид шрифтов всей страницы!

[1] Называемый также селектор `id`. — *Прим. пер.*

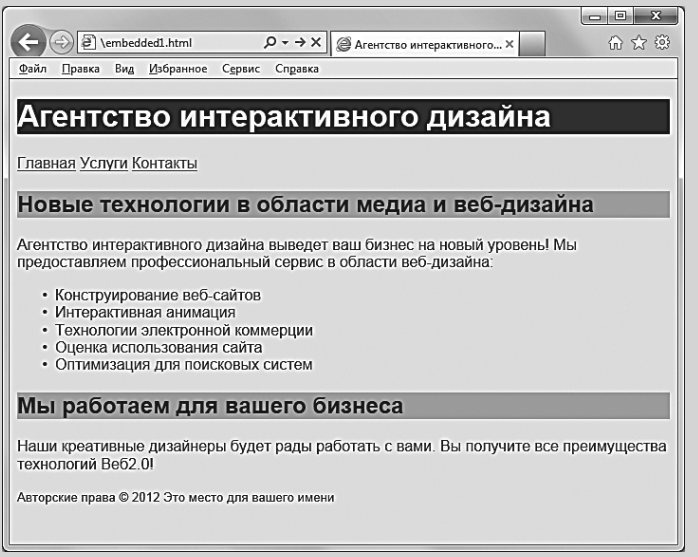


Рис. 3.9. Внешний вид текста, выполненного шрифтом без засечек

Сконфигурируйте селектор `h1`

Вы будете конфигурировать свойства `line-height` и `font-family`. Установите значение свойства `line-height` равным 200% — это немного увеличит интервалы до и после текста заголовка. (В главе 6 вы изучите другие свойства CSS, такие как отступы, границы и поля, которые все чаще используются для изменения окружения элементов.) Затем измените селектор `h1` с применением шрифта без засечек. Если имя шрифта содержит пробелы, используйте кавычки, как показано в листинге ниже. Так как общеизвестно, что блоки текста, написанные шрифтом без засечек, читаются легче, обычно именно этот шрифт используется при форматировании страниц или частей заголовков.

```
h1 { background-color: #191970;
color: #E6E6FA;
line-height: 200%;
font-family: Georgia, "Times New Roman", serif; }
```

Сохраните файл и откройте его в браузере. Если вы обнаружите, что текст «Агентство интерактивного дизайна» расположен слишком близко к левой границе, добавьте в теле страницы после открывающего тега `h1` комбинацию ` ` (специальный символ неразрывного пробела).

Сконфигурируйте селектор `h2`

Задайте правило CSS для использования той же самой гарнитуры шрифта, что и в селекторе `h1`, и отображения текста по центру.

```
h2 { background-color: #AEAED4;
color: #191970;
font-family: Georgia, "Times New Roman", serif;
text-align: center; }
```

Отредактируйте текст абзацев

Отредактируйте HTML-код и удалите элемент разрыва строки после первого предложения каждого абзаца; эти разрывы строк смотрятся некрасиво. Сконфигурируйте текст абзаца таким образом, чтобы он выглядел лишь немного меньшим, чем шрифт по умолчанию. Для этого установите значение свойства `font-size` равным `.90em`. Конфигурируйте отступ первой строки каждого абзаца. С помощью свойства `text-indent` задайте величину отступа равную `.3em`.

```
p { font-size: .90em;
text-indent: 3em; }
```

Сконфигурируйте неупорядоченный список

Отформатируйте текст неупорядоченного списка таким образом, чтобы он выглядел полужирным.

```
ul { font-weight: bold; }
```

Сохраните файл с именем *embedded2.html* и проверьте его в браузере. Ваша страница должна выглядеть так, как показано на рис. 3.10. Для примера вы можете просмотреть файл *Примеры\Глава_03\embedded2.html*.

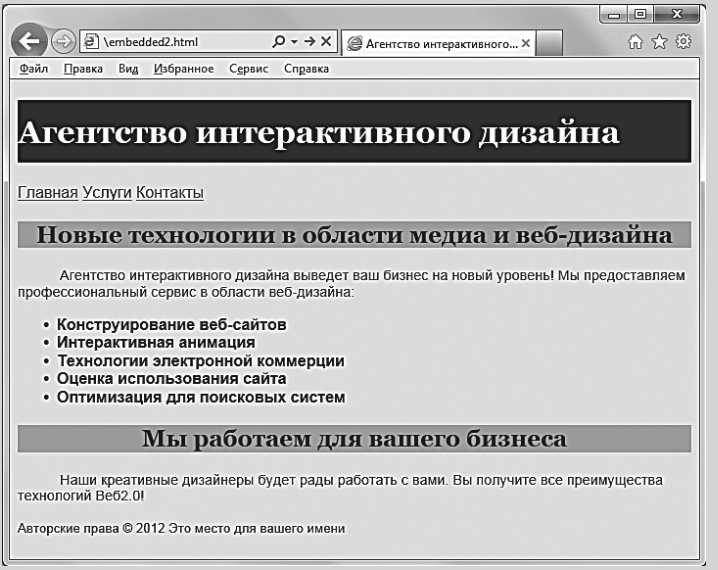


Рис. 3.10. Правила CSS конфигурируют свойства цветов и текста веб-страницы

CSS — весьма мощное средство: всего несколько строк кода существенно изменяют внешний вид веб-страницы. Вам, наверное, интересно, имеются ли в CSS другие средства для настойки? Например, вы не хотите, чтобы все абзацы выглядели одинаково. Вы можете использовать внутренние стили, но обычно это не самое эффективное средство. Следующий раздел познакомит вас с классами и идентификаторами, которые широко используются для конфигурирования специальных элементов страницы.

ЧаВо

СУЩЕСТВУЮТ ЛИ СРЕДСТВА ДЛЯ БЫСТРОГО НАЗНАЧЕНИЯ ОДНОГО И ТОГО ЖЕ СТИЛЯ БОЛЕЕ ЧЕМ ОДНОМУ HTML-ЭЛЕМЕНТУ ИЛИ БОЛЕЕ ЧЕМ ОДНОМУ КЛАССУ?

Да, вы можете применить одно и то же правило стиля нескольким селекторам (таким как HTML-элементы, классы или идентификаторы), перечислив селекторы перед правилом. В приведенном ниже образце кода размер шрифта 1em одновременно применяется к элементам абзац и строка:

```
p, li { font-size: 1em; }
```

3.6. Селекторы класса, идентификатора и потомка

Селекторы класса

Селектор класса, также называемый ***классом***, используется, когда требуется применить правила CSS к определенному классу элементов на веб-странице, но не связывать стиль с конкретным HTML-элементом. Обратите внимание, что последние два элемента неупорядоченного списка отображаются другим цветом. Это пример использования класса. Устанавливая стиль для класса, в качестве селектора задайте имя класса. В таблице стилей перед именем класса поставьте точку (`.`). Приведенный ниже код конфигурирует в таблице стилей класс `feature` с текстом красного цвета:

```
.feature { color: #FF0000; }
```

Стиль, заданный для нового класса, может применяться к любому HTML-элементу. Это делается с помощью ***атрибута класса***, такого как `class="feature"`. Не указывайте точку перед значением класса

в HTML-элементе, в котором он применяется. Следующий код задаст стиль класса `feature` элементу `li`:

```
<li class="feature">Оценка использования сайта</li>
```

Селекторы идентификатора

Используйте ***селекторы идентификаторов*** (или просто ***идентификаторы***), если вы планируете составить и применить правила CSS к единственной области веб-страницы. Например, вы хотите, чтобы абзац в нижней части (в нижнем колонтитуле) страницы *embedded2.html*, содержащий информацию об авторском праве, отображался мелким курсивом серого цвета (`#333333`). Так как на этой странице мог использоваться класс, для решения поставленной задачи более подходит идентификатор, если на вашей странице один нижний колонтитул. Например, вы можете создать стиль для идентификатора с именем `footer` для применения в области колонтитула мелкого курсива. Для применения стиля к идентификатору укажите символ # перед его именем в таблице стилей:

```
#footer { color: #333333;
font-size: .75em;
font-style: italic; }
```

Стиль, заданный для идентификатора `footer`, может применяться к любому HTML-элементу, к которому вы захотите его применить посредством использования ***атрибута идентификатора*** следующим образом `id="footer"`. Не указывайте символ # перед значением идентификатора в открывающем теге. Следующий код применяет стили идентификатора `footer` к элементу `div`:

```
<div id="footer">Этот абзац будет визуализироваться
с использованием стиля идентификатора footer</div>
```

Применение CSS с селектором идентификатора аналогично применению CSS с селектором класса. Обычно идентификатор используется для одиночного HTML-элемента, а класс — нескольких HTML-элементов.

Практическое задание 3.4

На этом практическом задании вы измените код CSS и HTML на странице Агентства интерактивного дизайна, настраивая области нави-

гации и колонтитула. Откройте файл *embedded2.html* в программе Блокнот (Notepad).

Сконфигурируйте область навигации

Область навигации будет акцентированной, если представить ее более крупным и жирным шрифтом. Создайте класс с именем `nav` и установите свойства `font-size` и `font-weight`.

```
.nav { font-weight: bold;
font-size: 1.25em; }
```

Измените открывающий тег `div` в области навигации. Добавьте атрибут класса, который связывает абзац (элемент `div`) с классом `nav` следующим образом:

```
<div class="nav"><a href="index.html">Главная</a>
<a href="services.html">Услуги</a>
<a href="contact.html">Контакты</a></div>
```

Сконфигурируйте область контента

Агентство интерактивного дизайна хотело бы обратить внимание пользователей на улучшения в юзабилити сайта и на предлагаемые компанией услуги по поисковой оптимизации. Создайте класс с именем `feature`, задающий красный цвет текста (`#ff0000`).

```
.feature { color: #ff0000; }
```

Измените последние два элемента неупорядоченного списка. Добавьте атрибут класса к открывающим тегам `li`, связав элементы списка с классом `feature`, как показано ниже:

```
<li class="feature">Оценка использования сайта</li>
<li class="feature">Оптимизация для поисковых систем</li>
```

Сконфигурируйте колонтитул

Создайте идентификатор с именем `footer`, задающий свойства `font-size` и `font-#footer { color: #333333;`

```
font-size: .75em;
font-style: italic; }
```

Измените открывающий тег `div` в области колонтитула. Добавьте атрибут идентификатора, который связывает абзац (элемент `div`) с идентификатором следующим образом:

```
<div id="footer">Авторские права &copy; 2012 Это место
для вашего имени</div>
```

Сохраните файл с именем *embedded3.html* и проверьте его в браузере. Ваша страница должна выглядеть так, как показано на рис. 3.11. Файл *embedded3.html* вы найдете на диске, в каталоге *Примеры\Глава_03*. Запомните, каким образом задаются стили класса и идентификатора.

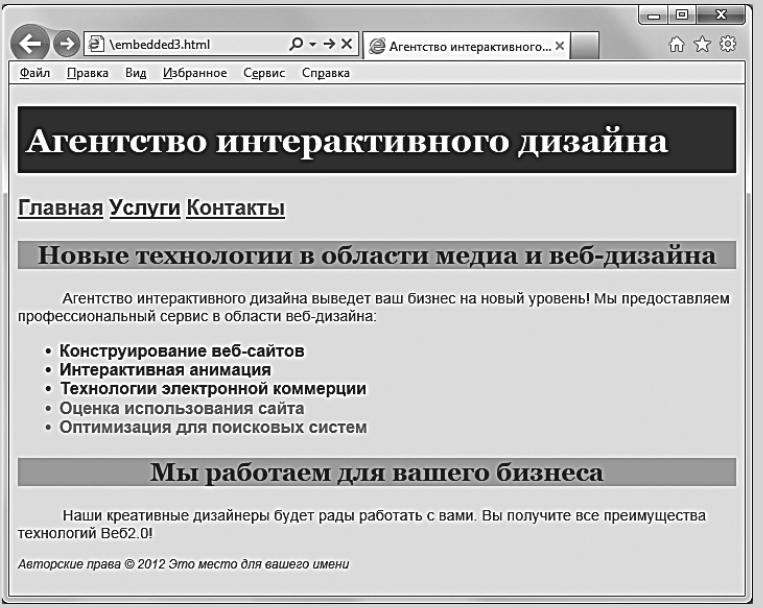


Рис. 3.11. Использование классов и идентификаторов

ЧаВо

КАК ПРАВИЛЬНО ИМЕНОВАТЬ КЛАССЫ И ИДЕНТИФИКАТОРЫ?

Можно использовать почти любые имена для классов CSS. Однако классы будут более описательными и их легче будет обслуживать в будущем, если они описывают структуру, а не способ ее форматирования. Например, название класса `largeBold`[1] потеряет свою наглядность, если дизайн страницы изменится таким образом, что эта область будет отображаться иначе, в то же время классы, имена которых отражают содержимое класса, такие как `logo`, `footer`, `content` или `subheader`, сохраняют свою наглядность вне зависимости от того, как эта область сконфигурирована. Другие советы по выбору имен классов:

- используйте короткие, но описательные имена;
- можете использовать как латинские буквы, так и цифры;
- избегайте пробелов внутри имен классов;
- имена классов не чувствительны к регистру, но стоит быть последовательным для того, чтобы обслуживание страницы было проще.

Последний совет, касательно классов CSS, — старайтесь не увлекаться. Не нужно создавать новый класс каждый раз, когда требует-

[1] В пер. с англ. — крупный, полужирный.

ся немного изменить текст. Заранее решите, как конфигурировать области страницы, укажите в коде классы и примените их. В результате получится организованная веб-страница.

Селекторы потомка

Применяйте ***селекторы потомка*** (или ***контекстные селекторы*** CSS для указания элемента в контексте его контейнера (или родительского элемента). Чтобы задать селектор потомка, укажите сначала селектор контейнера (им может быть селектор элемента, класса или идентификатора), а затем конкретный селектор, стиль которого вы меняете. Например, чтобы задать зеленый цвет текста только для элементов привязки, находящихся *внутри* идентификатора `footer`, созданного ранее, укажите в коде следующее правило стиля:

```
#footer a { color: #00ff00; }
```

Вы сможете попрактиковаться в создании селекторов потомка в главах 6 и 7. Следующий раздел посвящен новому HTML-элементу, который удобно использовать для конфигурирования областей страницы.

3.7. Элемент `span`

Как вы узнали из главы 2, элемент `div` конфигурирует секцию или раздел на веб-странице с пустыми строками снизу и сверху. Элемент `div` полезен, если вам нужно форматировать секцию, отделенную от остальной части страницы переносами строки. В противоположность сказанному выше, ***элемент* `span`** определяет раздел, физически *не* отделенный символами переноса строки от остальной страницы. Используйте элемент `span`, если вам требуется отформатировать область, расположенную внутри другой области, такой как `p`, `blockquote`, `li` и `div`.

Практическое задание 3.5

На этом практическом задании вы будете экспериментировать с элементом `span` для форматирования названия компании. Вы создадите новый класс и примените этот класс с помощью элемента `span`. Откройте файл *embedded3.html* в редакторе Блокнот (Notepad). После изменений ваша страница будет выглядеть так, как показано на рис. 3.12.

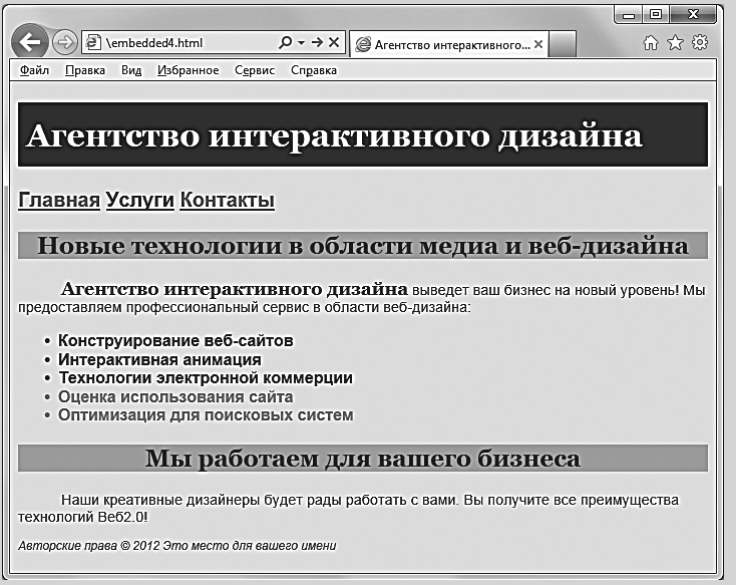


Рис. 3.12. Веб-страница, на которой использован элемент `span`

Сконфигурируйте название компании

Посмотрите на рис. 3.12 и обратите внимание на то, что наименование компании — Агентство интерактивного дизайна — выделено внутри абзаца жирным шрифтом с засечками. Для реализации этого форматирования вы напишите как CSS, так и HTML-код. Сначала создайте новое CSS-правило, которое конфигурирует класс с названием `companyname`, отформатированным полужирным шрифтом с засечками, с размером 1.25 em. Получится следующий код:

```
.companyname { font-weight: bold;
font-family: Georgia, "Times New Roman", serif;
font-size: 1.25em; }
```

Затем измените начало первого абзаца, используя элемент `span` для задания класса следующим образом:

```
<p><span class="companyname">Агентство интерактивного дизайна</span> выведет...
```

Сохраните файл и проверьте его в браузере. Страница должна выглядеть так, как показано на рис. 3.12. Также вы можете просмотреть файл *Примеры\Глава_03\embedded4.html*. Просмотрите исходный код страницы и проверьте код CSS и HTML. Обратите внимание, что все стили расположены на странице в одной локации. Поскольку глобальные стили указываются в определенном месте, их проще поддерживать в рабочем состоянии с течением времени, чем вну-

тренние стили. Также заметьте, что достаточно указать стили для селектора элемента `h2` только один раз (в разделе заголовка), и стиль `h2` будет применен к обоим элементам `h2`. Однако веб-сайты редко состоят только из одной страницы. Повторять правила CSS в разделе заголовка каждой страницы неэффективно и трудоемко. В следующем разделе вы познакомитесь с более эффективным подходом — конфигурированием внешней таблицы стилей.

3.8. Использование внешних таблиц стилей

Внешняя таблица стилей содержится в текстовом файле отдельно от HTML-документов. Элемент `link` — это автономный элемент, используемый в заголовках HTML-документов и обеспечивающий связь таблицы стилей с веб-страницей. Это дает возможность нескольким веб-страницам ссылаться на один и тот же файл внешней таблицы стилей. Файлы внешних таблиц стилей сохраняются с расширением *.css* и содержат только данные таблиц стилей и никаких HTML-элементов.

Достоинством этого подхода является то, что все стили конфигурируются одним файлом. То есть если необходимо провести изменения каких-либо стилей, изменяется только один файл, а не множество веб-страниц. На больших сайтах это позволяет веб-дизайнеру сэкономить массу времени, повышая производительность. Мы выполним несколько упражнений с этой полезной технологией.

Элемент `link`

Элемент **`link`** связывает внешнюю таблицу стилей с веб-страницей. Он указывается в разделе заголовка и представляет собой отдельный пустой элемент. В синтаксисе HTML5 элемент `link` размечается как `<link>`. В синтаксисе XHTML используется разметка `<link />`. С элементом применяются три атрибута: `rel`, `href` и `type`.

- значение атрибута **`rel`** — `stylesheet`;
- значение атрибута **`href`** — имя файла таблицы стилей;
- значение атрибута **`type`** — `"text/css"` (MIME-тип для CSS). Атрибут `type` не обязателен в HTML5, но необходим в XHTML.

Добавьте в раздел заголовка страницы следующий код, чтобы связать документ с файлом внешней таблицы стилей под именем *color.css*:

```
<link rel="stylesheet" href="color.css">
```

Практическое задание 3.6

Давайте попробуем применить внешние стили. Сначала создайте внешнюю таблицу стилей. Затем конфигурируйте веб-страницу, которая будет связана с этой таблицей стилей.

Создайте внешнюю таблицу стилей

Запустите текстовый редактор Блокнот (Notepad) и введите правила стиля, устанавливающие синий цвет фона и белый цвет текста. Сохраните файл с именем *color.css*. Код выглядит следующим образом:

```
Body {background-color: #FF0000;
color:#FFFFFF; }
```

На рис. 13.13 показан код внешней таблицы стилей *color.css* в программе Блокнот (Notepad). Обратите внимание, что в файле нет HTML-кода. Во внешней таблице стилей не прописываются HTML-элементы. В ней указываются только правила CSS (селекторы, свойства и значения).

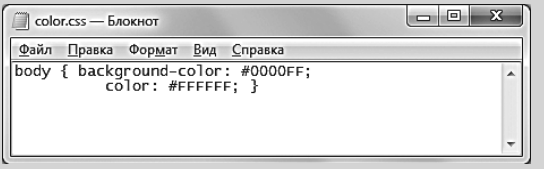


Рис. 3.13. Внешняя таблица стилей *color.css*

Для создания веб-страницы, показанной на рис. 3.14, откройте файл *Примеры\Глава_02\template.html* в программе Блокнот (Notepad). Измените элемент заголовка, добавьте элемент `link` в раздел заголовка и абзац — в раздел тела страницы, как показано в коде ниже (выделено полужирным):

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Внешние стили</title>
<meta charset="utf-8">
<link rel="stylesheet" href="color.css">
</head>
<body>
<p>На этой странице используется внешняя таблица стилей.</p>
</body>
</HTML>
```

Сохраните файл с именем *external.html* в той же папке, что и *color.css*. Проверьте файл в браузере. Страница должна выглядеть, как на рис. 3.14. Сравните свою страницу с файлом *Примеры\Глава_03\external.html*.

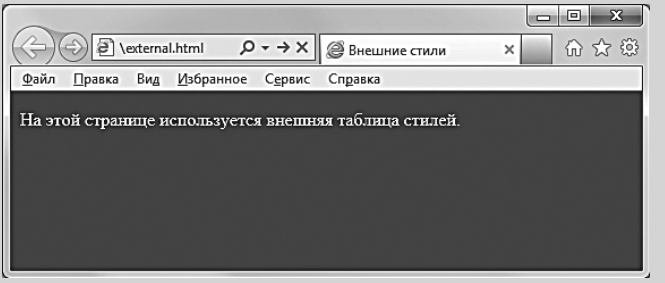


Рис. 3.14. Страница, связанная с внешней таблицей стилей *color.css*

Файл *color.css* может быть связан с любым количеством веб-страниц. Если вам понадобится изменить этот стиль форматирования, вам нужно будет внести коррективы только в этот файл (*color.css*), а не множество файлов (каждой страницы). Как уже говорилось ранее, этот метод может заметно повысить производительность при работе с многостраничными сайтами.

Это лишь маленький пример, но выигрыш от возможности иметь дело только с одним файлом заметен и для больших и для маленьких сайтов. На следующем практическом задании вы будете тренироваться в применении внешних стилей, изменяя главную страницу сайта агентства интерактивного дизайна.

Практическое задание 3.7

На этом практическом задании мы продолжим накапливать опыт в применении внешних таблиц стилей, создав файл внешней таблицы стилей с названием *trillium.css*. Мы изменим главную страницу сайта Агентства интерактивного дизайна, используя вместо глобальных стилей внешние, и свяжем вторую веб-страницу с внешней таблицей стилей *trillium.css*.

Вариант главной страницы сайта агентства интерактивного дизайна представлен на диске, прилагающемся к книге, в каталоге *Примеры\Глава_03*. Откройте файл *embedded4.html* в браузере. Вид страницы должен быть таким же, как на рис. 3.12 из практического задания 3.5.

Теперь, когда вы увидели, над чем мы будем работать, можно начинать. Запустите редактор Блокнот (Notepad) и сохраните ваш файл

с именем *index.html* в папке с названием *trillium*. Вы готовы преобразовать глобальную CSS во внешнюю.

Выделите правила CSS (все строки между открывающим и закрывающим тегами `style`, но не включая их). Выберите команду меню **Правка** ⇒ **Вырезать** (Edit ⇒ Cut) для удаления и копирования кода CSS в буфер обмена. Вы перенесете код CSS в новый файл. Запустите редактор Блокнот (Notepad) и вставьте код из буфера обмена (выбрав команду меню **Правка** ⇒ **Вставить** (Edit ⇒ Insert) или нажав сочетание клавиш **Ctrl+V**), а затем сохраните файл с именем *trillium.css*. На рис. 3.15 показан снимок этого файла в редакторе Блокнот (Notepad). Обратите внимание, что в файле *trillium.css* нет HTML-элементов, даже `style`. Файл содержит только правила CSS.

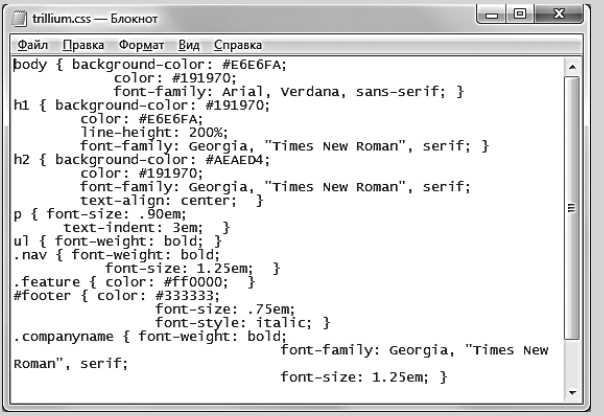


Рис. 3.15. Внешняя таблица стилей *trillium.css*

Теперь отредактируйте файл *index.html* в редакторе Блокнот (Notepad). Удалите закрывающий тег `</style>`. Замените открывающий тег `<style>` элементом `<link>`, чтобы связать страницу с таблицей стилей *trillium.css*. Код элемента `link` имеет вид:

```
<link href="trillium.css" rel="stylesheet">
```

Сохраните файл и проверьте его в браузере. Вид страницы должен быть таким же, как показано на рис. 3.12. Хотя внешний вид страницы не изменился, произошли изменения кода — на странице теперь используются внешние стили, а не глобальные.

Теперь вы свяжете вторую веб-страницу с этой же внешней таблицей стилей. В папке *Примеры\Глава_03* на диске содержится страница *services.html*. При просмотре этого файла в браузере страница будет выглядеть так, как изображено на рис. 3.16. Обратите внимание, что хотя структура страницы аналогична структуре главной страницы, стили текста и цветов на ней отсутствуют.

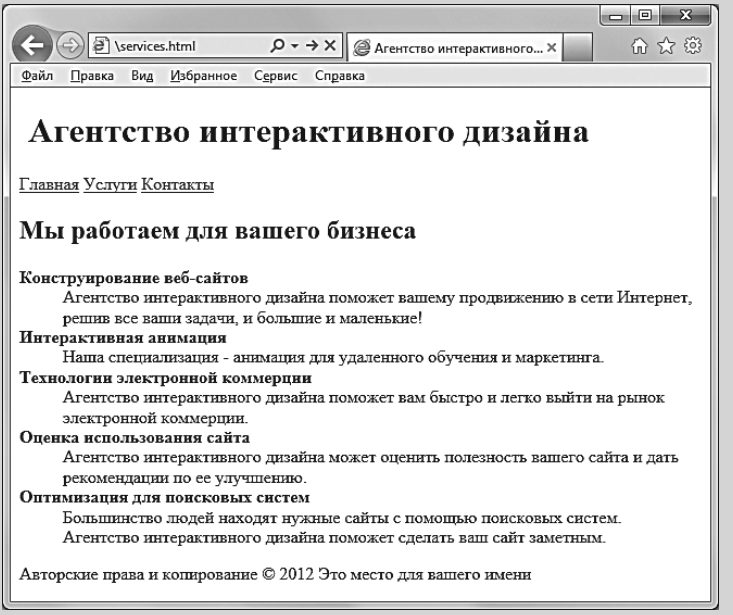


Рис. 3.16. Веб-страница *services.html* не связана с внешней таблицей стилей

Запустите редактор Блокнот (Notepad) и откройте в нем файл *services.html*. Если вы взгляните на HTML-код, то увидите, что эта страница готова для применения стилей из файла *trillium.css:* например, класс `nav` и идентификатор `footer` задействованы как атрибуты соответствующих областей — области навигации и колонтитула. Все, что осталось сделать вам, — это задействовать элемент `link`, чтобы связать веб-страницу *services.html* с внешней таблицей стилей *trillium.css*. Поместите следующую строку кода в раздел заголовка выше закрывающего тега `</head>`:

```
<link href="trillium.css" rel="stylesheet">
```

Сохраните файл и проверьте его в браузере. Вид страницы должен быть таким же, как на показано рис. 3.17 — правила CSS были применены!

Если в браузере щелкнуть мышью по гиперссылкам **Главная** и **Услуги**, можно переходить от страницы *index.html* к странице *services.html* и обратно. Этот пример содержится на диске, прилагающемся к книге, в папке *Примеры\Глава_03\3.7*.

Обратите внимание, что при применении внешних таблиц стилей, когда необходимо изменить цвета или шрифты на странице, изменения нужно проводить только во внешних таблицах стилей. Представьте себе, насколько более предсказуемым в этом случае станет сайт с большим количеством страниц. При изменении цвета или

шрифта вместо настройки сотен страниц нужно будет скорректировать только один файл — внешние таблицы стилей CSS. Очень важно, чтобы вы по мере приобретения опыта и повышения квалификации начинали свободнее ориентироваться в правилах разметки CSS.

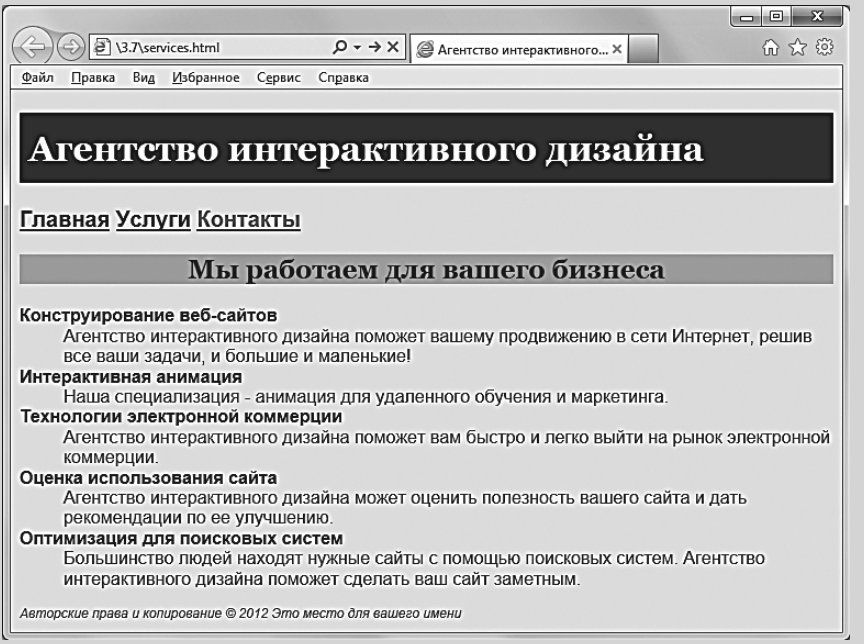


Рис. 3.17. Веб-страница *services.html* связана с внешней таблицей стилей

ЧаВо

ЕСЛИ Я НАЧИНАЮ КОНСТРУИРОВАТЬ НОВУЮ ВЕБ-СТРАНИЦУ ИЛИ САЙТ, С ЧЕГО МНЕ НАЧАТЬ?

При конструировании веб-страницы с использованием CSS полезно следовать таким советам:

- Проанализируйте дизайн страницы — убедитесь, что используются распространенные на компьютерах шрифты. Определите общие свойства (свойства по умолчанию для всей страницы) для таких компонентов, как шрифты и цвета, назначенные селектору элемента `body`.
- Определитесь с типовыми элементами структуры страницы (такими как `h1`, `h3` и т. д.) и опишите их стили, если они отличаются от используемых по умолчанию.
- Выделите различные области страницы, такие как логотип, область навигации, колонтитул и т. д. — и составьте список спе-

циальных конфигураций, необходимых для этих областей. Для конфигурирования этих областей вы можете создать классы или идентификаторы в таблицах CSS.

- Для проверки создайте пробную страницу с большинством запланированных элементов. При необходимости исправьте код CSS.
- Планируйте и проверяйте. Это две важные составляющие процесса конструирования веб-страниц.

3.9. Центрирование HTML-элементов с помощью CSS

Ранее в этой главе вы узнали, как выровнять текст по центру веб-страницы. А как центрировать саму веб-страницу? В распространенном варианте макета страницы, который легко создать с помощью всего нескольких строк кода CSS, все содержимое выравнивается по центру окна просмотра браузера. Тут важно задать элемент `div`, вмещающий или охватывающий полностью содержимое страницы. HTML-код следующий:

```
<body>
<div id="wrapper">
...здесь находится содержимое страницы...
</div>
</body>
```

Затем конфигурируйте правило CSS для данного контейнера. Как мы расскажем далее в главе 6, пустое пространство вокруг элемента называется ***полем*** (***margin***). Если речь идет об элементе `body`, полями считаются пробелы между содержимым страницы и краями окна браузера. Как можно догадаться, свойства `margin-left` и `margin-right` задают величину левого и правого полей соответственно. Значения полей можно установить равным `0` или `auto`, а также выраженным в пикселах, единицах `em` или процентах. Свойство `width` задает ширину блочного элемента. В приведенном ниже примере CSS-кода установлена ширина идентификатора **`wrapper`**, равная 700 пикселов, и он выровнен по центру.

```
#wrapper { width: 700px;
margin-left: auto;
margin-right: auto; }
```

Вы потренируетесь применять эту технику в следующем практическом задании.

Практическое задание 3.8

На этом практическом задании вы будете центрировать контент страницы с помощью CSS. В качестве исходного материала мы будем использовать файлы практического задания 3.7. Создайте новую папку с именем *3.8*. Найдите папку *Примеры\Глава_03\3.7* на диске, прилагающемся к книге, и скопируйте файлы *index.html*, *services.html* и *trillium.css* в папку *3.8*. Откройте файл *trillium.css* в текстовом редакторе. Создайте идентификатор с именем `wrapper`. Добавьте свойства стилей `margin-left`, `margin-right` и `width` в правила стилей следующим образом:

```
#container { margin-left: auto;
margin-right: auto;
width:80%; }
```

Сохраните файл. Откройте файл *index.html* в текстовом редакторе. Добавьте HTML-код, конфигурирующий элемент `div`, присвоенный идентификатору с именем `wrapper` и «оборачивающему» или заключающему в себе код тела страницы. При проверке вашего файла *index.html* в браузере он должен выглядеть так, как показано на рис. 3.18. Пример находится на диске, прилагающемся к книге, в папке *Примеры\Глава_03\3.8*.

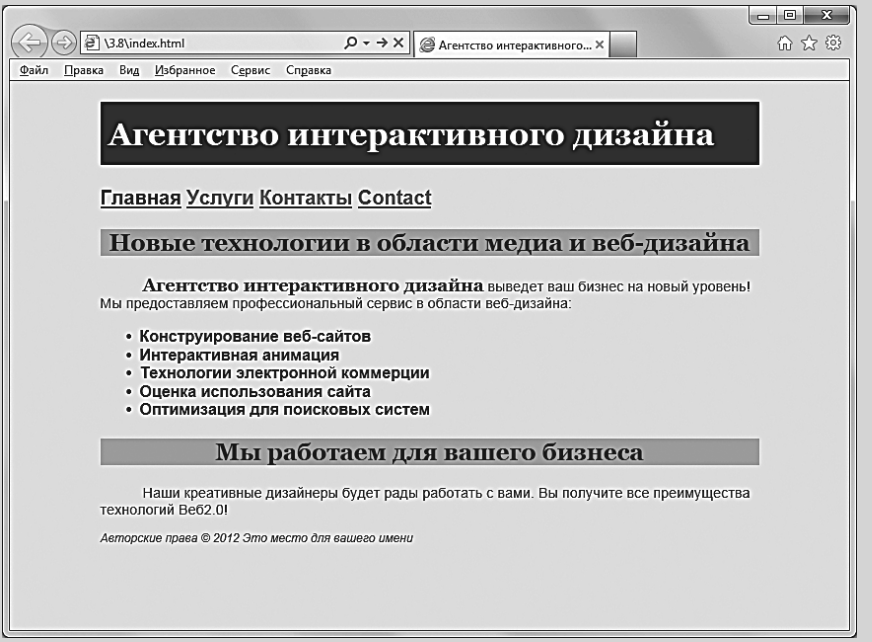


Рис. 3.18. Содержимое страницы выровнено по центру

ЧаВо

МОЖНО ЛИ БЫСТРО ДОБАВИТЬ КОММЕНТАРИЙ В КОД CSS?

Да. Комментарий можно легко добавить, если обернуть его символами "/*" и "*/", как показано ниже:

```
/* Настройка колонтитула */
#footer { font-size: .80em; font-style: italic; text-align: center; }
```

3.10. Каскадность стилей CSS

На рис. 3.19 схематически показан «каскад» или ***правило приоритетов***, в соответствии с которым применяются стили, начиная с наружного уровня (внешние стили) к внутреннему (HTML-атрибуты, указанные в коде страницы). Данный набор правил позволяет задавать сквозные стили для всей страницы, которые при необходимости замещаются более специфичными стилями (такими как глобальные или внутренние).

Рис. 3.19. «Каскад» правил CSS

Внешние таблицы стилей могут использоваться для различных страниц. Если веб-страница содержит как ссылки на внешние таблицы стилей, так и глобальные стили, сначала будут применяться внешние

стили, а затем глобальные. Это дает возможность веб-дизайнерам переопределять глобальные внешние стили на некоторых страницах.

Если веб-страница содержит как ссылки на глобальные стили, так и внутренние стили, сначала будут применяться глобальные стили, а затем внутренние. Это дает возможность веб-дизайнерам подавлять сквозные стили для некоторых элементов или классов.

Любой атрибут или элемент способен переопределять стили. Например, элемент `strong` будет иметь приоритет перед другими стилями шрифтов, заданными для конкретного элемента. Если элементу не заданы никакие стили, будут применяться стили браузера, назначенные по умолчанию. Однако внешний вид выбранных по умолчанию стилей может отличаться в зависимости от браузера и, возможно, результат вам не понравится. Поэтому старайтесь не использовать настройки браузера по умолчанию — конфигурируйте свойства текста и других элементов веб-страницы с помощью CSS.

Правил приоритетов придерживается не только общий каскад CSS, описанный выше, но и сами правила стилей. Правила стиля, применяемые к элементам более низкого уровня (например к абзацу), имеют приоритет перед стилями, назначенными элементам более высокого уровня (например, элементу `div`, содержащему данный абзац).

Ниже приведен пример каскада. Обратите внимание на CSS-код:

```
.content { font-family: Arial, sans-serif; }
p { font-family: "Times New Roman", serif; }
```

CSS содержит два правила стиля: правило, создающее класс с именем `content`, задающий тексту шрифт Arial (или какой-либо из шрифтов без засечек), и правило, задающее тексту шрифт Time New Roman (или какой-либо из шрифтов с засечками).

Код HTML-страницы содержит элемент `div` с несколькими элементами более низкого уровня, такими как заголовки и абзацы, как показано ниже:

```
<div class="content">
<h1>Основной заголовок</h1>
<p>Это абзац. Обратите внимание, что этот абзац
заключен в элемент div</p>
</div>
```

Браузер визуализирует страницу следующим образом:

1. Текст заголовка будет отображаться шрифтом Arial, так как он является частью элемента `div`, относящегося к классу `content`, и наследует свойства от класса родительского элемента (`div`). Это пример процедуры ***наследования***, согласно которой свойства CSS элемента-контейнера (например, свойства элемента `div` или `body`) передаются вложенному элементу. Свойства текста (семейство шрифтов, цвет и т. д.) обычно наследуются, а свойства блока (поля, отступ, ширина и т. д.) — нет. Подробный список свойств CSS с информацией о возможности наследования приведен на сайте **www.w3.org/TR/CSS21/propidx.html**.
2. Текст абзаца отображается шрифтом Time New Roman, так как браузер использует стили, заданные элементу наиболее низкого уровня (абзацу). Хотя абзац относится к классу `content` (и является дочерним), приоритет имеют локальные стили абзаца и именно они используются браузером.

Не расстраивайтесь, если правила стиля пока кажутся немного сложными. Попрактиковавшись, вы научитесь работать с CSS. У вас будет возможность потренироваться в применении «каскада» при выполнении следующего практического задания.

Практическое задание 3.9

В этом практическом задании вы будете экспериментировать с «каскадом», работая с веб-страницей, к которой применяются внешние, глобальные и внутренние стили.

1. Создайте новую папку и назовите ее *mycascade*.
2. Запустите текстовый редактор. Создайте новый файл. Сохраните его в папке *mycascade* под именем *site.css.* Вы создадите внешнюю таблицу стилей, задающую желтоватый оттенок фона страницы (`#FFFFCC`) и черный цвет текста (`#000000`). Код следующий:

```
body { background-color: #FFFFCC;
color: #000000; }
```

   Сохраните и закройте файл *site.css*.
3. Откройте в текстовом редакторе новый файл и сохраните его в папке *mycascade* под именем *mypage1.html.* Веб-страница будет связана с внешней таблицей стилей *site.css.* Примените глобальные стили, чтобы установить в качестве глобального цвета текста синий, а затем с помощью внутренних стилей задайте цвет

текста второго абзаца. В файле *mypage1.html* два абзаца текста. Код для файла следующий:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Каскадность в действии</title>
<meta charset="utf-8">
<link rel="stylesheet" href="site.css">
<style>
body { color: #0000FF;
}
</style>
</head>
<body>
<p>К этому абзацу текста применяются внешние и глобальные стили CSS &mdash; обратите внимание, что синий цвет текста, который настроен в глобальных стилях, имеет приоритет над черным цветом, настроенным во внешнем стиле.</p>
<p style="color:#FF0000">Внутренние стили форматируют текст этого абзаца красным цветом и имеют приоритет над глобальными и внешними стилями.</p>
</body>
</HTML>
```

Сохраните файл *mypage1.html* и откройте его в браузере. Страница должна быть похожа на пример, показанный на рис. 3.20. Сравните свой код с файлом *Примеры\Глава_03\3.9\mypage1.html.*

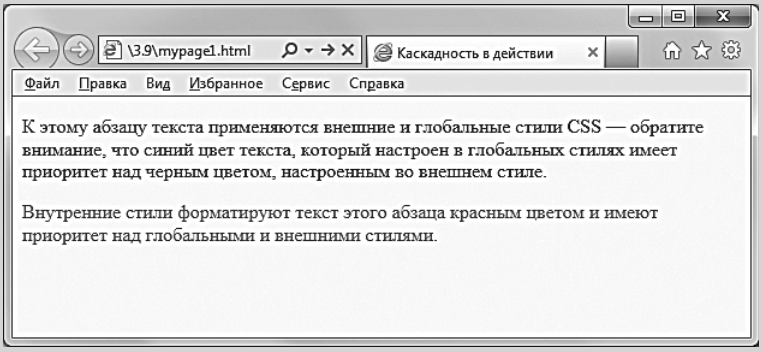


Рис. 3.20. Смешение внешних, глобальных и внутренних стилей

Найдите минутку, чтобы изучить веб-страницу *mypage1.html* и проанализировать ее исходный код. Внешняя таблица стилей задает желтый фон страницы. Глобальные стили конфигурируют синий цвет текста, заменивший черный цвет, указанный во внешней CSS. Для первого абзаца веб-страницы нет внутренних стилей, поэтому он наследует правила стилей, прописанные во внешней и глобальной таблицах стилей. Второй абзац содержит внутренний стиль, задающий красный цвет текста. Данная настройка имеет преимущественное значение по сравнению с внешними и глобальными стилями.

3.11. Валидация CSS

Консорциум W3C содержит на своем сайте[1] сервис проверки CSS на наличие ошибок. Вы сможете убедиться, что созданный вами код синтаксически правилен. Для веб-дизайнеров всего мира ***CSS-валидатор*** является инструментом проверки качества. Неправильно написанный код может привести к тому, что браузер будет отображать страницу медленнее, чем мог бы.

Практическое задание 3.10

На этом практическом задании вы будете использовать CSS-валидатор на сайте консорциума W3C для проверки внешних таблиц стилей. В этом примере используется файл *color.css*, созданный на практическом задании 3.6 (*Примеры\Глава_03\color.css*). Найдите файл *color.css* и откройте его в редакторе Блокнот (Notepad). Мы добавим ошибки в этот файл. Найдите правило стиля для селектора `body` и удалите первую букву `r` в имени свойства `background-color`. Удалите символ `#` из значения свойства `color`. Сохраните файл.

Теперь попытайте проверить файл color.css. Зайдите на страницу **jigsaw.w3.org/css-validator** и выберите вкладку **Проверить загруженный файл**. Щелкните мышью по кнопке **Обзор** и выберите файл *color.css* на вашем компьютере. Щелкните мышью по кнопке **Проверить**. Страница должна принять вид, показанный на рис. 3.21. Убедитесь в том, что две ошибки найдены. Указывается селектор с последующим комментарием к найденной ошибке.

Обратите внимание, что в первом сообщении на рис. 3.21 говорится, что свойство `background-color` не существует. Это должно подтолкнуть вас к проверке синтаксиса имени свойства. Отредактируйте файл *color.css*, исправив ошибку. Проверьте и повторно проведите

[1] jigsaw.w3.org/css-validator

валидацию страницы. Теперь страница будет выглядеть так, как показано на рис. 3.22, и на ней будет сообщение только об одной ошибке.

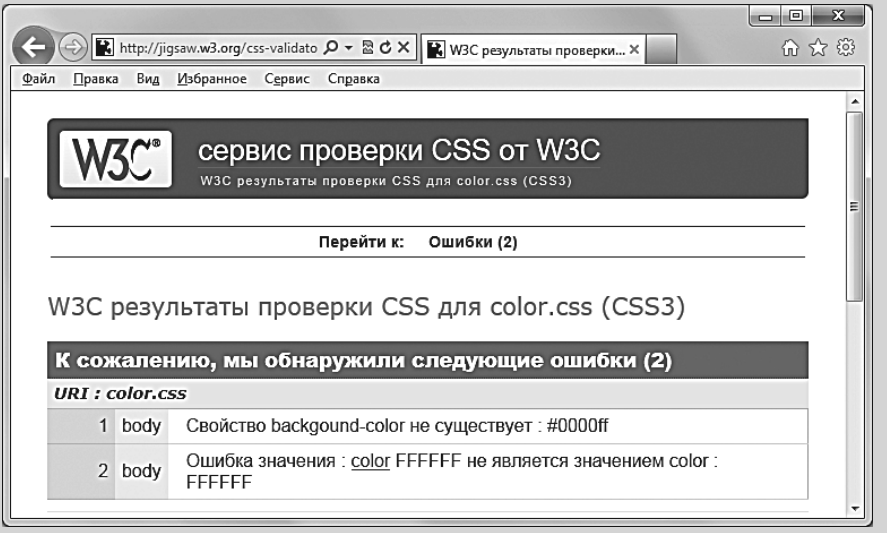


Рис. 3.21. Валидация выявила наличие ошибок

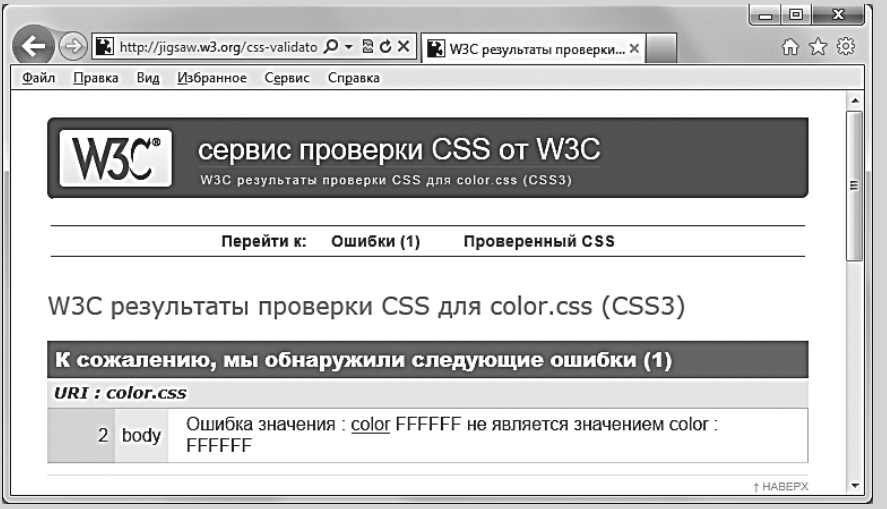


Рис. 3.22. Результаты проверки CSS, ниже указываются ошибки (и предупреждения, если они имеются)

Это сообщение напоминает вам, что `FFFFFF` не является значением цвета — считается, что вы знаете о том, что перед указанной последовательностью букв нужно добавить символ `#`, чтобы получить значение цвета `#FFFFFF`. Обратите внимание на то, каким образом

ниже сообщения об ошибке указываются корректные CSS. Исправьте ошибку, сохраните файл и вновь его проверьте.

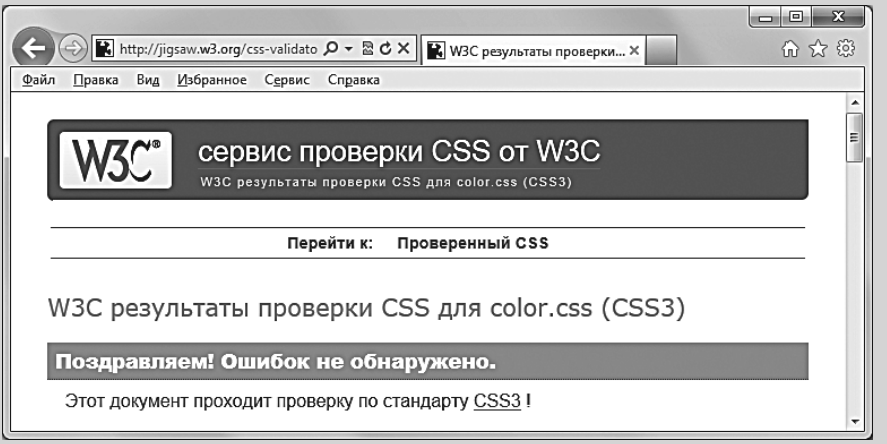


Рис. 3.23. Таблица CSS успешно прошла валидацию

Результат валидации должен выглядеть так, как показано на рис. 3.23. В нашем случае ошибки не обнаружены. Это означает, что ваш файл успешно прошел проверку валидатором. Поздравляем, ваш файл имеет допустимую структуру. Валидация правил стилей CSS должна стать для вас хорошей привычкой. Она поможет вам быстро найти фрагменты кода, которые нуждаются в исправлении, и укажет, какие правила стиля, управляющие браузером, вероятнее всего правильны. Валидация CSS — это одна из наиболее полезных процедур, используемых веб-дизайнерами.

Глава 4
ГРАФИЧЕСКИЕ ЭЛЕМЕНТЫ И РИСУНКИ

Цели главы

В этой главе вы узнаете следующее:

- как создавать на веб-страницах линии и границы, а также как их форматировать;
- как определить, когда следует использовать графику, и какая графика вам подойдет;
- как использовать элемент `image` для включения графики на веб-страницу;
- как оптимизировать изображения для Всемирной паутины;
- как сделать изображение фоном страницы;
- как присоединить к изображению гиперссылку;
- как задать скругление углов, тень блока, тень текста, прозрачность и градиенты с помощью CSS3;
- как конфигурировать цвета по модели RGBA с помощью CSS3;
- как применять элементы HTML5 для создания подписей под фигурами;
- как использовать HTML5-элементы `meter` и `progress`;
- где найти бесплатные и платные источники изображений;
- каким рекомендациям по веб-дизайну нужно следовать при использовании графики на веб-страницах.

Ключевым моментом конструирования веб-сайта является добавление интересной графики, соответствующей контенту сайта. Эта глава познакомит вас с использованием графики на сайтах.

Когда вы добавляете изображения на ваш веб-сайт, важно помнить, что не все пользователи могут увидеть их. Некоторые из пользователей

имеют нарушения зрения и пользуются специальными программами экранного доступа, которые зачитывают текстовое содержимое. Кроме этого следует иметь в виду, что поисковые системы, которые постоянно посылают своих поисковых агентов (роботов, пауков и пр.) путешествовать по веб, собирать информацию и составлять каталоги веб-страниц по их индексам и базам данных, не смогут обнаружить ваши рисунки. Поэтому как веб-дизайнер вы должны конструировать веб-страницы таким образом, чтобы они, несмотря на наличие изображений, могли использоваться и без них.

4.1. Конфигурирование линий и границ

Веб-дизайнеры часто используют графические элементы, такие как линии и границы (рамки), для разделения или выделения различных областей страницы. В этом разделе вы изучите две технологии конфигурирования линий на веб-станицах: элемент `rule` в HTML и свойства CSS — `border` и `padding`.

Горизонтальные линии

Горизонтальная линия визуально разделяет области страницы. Так как этот элемент не содержит никакого текста, он указывается как одиночный элемент, а не как пара, состоящая из открывающего и закрывающего тегов. Синтаксис XHTML для создания горизонтальной линии `<hr />`, а HTML5 — `<hr>` . В HTML5 горизонтальная линия обретает новое семантическое значение — указывает на окончание темы. По умолчанию горизонтальные линии выравниваются по центру внутри элемента-контейнера.

Практическое задание 4.1

Откройте в текстовом редакторе веб-страницу *Примеры\Глава_04\starter1.html*. Этот файл вам уже знаком. Это та же веб-страница, с которой вы работали, изучая главу 3. Добавьте элемент `hr` непосредственно над элементом `div`, содержащем колонтитул страницы.

Сохраните файл с именем *hr.html* и протестируйте его в браузере. Нижняя часть вашей страницы должна выглядеть так, как показано на рис. 4.1. Сравните результат вашей работы с файлом *Примеры\Глава_04\hr.html*.

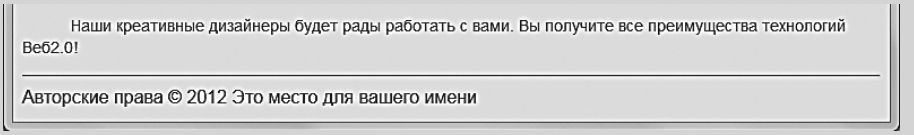


Рис. 4.1. Элемент `hr` создает горизонтальную линию через всю страницу

Хотя горизонтальные линии легко могут быть созданы средствами HTML, более современный подход — это конфигурирование элементов границ веб-страницы с помощью CSS.

Свойства `border`, `border-style` и `padding`

Задавая цвет фона элементам заголовка в главе 3, вы, возможно, уже заметили, что блочные HTML-элементы имеют форму прямоугольника. Эта особенность известна как блочная модель CSS, которую вы будете изучать в главе 6. А здесь мы сконцентрируемся на двух свойствах CSS, которые могут быть заданы для «блока» — `border` и `padding`.

Свойство `border`

Свойство **`border`** конфигурирует рамку, т.е. границу вокруг элемента. По умолчанию граница имеет ширину 0 и не отображается. Вы можете также задать свойства `border-width`, `border-color` и `border-style`. Более того, вы можете отдельно задать свойства верхней, правой, левой и нижней границам (`border-top`, `border-right`, `border-left` и `border-bottom` соответственно).

Свойство `border-style`

Свойство `border-style` задает тип линии, отображаемый на границе. Варианты форматирования включают следующие значения: `inset`, `outset`, `double`, `groove`, `ridge`, `solid`, `dashed` и `dotted`. Следует иметь в виду, что эти значения могут не поддерживаться некоторыми браузерами. На рис. 4.2 показано, как браузеры Firefox 4 и Internet Explorer 9 визуализируют линии, задаваемые различными значениями свойства `border-style`.

В таблице стилей CSS, задающих границы, показанные на рис. 4.2, используется цвет границы `#000033`, ширина границы 3 пиксела и различные значения свойства `border-style`. Правило стиля, конфигурирующее штриховую границу, может иметь такой вид:

```
.dashedborder { border-width: 3px;
border-style: dashed;
border-color: #000033; }
```

Упрощенная форма записи (нотация) позволяет конфигурировать все свойства границы в одном правиле стиля, просто перечисляя значения свойств `border-width`, `border-color` и `border-style`. Например:

```
.dashedborder { border: 3px dashed #000033 }
```

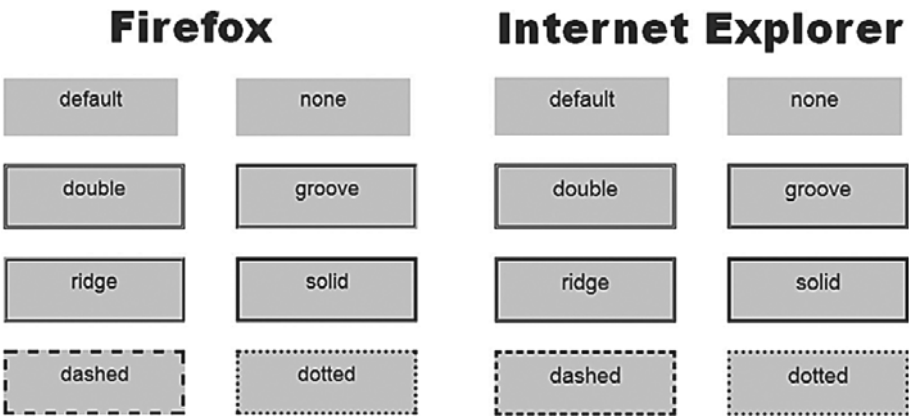


Рис. 4.2. Не все значения свойства `border-style` одинаково отображаются даже популярными браузерами

Свойство `padding`

Свойство **`padding`** конфигурирует пустое пространство между содержимым HTML-элемента (обычно это текст) и границей. По умолчанию значение свойства `padding` равно 0. Если вы задаете цвет фона элемента, этот цвет задается как границе, так и области содержимого. Вы будете учиться применять свойство `padding` в следующем практическом задании. На этом задании вам может потребоваться обратиться к табл. 4.1, в которой описаны все свойства CSS, рассматриваемые в главе 4.

Таблица 4.1. Новые свойства CSS, введенные в главе 4

Свойство	Описание	Значение
`background-attachment`	Конфигурирует, будет ли фоновое изображение прокручиваться вместе со страницей или останется на месте	`fixed`, `scroll` (по умолчанию)

Свойство	Описание	Значение
background-clip	Задает область фонового рисунка. Это свойство CSS3 поддерживается не во всех браузерах	padding-box, border-box или content-box
background-image	Фоновое изображение элемента	Для показа изображения введите значение url(*имяфайла*.gif), url(*имяфайла*.jpg) или url(*имяфайла*.png). Чтобы скрыть изображение, укажите значение none (установлено по умолчанию)
background-origin	Задает область позиционирования фонового рисунка. Это свойство CSS3 поддерживается не во всех браузерах	padding-box, border-box или content-box
background-position	Положение фонового рисунка	Два значения в процентах или пикселах. Первое значение задает положение по горизонтали, второе — положение по вертикали, отсчитываемые от верхнего левого угла блока, содержащего элемент. Также могут использоваться текстовые значения: left, top, bottom и right
background-repeat	Определяет повторяемость фонового рисунка	Текстовые значения: repeat (по умолчанию), repeat-y (повтор по вертикали), repeat-x (повтор по горизонтали), no-repeat (без повтора). Новые значения CSS3 (space, round) поддерживаются не во всех браузерах
background-size	Определяет размеры фонового рисунка	Две величины в процентах, пикселах или значения auto, contain или cover. Первое значение обозначает ширину, второе — высоту. Если указано только одно значение, вторым по умолчанию будет значение auto. Значение contain задает масштабирование изображения по горизонтали в соответствии с размером контейнера (соотношение сторон остается неизменным). Значение cover задает масштабирование изображения по вертикали в соответствии с размером контейнера (соотношение сторон остается неизменным)
border	Упрощенная нотация для задания значений border-width, border-color и border-style для элемента	Значения свойств border-width, border-color и border-style отделяются пробелами, например: border: 1px solid #000000;

Свойство	Описание	Значение
`border-bottom`	Упрощенная нотация для конфигурирования нижней границы элемента	Значения свойств `border-width`, `border-color` и `border-style` отделяются пробелами, например: `border-bottom: 1px solid #000000;`
`border-bottom-left-radius`	Конфигурирует скругление левого нижнего угла границы. Это свойство CSS3 поддерживается не во всех браузерах	Одно числовое значение в пикселах, единицах em или процентах, задающее радиус угла
`border-bottom-right-radius`	Конфигурирует скругление правого нижнего угла границы. Это свойство CSS3 поддерживается не во всех браузерах	Одно числовое значение в пикселах, единицах em или процентах, задающее радиус угла
`border-color`	Значение цвета всей границы вокруг элемента	Любое допустимое значение цвета
`border-left`	Упрощенная запись для конфигурирования левой границы элемента	Значения свойств `border-width`, `border-color` и `border-style` отделяются пробелами, например: `border-left: 1px solid #000000;`
`border-radius`	Конфигурирует скругленные углы элемента. Это свойство CSS3 поддерживается не во всех браузерах	От одного до четырех числовых значений в пикселах, единицах em или процентах, задающих радиусы углов. Если указывается одно значение, оно конфигурирует все четыре угла. Углы скругляются в таком порядке: левый верхний, правый верхний, правый нижний и левый нижний
`border-right`	Упрощенная нотация для конфигурирования правой границы элемента	Значения свойств `border-width`, `border-color` и `border-style` отделяются пробелами, например: `border-right: 1px solid #000000;`
`border-style`	Задает тип границы вокруг элемента	Текстовые значения `double`, `groove`, `inset`, `none` (по умолчанию), `outset`, `ridge`, `solid`, `dashed`, `dotted` и `hidden`
`border-top`	Упрощенная нотация для конфигурирования верхней границы элемента	Значения свойств `border-width`, `border-color` и `border-style` отделяются пробелами, например: `border-top: 1px solid #000000;`
`border-top-left-radius`	Конфигурирует скругление левого верхнего угла. Это свойство CSS3 поддерживается не во всех браузерах	Одно числовое значение в пикселах, единицах em или процентах, задающее радиус угла

Свойство	Описание	Значение
`border-top-right-radius`	Конфигурирует скругление правого верхнего угла. Это свойство CSS3 поддерживается не во всех браузерах	Одно числовое значение в пикселах, единицах em или процентах, задающее радиус угла
`border-width`	Ширина границы вокруг элемента	Числовые значения в пикселах (такие как `1px`) `thin`, `medium`, `thick`
`box-shadow`	Конфигурирует тень элемента. Данное свойство CSS3 поддерживается не во всех браузерах	Указывается от двух до четырех числовых значений в пикселах или единицах `em` для обозначения смещения по горизонтали и по вертикали, радиуса размытия (не обязательно) и размера (не обязательно), а также допустимое значение цвета. Чтобы задать внутреннюю тень, используйте ключевое слово `inset`
`height`	Высота элемента	Числовое значение в пикселах или процентах
`linear-gradient`	Конфигурирует линейное смешение оттенков и переход от одного цвета к другому. Данное свойство CSS3 поддерживается не во всех браузерах	Существуют различные варианты синтаксиса для указания начальных точек градиента и значений цветов. Например, следующий код задает двухцветный линейный градиент: `linear-gradient(#FFFFFF, #8FA5CE);`
`min-width`	Задает минимальную ширину элемента	Числовые значения (например в пикселах) или в процентах
`opacity`	Задает степень непрозрачности элемента. Данное свойство CSS3 поддерживается не во всех браузерах	Числовые значения от 1 (совершенно непрозрачный) до 0 (полностью прозрачный). Это свойство наследуется дочерними элементами
`padding`	Упрощенная запись для конфигурирования отступа между содержимым (контентом) элемента и его границей	1. Одиночное числовое значение в пикселах, единицах em или процентах задает одинаковую ширину отступа по всем сторонам элемента. 2. Два числовых значения в пикселах, единицах em или процентах задают: первое — ширину отступа сверху и снизу, второе — ширину отступа слева и справа. Например: `padding: 20px 30px;` 3. Три числовых значения в пикселах, единицах em или процентах. Первое задает величину верхнего отступа, второе — величину левого и правого отступов, а третье — нижнего отступа. Например, `padding: 5px 30px 10px;` 4. Четыре числовых значения в пикселах, единицах em или процентах задают ширину

Свойство	Описание	Значение
		отступа в следующем порядке: `padding-top`, `padding-right`, `padding-bottom`, `padding-left`
`padding-bottom`	Задает пустое пространство между содержимым элемента и его нижней границей	Числовые значения в пикселах, единицах em или процентах
`padding-left`	Задает пустое пространство между содержимым элемента и его левой границей	Числовые значения в пикселах, единицах em или процентах
`padding-right`	Задает пустое пространство между содержимым элемента и его правой границей	Числовые значения в пикселах, единицах em или процентах
`padding-top`	Задает пустое пространство между содержимым элемента и его верхней границей	Числовые значения в пикселах, единицах em или процентах
`radial-gradient`	Задает радиальное (круговое) смешение оттенков и переход от одного цвета к другому. Это свойство CSS3 поддерживается не во всех браузерах	Существуют различные варианты синтаксиса для указания начальных точек градиента и значений цветов. Например, следующий код задает двухцветный радиальный градиент: `radial-gradient(#FFFFFF,#8FA5CE);`
`text-shadow`	Задает эффект тени текста, отображаемому внутри элемента. Это свойство CSS3 поддерживается не во всех браузерах	От двух до четырех числовых значений в пикселах или единицах `em` для обозначения смещения по горизонтали и по вертикали, радиуса размытия (не обязательно) и расстояния распространения (не обязательно), а также допустимого значения цвета

Практическое задание 4.2

На этом практическом задании вы будете работать со свойствами `border` и `padding`. Откройте файл *Примеры\Глава_04\starter1.html* в программе Блокнот (Notepad). Вы измените правила стилей CSS для селекторов элементов `h1` и `h2`, а также идентификатора `footer`. Когда вы закончите, страница будет выглядеть так, как показано на рис. 4.3. Измените правила стилей CSS следующим образом:

1. Измените правила стилей CSS для селектора `h1`. Удалите из правил стиля свойство `line-height`, потому что вы будете конфигурировать отступ с использованием свойства `padding`. Добавьте правило стиля, задающее ширину отступа в 15 пикселов. Код будет выглядеть следующим образом:

```
padding: 15px;
```

2. Добавьте правило стиля для селектора `h2`, которое создаст штриховую линию шириной 2 пиксела с цветом `#191970` на нижней границе элемента. Код будет выглядеть следующим образом:

```
border-bottom: 2px dashed #191970;
```

3. Добавьте к идентификатору `footer` правила стиля, задающие тонкую сплошную линию с цветом `#aeaed4` для верхней границы и с расстоянием до текста 10 пикселов. Также отформатируйте текст нижнего колонтитула серым цветом и меньшим размером, а также курсивным начертанием. Код выглядит следующим образом:

```
#footer { border-top: thin solid #aeaed4;
padding-top: 10px;
color: #333333;
font-size: .75em;
font-style: italic; }
```

Отредактируйте HTML-код и назначьте элементу `div`, содержащему информацию нижнего колонтитула, идентификатор с именем `footer`. Сохраните файл с именем *border.html*.

Проверьте измененный файл в нескольких браузерах. Можно ожидать, что ваша страница будет выглядеть несколько по-разному в различных браузерах. На рис. 4.3 показан вид страницы в браузере Google Chrome. Образец приведен на диске, прилагающемся к книге, в файле *Примеры\Глава_04\border.html*.

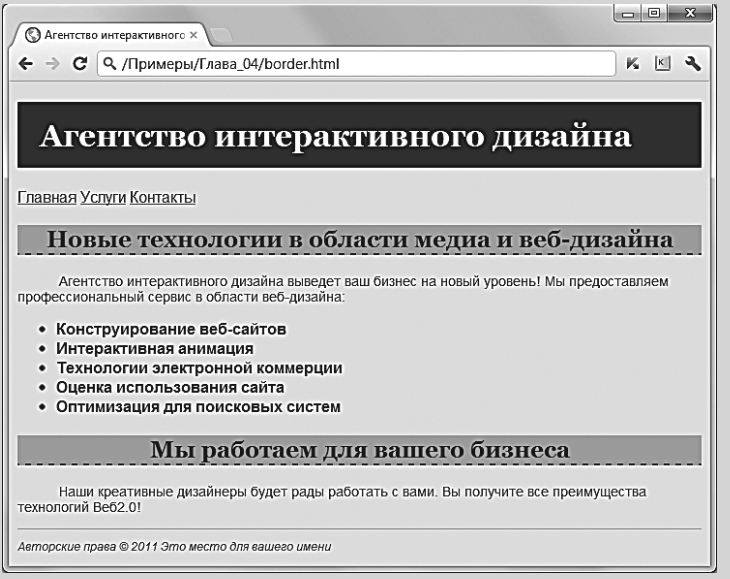


Рис. 4.3. Применение свойств `border` и `padding` делает страницу привлекательнее

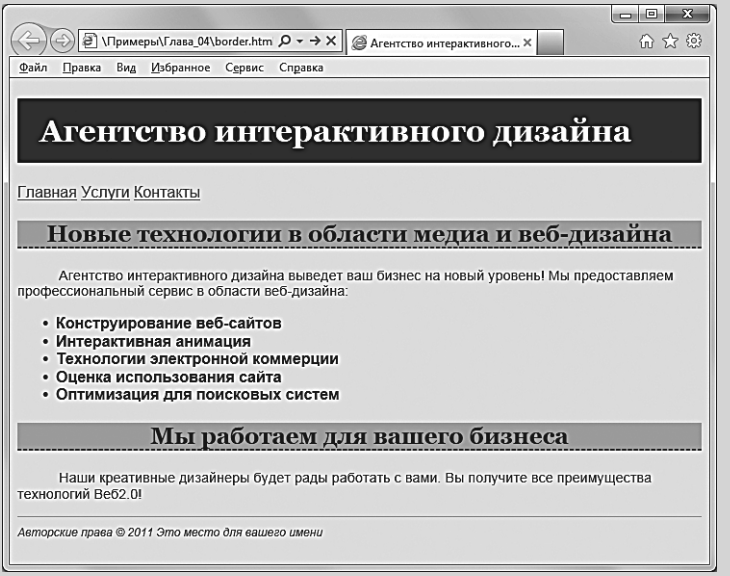


Рис. 4.4. Браузер Internet Explorer отображает штриховую границу несколько иначе, чем Google Chrome

ЧаВо

МОЯ СТРАНИЦА ВЫГЛЯДИТ ПО-РАЗНОМУ В РАЗНЫХ БРАУЗЕРАХ, ЧТО ДЕЛАТЬ?

Не следует думать, что ваша страница будет выглядеть одинаково в разных браузерах и даже во всех версиях одного и того же браузера. Различный вид страниц в разных браузерах — это огорчительная особенность повседневной работы разработчиков веб-сайтов. Однако есть и хорошая новость — разработчики браузеров начинают более последовательно выполнять требования стандартов W3C. Кроме того, организации, подобные Web Standards Project[1], лоббируют принятие стандартов в области разработки браузеров. Так что в будущем следует ожидать улучшения этой ситуации.

4.2. Типы графических изображений

Графика придает веб-странице привлекательный вид. В этом разделе рассматриваются такие виды используемых на веб-страницах графических форматов, как GIF, JPEG, PNG, а также обсуждаются их особенности. Кроме того, рассказывается о новом формате веб-графики — WepP.

[1] www.webstandards.org

Изображения в формате GIF

Формат графического обмена (***GIF***) более всего пригоден для графики, содержащей однородно закрашенные области, а также для простых изображений, таких как клипарты. Максимальное число цветов, поддерживаемое форматом GIF — 256. Для файлов изображений в формате GIF используется расширение *.gif*. На рис. 4.5 показан логотип, созданный в формате GIF.

Рис. 4.5. Логотип в формате GIF

Прозрачность

Формат GIF89A, применяемый для GIF изображений, поддерживает функцию ***прозрачности***. В графических редакторах, таких как Adobe Photoshop и GIMP, один цвет (обычно это цвет фона) может быть сделан прозрачным. Это позволяет создавать гармоничные сочетания элементов с фоном страницы или таблицы. На рис. 4.6 приведены два рисунка формата GIF, один без применения прозрачности, и второй с прозрачным фоном.

Рис. 4.6. Сравнение изображений формата GIF с применением прозрачности и без нее

Анимация

Анимированные изображения формата GIF содержат несколько слегка отличающихся друг от друга изображений (или кадров). Когда кадры быстро перелистываются на экране в нужном порядке, изображение начинает двигаться — анимированные GIF-изображения могут создаваться в таких графических приложениях, как Adobe Fireworks.

Сжатие (компрессия)

При сохранении файлов формата GIF используется ***сжатие без потерь***, т.е. никакие элементы изображения не теряются и при последующей визуализации на экране изображение выглядит так же, как и оригинал.

Оптимизация

Старайтесь не создавать медленно загружающихся страниц; графические файлы должны быть оптимизированы для Всемирной паутины. ***Оптимизация*** изображений — это процесс минимизации размера изображения до минимально возможной величины, которая все еще дает удовлетворительное качество изображения на экране, т.е. здесь необходимо добиться компромисса между размером файла и качеством изображения. GIF-файлы обычно оптимизируются путем уменьшения количества используемых цветов в таких графических редакторах, как Adobe Photoshop, Adobe Fireworks или GIMP.

Интерлейсинг

При создании изображений в формате GIF может использоваться ***интерлейсинг***, который изменяет способ отображения графики браузером. При обычном способе визуализации рисунка файл отрисовывается сверху вниз, причем верхняя часть изображения появляется на экране после того, как 50% файла, содержащего изображение, считано браузером. При использовании интерлейсинга, изображение выводится на экран постепенно и в это время выглядит тусклым. Сначала оно размытое, затем, по мере загрузки, постепенно становится яснее и четче. Если вы используете сложное GIF-изображение, используйте интерлейсинг для уменьшения воспринимаемого времени его загрузки.

Изображения в формате JPEG

Формат JPEG лучше всего подходит для представления фотографий и поэтому популярен среди фотографов. В отличие от формата GIF, JPEG поддерживает 16,7 миллионов цветов. Однако изображения в формате JPEG не могут иметь прозрачные области и быть анимированы. Файлы в формате JPEG обычно имеют расширения .jpg или .jpeg.

Сжатие (компрессия)

Другим отличием файлов JPEG от GIF является то, что при сжатии в формате JPEG происходит ***потеря*** информации об изображении.

То есть при сжатии некоторые пикселы исходного изображения теряются или удаляются, и выводящееся на экран изображение хотя и похоже на оригинал, но не совсем то, что было изначально.

Оптимизация

Необходимо добиваться компромисса между качеством изображения и степенью сжатия. Изображение с меньшей степенью сжатия имеет более высокое качество, но и больший размер файла, а изображение с большей степенью сжатия имеет более низкое качество, но меньший размер файла. Большинство программ для работы с графикой предоставляет возможность предварительной установки соотношения размер/качество и выбора файла с оптимальными характеристиками.

Когда вы делаете снимок цифровой фотокамерой, размер получаемого файла оказывается слишком большим для того, чтобы помещать его на веб-страницу. На рис. 4.7 показана фотография, оригинальный файл которой имеет размер 250 Кб. Это изображение было оптимизировано в графическом редакторе с качеством в 80% (размер файла 55 Кб).

Рис. 4.7. Изображение в формате JPEG с качеством 80% (размер файла 55 Кб)

На рис. 4.8 показано то же изображение с качеством 20% (размер файла 19 Кб), но в этом последнем случае качество оказалось непри-

емлемым. Качество ухудшается при уменьшении размера изображения. Появившаяся выраженная блочная структура изображения, на рис. 4.8, называется ***пикселизация*** и ее появления следует избегать.

Рис. 4.8. Изображение в формате JPEG с качеством 20% (размер файла 19 Кб)

Другой способ оптимизации изображений в формате JPEG — это использование уменьшенных копий изображения или ***миниатюр***. Часто миниатюра является гиперссылкой, при щелчке мышью по которой появляется более крупное изображение. На рис. 4.9 показан пример миниатюры.

Рис. 4.9. Миниатюра формата JPEG с размером файла 5 Кб

Прогрессивный JPEG-формат

При создании файла в формате JPEG может быть выбран ***прогрессивный*** режим. Этот режим аналогичен интерлейсингу GIF-файлов

в том, что прогрессивные JPEG-изображения также постепенно выводятся на экран и по мере загрузки становится все более и более четким.

Изображения в формате PNG

Формат PNG сочетает в себе все лучшее форматов GIF и JPEG и предполагается, что в будущем заменит формат GIF. Формат PNG (произносится как «пинг») поддерживает миллионы цветов, различные уровни прозрачности и сжатие без потерь (табл. 4.2). Изображения формата PNG также могут использовать интерлейсинг.

Таблица 4.2. Обзор распространенных типов графических файлов для Всемирной паутины

Тип изображения	Расширение файла	Сжатие	Прозрачность	Анимация	Цвета	Прогрессивный режим
GIF	*.gif*	Без потери качества	Да	Да	256	Интерлейсинг
JPEG	*.jpg* или *.jpeg*	С потерей качества	Нет	Нет	Миллионы	Прогрессивный
PNG	*.png*	Без потери качества	Да	Нет	Миллионы	Интерлейсинг

Новый формат WebP

Новый формат изображений ***WebP***, разработанный корпорацией Google, предлагает улучшенные возможности сжатия фотографий с потерей качества, однако говорить о применении его на веб-сайтах еще рано. Изображения в формате WebP (произносится «веппи») на данный момент поддерживаются только браузером Google Chrome. Узнать больше об этом формате можно на странице **code.google.com/speed/webp**.

4.3. Элемент изображения

Элемент изображения применяется для размещения изображений на веб-странице. Этими изображениями могут быть фотографии, баннеры, логотипы компании, кнопки навигации и прочее — выбор ограничен только вашей творческой фантазией.

Элемент изображения используется как одиночный элемент, а не как пара, состоящая из открывающего и закрывающего тегов. Он

считается пустым элементом. В синтаксисе XHTML используйте тег <img />, а в синтаксисе HTML5 — тег <img>. Например, чтобы поместить на страницу изображение с именем файла *logo.gif*, нужно ввести такой код:

```
<img src="logo.gif" height="200" width="500"
alt="Название моей компании">
```

***Атрибут* `src`** указывает на имя файла изображения. ***Атрибут* `alt`** служит для отображения замещающего текста (обычно описания изображения). Если использовать атрибуты ширины и высоты со значениями, точно соответствующими или приблизительно равными величине изображения, браузер зарезервирует для вашего изображения область нужного размера.

В таблице 4.3 перечислены атрибуты элемента `img` и их значения. Наиболее часто используемые атрибуты выделены жирным шрифтом.

Таблица 4.3. Атрибуты элемента `img`

Атрибут	Значение
`align`	`right`, `left` (по умолчанию), `top`, `middle`, `bottom` (устарел; используйте вместо него CSS-свойства `float` или `position` (см. главу 6))
`alt`	Замещающий текст, описывающий рисунок
`border`	Ширина границ рисунка в пикселах; значение **`border="0"`** скрывает границы гиперссылки-изображения. Устарел, используйте вместо него CSS-свойство `border`
`height`	Высота изображения в пикселах
`hspace`	Ширина пустого пространства слева и справа от изображения; устарел, используйте вместо него CSS-свойство `padding`
`id`	Имя, состоящее из латинских букв и цифр, начинающееся с буквы, без пробелов, — имя должно быть уникальным, не использующимся нигде больше в этом HTML-документе
`name`	Имя, состоящее из латинских букв и цифр, начинающееся с буквы, без пробелов, — этот атрибут определяет имя изображения, поэтому к нему должен быть обеспечен легкий доступ со стороны клиента, использующего такие языки, как JavaScript. Устарел, используйте атрибут `id`
`src`	URL-адрес или имя файла изображения
`title`	Текст, содержащий полезную информацию об изображении, обычно более подробный, чем замещающий текст
`vspace`	Ширина пустого пространства вверху и внизу изображения; устарел, используйте вместо него CSS-свойство `padding`
`width`	Ширина изображения в пикселах

Обратите внимание, что некоторые атрибуты в табл. 4.3 помечены как устаревшие. Они устарели по отношению к спецификации HTML5, хотя все еще действуют в XHTML, поэтому, вероятно, встретятся вам в коде существующих веб-страниц. При выполнении заданий книги вы будете задействовать функции этих устаревших атрибутов с помощью CSS.

ЧаВо

ЧТО ДЕЛАТЬ, ЕСЛИ Я НЕ ЗНАЮ ВЫСОТУ И ШИРИНУ ИЗОБРАЖЕНИЯ?

Большинство графических редакторов отображают информацию о размере (высоте и ширине) изображения. Если у вас установлено графическое приложение, например, Adobe Photoshop или Adobe Fireworks, запустите его и откройте ваше изображение. Это приложение сразу укажет вам его высоту и ширину.

Если вы не располагаете нужным приложением для работы с графикой, то можете определить размер изображения с помощью браузера. Откройте изображение на веб-странице. Щелкните правой кнопкой мыши по изображению, откроется контекстное меню. Выберите команду меню **Свойства** (Properties), в этом окне будут указаны размеры изображения. (***Внимание:*** если высота и ширина заданы кодом веб-страницы, будут отображаться именно эти величины, хотя реальный размер на самом деле может быть другим.)

Изображения-ссылки

Легко превратить изображение в гиперссылку. Чтобы это сделать, вам нужно только указать элементы привязки с двух сторон элемента изображения. Например, чтобы назначить ссылку изображению *home.gif*, используется следующий HTML-код:

```
<a href="index.html"><img src="home.gif"
height="19" width="85" alt="Главная"></a>
```

Если какой-либо рисунок выполняет функции гиперссылки, по умолчанию вокруг него отображается граница синего цвета. Если вы не хотите, чтобы эта граница отображалась, то можете присвоить атрибут `border="0"` элементу `img`:

```
<a href="index.html"><img src="home.gif" height="19"
width="85" alt="Главная" border="0"></a>
```

Более современный подход к использованию CSS для конфигурирования границ заключается в применении селектора `img`. Этот метод

будет продемонстрирован на следующем практическом задании, когда вы будете добавлять ссылки к изображению.

Практическое задание 4.3

В этом практическом задании вам нужно будет добавить на веб-сайт графический баннер-логотип и кнопки навигации в виде картинок. Затем вы настроите кнопки-изображения, работающие как графические ссылки. Создайте новую папку под названием *4.3*. Изображения для этого практического задания находятся в папке *Примеры\Глава_04\starters* на диске, прилагающемся к книге. Скопируйте файлы *trilliumbanner.jpg*, *home.gif*, *services.gif* и *contact.gif* в папку *4.3*. Базовый вариант главной страницы Агентства интерактивного дизайна уже готов и доступен на диске, прилагающемся к книге. Скопируйте файл *Примеры\Глава_04\starter2.html* в папку *4.3*. Откройте страницу *starter2.html* в браузере. Обратите внимание, что цветовая схема, содержащая только оттенки зеленого, была задана с помощью правил CSS. По окончании практического задания ваша страница должна быть похожа на отображенную на рис. 4.10.

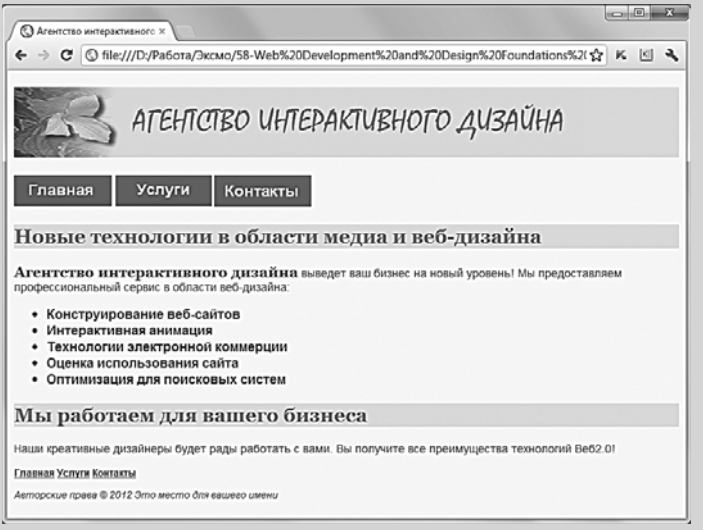


Рис. 4.10. Новая главная страница Агентства с добавленным логотипом

Запустите текстовый редактор и откройте файл *starter2.html*, находящийся в папке *4.3*.

1. Сконфигурируйте изображение логотипа:

- Замените текст, содержащийся между открывающим и закрывающим тегами `h1`. Укажите в коде элемент `img`, чтобы отобразить

в этой области изображение *trilliumbanner.jpg*. Не забудьте добавить атрибуты `src`, `alt`, `height` и `width`. Пример кода следующий:

```
<img src="trilliumbanner.jpg" alt="Агентство интерактивного дизайна" width="700" height="86">
```

- Отредактируйте глобальные правила CSS так, чтобы высота селектора элемента `h1` совпадала с высотой изображения. Добавьте следующее правило стилей:

```
height: 86px;
```

2. Сконфигурируйте графические ссылки. Обратите внимание, что элементы привязки уже прописаны в коде; вам просто нужно превратить текстовые ссылки в графические. Если навигация по странице реализована на основе мультимедийных объектов, таких как изображения, не все пользователи смогут увидеть их (или отображение рисунков может быть отключено в их браузере). Для обеспечения средств навигации, доступных для всех, настроим простые текстовые навигационные ссылки в области колонтитула следующим образом:

- скопируйте элемент `div`, содержащий навигационную область, в нижнюю часть страницы, вставив его выше колонтитула страницы;
- измените правила стиля класса `nav`. Измените размер шрифта, установив его равным `.75em`.

3. Теперь перейдите в верхнюю область навигации. Замените текст между каждой парой тегов привязки элементами изображения. Используйте изображение в файле *home.gif* для связи с *index.html*, *service.gif* для связи с *services.html* и *contact.gif* для связи с *contact.html*. Код в первом случае имеет вид:

```
<a href="index.html"><img src="home.gif" alt="Главная" width="120" height="40"></a>
```

4. Создайте новые правила стиля, конфигурирующие селектор `img` без границы:

```
img {border:none}
```

Сохраните файл с именем *index.html* в папке *4.3*. Запустите браузер и проверьте страницу. Она должна выглядеть так, как показано рис. 4.10.

Примечание: если изображение не отображается на веб-странице, убедитесь, что файлы *trilliumbanner.jpg, home.gif, services.*gif и *contact.gif* сохранены в папку *4.3* и их названия в элементе `img` указаны верно.

Изменяйте размер окна браузера и обратите внимание, что когда окно становится маленьким, изображения смещаются. Чтобы это предотвратить, добавьте к селектору `body` новое правило стиля, которое устанавливает минимальную ширину страницы. В этом случае в окне браузера автоматически будет появляться полоса прокрутки,

если пользователь уменьшит окно до величины, меньшей заданного минимального размера:

```
min-width: 700px;
```

Сохраните и снова проверьте страницу. Пример файла вы найдете на диске, прилагающемся к книге, в папке *Примеры\Глава_04\4.3*.

ЧаВо

ЧТО ДЕЛАТЬ, ЕСЛИ МОЕ ИЗОБРАЖЕНИЕ НЕ ОТОБРАЖАЕТСЯ?

Если изображение не отображается на веб-странице, следует проверить следующее:

- Действительно ли файл вашего изображения находится в папке веб-сайта? Проверьте еще раз.
- Правильно ли написан HTML- и CSS-код? Проверьте ваш код на отсутствие распространенных ошибок, таких как использование `scr` вместо `src` или отсутствие кавычек.
- Правильно ли в HTML-коде написано имя файла, который вы собираетесь использовать в качестве фона или атрибута `src`? Будьте внимательны к мелочам и логике построения кода.

ЧаВо

КАК ПРАВИЛЬНО ИМЕНОВАТЬ ФАЙЛЫ ИЗОБРАЖЕНИЙ?

Советы по выбору имен файлов:

- используйте строчные латинские буквы;
- не используйте в именах файлов точки и пробелы;
- не изменяйте расширение файла (должно быть *.gif*, *.jpg*, *.jpeg* или *.png*);
- используйте короткие, но описательные имена. Примеры имен: Имя *v1.gif*, по-видимому, слишком короткое, *neobychnyjvelosipednafonetravki.gif* — слишком длинное, а *velik.gif* — в самый раз.

Оптимизация изображений для Всемирной паутины

Фотографии, сделанные цифровым фотоаппаратом, слишком велики (как их размер в пикселах, так и размер файла) для отображения на веб-странице. Помните, что для графической оптимизации нужно достичь баланса между качеством изображения и размером файла. Это процесс создания изображения, визуализируемого браузером с хорошим качеством, но имеющего меньший размер файла. Чтобы оптимизировать изображение для размещения во Всемирной паутине, профессионалы

часто используют приложения Adobe Photoshop и Adobe Fireworks. Популярностью пользуется программа GIMP[1] — графический редактор с открытым исходным кодом, работающий на различных платформах. Кроме того, на сайте **pixlr.com/editor** предлагается бесплатный и простой в использовании онлайн графической редактор Pixlr.

Практическое задание 4.4

На данном практическом задании вы будете конфигурировать на веб-странице изображение с подписью. Фотография, используемая при выполнении задания, находится в папке *Примеры\Глава_04\starters*. Сохраните файл *myisland.jpg* в папку *caption.*

Шаг 1: запустите текстовый редактор. Выберите команду меню **Файл** ⇒ **Открыть** (File ⇒ Open), чтобы открыть для изменения файл *Примеры\Глава_02\template.html*. Измените элемент заголовка. Добавьте в раздел тела страницы элемент изображения, чтобы отобразить файл *myisland.jpg*, следующим образом:

```
<img src="myisland.jpg" alt="Тропический остров"
height="480" width="640">
```

Сохраните файл в папку *caption* с именем *index.html.* Запустите браузер и протестируйте страницу. Она должна быть похожа на показанную на рис. 4.11.

Рис. 4.11. На веб-странице отображается фотография

[1] gimp.org

Шаг 2: сконфигурируйте подпись и границу изображения. Для этого запустите текстовый редактор и откройте файл веб-страницы. Добавьте в раздел заголовка правила CSS, которые конфигурируют идентификатор `figure` шириной 640 пикселов, имеющий границу, отступы, равные 5 пикселам, и текст, выровненный по центру и набранный шрифтом Impact (или установленным по умолчанию шрифтом семейства Fantasy). Используется следующий код:

```
<style>
#figure { width: 640px;
border: 1px solid #000000;
padding: 5px;
text-align: center;
font-family: Impact, fantasy; }
</style>
```

Отредактируйте раздел тела страницы и добавьте элемент `div`, который будет содержать изображение. Под изображением в элементе `div` добавьте текст "`Тропический остров`". Назначьте элементу `div` идентификатор `figure`. Сохраните файл в папку *mycaption* под именем *index.html*. Запустите браузер и протестируйте страницу. Она должна быть похожа на показанную на рис. 4.12. Пример можно найти в папке *Примеры\Глава_04\caption* на диске, прилагающемся к книге.

Рис. 4.12. CSS задает местоположение границы и подписи под изображением

4.4. Визуальные элементы HTML5

В практическом задании 4.4 вы конфигурировали на веб-странице изображение и подпись к нему. В качестве контейнера для изображения и текстовой подписи использовался элемент `div`. В этом разделе вы изучите подход, при котором применяются новые элементы HTML5 и необходима современная версия браузера Safari, Firefox, Chrome, Opera или Internet Explorer (не ниже версии 9). Вы будете работать с новыми элементами языка HTML5 `figure` и `figcaption`.

Элемент `figure`

Блочный ***элемент*** **`figure`** содержит самостоятельную единицу контента, такую как изображение, вместе с необязательным элементом `figcaption`.

Элемент `figcaption`

Блочный ***элемент*** **`figcaption`** предоставляет подпись к изображению.

Практическое задание 4.5

В этом практическом задании вы конфигурируете область веб-страницы, содержащую изображение и подпись к нему, используя элементы `figure` и `figcaption` языка HTML5. Изображения для практического задания находятся в папке *Примеры\Глава_04\starters* на диске, прилагающемся к книге. Сохраните файл *myisland.jpg* в папку под названием *caption2*.

Шаг 1: запустите текстовый редактор и откройте в нем файл *Примеры\Глава_02\template.html*. Измените элемент заголовка. Добавьте в раздел тела страницы элемент изображения, чтобы отобразить файл *myisland.jpg* следующим образом:

```
<img src="myisland.jpg" alt="Тропический остров"
height="480" width="640">
```

Сохраните файл в папку *caption2* под именем *index.html.* Запустите браузер и протестируйте страницу. Она должна быть похожа на показанную на рис. 4.11.

Шаг 2: задайте подпись и границу изображения. Для этого запустите текстовый редактор и откройте файл веб-страницы. Добавьте в раздел заголовка правила CSS, которые конфигурируют идентификатор `figure` шириной 640 пикселов, имеющий границу и отступы, равные 5 пикселам. Сконфигурируйте селектор элемента `figcaption` так, чтобы текст в нем оказался выровнен по центру и отформати-

рован шрифтом Impact (или установленным по умолчанию шрифтом семейства Fantasy). Используется следующий код:

```
<style>
figure { width: 640px; border: 1px solid #000000;
padding: 5px; } figcaption { text-align: center;
font-family: Impact, fantasy; }
</style>
```

Отредактируйте раздел тела страницы. Под изображением добавьте элемент `figcaption`, содержащий текст `"Тропический остров"`. Сконфигурируйте элемент `figure` , содержащий изображение и подпись. Код для этого следующий:

```
<figure>
<img src="island.jpg" width="640" height="480"
alt="Tropical Island">
<figcaption>
Тропический остров
</figcaption>
</figure>
```

Сохраните файл в папку *mycaption2* под именем *index.html*. Запустите браузер и протестируйте страницу. Она должна быть похожа на показанную на рис. 4.13. Пример можно найти на диске, прилагающемся к книге, в папке *Примеры\Глава_04\caption2*.

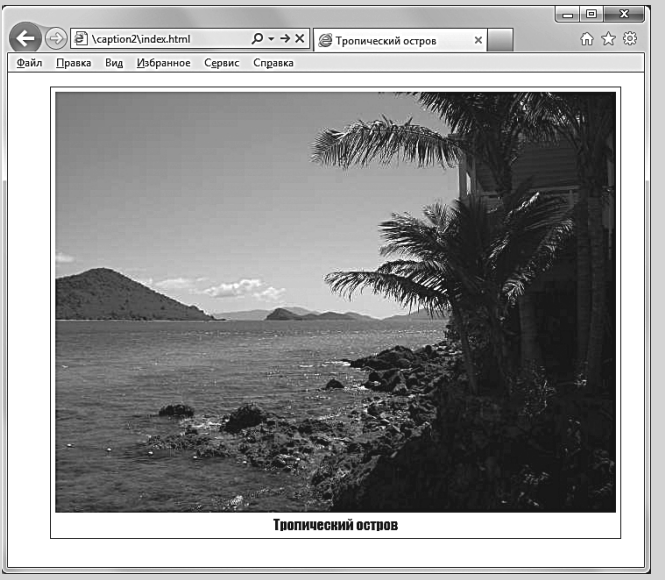


Рис. 4.13. На этой веб-странице были применены элементы `figure` и `figcaption` языка HTML5

Вам, наверное, интересно, зачем создавать эти новые элементы, если такого же оформления можно было добиться, используя в качестве контейнера элемент `div`. Дело в семантике. Элемент `div` полезен, но уж очень универсален. Когда применяются элементы `figure` и `figcaption`, структура контента четко определена. Однако если вы занимаетесь созданием страниц для коммерческих веб-сайтов, воздержитесь от использования новых элементов `figure` и `figcaption`, пока большинство браузеров не начнет поддерживать разметку HTML5.

Элемент `meter`

***Элемент* `meter`** визуально отображает величину числового значения в известном диапазоне, обычно как часть гистограммы. Элемент `meter` задается несколькими атрибутами, включающими `value` (отображаемое значение), `min` (наименьшее возможное значение в диапазоне) и `max` (наибольшее возможное значение в диапазоне). Следующий фрагмент кода (из файла *Примеры\Глава_04\meter.html*) определяет отображение ежемесячного отчета об использовании браузера, показывающий число посещений страницы.

```
<h1>Ежемесячный отчет</h1>
<meter value="14417" min="0" max="14417">14417</
meter>14,417 Всего посещений<br>
<meter value="7000" min="0" max="14417">7000</
meter> 7,000 Firefox<br>
<meter value="3800" min="0" max="14417">3800</
meter> 3,800 Internet Explorer<br>
<meter value="2062" min="0" max="14417">2062</
meter> 2,062 Chrome<br>
<meter value="1043" min="0" max="14417">1043</
meter> 1,043 Safari<br>
<meter value="312" min="0" max="14417">312</meter>
   312 Opera<br>
<meter value="200" min="0" max="14417">200</meter>
   200 прочие<br>
```

На рис. 4.14 показана страница, отображенная в браузере Opera. Элемент `meter` языка HTML5 предоставляет очень удобный способ отображения гистограммы на веб-странице. Чтобы получить информацию о поддержке этого элемента браузерами на сегодняшний день, посетите сайт **caniuse.com**.

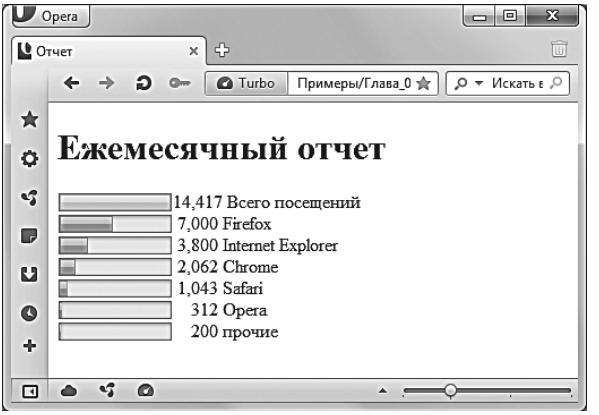


Рис. 4.14. Элемент `meter`, визуализированный в браузере Opera

Элемент `progress`

Элемент `progress` позволяет отобразить индикатор, графически представляющий числовое значение в определенном диапазоне. Элемент `progress` конфигурируется атрибутами `value` (отображаемое значение) и `max` (наибольшее допустимое значение). Следующий фрагмент кода (из файла *Примеры\Глава_04\progress.html*) демонстрирует, что задача выполнена на 50%:

```
<h1>Отчет о результатах</h1>
<progress value="5000" max="10000">5000</progress>
Результат продвижения к цели
```

На рис. 4.15 показана страница, отображенная в браузере Chrome. Чтобы получить информацию о поддержке этого элемента браузерами на сегодняшний день, посетите сайт **caniuse.com**.

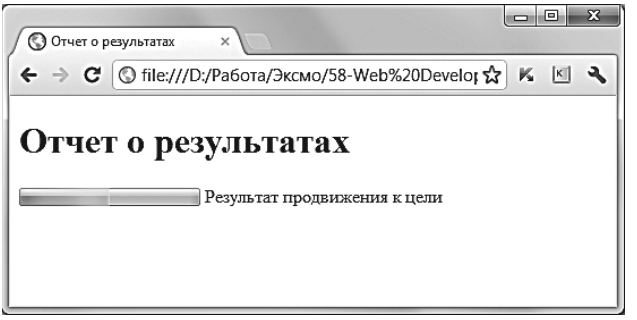


Рис. 4.15. Элемент `progress`, визуализированный в браузере Chrome

4.5. Фоновые изображения

В главе 3 объяснялось, как использовать CSS-свойство `background-color` для назначения цвета фона веб-страниц. Например, следующая запись установит для страницы мягкий желтый цвет:

```
body { background-color: #ffff99; }
```

Свойство `background-image`

Назначение фонового рисунка производится с помощью свойства CSS **`background-image`**. Приведенный ниже пример задает в качестве фонового изображения рисунок из файла *texture1.png*, находящегося в той же папке, что и сама страница.

```
body { background-image: url(texture1.png); }
```

Использование фонового изображения

Можно задать не только цвет фона, но и фоновое изображение. Сначала будет отображаться фоновый цвет, заданный свойством `background-color`, затем будет загружаться фоновое изображение, постепенно заполняя страницу.

Если назначить и фоновый цвет, и фоновое изображение, ваша страница будет лучше восприниматься. А если по какой-то причине фоновое изображение не загрузится, цвет фона обеспечит необходимый контраст с цветом текста. Если фоновое изображение меньше, чем окно браузера, а его конфигурация задана таким образом, что изображение не заполняет все окно автоматически, то фоновый цвет будет отображаться в области, не занятой изображением. CSS-код для одновременного задания цвета фона и фонового рисунка имеет вид:

```
body { background-color: #99cccc;
background-image: url(background.jpg); }
```

Отображение фонового изображения браузером

Может показаться, что графический объект, используемый в качестве фонового рисунка, должен иметь размеры, близкие к размеру окна браузера. На самом деле фоновый рисунок часто намного меньше размера окна браузера. Обычно фоновое изображение имеет форму длинного, узкого прямоугольника или маленького квадрата. По умолчанию веб-браузер заполняет или «мостит» рисунок в окне так, чтобы вся область была закрыта изображением, как показано на рис. 4.16 и 4.17. Размер изображений невелик, чтобы они загружались так быстро, как только возможно.

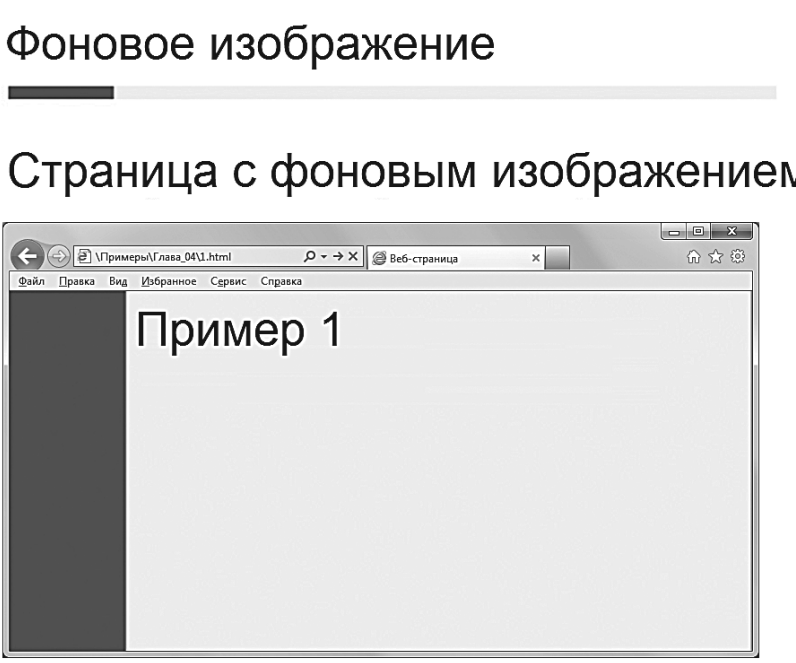


Рис. 4.16. Заполнение окна браузера с использованием изображения в форме узкого длинного прямоугольника

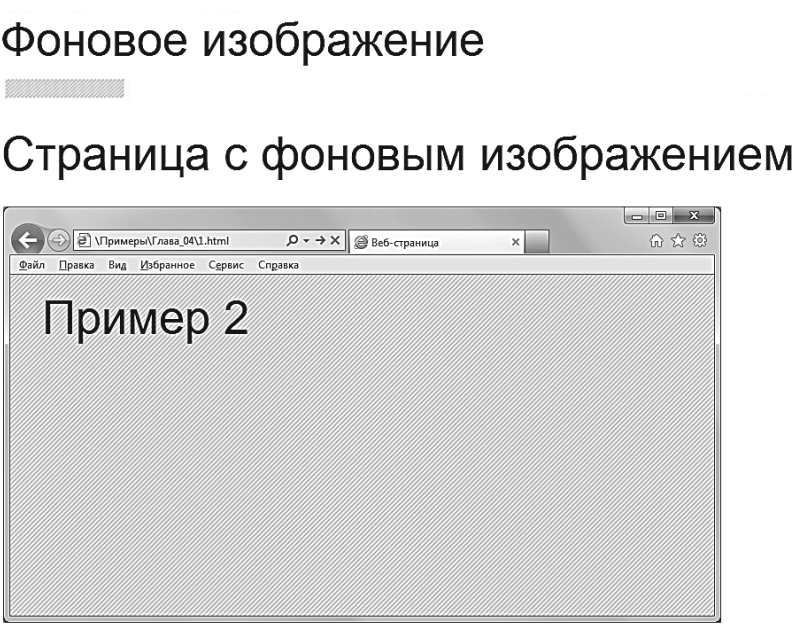


Рис. 4.17. Заполнение окна браузера с использованием изображения в форме маленького прямоугольника

Свойство `background-repeat`

По умолчанию все окно браузера заполняется небольшим фоновым графическим элементом, заданным как фоновое изображение данной страницы. Такое форматирование может применяться и для других элементов, таких как заголовки, абзацы и пр. Вы можете изменить характер заполнения с помощью свойства CSS `background-repeat`. ***Свойство* `background-repeat`** может принимать несколько значений `repeat` (по умолчанию), `repeat-x` (повтор по горизонтали), `repeat-y` (повтор по вертикали) и `no-repeat` (без повторов). На рис. 4.18 показаны примеры применения существующих элементов фоновых изображений с использованием различных значений свойства `background-repeat`.

Рис. 4.18. Примеры использования свойства CSS `background-repeat`

CSS3 предоставляет дополнительные значения свойства `background-repeat`, которые поддерживаются еще не всеми браузерами:

- `background-repeat: space;`

  Повторяет фоновое изображение без обрезки (или урезания) его частей благодаря тому, что регулирует пустое пространство вокруг повторяющихся изображений.

- `background-repeat: round;`

  Повторяет фоновое изображение и масштабирует (настраивает) его размеры так, чтобы избежать обрезки.

Свойство `background-position`

С помощью свойства `background-position` можно указать в качестве местоположения фонового рисунка не только установленный по умолчанию верхний левый угол, но и любую другую позицию. Значения свойства `background-position` могут быть выражены в процентах, пикселах или вариантах `left`, `top`, `center`, `bottom` и `right`. Первое значение указывает положение по горизонтали. Второе — положение по вертикали. Если указано только одно значение, второе устанавливается по умолчанию как `center`.

Рис. 4.19. Отображение фонового рисунка цветка с правой стороны было задано с помощью CSS

На рис. 4.19 небольшое изображение цветка было помещено с правой стороны элемента с использованием следующего правила стилей:

```
h2 { background-image: url(trilliumbg.gif);
background-position: right;
background-repeat: no-repeat; }
```

Практическое задание 4.6

Давайте потренируемся применять фоновые изображения. В ходе этого задания будет изменен файл *index.html*, созданный на предыдущем практическом задании (см. рис. 4.10). Вы сконфигурируете селектор элемента `h2`, добавив фоновое изображение, которое не будет повторяться. Найдите на диске, прилагающемся к книге, в папке *Примеры\Глава_04\starters* файл *trilliumbullet.gif*. Сохраните файл в папке с именем *4.6*. После того как вы выполните задание, ваш файл будет выглядеть так, как показано на рис. 4.20.

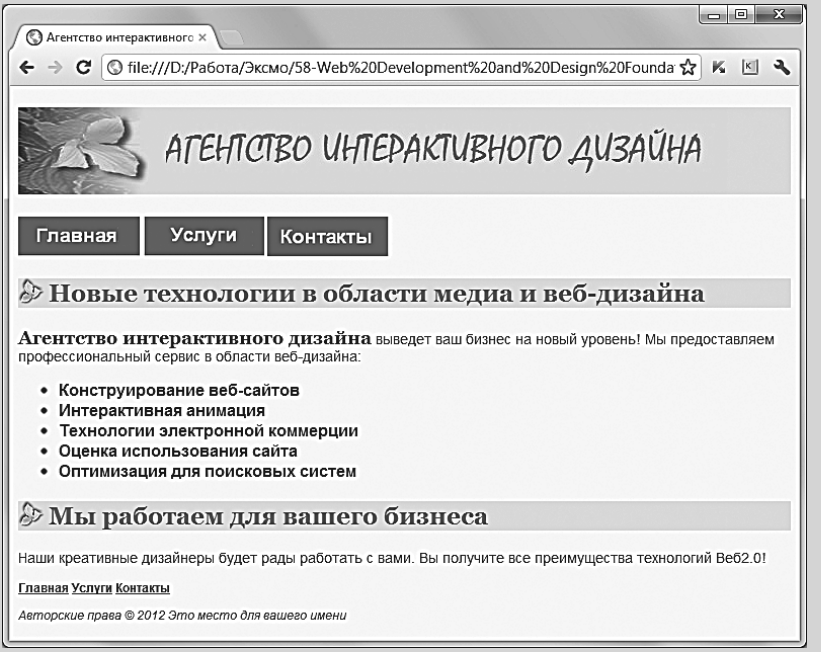


Рис. 4.20. Фоновое изображение в области `h2` конфигурировано с помощью свойства `background-repeat`: `no-repeat`

1. Откройте файл *index.html* в программе Блокнот (Notepad). Измените правило стиля для селектора элемента `h2`, сконфигурировав свойства `background-repeat` и `background-image`. В качестве фонового рисунка используйте изображение из файла *trilliumbullet.gif*. Установите фон так, чтобы он не повторялся. Правила стилей селектора элемента `h2` следующие:

```
h2 { background-color:#d5edb3;
color:#5c743d;
font-family: Georgia, "Times New Roman", serif;
background-image: url(trilliumbullet.gif);
background-repeat: no-repeat; }
```

2. Сохраните файл с именем *index.html*. Запустите его в браузере и проверьте страницу. Можно заметить, что текст в элементе `h2` отображается поверх фонового изображения. В этом случае страница будет выглядеть привлекательнее, если увеличить пространство или отступ перед текстом, отображаемым в элементах `h2`. Чтобы увеличить пустое пространство с левой стороны элемента, примените CSS-свойство `padding-left`. Добавьте к селектору элемента `h2` следующее описание:

```
padding-left: 30px;
```

3. Сохраните страницу и протестируйте ее. Она должна выглядеть так, как показано на рис. 4.20. Этот пример находится на диске, прилагающемся к книге, в папке *Примеры\Глава_04\4.6*.

ЧаВо

КАК ПОСТУПИТЬ, ЕСЛИ ФАЙЛЫ С МОИМИ ИЗОБРАЖЕНИЯМИ НАХОДЯТСЯ В ОТДЕЛЬНОЙ ПАПКЕ?

Бывает полезно организовать веб-страницу таким образом, чтобы рисунки находились в отдельной папке. На рис. 4.21 представлена такая структура, когда файлы рисунков в форматах GIF и JPEG расположены в отдельной папке с именем *images*. В этом случае вам нужно поместить ссылку на папку с рисунками в ваш HTML- или CSS-код. Это делается следующим образом:

- В CSS-код добавляется такая строка, загружающая изображение *background.gif*, находящееся в папке *images*:

```
body { background-image:
url(images/background.gif); }
```

- HTML-код для отображения на странице изображения с именем *logo.jpg* из папки *images* имеет вид:

```
<img src="images/logo.jpg" alt="Логотип" width="588"
height="120">
```

Рис. 4.21. Графические файлы содержатся в папке *images*

Свойство `background-attachment`

Применяйте свойство `background-attachment`, чтобы определить, должно ли фоновое изображение оставаться неподвижным или прокручиваться вместе со страницей в окне просмотра браузера. Значением свойства `background-attachment` может быть `fixed` или `scroll` (по умолчанию).

4.6. Дополнительные возможности при работе с изображениями

В этом разделе демонстрируются дополнительные приемы работы с изображениями на веб-страницах. Обсуждаемые темы включают карты изображений, значки веб-сайтов, разрезание изображений и CSS-спрайты.

Карты изображений

Карта изображений — это изображение, которое может использоваться в качестве одной или нескольких гиперссылок. Карта изображений содержит как минимум одну, а чаще несколько ***активных областей***, активизируемых щелчком мыши и связанных с другими веб-страницами, файлами или веб-сайтами. Вы много раз пользовались картами изображений, не подозревая об этом. Частое применение карт изображений — это создание интерактивных карт, выбор точки на которой производится мышью. На рис. 4.22 изображена страница сайта **nerrs.noaa.gov**, представляющая собой карту островов. С помощью этой карты посетители сайта выбирают интересующий их остров.

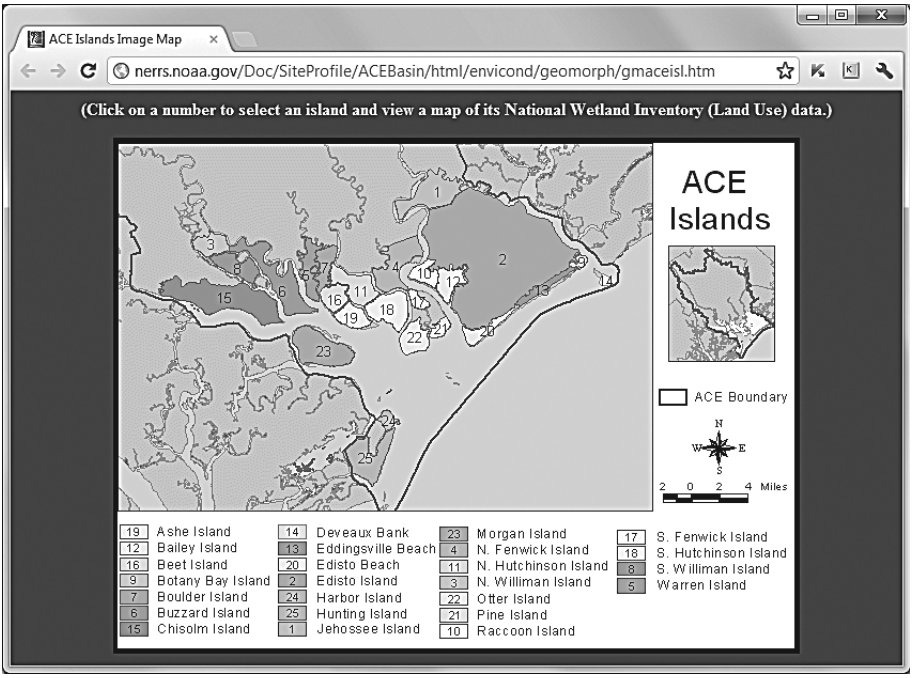


Рис. 4.22. На этом сайте для выбора острова используется карта изображений

Большинство приложений для верстки веб-страниц, таких как Adobe Dreamweaver, имеют инструменты, помогающие быстро и легко создавать карты изображений. Если вы не имеете возможности пользоваться такими программами, наиболее сложной частью для вас будет задание координат областей гиперссылок. Координаты активной области веб-страницы представляют собой пары чисел: первая равна расстоянию в пикселах от верхней границы страницы до выбранной области, вторая — расстоянию от левой границы. Если вы работаете вместе с полиграфистами, они помогут вам определить эти расстояния. Другой вариант — открыть ваше изображение в каком-либо графическом приложении, таком как Adobe Photoshop, Adobe Fireworks или GIMP, в которых можно посмотреть приблизительные координаты. Данные значения координат можно изменять при работе с HTML-кодом веб-страницы. Активизируемая область может быть прямоугольной, круглой или многоугольной. При работе с картами изображений используются два элемента — `map` и `area`. ***Элемент*** **`map`** является контейнерным элементом и используется в начале и в конце карты изображений. ***Атрибут*** **`name`** связывает элемент `map` с соответствующим рисунком. Элемент изображения использует ***атрибут*** **`usemap`** для указания на то, какой элемент `map` карты имеется в виду. Например, код `<img src="fishingboat.jpg" usemap="#boat">` будет связан с картой изображений, описанной кодом `<map name="boat" id="boat">`. Атрибут `id` присваивает уникальный идентификатор области карты.

Элемент **`area`** предназначен для указания границ области карты и используется с атрибутами `shape`, `coords`, `alt` и `href` (синтаксис XHTML — `<area />`). В таблице 4.4 перечислено, какие типы координат (`cords`) требуются для каждой формы активизируемой области.

Таблица 4.4. Координаты формы

Форма	Координаты	Разъяснение
Прямоугольная (`rect`)	"x1, y1, x2, y2"	Координаты x1, y1 соответствуют верхнему левому углу прямоугольной области. Координаты x2, y2 соответствуют нижнему правому углу прямоугольной области
Круглая (`circle`)	"x, y, r"	Координаты x1, y1 соответствуют центру круглой области. Значение r — радиус круга в пикселах
Многоугольная (`polygon`)	"x1, y1, x2, y2, x3, y3" и т. д.	Каждая пара значений (x, y) соответствует одному углу многоугольной области

Далее будет рассматриваться только прямоугольная область карты изображений. В этом случае значение атрибута `shape` — `rect` для задания активной области должно содержать следующие координаты в пикселах: расстояние от левой кромки страницы до верхнего левого угла активизируемой области, расстояние от верхней кромки страницы до верхнего левого угла области, расстояние от левой кромки страницы до нижнего правого угла области и расстояние от верхней кромки страницы до нижнего правого угла области.

На рис. 4.23 приведено изображение рыбацкой лодки. Штриховым прямоугольником обозначена активная область.

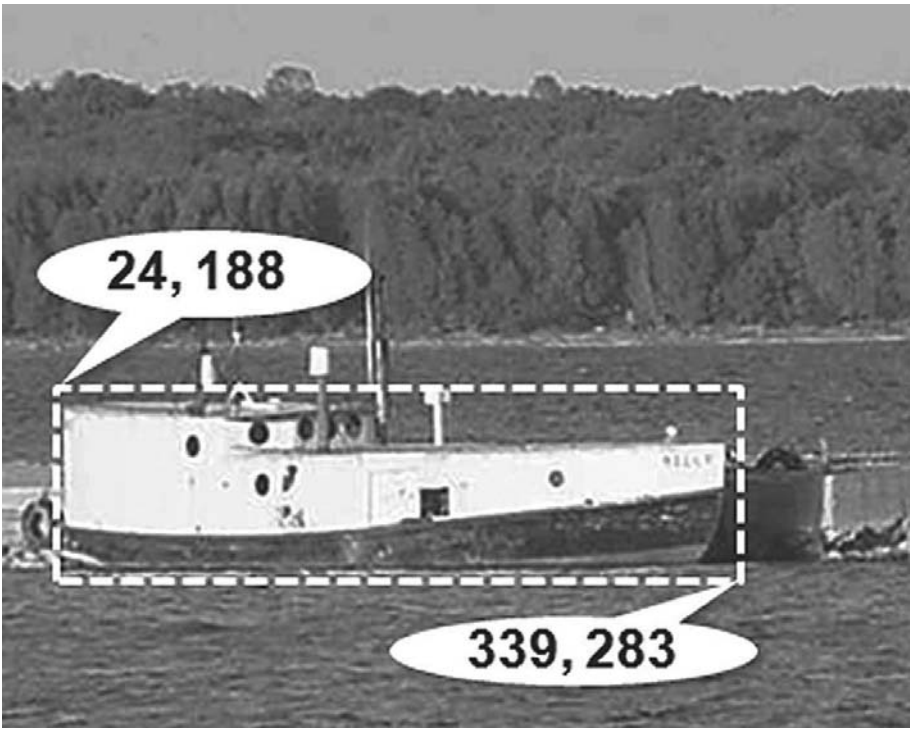


Рис. 4.23. Пример карты изображений

Приведенные на рисунке величины (24, 188) указывают, что расстояние от левой кромки страницы до верхнего левого угла активизируемой области равно 24 пиксела, а расстояние от верхней кромки страницы до верхнего левого угла области — 188 пикселов. Пара координат у нижнего правого угла (339, 283) соответствует расстоянию от левой кромки страницы до нижнего правого угла активизируемой области в 389 пик-

селов и расстоянию от верхней кромки страницы до нижнего правого угла области в 283 пиксела. HTML-код, создающий карту изображений, имеет следующий вид:

```
<map name="boat" id="boat">
<area href=" http://www.doorcountyvacations.
com " shape="rect" coords="24, 188, 339, 283"
alt="Рыбалка в графстве До">
</map>
<img src="fishingboat.jpg" usemap="#boat"
alt="Рыболовное судно" width="416"
height="350">
```

Большинство разработчиков веб-сайтов сами не программируют карты изображений. Как уже упоминалось выше, проще всего создавать их, используя приложения для верстки веб-страниц. Эту возможность предоставляют также условно-бесплатные программы, такие как CoffeeCup Image Mapper[1] и Adobe Dreamweaver[2].

Значки веб-сайтов

Замечали ли вы когда-нибудь маленькие значки в адресной строке или на ярлыках браузера? Это ***значки веб-сайтов***, также называемые ***favicon***, фавиконами или персональными иконками, представляющие собой квадратные изображения (16×16 пикселов или 32×32 пиксела), связанные с определенной веб-страницей. Значки, подобные показанным на рис. 4.24, могут отображаться в адресной строке, на вкладках в окне браузера, на ярлыках закладок или в списке избранного.

Значки веб-сайтов можно создавать, используя графические приложения, такие как Adobe Fireworks, или различные веб-сайты **www.favicongenerator.com**, **www.favicon.cc** или **www.freefavicon.com**. В ранних версиях Internet Explorer (например, версии 5 и 6) предполагается, что файл значка должен иметь имя *favicon.ico* и располагаться в корневом каталоге веб-сервера. Более современный подход к этому вопросу — связать файл с веб-страницей с помощью ссылки. Вспомните, что в главе 3 вы программировали код элемента `link` в разделе заголовка веб-страницы, связывающий страницу с внешней таблицей стилей.

[1] www.coffeecup.com
[2] www.adobe.com/ru/products/dreamweaver.html

Вы также можете использовать элемент `link` для того, чтобы связать со страницей файл значка. Для этого используются три атрибута: `rel`, `href` и `type`. Значение атрибута `rel` — `icon`. Значением атрибута `href` является имя файла с изображением значка. Значением атрибута `type` является MIME-описание изображения и по умолчанию для файлов *.ico* равно `image/x-icon`.

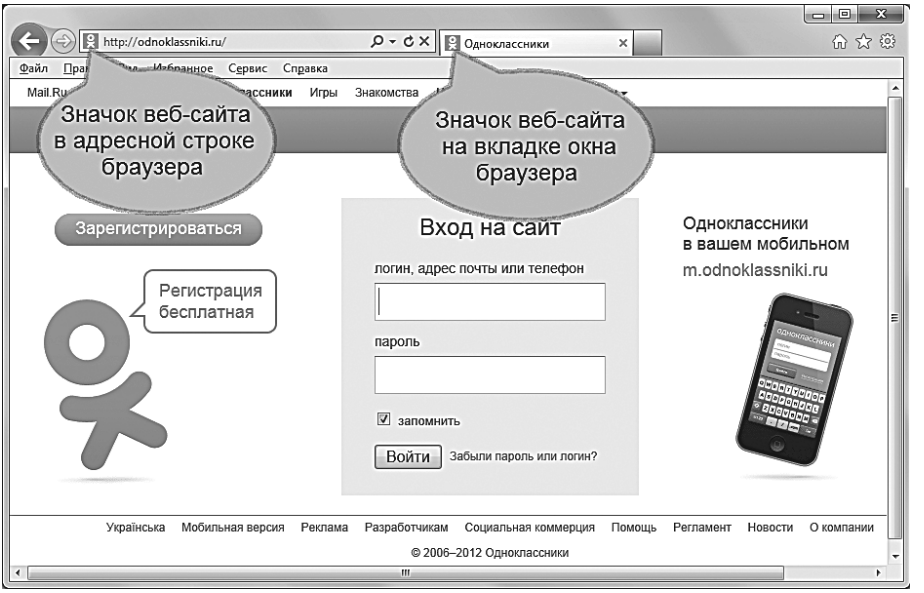


Рис. 4.24. Значки веб-сайтов отображаются в адресной строке и на вкладках окна браузера

HTML-код, связывающий значок *favicon.ico* со страницей, имеет вид:

```
<link rel="icon" href="favicon.ico" type="image/x-
icon">
```

Обратите внимание, что для того чтобы ваш код был совместим с браузером Internet Explorer и соответствовал требованиям к синтаксису, устанавливаемым Microsoft, необходимо добавить еще одну строку.

```
<link rel="shortcut icon" href="favicon.ico"
type="image/x-icon">
```

Необходимо также учитывать, что отображение значков сайтов не вполне корректно поддерживается веб-браузерами. Вам может потребоваться публикация ваших файлов во Всемирной паутине (см. при-

ложение Г), чтобы значки отображались даже наиболее современными версиями Internet Explorer. Другие браузеры, такие как Firefox, Safari, Chrome и Opera, отображают значки более предсказуемо. Они также поддерживают форматы изображений *.gif* и *.png*.

Практическое задание 4.7

Давайте потренируемся создавать значки сайтов. Используйте файл *favicon.ico* из папки *Примеры\Глава_04\starters* на диске, прилагающемся к книге. В качестве основы также возьмите файлы с прошлого практического задания, хранящиеся в папке *Примеры\Глава_04\4.6.*

1. Запустите программу Блокнот (Notepad) и откройте файл *index.html*. Добавьте в раздел заголовка веб-страницы следующие строки:

```
<link rel="icon" href="favicon.ico"
type="image/x-icon">
<link rel="shortcut icon" href="favicon.ico"
type="image/x-icon">
```

2. Сохраните страницу под именем *index.html*. Запустите браузер Firefox и протестируйте страницу. На вкладке в браузере Firefox вы должны увидеть маленький цветок, как показано на рис. 4.25. Пример вы найдете на диске, прилагающемся к книге, в папке *Примеры\Глава_04\4.7*.

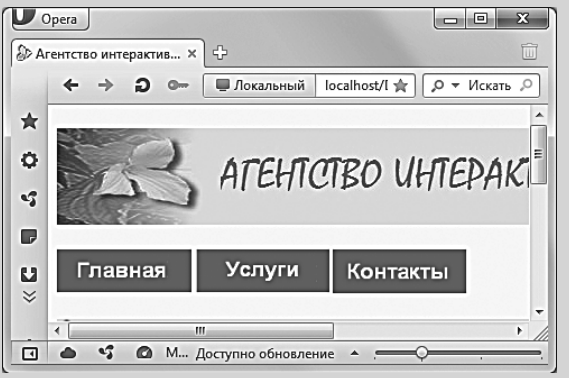


Рис. 4.25. На вкладке в браузере Firefox отображается значок сайта

Разрезание изображения

Полиграфисты и веб-дизайнеры могут создавать весьма сложные изображения. Некоторые части этих изображений было бы лучше представить

в формате GIF, чем в JPEG. Другие части, напротив, было бы лучше представить в формате JPEG, чем в GIF. Путем ***разделения*** одного сложного изображения на несколько более простых, меньшего размера, можно оптимизировать процесс отображения изображения в целом. Возможно, вы захотите добавить к большому сложному изображению такой дополнительный эффект, как прокрутка части изображения с помощью мыши. В этом случае требуется отдельный доступ к различным частям изображения и, следовательно, изображение необходимо разделить. Разделенное изображение представляется в виде нескольких графических файлов, форматируемых с помощью HTML-таблиц. Большинство графических приложений, таких как Adobe Fireworks и Adobe Photoshop, предоставляют возможность по разделению графики, автоматически создавая для вас HTML-таблицы. Дополнительная информация по разделению изображений приведена на сайте **HTMLbook.ru/content/razrezanie-i-skleyka-izobrazheniy**.

CSS-спрайты

CSS-спрайты — современная техника оптимизации использования изображений на веб-страницах. Спрайт — это графический файл, содержащий несколько небольших рисунков, которые задаются как фоновые изображения для различных элементов веб-страницы. Управление положением изображения осуществляется с помощью свойств CSS `background-image`, `background-repeat` и `background-position`. Поскольку изображение всего одно, уменьшается время его загрузки (для составного изображения браузеру нужно сделать всего один http-запрос, а не несколько, как для отдельных более маленьких изображений). Вы будете работать с CSS-спрайтами в главе 7.

4.7. Источники графических файлов и советы по работе с графикой

Источники графических файлов

Изображения могут быть получены различными способами: вы можете создавать их с помощью различных графических приложений, скачивать с бесплатных веб-сайтов, покупать и загружать с тематических сайтов, приобретать коллекции графики на CD и DVD-дисках, делать цифровые фотографии, сканировать изображения, а также пользоваться услугами графических дизайнеров. Наиболее популярные графические приложения — это Adobe Fireworks и Adobe Photoshop (платные).

Популярные бесплатные приложения для работы с графикой: GIMP[1], Google Picasa[2] (включает веб-сервис) и Pixlr[3] (онлайн-приложение). Обычно эти приложения содержат программу обучения и образцы рисунков, которые помогут вам начать работу с изображениями. На сайте **photoshop.demiart.ru/advanced_logo.sHTML** опубликован урок использования программы Adobe Photoshop для создания логотипов.

Акцент на этичность

Главное, чего вы, безусловно, не должны делать, — это загружать изображения без разрешения его автора. Все права на размещенные на веб-сайтах материалы защищены законом об авторском праве (даже если на странице нет соответствующего значка — ©) и не могут бесплатно использоваться без разрешения автора.

Имеется большое количество сайтов с бесплатной графикой, однако часто эта графика предлагается бесплатно только на условиях некоммерческого использования. Запустите поисковую систему, задав ключевые слова «бесплатные лицензионные изображения» или «стоковые изображения» и вы получите больше графики, чем будете в состоянии просмотреть. Ниже приведены некоторые из сайтов с бесплатными и недорогими изображениями (и не только), которые могут быть вам полезными:

- Клипарты Microsoft: **office.microsoft.com/ru-ru/images/**
- DepositPhotos: **ru.depositphotos.com/**
- Поисковая система бесплатных фотографий: **www.everystockphoto.com**
- Free Images: **www.freeimages.co.uk**
- Free Stock Solution: **www.tssphoto.com**
- SuperStock: **www.superstock.com**
- iStockphoto: **russki.istockphoto.com**

Также можно создавать баннеры и изображения кнопок прямо во Всемирной паутине. Ряд сайтов предоставляют такую возможность — в обмен иногда необходимо позволить разместить бесплатную рекламу с использованием ваших изображений, иногда предлагается платное членство, но часть из них полностью бесплатны. Для поиска таких сайтов воспользуйтесь поисковой системой с запросом «создание бесплатных баннеров онлайн»:

[1] gimp.org
[2] picasa.google.com
[3] pixlr.com/editor

- Animation Online: **www.animationonline.com**
- Web 2.0 LogoCreator: **www.creatr.cc/creatr**
- Cooltext.com: **www.cooltext.com**
- Ad Designer.com: **www.addesigner.com**

Советы по использованию графических файлов

Изображения помогут вам придать вашему сайту привлекательный, интересный вид. Однако они могут и ухудшить мнение пользователей о сайте, если будут загружаться слишком медленно, разочаровывая посетителей.

Используйте те же изображения многократно

Если изображение с вашего сайта требуется для веб-страницы, оно сохраняется в кэше браузера на жестком диске посетителя. При повторном запросе этого файла, он будет загружаться с диска, а не из Всемирной паутины. В результате все страницы, содержащие это изображение, будут загружаться быстрее. Таким образом, для логотипа и навигационных кнопок для всех страниц вам следует использовать одни и те же файлы, а не создавать новые для каждой страницы.

Учитывайте соотношение размер/качество

При создании изображения в графическом приложении можно сохранять его с различным качеством. При этом имеется соответствие между качеством изображения и размерами файла — чем лучше качество, тем больше файл. Выбирайте файлы с минимальным размером и все еще удовлетворительным качеством изображения. Возможно, для оптимального выбора вам придется поэкспериментировать со своей графикой. Также будьте внимательны к графическим файлам, созданными другими дизайнерами — файлы с размером более 300 Кб использовать не следует.

Учитывайте время загрузки изображений

Используйте изображения на веб-странице с осторожностью — на их загрузку требуется время. Оптимизируйте изображения, чтобы уменьшить размеры их графических файлов. Также вы можете воспользоваться ресурсом **2ip.ru/time-calc/** для расчета скорости загрузки файла определенного размера при любой указанной вами скорости подключения.

Правильно выбирайте разрешение

Веб-браузеры отображают графику с относительно низким разрешением — 72 или 96 пиксела на дюйм (ppi). Большинство цифровых фотокамер и сканеров способны создавать изображения с гораздо лучшей разрешающей способностью. И конечно, большее разрешение означает больший размер файла. Несмотря на то, что браузеры не способны отображать графику высокого разрешения, для передачи больших файлов все больше используются широкополосные сети.

Будьте осторожны с цифровыми фотографиями и сканированными изображениями, используйте разрешение, соответствующее разрешению браузера. В противном случае рисунок шириной в 1 дюйм и разрешением 150 ppi может иметь ширину около 2 дюймов, если его отобразить в браузере на мониторе с разрешением 72 ppi.

Задавайте размеры

Будьте внимательны при использовании атрибутов `height` и `width` элемента `image`. Это позволит браузеру резервировать нужное место на странице для ваших рисунков, и их загрузка будет производиться быстрее. Не пытайтесь изменять размер изображения с помощью атрибутов `height` и `width`. Хотя это и сработает, страница будет загружаться медленнее, и качество будет хуже. Вместо этого при необходимости изменяйте размер с помощью графических приложений.

Учитывайте яркость и контраст

Термин ***гамма*** обозначает яркость и контраст изображения на мониторе. Мониторы компьютеров с операционными системами Windows и OS X по умолчанию используют разные настройки цветовой гаммы (Windows — 1.8, OS X — 2.2). Изображение, имеющее хороший контраст на мониторе компьютера с системой Windows, может выглядеть слегка блеклым на мониторе с системой OS X. А изображение, созданное для монитора системы OS X, может выглядеть несколько темноватым на мониторе с операционной системой Windows. И даже монитор соответствующей операционной системы может иметь параметры отображения, несколько отличные от заданных по умолчанию. Разработчик веб-сайтов не имеет возможности изменять гамму монитора пользователя, но он должен знать, что по перечисленным выше причинам его изображения будут выглядеть по-разному на разных платформах.

4.8. Визуальные эффекты CSS3

В этом разделе освещены новые свойства CSS3, обеспечивающие визуальные эффекты на веб-страницах, включая обрезку и масштабирование фона, множественные фоновые изображения, скругленные углы, тени блока и текста, эффекты непрозрачности, прозрачные цвета RGBA и градиенты.

Свойство `background-clip`

Свойство **`background-clip`** спецификации CSS3 ограничивает отображение фонового рисунка посредством установки следующих значений:

- `content-box` (ограничивает отображение областью, находящейся позади содержимого веб-страницы);
- `padding-box` (ограничивает отображение областью, находящейся позади содержимого веб-страницы и отступами);
- `border-box` (установлено по умолчанию; ограничивает отображение областью, находящейся позади содержимого веб-страницы, отступов и границы. От свойства `padding-box` отличается тем, что рисунок отображается позади границы, которая задана прозрачной).

Свойство `background-clip` поддерживается современными браузерами, хотя приложение Safari, вместо `background-clip`, прописанного в черновом варианте спецификации, требует собственное свойство `-webkit-background-clip`. Имейте в виду, что если вы укажете в коде нестандартное свойство, ваш код CSS не пройдет проверку валидатором W3C. На рис. 4.26 показано фоновое изображение, вывод которого задан с помощью значения `content-box`. Пример страницы доступен на диске, прилагающемся к книге, в папке *Примеры\Глава_04\clip*.

Рис. 4.26. Свойство CSS3 `background-clip`

Используется следующая таблица CSS:

```
.test { background-image: url(myislandback.jpg);
-webkit-background-clip: content-box;
background-clip: content-box;
width: 400px;
padding: 20px;
border: 1px solid #000; }
```

Свойство `background-origin`

Свойство **`background-origin`** спецификации CSS3 управляет размещением фонового изображения посредством следующих значений:

- `content-box` (размещает относительно области содержимого страницы);
- `padding-box` (установлено по умолчанию; размещает относительно области отступа);
- `border-box` (размещает относительно области границы).

Рис. 4.27. Свойство `background-origin`

На рис. 4.27 показано фоновое изображение, местоположение которого задано в правом нижнем углу блока содержимого страницы. Пример страницы можно найти на диске, прилагающемся к книге, в папке *Примеры\Глава_04\origin*.

Используется следующая таблица стилей:

```
.test { background-image: url(trilliumsolo.jpg);
background-origin: content-box;
background-repeat: no-repeat;
background-position: right bottom;
width: 400px;
padding: 20px;
border: 1px solid #000; }
```

Возможно, вы заметили, что, конфигурируя фоновые изображения, часто используют несколько свойств CSS. Обычно свойства работают сообща. Однако имейте в виду, что свойство `background-origin` не может быть задействовано, если свойству `background-attachment` присвоено значение `fixed`.

Свойство `background-size`

Свойство **`background-size`** спецификации CSS3 можно использовать для изменения размера или масштабирования фонового изображения. Значениями свойства `background-size` могут быть:

- пара процентных значений (ширина, высота);
- пара числовых значений в пикселах (ширина, высота);
- `auto`, `contain` или `cover`.

Если указать только одно числовое или процентное значение, вторым по умолчанию будет установлен вариант `auto`. Значение `contain` задает масштабирование изображения по вертикали до соответствия размеру контейнера (соотношение сторон не меняется). Значение `cover` задает масштабирование изображения по горизонтали до соответствия размеру контейнера (соотношение сторон не меняется). На рис. 4.28 (см. на диске, прилагающемся к книге, папку *Примеры\Глава_04\size*) фоновое изображение было сконфигурировано так, чтобы оно заполнило контейнер по вертикали:

```
div { background-image: url(myislandback.jpg);
background-size: cover;
background-repeat: no-repeat;
width: 200px;
padding: 20px; }
```

Рис. 4.28. Свойство CSS3 `background-size`

Применяя новые свойства CSS3, помните, что они поддерживаются только современными версиями браузеров. Убедитесь, протестировав код без свойств спецификации CSS3, что страницы легкочитаемы и удобны для пользователя. Последнюю информацию о поддержке браузерами свойств CSS3 можно получить, посетив сайт **www.quirksmode.org/css/contents.html**.

Множественные фоновые изображения

Теперь рассмотрим применение на веб-странице множественных фоновых изображений. И хотя модуль Backgrounds and Borders спецификации CSS3 все еще находится в стадии доработки, последние версии наиболее популярных веб-браузеров поддерживают использование множественных изображений.

На рис. 4.29 показана веб-страница с двумя фоновыми изображениями, заданными в селекторе элемента `body`: повторяющийся градиент, растянутый во все окно браузера, и рисунок цветка, отображающийся один раз в правом углу нижнего колонтитула.

Рис. 4.29. Браузер Firefox отображает множественные фоновые изображения

Примените свойство `background`, чтобы конфигурировать множественные фоновые изображения. Определение каждого изображения отделено запятой. Можно добавить значения свойства, чтобы указать

местоположение изображения и повторяемость, однако это не обязательно. Для свойства `background` достаточно сокращенной нотации: перечислите значения, необходимые для правильной работы таких свойств, как `background-position` и `background-repeat`.

Прогрессивное улучшение

Множественные фоновые изображения сегодня поддерживают последние версии браузеров Firefox, Chrome, Safari, Opera и Internet Explorer. Имейте в виду, что они не поддерживаются в версиях браузера Internet Explorer ранее 9. Вы примените технику ***прогрессивного улучшения***, которую веб-разработчик и приверженец HTML5 Кристиан Хэйлман описывает так: «Начните с удобной функциональности, затем шаг за шагом обогащайте опыт пользователей, тестируя, поддерживаются ли улучшения перед тем, как их применять». Другими словами, начните с веб-страницы, которая правильно отображается в большинстве браузеров, а затем добавляйте новые дизайнерские техники, к примеру, множественные фоновые изображения, чтобы они улучшали внешний вид страницы для посетителей, пользующихся браузерами, поддерживающими эту новую технику.

Чтобы добиться прогрессивного улучшения при использовании множественных фоновых изображений, сначала задайте отдельное свойство `background-image` с одним изображением (которое визуализируется в большинстве браузеров), а потом — свойство `background` с множественными изображениями (которые визуализируются браузерами, поддерживающими это свойство, и игнорируются остальными).

Практическое задание 4.8

Мы потренируемся конфигурировать множественные фоновые изображения. В этом практическом задании вы настроите селектор элемента `body`, чтобы отобразить на веб-странице множественные фоновые изображения. Из папки *Примеры\Глава_04\starters* на диске, прилагающемся к книге, скопируйте файлы *trilliumgradient.png* и *trilliumfoot.gif*. Сохраните изображения в папку *4.8.* Запустите текстовый редактор и откройте файл *index.html* из прошлого практического задания.

1. Модифицируйте правило стилей для селектора элемента `body`. Задайте отображение файла *trilliumgradient.png* с помощью свойства `background-image`. Правило стилей будет применено браузерами, не поддерживающими множественные фоновые

изображения. Задайте отображение файлов *trilliumgradient.png* и *trilliumfoot.gif* с помощью свойства `background`. Изображение *trilliumfoot.gif* не должно повторяться и будет отображаться в правом нижнем углу. Правила стилей селектора `body` следующие:

```
body { background-color: #f4ffe4; color: #333333;
font-family: Arial; Verdana, sans-serif;
min-width: 700px;
background-image: url(trilliumgradient.png);
background: url(trilliumfoot.gif) no-repeat bottom right,
url(trilliumgradient.png); }
```

2. Сохраните страницу под именем *index.html*. Запустите браузер и протестируйте страницу. Она будет выглядеть по-разному, в зависимости от того, какой браузер используется. *Примечание*: убедитесь, что CSS-валидатор консорциума W3C настроен на версию CSS3. Для этого перед поверкой кода выберите в раскрывающемся списке **Профиль** вариант **CSS3**.
3. Обычно существует несколько способов дизайна веб-страницы. Возьмем расположение цветка в области нижнего колонтитула страницы. Почему бы не задать градиент как фоновое изображение селектора элемента `body`, а цветок — как фон нижнего колонтитула? При такой конфигурации страница будет отображаться почти одинаково во всех современных популярных браузерах. Давайте попробуем. Отредактируйте файл *index.html*. Удалите свойство `background` из селектора `body`. Пример кода следующий:

   ```
   body { background-color: #f4ffe4; color: #333333;
   background-image: url(trilliumgradient.png); }
   ```

   Далее конфигурируйте изображение *trilliumfoot.gif* в качестве фонового для селектора `#footer`. Задайте значение высоты достаточно большое, чтобы изображение было видно. Код следующий:

   ```
   #footer { font-size: .75em; font-style: italic;
   background-image: url(trilliumfoot.gif);
   background-repeat: no-repeat;
   background-position: right;
   height: 90px; }
   ```

4. Сохраните страницу под именем *index2.html*. Запустите браузер и протестируйте файл. Во всех современных популярных браузерах страница должна выглядеть, как изображенная на рис. 4.29. Примеры этого практического задания можно найти на диске, прилагающемся к книге, в папке *Примеры\Глава_04\4.8.*

Скругление углов

Работая с границами и блочной моделью, вы, возможно, замечали, как много прямоугольников на веб-страницах! В CSS3 введено новое свойство `border-radius`, которое можно использовать для скругления углов прямоугольников. Значения свойства `border-radius` включают от одного до четырех числовых значений в пикселах, единицах em или процентах, задающих радиус углов. Если указывается одно значение, оно конфигурирует все четыре угла. Если указываются четыре значения, углы конфигурируются в таком порядке: левый верхний, правый верхний, правый нижний и левый нижний. Углы можно конфигурировать по-отдельности, применив свойства `border-bottom-left-radius`, `border-bottom-right-radius`, `border-top-left-radius` и `border-top-right-radius`.

Однако при конфигурировании скругленных углов возникают сложности. Разработчики движков визуализации страниц в браузерах, таких как WebKit (используемый в программах Safari и Google Chrome) и Gecko (используемый в Firefox и других браузерах корпорации Mozilla), создали собственные свойства для скругления углов. Кроме того, Internet Explorer 9 — первая версия этого браузера, поддерживающая свойство `border-radius`. Поэтому для скругления углов вам придется указать в коде три разных определения стилей:

- `-webkit-border-radius` (для браузеров с движком WebKit);
- `-moz-border-radius` (для браузеров с движком Gecko);
- `border-radius` (синтаксис чернового варианта спецификации W3C).

Постепенно все браузеры станут поддерживать CSS3 и свойство `border-radius`, поэтому укажите его в коде в последнюю очередь. Определения CSS3 для установки границы со скругленными углами показаны в следующем фрагменте кода. Если вы хотите отобразить видимую границу, задайте свойство `border`. Затем установите значения трех свойств `border-radius` в диапазоне до 20 пикселов для достижения лучшего результата. Например:

```
border: 3px ridge #330000;
-webkit-border-radius: 15px;
-moz-border-radius: 15px;
border-radius: 15px;
```

На рис. 4.30 можно увидеть этот код в действии (папка *Примеры\Глава_04\box.html* на диске, прилагающемся к книге). Помня о прогрессивном улучшении, имейте в виду, что пользователи, открывающие страницу в версиях браузера Internet Explorer ранее 9, увидят прямые углы вместо скругленных. Однако на функциональность и юзабилити веб-страницы это не повлияет.

Рис. 4.30. Скругленные углы сконфигурированы с помощью CSS

Имейте в виду, что если вы указываете в коде нестандартные свойства, CSS не пройдет проверку валидатором W3C. Также помните, что есть и другой способ создать скругленные углы — нарисовать в графическом редакторе прямоугольник со скругленными углами и использовать его в качестве фонового изображения. Однако, как только CSS3 станет широко поддерживаемым стандартом, способ реализации скругленных углов станет более распространенным и эффективным.

Практическое задание 4.9

В ходе этого практического задания вы настроите в разделе заголовка область логотипа, в которой будут использоваться фоновое изображение и скругленные углы. Готовая веб-страница будет похожа на изображенную на рис. 4.31.

1. Создайте новую папку под именем *coffeehouse*. Скопируйте в нее файлы *lighthouselogo.jpg* и *background.jpg* из папки *Примеры\Глава_04\starters*. Скопируйте файл *Примеры\Глава_04\starter3.html* в папку *coffeehouse*. Запустите браузер и откройте веб-страницу *starter3.html,* показанную на рис. 4.32.
2. Запустите текстовый редактор и откройте файл *starter3.html.* Сохраните его под именем *index.html*. Отредактируйте глобальную таблицу CSS и добавьте к селектору `h1` следующие правила стилей, конфигурирующие файл *lighthouselogo.jpg* как неповторяющееся фоновое изображение: установите значение высоты,

равное 100 пикселам, ширины — 700 пикселам, размер шрифта — 3 em, левый отступ — 150 пикселов, верхний отступ — 30 пикселов и сплошную границу темно-синего цвета (#000033) со значением радиуса, равным 15 пикселам.

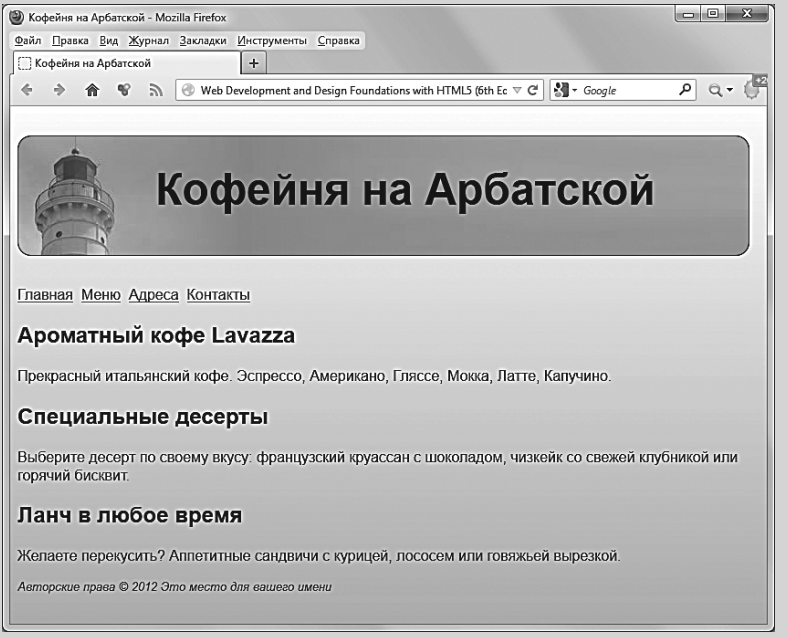


Рис. 4.31. Веб-страница с заданной областью логотипа

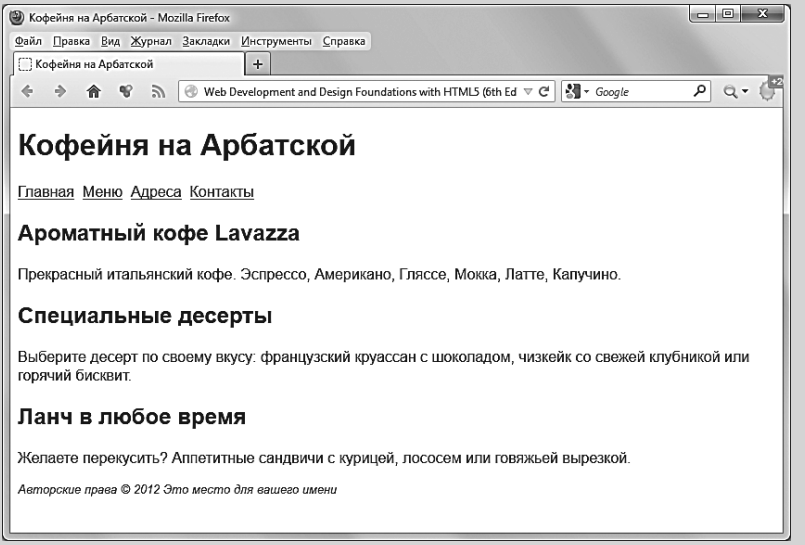


Рис. 4.32. Файл *starter3.html*

Определения стилей следующие:

```
h1 { background-image: url(lighthouselogo.jpg);
background-repeat: no-repeat;
height: 100px; width: 700px; font-size: 3em;
padding-left: 150px; padding-top: 30px;
border: 1px solid #000033;
-webkit-border-radius: 15px;
-moz-border-radius: 15px;
border-radius: 15px; }
```

3. Сохраните файл. Когда вы будете тестировать страницу *index.html* в браузере, поддерживающем скругленные углы, она должна быть похожа на изображенную на рис. 4.31. В противном случае углы логотипа будут прямыми, но веб-страница по-прежнему будет комфортна для просмотра. Сравните свой вариант с примером на диске, прилагающемся к книге (*Примеры\Глава_04\coffeehouse\index.html*).

ЧаВо

ПОСЛЕ ДОБАВЛЕНИЯ НОВЫХ СВОЙСТВ МОЙ CSS-КОД БОЛЬШЕ НЕ ПРОХОДИТ ТЕСТ W3C-ВАЛИДАТОРА. ЧТО МНЕ ДЕЛАТЬ?

Поскольку помимо свойства CSS3 `border-radius` вы используете собственные свойства браузеров `-webkit-border-radius` и `-moz-border-radius`, ваш CSS-код не пройдет валидацию, но страницы будут отображаться в современных браузерах во всей своей красе. В данном случае сознательно было отдано предпочтение визуальному отображению вместо использования абсолютно верного синтаксиса. Через некоторое время все браузеры будут поддерживать свойство `border-radius`, вы сможете удалить собственный код, и ваши CSS пройдут тест валидатора W3C с профилем CSS3. А пока, если вы специально применяете нестандартные свойства, чтобы улучшить веб-страницы, не волнуйтесь, когда тест W3C-валидатора сообщает об ошибках, связанных с этими свойствами.

ЧаВо

ПОЧЕМУ НЕКОТОРЫЕ БРАУЗЕРЫ ИСПОЛЬЗУЮТ СОБСТВЕННЫЕ СВОЙСТВА?

Поддерживая новые CSS-эффекты или предложенные свойства, производители браузеров обычно используют префикс, чтобы свойство определялось как дополнение (расширение) браузера.

Со временем W3C-спецификация свойства становится стабильной и разработчики браузера в конце концов реализуют поддержку официальной спецификации. Поскольку каскад CSS выполняется сверху вниз, принято сначала перечислять в коде правила стилей, содержащие собственные свойства браузеров, в потом стандартное свойство W3C.

Свойство `box-shadow`

Свойство **`box-shadow`** спецификации CSS3 может быть использовано для создания эффекта тени блочных элементов, таких как элемент абзаца или `div`. Разработчики движков WebKit и Gecko для визуализации страниц в браузерах ввели собственные свойства для создания эффекта тени блока. Для создания тени вам придется указать в коде три определения стилей и внести от трех до пяти значений для каждого:

- числовое значение в пикселах смещения тени по горизонтали: положительное значение размещает тень справа, отрицательное значение — слева;
- числовое значение в пикселах смещения тени по вертикали: положительное значение размещает тень снизу, отрицательное — сверху;
- числовое значение в пикселах радиуса размытия (необязательно): более высокие значения задают более сильное размытие и нечеткую тень, 0 задает резкую тень;
- числовое значение в пикселах размера тени (необязательно): положительные значения увеличивают тень, а отрицательные уменьшают;
- допустимое значение цвета.

Пример ниже задает тень темно-серого цвета со смещением по горизонтали и вертикали на 5 пикселов и радиусом размытия, значение которого также равно 5 пикселам:

```
-webkit-box-shadow: 5px 5px 5px #828282;
-moz-box-shadow: 5px 5px 5px #828282;
box-shadow: 5px 5px 5px #828282;
```

Постепенно все браузеры начнут поддерживать CSS3 и официальное свойство `box-shadow`, поэтому укажите это свойство последним в коде. Обратите внимание, что если вы добавляете нестандартные свойства, ваша таблица CSS не пройдет проверку валидатора.

Добавьте необязательное значение `inset`, чтобы конфигурировать внутреннюю тень. Например:

```
-webkit-box-shadow: inset 5px 5px 5px #828282;
-moz-box-shadow: inset 5px 5px 5px #828282;
box-shadow: inset 5px 5px 5px #828282;
```

Свойство `text-shadow`

Свойство **`text-shadow`** спецификации CSS3 задает эффект тени для текста и поддерживается последними версиями современных браузеров, за исключением Internet Explorer (на момент написания книги). Для данного свойства требуются четыре значения:

- числовое значение в пикселах смещения тени по горизонтали: положительное значение размещает тень справа, отрицательное значение — слева;
- числовое значение в пикселах смещения тени по вертикали: положительное значение размещает тень снизу, отрицательное — сверху;
- числовое значение в пикселах радиуса размытия: более высокие значения задают более сильное размытие и нечеткую тень, 0 задает резкую тень;
- допустимое значение цвета.

Ниже приведен пример:

```
text-shadow: 3px 3px 3px #666;
```

Практическое задание 4.10

В ходе этого практического задания вы конфигурируете выровненную по центру область содержимого страницы и примените свойства `text-shadow` и `box-shadow`. Готовая веб-страница будет похожа на изображенную на рис. 4.33.

Запустите текстовый редактор и откройте файл *Примеры\Глава_04\starter3.html,* показанный на рис. 4.32. Сохраните его в папку *coffee-house* под именем *shadow.html*.

1. Задайте выравнивание содержимого по центру, установите значение ширины, равное 800 пикселам, белый цвет фона и небольшой отступ.

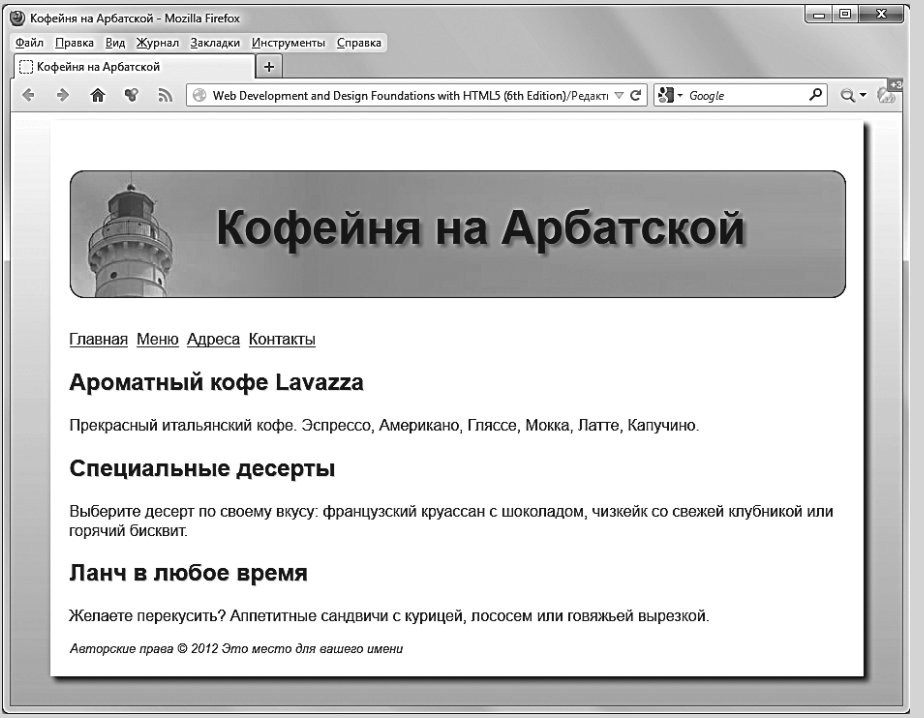


Рис. 4.33. Выровненный по центру нейтральный фон вместе с эффектом тени придает объем странице

- Измените HTML-код. Конфигурируйте элемент `div`, назначьте ему идентификатор `container`, который содержит код в разделе тела страницы. На следующей строке после тега `<body>` укажите тег `<div>`. Закрывающий тег `</div>` введите на отдельной строке перед тегом `</body>`.
- Отредактируйте глобальную таблицу CSS, чтобы задать новый селектор идентификатора `container`. Задайте белый цвет фона и отступ величиной в 20 пикселов. Вспомните определения стилей, выравнивающие содержимое по центру, о которых говорилось в главе 3. Используйте свойства `width`, `margin-left` и `margin-right`, как показано ниже:

```
#container { background-color: #ffffff;
padding: 20px;
width: 800px;
margin-right: auto;
margin-left: auto; }
```

2. Отредактируйте глобальную таблицу CSS и добавьте к селектору `#container` следующие определения стилей, чтобы настроить тень блока:

```
-webkit-box-shadow: 5px 5px 5px #1e1e1e;
-moz-box-shadow: 5px 5px 5px #1e1e1e;
box-shadow: 5px 5px 5px #1e1e1e;
```

3. Добавьте к селектору элемента `h1` следующее определение стилей, чтобы конфигурировать тень текста темно-серого цвета:

```
text-shadow: 3px 3px 3px #666;
```

4. Добавьте к селектору элемента `h2` следующее определение стилей, чтобы конфигурировать тень текста светло-серого цвета без размытия:

```
text-shadow: 1px 1px 0 #ccc;
```

Сохраните файл. Когда вы откроете файл *shadow.html* в браузере, страница должна быть похожа на показанную на рис. 4.33, если ваш браузер поддерживает свойства `box-shadow` и `text-shadow`. В противном случае тени не отобразятся, но веб-страница все равно будет функциональна. Образец вы найдете на диске, прилагающемся к книге, в папке *Примеры\Глава_04\coffeehouse\shadow.html*.

С релизом каждой новой версии браузеров меняется и поддержка новых спецификаций. Тщательную проверку страниц ничто не заменит. Однако некоторые ресурсы предоставляют список поддерживаемых свойств. Эту информацию можно найти на следующих веб-сайтах:

- **caniuse.com/#agents=desktop&cats=CSS**
- **www.findmebyip.com/litmus**
- **www.quirksmode.org/css/contents.html**
- **www.impressivewebs.com/css3-click-chart**

Свойство `opacity`

Свойство **`opacity`** спецификации CSS3 задает степень непрозрачности фонового цвета. Диапазон значений непрозрачности от 0 (полностью прозрачный) до 1 (совершенно непрозрачный). Пример использования свойства `opacity` для конфигурирования белого фона со значением непрозрачности 60% приведен на рис. 4.34.

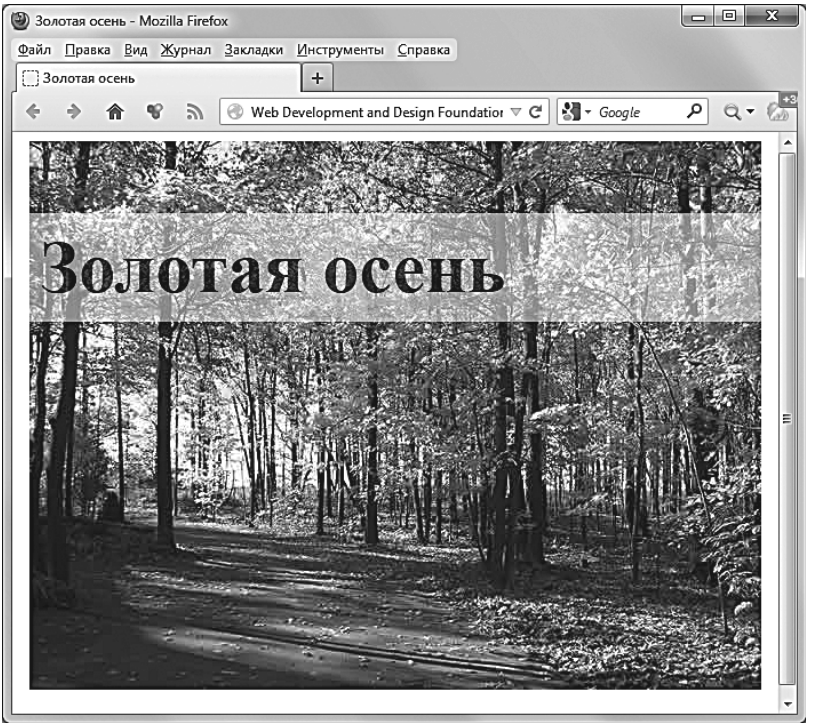


Рис. 4.34. Фон элемента `h1` со значением непрозрачности 60%

Практическое задание 4.11

В ходе этого практического задания вы будете работать со свойством `opacity`, реализуя веб-страницу, показанную на рис. 4.34.

1. Создайте новую папку с именем *opacity*. Скопируйте в нее файл *fall.jpg* из папки *Примеры\Глава_04\starters*. Запустите текстовый редактор и откройте файл *Примеры\Глава_02\template.html.* Сохраните его в папку *opacity* под именем *index.html.* Замените заголовок страницы на «Золотая осень».
2. Далее создадим структуру веб-страницы с элементом `div`, содержащим элемент `h1`. Добавьте в раздел тела веб-страницы следующий код:

```
<div id="content">
<h1>Золотая осень</h1>
</div>
```

3. Теперь добавьте в раздел заголовка элемент `style` и задайте глобальные стили CSS. Создайте идентификатор `content`, чтобы отобразить файл *fall.jpg* как неповторяющееся фоновое изо-

бражение. У идентификатора `content` также имеются значения: ширина — 640 пикселов, высота — 480 пикселов, левое и правое поля устанавливаются автоматически (они выровняют объект по центру в окне просмотра браузера) и верхний отступ величиной 20 пикселов. Код следующий:

```
#content { background-image: url(fall.jpg);
background-repeat: no-repeat;
margin-left: auto;
margin-right: auto;
width: 640px;
height: 480px;
padding-top: 20px;}
```

4. Теперь задайте селектору элемента `h1` белый фон со значением непрозрачности 0,6. Установите размер шрифта равный 4 em и величину отступа 10 пикселов. Пример кода:

```
h1 { background-color: #FFFFFF;
opacity: 0.6;
font-size: 4em;
padding: 10px; }
```

5. Сохраните файл. При тестировании страницы *index.html* в браузере, поддерживающем свойство `opacity`, она будет похожа на показанную на рис. 4.34. На диске, прилагающемся к книге, в папке *Примеры\Глава_04\opacity\index.html* вы найдете пример страницы. Если вы внимательно изучите веб-страницу, то увидите, что свойством `opacity` задана частичная прозрачность цвета фона и цвета текста элемента `h1`. Свойство непрозрачности наследуется и будет оказывать влияние на все элементы, содержащиеся в элементе `h1`. Поэкспериментируйте, меняя значение непрозрачности, и посмотрите, что получится.

   На рис. 4.35 показана веб-страница, отображенная в браузере Internet Explorer 8, который не поддерживает свойство `opacity`. Обратите внимание, что ее внешний вид отличается, но страница по-прежнему функциональна. Браузер Internet Explorer, начиная с версии 9, поддерживает свойство `opacity`, а более ранние версии задействуют собственное свойство — `filter`, задающее уровень непрозрачности в диапазоне от 1 (прозрачный) до 100 (непрозрачный). Пример доступен на диске, прилагающемся к книге (*Примеры\Глава_04\opacity\opacityie.html*). Правило CSS для свойства `filter` выглядит следующим образом:

```
filter: alpha(opacity=60);
```

Рис. 4.35. Браузер Internet Explorer 8 не поддерживает свойство `opacity`

Цветовая модель RGBA

Спецификация CSS3 обеспечивает для свойства цвета новый синтаксис ***цветовой модели RGBA***, поддерживающей прозрачность. Необходимы четыре значения: красного, зеленого и синего цветов, а также альфа (прозрачность). Модель RGBA не использует шестнадцатеричные цветовые значения. Вместо этого задаются десятичные цветовые значения. Примеры см. в файле *Примеры\Глава_04\RGBA.jpg* на диске, прилагающемся к книге. Значения красного, зеленого и синего цветов должны быть десятичными и находиться в диапазоне от 0 до 255. Значение альфа должно быть числом от 0 (прозрачный) до 1 (непрозрачный). На рис. 4.36 показана веб-страница с текстом, которому задан небольшой уровень прозрачности.

Практическое задание 4.12

В ходе этого практического задания вы создадите прозрачный текст, верстая код веб-страницы, показанной на рис. 4.36.

1. Запустите текстовый редактор и откройте файл, который вы создали в прошлом практическом задании (он также доступен на диске, прилагающемся к книге, в папке *Примеры\Глава_04\opacity\index.html*). Сохраните файл под именем *rgba.html*.

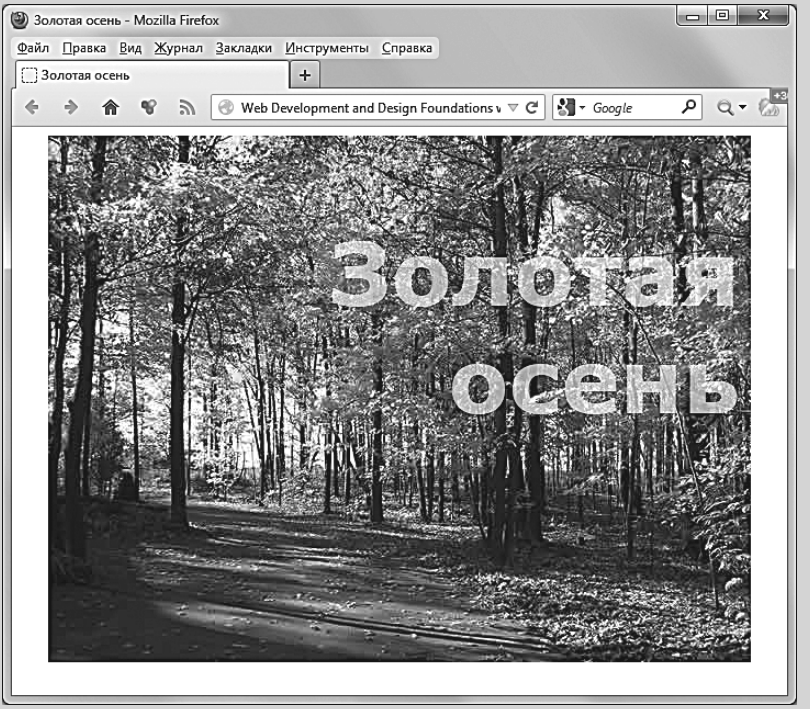


Рис. 4.36. Цветовая модель RGBA задает прозрачность тексту

2. Удалите текущие определения стилей селектора элемента `h1`. Вы создадите новые правила стилей селектора `h1`, чтобы конфигурировать отступ справа величиной 10 пикселов и выровненный по правому краю текст с уровнем непрозрачности 70%, набранный шрифтом без засечек размером 5 em. Поскольку не все браузеры поддерживают цвет RGBA, вы зададите свойство цвета дважды. Первым будет стандартное цветовое значение, поддерживаемое всеми современными браузерами. Второе будет задавать цвет RGBA. Более ранние версии браузеров не распознают цвета RGBA и игнорируют их. Актуальные версии браузеров распознают оба определения цветовых стилей и применят их в том порядке, в каком они указаны в коде, поэтому в результате эффект прозрачности будет реализован. Код CSS следующий:

```
h1 { color: #ffffff;
color: rgba(255, 255, 255, 0.7);
font-family: Verdana, Helvetica, sans-serif;
font-size: 5em;
padding-right: 10px;
text-align: right; }
```

3. Сохраните файл. При тестировании страницы *rgba.html* в браузере, поддерживающем цвета RGBA, она будет похожа на страницу, показанную на рис. 4.36. На диске, прилагающемся к книге, вы найдете пример (*Примеры\Глава_04\opacity\rgba.html*). Если вы используете браузер, не поддерживающий это свойство, к примеру, Internet Explorer 8 (или более раннюю версию), вы увидите белый текст вместо прозрачного. Версия 9 программы Internet Explorer поддерживает цветовую модель RGBA, а ранние версии реализуют собственное свойство `filter`, пример использования которого вы увидите, открыв файл *Примеры\Глава_04\opacity\rbgaie.html*.

ЧаВо

ДЛЯ ЧЕГО НУЖНА ЦВЕТОВАЯ МОДЕЛЬ HSLA?

Существует еще один новый метод конфигурирования цвета с помощью CSS3, называемый ***моделью HSLA***. Аббревиатура HSLA означает оттенок (hue), насыщенность (saturation), яркость (lightness) и альфа (alpha). Это способ представить цвета иначе, чем в системе RGB, обычно используемой веб-дизайнерами. Модель HSLA пока поддерживается не всеми браузерами. Более подробную информацию можно найти на следующих ресурсах:

- **www.w3.org/TR/2003/CR-css3-color-20030514/#hsla-color**
- **www.useragentman.com/blog/2010/08/28/coding-colors-easily-using-css3-hsl-notation**
- **css-tricks.com/yay-for-hsla**

Градиенты

Спецификация CSS3 обеспечивает возможность конфигурировать цвет как градиент, т.е. как плавный переход оттенков одного цвета в другой. Градиент в CSS3 в качестве фонового цвета задается исключительно с помощью таблицы стилей, графический файл не нужен. Это дает веб-дизайнерам большую гибкость и позволяет повысить пропускную способность за счет времени, которое требовалось бы на загрузку файла с изображением градиента для фона. Звучит потрясающе? Это так, но есть подвох: движки для визуализации страниц в браузерах, WebKit и Gecko, используют для обработки CSS-градиентов собственный синтаксис. Консорциум W3C добавил поддержку градиентов в модуль Image Value and Replaced Content (черновой вариант), но на момент напи-

сания книги браузеры еще не задействовали этот синтаксис. В данном разделе представлен пример градиента CSS3, а также ссылки на ресурсы для дальнейшего изучения.

На рис. 4.33 показана веб-страница с фоновым градиентом, представляющим собой файл в формате JPG, созданный в графическом редакторе. Веб-страница, показанная на рис. 4.37 (см. файл *Примеры\Глава_04\coffeehouse\gradient.html*), не использует файл JPG в качестве фонового изображения — правила стилей CSS3 воссоздали линейный градиент.

Градиенты и прогрессивное улучшение

Очень важно, реализуя градиенты с помощью правил CSS3, не забывать о прогрессивном улучшении. Задайте запасное свойство `background-color` или `background-image`, которые будут визуализироваться браузерами, не поддерживающими градиенты CSS3.

На рис. 4.37 цвет фона задан такой же, как и конечный цвет градиента.

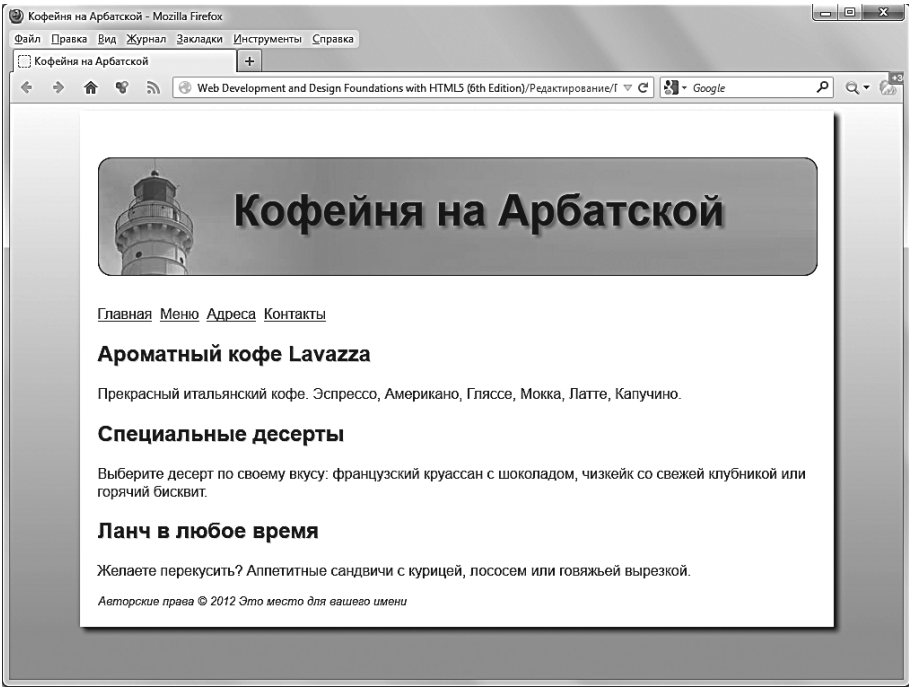


Рис. 4.37. Фоновый градиент был задан с помощью CSS3 без использования графического файла

Настройка градиентов с помощью CSS3

Чтобы настроить фоновый градиент, укажите в коде четыре определения стилей:

- `-webkit-gradient` (для браузеров на движке Webkit);
- `-moz-linear-gradient` (для браузеров на движке Gecko);
- `filter` (для браузера Internet Explorer);
- `linear-gradient` (синтаксис черновика W3C).

Задайте градиент как значение свойства `background-image` (кроме браузера Internet Explorer, где используется свойство `filter`). Следующий код CSS сначала задает цвет фона (для браузеров, не поддерживающих градиент), а затем — линейный градиент, переходящий из белого цвета (#FFFFFF) в синий (#8FA5CE):

```
background-color: #8FA5CE;
background-image: -webkit-gradient(linear, left
top, left bottom, from(#FFFFFF), to(#8FA5CE));
background-image: -moz-linear-gradient(top,
#FFFFFF, #8FA5CE);
filter: progid:DXImageTransform.Microsoft.gradient
(startColorstr=#FFFFFFFF, endColorstr=#FF8FA5CE);
background-image: linear-gradient(#FFFFFF,
#8FA5CE);
```

Синтаксис для браузеров на движке WebKit

Изучите синтаксис кода для браузеров на движке WebKit, используемого браузерами Safari и Google Chrome. В самой простой форме в значении свойства `-webkit-gradient` перечисляются тип градиента (линейный или радиальный), начальная и конечная точки, начальный и конечный цвета.

Синтаксис для браузеров на движке Gecko

Изучите синтаксис кода для браузеров на движке Gecko, который используется такими браузерами, как Firefox и Flock. В значении свойства `-moz-linear-gradient` перечисляются начальная точка градиента и точки цвета (в данном случае оттенка два: #FFFFFF и #8FA5CE).

Синтаксис для браузера Internet Explore

Браузер Internet Explorer 9 и более ранние его версии вместо свойства `background-image` применяют свойство `filter`. Значения цвета — восьмизначные. Первые два числа — это значение альфа (#00 — прозрачный; #FF — непрозрачный). Следующие шесть чисел обозначают шестнадцатеричное значение цвета.

Синтаксис Консорциума W3C

В синтаксисе W3C используются различные функции для линейного (`linear-gradient`) и радиального (`radial-gradient`) градиентов. В основном формате для двухцветного линейного градиента указываются значения обоих цветов (в этом примере #FFFFFF и #8FA5CE). Как показано в примерах кода, синтаксис для конфигурирования градиента меняется в зависимости от движка браузера. Ожидается, что со временем все браузеры будут поддерживать синтаксис W3C, поэтому указывайте это определение в коде последним. Помните, что собственный синтаксис свойств браузеров в этом разделе нестандартный. Ваш код CSS не пройдет проверку валидатором W3C, если вы используете эти свойства.

Потренируйтесь в верстке CSS-кода градиентов на сайтах **www.westciv.com/tools/gradients**, **www.colorzilla.com/gradient-editor** и **gradients.glrzad.com.** Для получения дополнительной информации о синтаксисе CSS для реализации градиентов посетите соответствующие вебсайты:

- Webkit: **webkit.org/blog/175/introducing-css-gradients**
- Mozilla: **developer.mozilla.org/en/CSS/-moz-linear-gradient**
- Internet Explorer: **msdn.microsoft.com/en-us/library/ms532997(VS.85,loband).aspx**
- W3C: **dev.w3.org/csswg/css3-images/#gradients**

Глава 5
ВЕБ-ДИЗАЙН

Цели главы

В этой главе вы узнаете следующее:

- какие типы структур наиболее часто используются для создания веб-сайтов;
- принципы визуального дизайна;
- как создавать дизайн для своей целевой аудитории;
- как создавать понятные, простые элементы навигации;
- как улучшить читаемость текста ваших веб-страниц;
- как правильно использовать графику на веб-страницах;
- как применить к веб-страницам концепцию универсального дизайна;
- какие технологии применяются для создания макетов веб-страниц;
- как применить ваши практические знания при проектировании сайтов.

Как посетитель веб-сайтов вы, вероятно, знаете, что некоторые сайты имеют привлекательный вид и ими легко пользоваться, другие же неудобны и раздражают своей примитивностью. Чем же отличаются хорошие сайты от плохих? В этой главе даются практические рекомендации по дизайну веб-сайтов. Темы главы включают организацию сайта, навигацию по сайту, дизайн страниц сайта, текста и графики.

5.1. Создание дизайна для целевой аудитории

Независимо от ваших личных предпочтений, создаваемый вами сайт должен соответствовать запросам ***целевой аудитории*** — людей, которые будут его посещать. Аудитория, для которой предназначается сайт, может быть конкретной, к примеру, дети, студенты, молодые пары, пожилые люди или же сайт может быть ориентирован на любых пользователей. Намерения и цели ваших посетителей различны: они могут случайно искать информацию, проводить исследование для учебы или работы, сравнивать магазины, заниматься поиском работы и т. д. Дизайн веб-сайта должен привлекать целевую аудиторию и удовлетворять ее потребности.

Рис. 5.1. Захватывающая графика притягивает

Например, сайт NASA (Национального агентства по аэронавтике и исследованию космического пространства) **www.nasa.gov** (рис. 5.1), насыщенный графикой, выглядит совсем иначе, чем преимущественно текстовый сайт ЦРУ — Центрального Разведывательного Управления, **www.cia.gov/ru/index.html** (рис. 5.2).

Рис. 5.2. Преимущественно текстовая веб-страница

Первый сайт сразу притягивает внимание и захватывает вас, приглашает изучить его подробнее. Второй, с полезной информацией, вынесенной на первый план, позволяет быстро включиться в работу. Имея в виду свою целевую аудиторию, рассмотрим некоторые наиболее общие рекомендации по дизайну веб-сайтов.

5.2. Структура веб-сайта

Каким образом посетители будут перемещаться по сайту? Как они найдут то, что им нужно? Это в большой степени определяется структурой сайта или, иначе говоря, его архитектурой. Существуют три распространенных типа структуры сайта:

- иерархическая;
- линейная;
- хаотичная (также называемая «решеткой» или «паутиной»).

Схематически изображенная структура сайта называется ***картой веб-сайта***. Создание карты сайта является первым шагом к его разработке (более подробно об этом см. в главе 10).

Иерархическая структура

Большинство веб-сайтов имеет ***иерархическую структуру***. Карта такого сайта, показанная на рис. 5.3, имеет четко выраженную главную страницу, связанную со всеми основными разделами сайта.

Рис. 5.3. Иерархическая структура сайта

Внутренние страницы сайта добавляются по мере необходимости. На панели навигации главной страницы и страниц первого уровня в иерархической структуре сайта обычно указываются гиперссылки на все остальные страницы.

Важно знать о возможных ошибках в построении иерархической структуры: например, структура, показанная на рис. 5.4, чересчур плоская — в ней слишком много главных разделов.

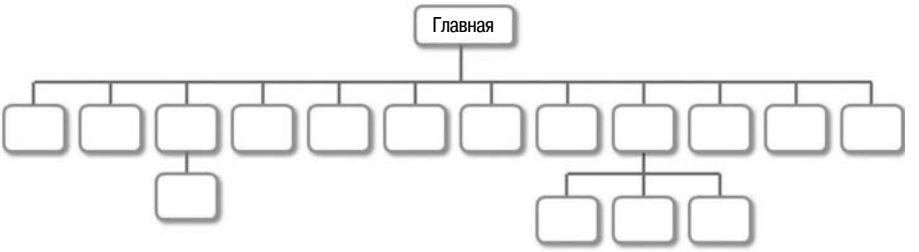


Рис. 5.4. Структура этого сайта чересчур плоская — в ней слишком много главных разделов

В структуре такого сайта целесообразно выделить более узкие и легче управляемые темы или информационные группы. Этот процесс называется ***разделением***. В случае веб-сайта каждая группа представляет собой отдельную страницу. Нельсон Кован, психолог-исследователь из Университета Миссури, установил, что человек способен запомнить в кратковременной памяти только около 4 единиц или групп единиц информации, к примеру, три группы цифр, составляющие телефонный номер[1]. Руководствуясь этим принципом, следите за количеством основных ссылок навигации и старайтесь визуально разделить их на несколько разделов таким образом, чтобы в каждом разделе было не более 4 ссылок.

Другой ошибкой было бы создавать слишком глубокую структуру, подобную показанной на рис. 5.5. Дизайнерское «правило трех щелчков» гласит, что пользователь должен иметь возможность попасть с любой страницы сайта на любую другую его страницу, задействовав не более трех гиперссылок. Иначе говоря, если посетитель не сможет за три щелчка найти то, что ему нужно, он будет разочарован и может покинуть сайт. Конечно, на больших сайтах это правило очень трудно выполнить, но, по большому счету, целью дизайнера является такая организация сайта, чтобы посетитель мог легко перемещаться внутри него.

[1] web.missouri.edu/~cowann/ research.html

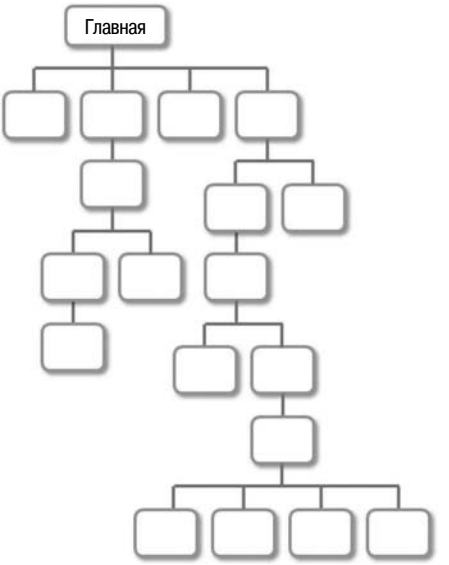


Рис. 5.5. Структура этого сайта слишком глубокая

Линейная структура

Линейная структура (рис. 5.6) полезна для организации сайтов или ряда веб-страниц, предоставляющих услуги по обучению, туры или презентации, которые должны рассматриваться последовательно.

Рис. 5.6. Линейная организация сайта

При линейной структуре сайта страницы отрываются последовательно одна за другой. Некоторые сайты, использующие в основном иерархическую структуру, применяют линейную организацию в отдельных небольших областях.

Хаотичная структура

При ***хаотичной структуре*** (также называемой «решеткой» или «паутиной») не создается четкого пути для перемещения по сайту, что показано на рис. 5.7. В этом случае часто не бывает выраженной глав-

ной страницы и заметной структуры. Хаотичная структура не так распространена, как иерархическая или линейная, и обычно встречается на художественных сайтах и сайтах, при создании которых стояла цель сделать их оригинальными и отличными от других. Этот тип структуры обычно не используется на коммерческих сайтах.

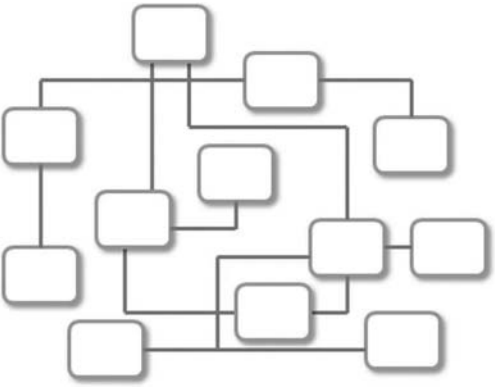


Рис. 5.7. Хаотичная структура сайта

ЧаВо

КАК ЛУЧШЕ ОРГАНИЗОВАТЬ КАРТУ САЙТА?

Иногда бывает трудно создать карту для веб-сайта. Некоторые дизайнеры собираются в комнате с пустыми стенами и упаковками стикеров. Они пишут названия тем и подтем на стикерах. Затем они прикрепляют эти записки на стены и обсуждают их до тех пор, пока структура сайта не станет понятной, и вся группа не придет к согласию. Если вы работаете не в группе, можно попытаться проделать это в одиночку, а затем обсудить выбранную вами структуру с другом или одногруппником.

5.3. Принципы визуального дизайна

Существуют четыре принципа, которые следует применять для дизайна почти всей графики: ***повторяемость***, ***контраст***, ***приближенность*** и ***выравнивание***. Независимо от того, что вы проектируете — веб-страницу, кнопку, логотип, обложку DVD-диска, брошюру или интерфейс программы, — использование перечисленных принципов помогает создать хороший внешний вид и впечатление вашего проекта и определить, дойдут ли ваши идеи по назначению.

Повторяемость: повторение элементов дизайна на всем сайте

Применяя принцип ***повторяемости*** дизайнер повторяет один или несколько графических элементов в разных местах сайта. Повторение элементов объединяет сайт. Все повторяемые элементы, будь это цвет, форма, шрифт или изображение, помогают объединить дизайн.

Контраст: добавление эмоциональности и привлечение внимания

Для реализации принципа ***контрастности*** дизайнер должен использовать существенно различающиеся элементы (добавить контраста), чтобы сделать дизайн более интересным и привлечь внимание к сайту. При проектировании веб-сайтов должен быть обеспечен контраст между цветом фона и текста.

Приближенность: группирование

Применение принципа ***приближенности*** означает, что связанные элементы располагаются рядом друг с другом. Между элементами, не связанными общей темой, оставляется свободное пространство. Размещение элементов интерфейса близко друг к другу позволяет понять логику размещения на сайте информации или функциональных кнопок.

Выравнивание: выравнивание элементов для создания единства стиля

Следующий принцип дизайна, позволяющий создавать привлекательные сайты, — это ***выравнивание***. Используя этот принцип, дизайнер организует страницу таким образом, что каждый элемент оказывается выровненным (по вертикали или по горизонтали) относительно других элементов страницы.

Повторяемость, контраст, приближенность и выравнивание — это четыре принципа, которые помогут вам значительно улучшить графический дизайн веб-страницы. Эффективное использование этих принципов сделает вашу страницу профессиональной, и она будет выглядеть более понятной посетителям сайта. Помните об этих принципах при разработке дизайна и проектировании веб-сайтов.

5.4. Методы обеспечения доступности

В главе 1 вы познакомились с концепцией ***универсального дизайна***. Центр универсального дизайна определяет универсальный дизайн как «дизайн товаров и сред, позволяющий всем людям максимально использовать их без адаптации или специального дизайна».

Кто выигрывает от повышения доступности?

Рассмотрим следующие ситуации:

- Мария, молодая женщина-инвалид двадцати с небольшим лет — она не может работать мышью, а с клавиатурой справляется с трудом. Доступные веб-страницы, спроектированные так, что они функционируют без использования мыши, позволяют Марии получить доступ к контенту.
- Леонид, студент колледжа. Он глухой и хочет быть веб-дизайнером. Субтитры к аудио/видео контенту и транскрипции позволяют Леониду получить доступ к контенту.
- Денис, мужчина средних лет, имеет коммутируемое подключение к Интернету, который он использует для работы. Замена изображений текстом, а мультимедийных файлов — транскрипциями позволяют ему получить более быстрый доступ к контенту.
- Надежда, женщина в возрасте, с возрастной макулярной дегенерацией, испытывающая затруднения при чтении мелкого текста. Ей будет легче читать веб-страницы, разработанные так, чтобы их текст можно было увеличить в браузере.
- Карина, студентка колледжа, подключается к Интернету через смартфон. Доступный контент, организованный в виде заголовков и списков, облегчит Карине поиск информации в Сети с помощью мобильного устройства.
- Павел, практически слепой тридцатилетний мужчина, которому нужен Интернет для работы. Прочитать контент ему помогут веб-страницы, разработанные, чтобы быть доступными для всех (к примеру, организованные в виде заголовков и списков, отображающие текстовое описание для гиперссылок, предоставляющие замещающий текст вместо изображений и позволяющие работать без мыши).

Все эти люди выиграют, если веб-страницы будут создаваться с использованием принципов доступности. Такие страницы обычно более

удобны в использовании для всех — даже люди без каких-либо физических недостатков с широкополосным доступом в Интернет выиграют от высокого качества представления материала и правильно выбранной структуры хорошо спроектированного сайта (рис. 5.8).

Рис. 5.8. Если веб-страница доступна, выигрывают все

Доступный дизайн может помочь при индексации сайта поисковыми системами

Программы поисковых систем (обычно называемые ботами или поисковыми пауками) путешествуют по Всемирной паутине и переходят по ссылкам на веб-сайты. Роботам поисковых систем проще обнаружить доступный веб-сайт с описательными названиями страниц, хорошо организованными заголовками, списками, описательными гиперссылками и альтернативными текстовыми пояснениями к изображениям — и в результате сайт может получить более высокий рейтинг.

Доступность — это правильно

Интернет и веб-технологии глубоко проникли в нашу культуру, и в России их общедоступность уже охраняется законом. ГОСТ Р 52872-2007 требует, чтобы информация веб-сайтов, включая веб-страницы го-

сударственных организаций, была доступна людям с физическими недостатками.

Рекомендации по методам обеспечения доступности, упоминаемые в этой книге, направлены на приведение создаваемых веб-сайтов в соответствие с требованиями ГОСТа Р 52872-2007 и Руководством по обеспечению доступности веб-контента (WCAG) 2.0, рекомендованным подразделением WAI — Web Accessibility Initiative (Инициативная группа по веб-доступности) консорциума W3C Web. Следующие четыре принципа должны соблюдаться для того, чтобы сайт соответствовал стандарту WCAG 2.0 — воспринимаемость, управляемость, понятность и надежность.

1. Содержимое сайта должно быть **воспринимаемым**. Воспринимаемое содержимое легко увидеть или услышать. Любой графический или мультимедийный контент должен быть доступен в текстовом формате, к примеру, в виде текстовых описаний к изображениям, субтитрам к видео и транскрипциям аудиофайлов.
2. Компоненты интерфейса содержимого должны быть **управляемыми**. Управляемое содержимое включает в себя формы навигации или другие интерактивные свойства, которые можно использовать или которыми можно управлять с помощью мыши или клавиатуры. При создании мультимедийного контента следует избегать частого мигания изображения, так как вспышки могут вызвать у некоторых людей эпилептический припадок.
3. Контент и управление должны быть **понятны**. Понятное содержимое веб-страниц легкочитаемо, огранизовано по единому образцу и при необходимости отображает полезные сообщения об ошибках.
4. Содержимое сайта должно быть достаточно **надежным**, чтобы работать с пользовательскими устройствами, существующими в настоящее время, и с теми, которые появятся в будущем, включая вспомогательные технологии. Надежный контент создается в соответствии с рекомендациями консорциума W3C и должен быть совместим с различными операционными системами, браузерами и вспомогательными технологиями, такими как приложения экранного доступа.

С подробным описанием положений WCAG 2.0 можно ознакомиться на сайте **accessibility.ru/docs/WCAG20Russian031109.html**. Инструкции, включенные в WCAG 2.0, разделены по степени соответствия стандартам на 3 уровня: уровень A, уровень AA и уровень AAA.

Разработка доступных веб-сайтов — это важный момент работы веб-дизайнера. Компания WebAim предлагает перечень контрольных вопросов, касающихся WCAG2.0, с полезными советами на сайте **webaim.org/standards/wcag/checklist**. Университет Торонто предоставляет на своем сайте бесплатный валидатор доступности **(achecker.ca/checker/index.php)**.

И наконец, ГОСТ Р 52872-2007 требует, чтобы в случае, если сайт не может соответствовать требованиям по доступности, создавалась и регулярно обновлялась отдельная текстовая версия сайта.

Работая с этой книгой и создавая страницы в ходе практических заданий, вы научитесь добавлять в них элементы доступности. Из глав 2–4 вы уяснили, насколько важны теги заголовка страницы, теги раздела заголовка, текстовые описания для гиперссылок и замещающий текст для изображений. Вы уже на пути к созданию доступных веб-страниц!

5.5. Написание текстов для Всемирной паутины

Пространные предложения и описания привычны для учебников и любовных романов, но совершенно неуместны на веб-странице. Большие блоки текста или длинные абзацы трудно читать в Интернете. Следующие советы помогут вам сделать веб-страницы легкочитаемыми.

Структурируйте контент на странице

Эксперт по веб-юзабилити Якоб Нильсен утверждает, что люди на самом деле не читают веб-страницы полностью, а только просматривают их. Структурируйте текст на веб-страницах так, чтобы его можно было быстро просмотреть. Будьте кратки. С помощью заголовков, подзаголовков, небольших абзацев и неупорядоченных списков структурируйте контент веб-страницы так, чтобы он был легкочитаем и посетители могли быстро найти нужную информацию.

Также обращайте внимание на длину строк. По возможности используйте отступы и колонки. Изучите примеры размещения текста на веб-странице на рис. 5.15. Заголовки и области навигации можно выравнивать по центру, однако старайтесь не центрировать весь текст — абзац центрированного текста сложнее прочитать, чем текст, выровненный по левому краю.

Текст в гиперссылках

Помещайте гиперссылки на ключевых словах или описательных выражениях — не создавайте гиперссылки длиной в целое предложение. Избегайте использовать выражения типа «щелкните здесь» — в наше время пользователи сами знают, что делать. Также помните, что все чаще люди используют сенсорные дисплеи, где им нужно провести пальцем или легонько стукнуть, а не щелкнуть.

Навыки чтения

Стиль изложения информации на сайте должен соответствовать навыкам чтения целевой аудитории. Используйте лексику, к которой они привыкли. Компания Juicy Studio предлагает бесплатный онлайн-тест на читабельность на сайте **juicystudio.com/services/readability.php**.

Используйте общепринятые шрифты

Используйте общепринятые шрифты, например, Arial, Verdana или Times New Roman. Помните, что для того чтобы ваши шрифты отображались на компьютере пользователя, они должны быть установлены на нем. Ваша страница сможет выглядеть великолепно со шрифтом Gill Sans, сжатым, очень жирным, но если у посетителя вашего сайта такого шрифта нет, браузер использует шрифты, установленные по умолчанию. Ознакомьтесь со списком «безопасных» шрифтов Всемирной паутины, приведенным на сайте **www.ampsoft.net/webdesign-l/WindowsMacFonts.html.**

Размер и толщина шрифтов

Учитывайте, что на компьютерах Mac шрифты имеют меньший размер, чем на компьютерах с операционной системой Windows. Даже при использовании одной платформы на разных компьютерах Windows размер шрифта может быть разным при отображении страниц в разных браузерах. Предусмотрите создание прототипа веб-страницы с настройками ваших шрифтов и их тестирование в разных браузерах и при разных разрешениях экрана. Выделяйте важную информацию **полужирным шрифтом** или *курсивом*. Однако будьте осторожны и не выделяйте полужирным шрифтом все подряд. С таким же успехом можно вообще ничего не выделять.

Контрастные цвета шрифтов

Используйте подходящие цветовые схемы. Начинающие веб-дизайнеры часто выбирают цвета, которые они никогда не хотели бы видеть в своем гардеробе. Простой способ выбора цветов, которые были бы достаточно контрастными и при этом хорошо сочетались, — это позаимствовать их из изображения или логотипа, которые вы предполагаете использовать на сайте.

Орфография и грамматика

Множество сайтов сегодня содержат грамматические ошибки. У большинства приложений для верстки веб-страниц, таких как Adobe Dreamweaver, есть встроенные программы для проверки орфографии, используйте их возможности. Также убедитесь, что вы досконально вычитали и протестировали ваш сайт. Очень хорошо, если вы сможете найти другого веб-разработчика — вы будете проверять его сайты, а он — ваши. Всегда легче увидеть чужие ошибки, чем свои.

5.6. Использование цвета

Подбор цветов

Вы, возможно, задавались вопросом, как выбрать цвета, которые будут отображаться на веб-странице. Самый простой способ — применить монохроматическую цветовую схему (все оттенки и тона одного цвета в сочетании с нейтральными белым, черным и/или серым цветами). Для подбора цветов монохроматической цветовой схемы попробуйте применить программу Color Blender на сайте **meyerweb.com/eric/tools/colorblend.** Другой способ — создать цветовую схему на основе фотографии или изображения. Если у вас есть любимый цвет и вы хотите создать цветовую схему, опираясь на него, посетите один из указанных сайтов, предлагающих варианты цветовых схем:

- Colors on the Web: **colorsontheweb.com/colorwizard.asp**
- Kuler: **kuler.adobe.com**
- Lee Street Management: **www.leestreet.com/QuickColor.swf**
- Color Scheme Designer: **colorschemedesigner.com**
- ColorJack: **www.colorjack.com/articles/color_formulas.html**

Цвет и доступность

Цвета способны помочь вам создать привлекательную веб-страницу, однако помните, что не все посетители смогут увидеть или различить цвета. Некоторые посетители будут пользоваться программой экранного доступа и не заметят ваши цвета, поэтому информация должна точно передаваться даже если цвета не видны.

Ваш выбор цвета имеет решающее значение. Например, красный текст на синем фоне обычно трудно прочитать. Также старайтесь не использовать сочетания красного и зеленого или коричневого и фиолетового цветов потому, что пользователям, страдающим цветовой слепотой, возможно, будет нелегко их различить. По данным компании Vischeck[1], каждый двадцатый человек в мире страдает от той или иной формы нарушений восприятия цвета. Чтобы узнать, как цвета на вашей веб-странице воспринимаются пользователями, страдающими цветовой слепотой, посетите сайт **www.vischeck.com/vischeck/vischeckURL.php**. Большинству людей проще различать белый, черный цвета и оттенки голубого и желтого.

Выбирайте контрастные цвета фона и текста. В положениях WCAG2.0 для стандартного текста рекомендуется контрастность 4,5:1. Если текст набран шрифтом крупного размера, контрастность должна быть ниже — 3:1. Онлайн-проверка цветового контраста Джонатана Снука[2] поможет вам определить уровень контрастности цветов текста и фона.

Более подробную информацию об эффективном использовании цветов вы найдете на сайте компании Lighthouse International[3].

При выборе цвета важно учитывать предпочтения целевой аудитории. Этому аспекту веб-дизайна посвящен следующий раздел.

Цвета и целевая аудитория

Выбирайте цвета, привлекающие целевую аудиторию. Наиболее молодая аудитория, дети и подростки, любит яркие живые цвета, анимацию и эффекты интерактивности.

Подростки более старшего возраста и молодежь в возрасте около 20 лет предпочитает темные цвета фона с включениями яркого цве-

[1] www.vischeck.com/vischeck
[2] snook.ca/technical/colour_contrast/colour.html
[3] lighthouse.org/accessibility/design/accessible-print-design/effective-color-contrast

та, музыкой и динамичной навигацией. Для более старшей целевой аудитории подойдут светлый фон, четкие изображения и крупный текст.

Если вы хотите обратиться ко всем пользователям Интернета, а не к ограниченной аудитории, возьмите в качестве образца использования цвета популярные сайты Amazon.com и eBay.com. На этих сайтах используется нейтральный белый цвет с включением ярких цветовых пятен, добавляющих сайту притягательности и обозначающих основные области страницы. Об использовании белого цвета для фона веб-страниц говорится и в книге Якоба Нильсена и Мари Тахир «Дизайн Web-страниц. Анализ удобства и простоты использования 50 узлов»[1], где анализируется 50 лучших сайтов. В книге приводятся данные, что 84% из рассмотренных сайтов используют для фона белый цвет, и 72% этих сайтов используют черный цвет для текста. Такой выбор цветов максимизирует контраст между фоном и текстом, обеспечивая наилучшую читаемость.

5.7. Размещение графических и мультимедийных элементов

Как показано на рис. 5.1, захватывающее изображение может стать притягательным элементом веб-страницы. Однако не полагайтесь только на рисунки. Некоторые люди не смогут увидеть их, возможно, их браузер настроен таким образом, что он не отображает графику или они используют специальные программы экранного доступа. Хорошо также добавить текст, описывающий наиболее важные идеи или ключевые моменты, которые вы хотите донести до посетителей сайта. В этом разделе вы изучите рекомендуемые техники использования на веб-страницах графики и мультимедийных элементов.

Размер файла и изображения имеют значение

Старайтесь, чтобы разрешение изображения и размер его файла были как можно меньше. Старайтесь показывать только то, что необходимо для понимания ваших идей. Кадрируйте изображения и создавайте миниатюры, ссылающиеся на изображения большего размера.

[1] *Нильсен Я., Тахир М.* Дизайн Web-страниц. Анализ удобства и простоты использования 50 узлов. М. : Вильямс, 2002.

Сглаживание изображений текста

При **сглаживании** вблизи краев добавляются промежуточные цвета, сглаживающие зубцы, образующиеся на цифровых изображениях. Для создания изображений сглаженного текста можно использовать графические редакторы Adobe Photoshop и Adobe Fireworks. Изображение на рис. 5.9 было создано с применением сглаживания. На рис. 5.10 показано изображение, созданное без сглаживания: обратите внимание на зубцы по краям.

Рис. 5.9. Сглаженный текст

Рис. 5.10. Этот рисунок не был сглажен. У буквы «A» видны зубчатые края

Используйте мультимедийные файлы только при необходимости

Анимацию и мультимедийные объекты стоит использовать, только если это улучшает качество сайта. Не добавляйте анимированные файлы в формате Flash (см. главу 11) или GIF просто потому, что они у вас

есть. Ограничьте использование анимированных элементов. Вставляйте анимацию, только если она повысит эффективность восприятия страницы. Подумайте об ограничении времени воспроизведения анимации.

В целом для молодежной аудитории анимация более привлекательна, чем для пользователей старшего возраста. Однако качественно выполненная навигационная анимация или анимация, описывающая товар или услугу, могут показаться привлекательными почти любой целевой группе. Чтобы сделать веб-страницы визуально более интересными и интерактивными, во Всемирной паутине часто используют объекты в формате Adobe Flash. Вы будете создавать Flash-анимацию для веб-страницы в главе 11.

Предоставьте замещающий текст

Как уже говорилось в главе 4, каждое изображение на веб-странице должно сопровождаться замещающим текстом. Этот текст может отображаться на мобильных устройствах; появляться на короткий промежуток времени, когда изображение медленно загружается или когда в браузере отключена функция показа изображений. Также замещающий текст произносится вслух, если человек использует программу экранного доступа. На рис. 5.11 для отображения замещающего текста вместо изображения используется дополнение Web Developer в браузере Firefox.

Рис. 5.11. Дополнение Web Developer для браузера Firefox отображает замещающий текст над изображением

Для удовлетворения требований доступности предоставьте также замещающий текст для мультимедийных, видео- и аудиообъектов. Текстовая транскрипция аудиозаписи будет полезна не только людям с нарушениями слуха, но и тем, кто предпочитает читать новую информацию. Кроме того, транскрипцию текста могут обработать поисковые роботы при индексации и категоризации сайта. Субтитры и титры помогают обеспечивать доступность видеосайтов.

5.8. Дизайн навигационных элементов

Иногда разработчики веб-сайтов настолько захвачены своей работой, что не заботятся об удобной навигации, и в результате новый посетитель, бродящий по сайту, не знает, где щелкнуть и как найти нужную информацию. Поэтому очень полезно четко обозначать кнопки навигации. Для максимального удобства на всех страницах они должны находиться на одном и том же месте.

Навигационные панели

Понятная **навигационная панель**, неважно, текстовая или графическая, дает посетителям знать, где они находятся и куда они могут отсюда переместиться. Обычно панель навигации по сайту располагается горизонтально под логотипом или вертикально с левой стороны страницы. Вертикальная панель навигации с правой стороны страницы встречается реже, потому что эта область может оказаться обрезанной при низком разрешении экрана.

Навигация из «хлебных крошек»

Якоб Нильсен, известный специалист по использованию и созданию веб-сайтов советует использовать для больших сайтов то, что он называет «***следом из хлебных крошек***»[1], который указывает последовательность веб-страниц, посещенных пользователем в ходе данной сессии. Обычно дизайнеры располагают «след из хлебных крошек» в верхней части главной области контента, и он указывает, какие страницы посетитель просмотрел за время посещения.

Использование графики для навигации

Иногда для навигации вместо текста используется графика. Когда вместо текстовых гиперссылок навигацию обеспечивают графические ссылки, доступность достигается с помощью двух техник:

- каждый элемент страницы конфигурируется с альтернативным текстовым описанием (см. главу 4);
- страница конфигурируется с текстовыми гиперссылками в разделе нижнего колонтитула.

[1] Также называемый ***навигационной цепочкой***, см. ***ru.wikipedia.org/wiki/Навигационная_цепочка*** — *Прим. пер.*

Пропуск повторяющейся навигации

Предоставьте способ избежать повторения ссылок навигации. Пользователи, не страдающие нарушениями зрения или слуха, могут легко проанализировать веб-страницу и быстро сосредоточиться на ее содержимом. Однако длинные, повторяющиеся панели навигации быстро утомляют, если для работы со страницей используется программа экранного доступа или клавиатура.

Подумайте о том, чтобы добавить ссылку «Пропустить навигацию» или «Перейти к содержимому» перед основной панелью навигации, по которой пользователь переходил бы к определенному фрагменту (см. главу 7) в начале раздела содержимого веб-страницы.

Динамическая навигация

При посещении веб-сайтов вы, возможно, сталкивались с меню навигации, отображающими дополнительные варианты при наведении указателя мыши на определенный элемент страницы. Это динамическая навигация, предоставляющая способ предложить посетителям множество вариантов выбора, при этом не ошеломляя их количеством. Вместо того чтобы одновременно показывать все ссылки навигации, элементы меню отображаются динамически (обычно с помощью сочетания HTML и CSS) по мере надобности.

Дополнительные элементы становятся доступны, когда указывается или выбирается связанный с ними элемент основного меню. Посетите сайт компании Dynamic Drive[1], чтобы найти образец кода, который можно использовать для создания динамической навигации на собственных веб-страницах.

Карта сайта

Даже при наличии четкой и последовательной навигации пользователи иногда могут заблудиться на крупных веб-сайтах. Карта сайта предоставляет план структуры веб-сайта с гиперссылками на все основные страницы. Она может помочь пользователям найти другой путь к нужной им информации, как показано на примере веб-сайта оператора сотовой связи Tele2 на рис. 5.12.

[1] www.dynamicdrive.com/dynamicindex1

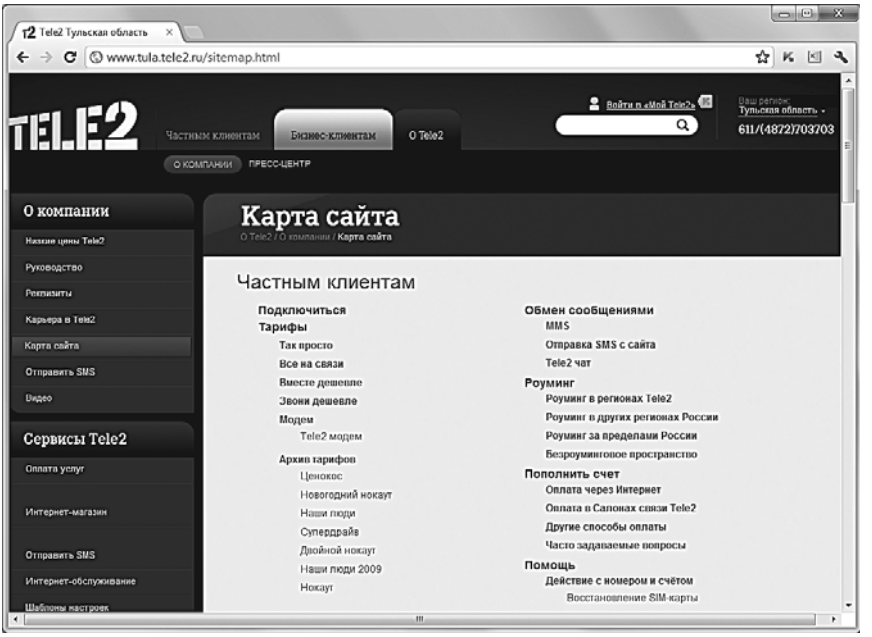


Рис. 5.12. Большой сайт предоставляет посетителям систему поиска и карту сайта

Система поиска по сайту

Обратите внимание на систему поиска в правом верхнем углу веб-страницы на рис. 5.12. Эта система поиска дает возможность найти на сайте информацию, которая не сразу может быть найдена с помощью карты сайта.

5.9. Дизайн макетов веб-страниц

Блок-схемы и макеты веб-страниц

Блок-схема — это набросок или проект веб-страницы, показывающий структуру основных элементов, таких как логотип, область навигации, основное содержимое и колонтитул (без детального дизайна). Блок-схемы используются в процессе дизайна при экспериментировании с различными макетами веб-страниц, разработке структуры и навигации сайта и обеспечении основ взаимодействия между участниками проекта. Обратите внимание, что само содержимое (текст, изображения, логотип и навигацию) не нужно помещать в блок-схему. На рис. 5.13 показана

схема веб-страницы, содержащей логотип, область навигации, область содержимого (с заголовками, подзаголовками, изображениями, абзацами и неупорядоченными списками) и область нижнего колонтитула.

В главе 6 вы узнаете, как с помощью каскадных таблиц стилей (CSS) конфигурировать веб-страницы с несколькими колонками. Иногда макет главной страницы отличается от макета, используемого для страниц с контентом сайта. Даже в этом случае создать более цельный веб-сайт помогут выполненные в одном стиле логотип и цветовая схема.

Рис. 5.13. В этой блок-схеме макета веб-страницы используются изображения и колонки разной ширины

Техники дизайна макетов веб-страниц

Теперь, когда вы познакомились с основами дизайна веб-страницы, самое время рассмотреть три наиболее известных метода создания макета веб-страницы: фиксированный, эластичный и резиновый.

Фиксированный макет

В фиксированном макете контент страницы выстраивается вдоль левого поля, и при ее форматировании используются блочные элементы фиксированной ширины. К числу веб-сайтов, применяющих эту технику, относится сайт **www.3dnews.ru**, представленный на рис. 5.14.

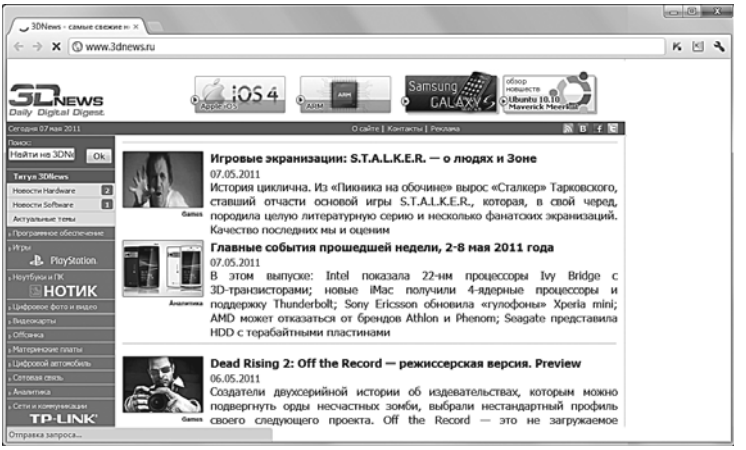


Рис. 5.14. Пример фиксированного макета

Эластичный макет

При использовании эластичного макета содержимое страницы центрируется, при этом оно может иметь постоянную ширину или ширина может задаваться в процентах, например 80%. Независимо от разрешения содержимое страницы расположено по центру с одинаковыми полями слева и справа. Сайт журнала «Русский репортер»[1], показанный на рис. 5.15, использует эластичный дизайн. Другие сайты, в настоящее время использующие эту технику, это **exler.ru** и **domolink.ru**.

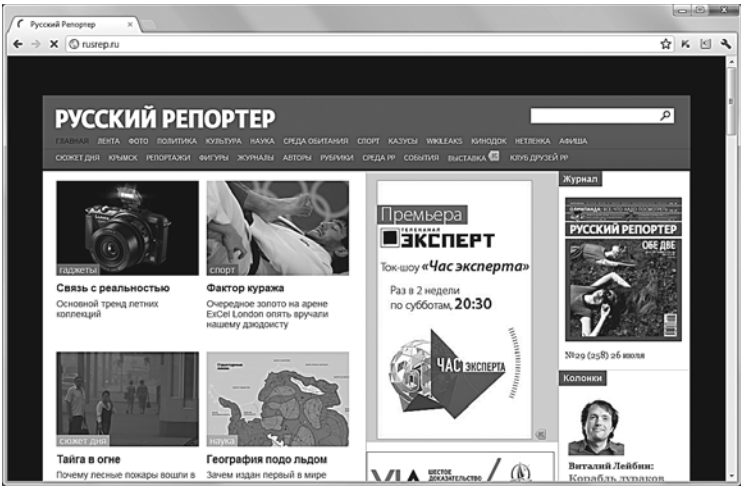


Рис. 5.15. Пример эластичного макета

[1] rusrep.ru

Эластичный дизайн обычно используется потому, что такие веб-страницы привлекательнее всего смотрятся при различных разрешениях экрана.

Резиновый макет

При использовании **резинового макета** создается растягиваемая веб-страница, содержимое которой занимает все 100% окна браузера, независимо от разрешения экрана. Справа и слева нет пустых полей — содержимое, разделенное на колонки, заполняет все пространство, каковы бы ни были размеры окна. Недостаток такого макета страницы в том, что при высоком разрешении экрана строки текста могут занимать почти все окно браузера, и их нелегко будет прочитать. В качестве примера использования этого метода на рис. 5.16 показана веб-страница интернет-магазина Softkey[1]. Этот метод также использован на веб-сайтах **www.amazon.com** и **www.ozon.ru**.

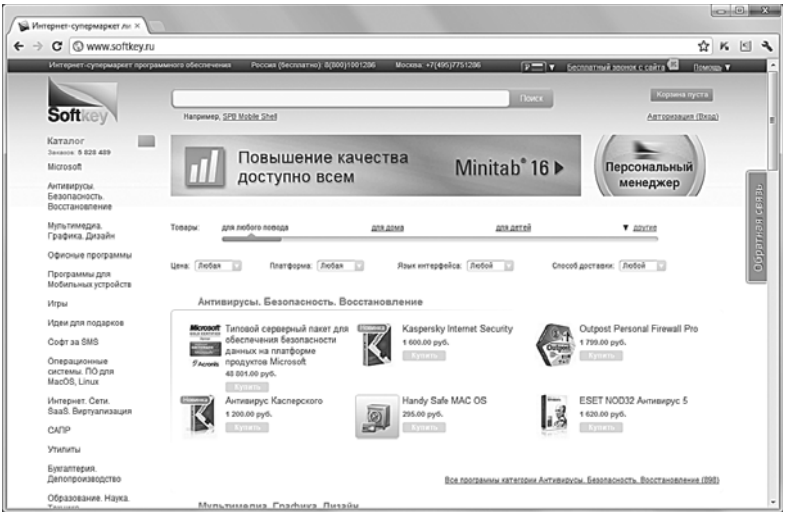


Рис. 5.16. Пример резинового макета

Сайты, сконструированные с использованием фиксированных, эластичных и резиновых дизайнов, могут быть найдены во Всемирной паутине повсюду. При применении методов фиксированных и эластичных дизайнов, использующих элементы фиксированной ширины, веб-дизайнер максимально контролирует компоновку страницы, но при ото-

[1] www.softkey.ru

бражении на экране с высоким разрешением на такой странице будут создаваться большие пустые области. Контент страниц с резиновым дизайном может становиться неудобочитаем при просмотре на экране с высоким разрешением, потому что страница, растягивается для заполнения всего окна браузера.

Дизайн для мобильного Интернета

Доступ ко Всемирной паутине с мобильных телефонов, смартфонов и планшетов позволяет быть всегда онлайн. Исследовательская фирма eMarketer[1] прогнозирует значительный рост мобильного доступа в Интернет и предполагает, что в 2013 году количество пользователей такого доступа увеличится до 134,3 млн. Учитывая этот рост, становится все важнее разрабатывать веб-страницы, доступные и удобные для пользователей мобильного Интернета.

Существует несколько точек зрения на лучший способ достижения этой цели, включая разработку нового мобильного сайта с помощью домена высшего уровня .mobi (см. главу 1 о доменах высшего уровня), создание отдельного сайта, размещенного на вашем текущем домене и предназначенного для мобильных пользователей, и использование CSS для создания таблиц стилей наряду с новыми мультимедийными особенностями спецификации CSS3 для настройки текущего веб-сайта таким образом, чтобы его можно было просматривать на мобильных устройствах. Вы потренируетесь применять эти способы верстки правил CSS в главе 7. Какой бы метод вы ни выбрали, при разработке мобильных сайтов помните следующее:

- **Малый размер экрана.** Распространенные размеры экрана мобильного телефона включают 320×240, 320×480, 480×800 и 640×960 (Apple iPhone 4). Даже на большом телефоне это пространство невелико!
- **Низкая пропускная способность (низкая скорость подключения).** Хотя использование более быстрых 3G и 4G сетей становится все более распространенным, многие пользователи мобильных устройств сталкиваются с низкой скоростью соединения. На обычном сайте на изображения тратится значительная часть пропускной способности. В зависимости от тарифного плана, некоторые пользователи мобильного Интернета платят за каждый килобайт. Помните об этом и удалите ненужные изображения.

[1] emarketer.com

- **Проблемы со шрифтами, цветами и мультимедиа.** Поддержка шрифтов на мобильных устройствах может быть весьма ограничена. Задайте размер шрифта, используя единицы `em` или процентное значение. Включите в таблицу стилей общие названия семейств шрифтов. Поддержка цветов на мобильных устройствах может также быть ограничена. Тщательно выбирайте цвета, чтобы максимально усилить контраст. Многие мобильные устройства не поддерживают файлы Adobe Flash.
- **Неудобное управление, ограниченные возможности процессора и памяти**. Несмотря на то что смартфоны с сенсорным управлением становятся все более популярными, у многих мобильных пользователей не будет доступа к управлению, похожему на управление мышью. Обеспечьте доступ к клавиатуре, чтобы помочь этим пользователям. Скорость процессора мобильного устройства и объем памяти увеличиваются, однако они по-прежнему не могут сравниться с ресурсами настольного компьютера. Хотя эта проблема не будет актуальна для веб-сайтов, которые вы создадите в ходе заданий данной книги, помните о ней в будущем, когда продолжите развивать свои навыки и создавать веб-приложения.
- **Функциональность.** Обеспечьте легкий доступ к функциям вашего сайта с помощью заметных гиперссылок или кнопки поиска. Мобильная версия сайта Ozon.ru (рис. 5.17) довольно скудна по сравнению с обычной (полнофункциональной) версией ресурса.

Рис. 5.17. Вид сайта Ozon.ru на мобильном устройстве

5.10. Другие приемы дизайна веб-страниц

Время загрузки

Самое худшее, что может случиться с вами, — это если посетитель уйдет с сайта, не дождавшись, когда он полностью загрузится! Убедитесь в том, что вы сделали все возможное для того, чтобы сайт загружался быстро. Эксперт по юзабилити Якоб Нильсен сообщает, что обычно посетители не хотят ждать больше 10 секунд. При скорости обмена 56 Кб/с браузеру требуется менее 9 секунд для полного отображения страницы (вместе со связанными файлами) с размером в 60 Кб.

Согласно прогнозам аналитического агентства AC&M[1], на конец 2012 года процент российских пользователей Интернета с широкополосной связью (кабельный Интернет, DSL и пр.) составляет более 19 млн. 40% домохозяйств подключены к сетям широкополосного доступа к Интернету. Но даже при учете этого факта необходимо помнить об оставшихся 60% домов, не имеющих доступа к широкополосному Интернету.

Рекомендованная величина размера сайта 60 Кб — это предел: лучше, если ваша главная страница вместе со связанными с ней мультимедийными файлами будет еще меньше. Подходить к предельной величине можно, только если вы уверены в том, что ваши посетители настолько нуждаются в данной на вашем сайте информации, что готовы ждать дольше. На графике, приведенном на рис. 5.18, сравнивается время загрузки файлов в зависимости от размера файла и скорости передачи[2].

Один из способов определить время загрузки веб-страницы — это посмотреть размеры ваших файлов, например, в программе Проводник (Windows Explorer) или Finder (в операционной системе OS X). Вычислите полный размер страницы, прибавив все изображения и мультимедийные файлы. Если суммарная величина больше 60 Кб и есть вероятность, что ваша аудитория не имеет широкополосного доступа к Интернету, пересмотрите ваш дизайн. Подумайте, действительно ли вам нужны все изображения для того, чтобы донести ваши идеи до аудитории? Возможно, стоит оптимизировать использованные вами изображения или разделить содержимое на несколько страниц.

---

[1] www.acm-consulting.com/data-downloads/cat_view/16-broadband.html

[2] Вычисления выполнены на веб-сайте **www.t1shopper.com/tools/calculate/downloadcalculator.php**

Рис. 5.18. Время загрузки файлов и скорость интернет-соединения

А популярные приложения верстки веб-страниц, такие как Microsoft Expression Web и Adobe Dreamweaver, помогут вам определить время загрузки для разных скоростей обмена.

Воспринимаемое время загрузки — это время загрузки веб-сайта по ощущениям его посетителя. Так как посетители часто уходят, если сайт загружается слишком долго, важно уменьшить ощущение, что человек ждет долго (чувство ожидания). Обычно это делается путем оптимизации изображений с помощью CSS-спрайтов (см. главу 7) и деления большой страницы на несколько страниц меньшого размера. Популярные инструменты для создания веб-страниц, такие как Microsoft Expression Web или Adobe Dreamweaver, могут рассчитать время загрузки при различных скоростях соединения.

Размещение на верхней половине полосы

Размещение наиболее важной информации на ***верхней половине полосы*** — это технология, позаимствованная в индустрии новостей. Когда газета лежит на прилавке или в газетном киоске в ожидании продажи, видна информация, помещенная в ее верхней части. Издатели заметили, что газета лучше продается, если в этих местах помещается самая важная, интересующая публику информация. Вы можете использовать этот прием для привлечения и удерживания посетителей на вашем веб-сайте. Разместите интересную информацию на видном месте страницы — на том, которое посетитель увидит, не прокручивая страницу. При самом популярном разрешении экрана 1024×768 пикселов размер такой обла-

сти (при учете, что часть экрана занимают меню и средства управления) составит около 600 пикселов.

Воздух

Термин ***воздух*** (white space) также заимствован из издательского дела. Размещение воздуха (поскольку бумага обычно белая) вокруг блоков текста облегчает его чтение. Размещение воздуха вокруг рисунков позволяет сделать их более заметными. Отделяйте ваши блоки текста от рисунков пустым пространством. Как определить, какой ширины воздух оставлять? Это определяется опытным путем. Экспериментируйте и подбирайте вариант, наиболее подходящий вашей целевой аудитории.

Не используйте горизонтальную прокрутку

Чтобы облегчить посетителям вашего сайта его просмотр и использование, старайтесь не создавать слишком широкие страницы, не помещающиеся по ширине в окно браузера. На таких страницах посетитель вынужден использовать линейку горизонтальной прокрутки. Однако в наши дни одним из распространенных разрешений экрана является значение 1024×768 пикселов[1]. Камерон Молл высказывает мнение[2], что оптимальная ширина страницы при разрешении 1024×768 пикселов составляет 960 пикселов. При этом имейте в виду, что не у всех ваших посетителей браузер будет полностью развернут. Тем не менее, конструируя сайт, размещайте наименее важную информацию справа.

Наиболее значимые области веб-страницы

В сфере недвижимости говорят, что три наиболее важных фактора, касающихся недвижимости, — это местоположение, местоположение и еще раз местоположение. Для веб-страницы тоже важно положение таких ключевых элементов, как логотип, заголовки и элементы навигации. Исследование[3] того, как человек рассматривает страницу, проведенное Пойнтеровским институтом[4], показало, что «сначала глаз че-

[1] gs.statcounter.com/#resolution-ww-monthly-201202-201208-bar
[2] www.cameronmoll.com/archives/001220.html
[3] www.webprojects.ru/publications/usability/9/
[4] Также следует обратить внимание на информацию со страницы **ido.tomsk.ru/forum/archive/index.php?t-106.html** — *Прим. пер.*

ловека фиксирует верхний левый угол страницы и какое-то время рассматривает его, а затем начинает двигаться слева направо». Это делает наиболее значимыми областями верхний левый угол и центральную верхнюю часть страницы. Следует избегать размещения важной информации и навигационных элементов далеко справа — при некоторых параметрах разрешения экрана эта область может не отображаться сразу после загрузки страницы в браузере.

Поддержка браузерами

Если вы создаете сайт не только для просмотра в Интранете внутри корпорации, готовьтесь к тому, что посетители будут открывать ваш веб-сайт в самых разных браузерах. То, что ваш веб-сайт великолепно выглядит в вашем браузере, не означает, что он будет так же хорош во всех браузерах. Согласно анализу на сайте StatCounter[1], самым популярным браузером все еще является Microsoft Internet Explorer, а браузеры с открытым исходным кодом Firefox и Google Chrome обретают все большую известность. По данным исследования, 33,5% посетителей вебсайтов используют браузер Internet Explorer (разные версии), 31,8% — Google Chrome, 24,7% — Firefox, 6,9% используют браузер Safari, 1,8% — браузер Opera, а оставшийся 1% — прочие браузеры.

Применяйте принцип прогрессивного улучшения. Разрабатывайте сайт так, чтобы он выглядел великолепно в браузере, наиболее часто использующемся вашей целевой аудиторией, а затем добавляйте улучшения с помощью CSS3 и/или HTML5, которые будут отображаться в современных браузерах. Всегда стремитесь протестировать ваши веб-страницы в наиболее известных браузерах для операционных систем OS X и Windows. На момент написания этой книги это были: Firefox 4, Internet Explorer 9, Safari (версия 4 для OS X и 5 для операционной системы Windows), Opera 11 и Google Chrome11. Многие компоненты веб-страниц, включая устанавливаемые по умолчанию размер шрифта и поля, отличаются для разных браузеров, версий браузера и операционных систем.

Также вы можете протестировать свои страницы на других типах устройств, таких как планшеты и смартфоны. Opera предлагает бесплатный эмулятор мобильного браузера Opera Mini на сайте **www.opera.com/mobile/demo**.

[1] gs.statcounter.com/#browser-ww-monthly-201202-201208-bar

Разрешение экрана

Посетители вашего сайта используют самые различные разрешения экрана. Согласно исследованиям, проведенным компанией StatCounter[1], наиболее часто используемые разрешения — 1366×768 (19,9%), 1024×768 (18%), 1280×800 (12,5%) и 1280×1024 (7,2%). Степень мобильного использования разных сайтов варьируется в зависимости от сайта, но ожидается, что она будет увеличиваться, поскольку растет использование смартфонов и планшетов. Учитывайте, что у мобильных устройств совсем небольшое разрешение: 240×320, 320×480 или 480×800. Популярные модели планшетов предлагают чуть более высокое разрешение экрана: Apple iPad (1024×768), Motorola Xoom (1280×800), Samsun Galaxy Tab (1200×800), Blackberry Playbook (1024×600). В главе 7 вы изучите технику конфигурирования веб-страниц, хорошо отображаемых при различных разрешениях экрана.

5.11. Таблица проверки аспектов веб-дизайна

В таблице 5.1 приведен перечень аспектов веб-дизайна, рекомендованных для контроля. Это будет ваше руководство к действию, которое поможет создавать легкочитаемые, полезные и доступные веб-сайты.

Таблица 5.1. Перечень аспектов веб-дизайна, рекомендованных для контроля[2]

Макет веб-страницы		
□	1	Учтена целевая аудитория
□	2	Созданы единообразные заголовки и логотип
□	3	Созданы единообразные элементы навигации
□	4	Имеется информативный заголовок с именем компании/организации/сайта
□	5	Внизу страницы имеются данные по копирайту, дата последнего обновления и контактный адрес электронной почты
□	6	Использованы основные принципы дизайна: повторяемость, контраст, приближенность и выравнивание
□	7	Страница не требует горизонтальной прокрутки при разрешении экрана 1024×768 и более высоких значениях разрешения
□	8	Текст, графика и пробелы сбалансированы
□	9	Обеспечен хороший контраст между фоном и текстом

[1] gs.statcounter.com/#resolution-ww-monthly-201202-201208-bar

[2] Право на копирование Перечня аспектов веб-дизайна принадлежит ресурсу terrymorris.net. Копирование без разрешения запрещено.

□	10	Повторяющаяся информация (заголовок, логотип, навигационная область) занимает не более четверти или трети окна браузера при разрешении 1024×768
□	11	Главная страница содержит интересную, притягивающую посетителя информацию в открывающейся в первый момент части страницы (до ее прокручивания вниз) при разрешении 1024×768 пикселов
□	12	Главная страница загружается не более 10 секунд при коммутируемом соединении
Совместимость с браузером		
□	1	Страница отображается в браузере Internet Explorer (версия 8 и поздние)
□	2	Страница отображается в браузере Firefox (версия 4 и поздние)
□	3	Страница отображается в браузере Google Chrome
□	4	Страница отображается в браузере Safari (платформы Macintosh и Windows)
□	5	Страница отображается в браузере Opera (версия 10 и поздние)
□	6	Страница отображается на мобильных устройствах
Навигация		
□	1	Главные навигационные элементы обозначены четко и единообразно
□	2	Пользование навигационными элементами легко для целевой аудитории
□	3	Кроме главных навигационных элементов, выполненных в виде изображений, анимации или динамических элементов, для обеспечения доступности сайта внизу страницы помещены текстовые навигационные элементы
□	4	Использованы дополнительные навигационные возможности, такие как карта сайта, ссылки на переход к содержимому сайта, «хлебные крошки»
□	5	Все гиперссылки работают
Цвета и графика		
□	1	Количество цветов, использованных на странице для фона и текста, не превышает 3–4
□	2	Цвета используются единообразно
□	3	Цвет фона достаточно контрастен по отношению к тексту
□	4	Не только цвет используется для повышения воспринимаемости
□	5	Использованные цветовая схема и графика улучшают внешний вид сайта, а не отвлекают от его содержания
□	6	Графика оптимизирована для Всемирной паутины и не слишком замедляет загрузку сайта
□	7	Каждое изображение служит определенным целям
□	8	Элемент `image` использует атрибут `alt` для конфигурирования альтернативного текста, отображающегося в случае, если браузер или пользовательское устройство не поддерживает графику (доступность)
□	9	Анимированные изображения не отвлекают от контента сайта и не повторяются или повторяются в течение небольшого периода времени
Мультимедийный контент (см. главу 11)		
□	1	Использование каждого аудио/видео/анимации оправдано
□	2	Аудио/видео/анимация скорее привлекают внимание к сайту, чем отвлекают от его содержания

□	3	Для каждого аудио- или видеообъекта предусмотрены заголовки
□	4	Указано время загрузки аудио- или видеообъекта
□	5	Предусмотрены ссылки на проигрывающие устройства для мультимедийных файлов
Представление содержания страницы		
□	1	Использованы распространенные шрифты, такие как Arial или Times New Roman
□	2	Стиль текстов соответствует принятым в веб: заголовки, круглые маркеры, короткие предложения в коротких абзацах, использование «воздуха» и пр.
□	3	Шрифты, размеры шрифтов и их цвет используются единообразно
□	4	Контент сайта значимый, полезный и информативный
□	5	Контент сайта организован единообразно
□	6	Нужную информацию легко найти (с минимальным количеством щелчков)
□	7	Правильно указана дата последнего обновления и дата копирайта
□	8	Сайт не содержит устаревшей информации
□	9	В тексте нет орфографических и грамматических ошибок
□	10	Нет таких надписей, как «Щелкните здесь», в тексте для гиперссылок
□	11	При использовании стандартных цветов для ссылок ссылка на открытую в настоящее время страницу выделяется цветом
□	12	Если для передачи информации используются графические или мультимедийные средства, создан их текстовый эквивалент (в целях обеспечения доступности)
Работоспособность		
□	1	Все внутренние гиперссылки работают
□	2	Все внешние гиперссылки работают
□	3	Все функции форм работают так, как ожидается
□	4	Отсутствуют ошибки JavaScript (см. главы 11 и 14) на страницах
Доступность		
□	1	Если главные навигационные элементы выполнены в виде изображений, анимации или динамических элементов, внизу страницы помещены текстовые навигационные элементы
□	2	Не только цвет используется для повышения воспринимаемости
□	3	Элемент `image` использует атрибут `alt` для конфигурирования замещающего текста
□	4	Для каждого аудио- или видеообъекта предусмотрены заголовки
□	5	Используются атрибуты, улучшающие доступность: `alt`, `longdesc`, `title` и в необходимых местах дается краткая текстовая информация
□	6	На сайте, использующем фреймы, заголовки фреймов, а также значимый текст размещаются вне фреймов
□	7	Для помощи программам экранного доступа атрибуты HTML-элементов `lang` и `xml:lang` указывают язык страницы

Глава 6
РАЗРАБОТКА МАКЕТА СТРАНИЦЫ

Цели главы

В этой главе вы узнаете следующее:

- как описать и использовать блочную модель CSS;
- как задавать поля с помощью CSS;
- как создавать плавающие элементы с помощью CSS;
- как использовать абсолютное и относительное позиционирование, используя CSS;
- как создавать с помощью CSS макет веб-страницы с двумя колонками;
- как конфигурировать средства навигации в виде неупорядоченного списка и стилизовать их с помощью CSS;
- как делать гиперссылки интерактивными, используя псевдоклассы CSS;
- как конфигурировать веб-страницы, используя структурные элементы HTML5, включая `section`, `header`, `hgroup`, `nav`, `aside` и `footer`.

Вы уже конфигурировали макет веб-страницы, выровненной по центру с помощью CSS. В этой главе мы увеличим ваш арсенал методов создания макетов страниц с применением CSS и начнем с блочной модели. Вы изучите, как конфигурировать и позиционировать плавающие элементы с помощью CSS; узнаете, как использовать правила CSS для добавления интерактивности гиперссылкам с помощью псевдоклассов и конфигурировать средства навигации в виде неупорядоченных списков. Вы научитесь использовать в коде новые элементы языка HTML5, задающие структуру содержимого веб-страницы.

6.1. Блочная модель

Каждый элемент в документе является прямоугольным блоком. Как показано на рис. 6.1, этот блок представляет собой область содержания (контента), окруженную отступами, границей и полями. Это и есть ***блочная модель*** элемента.

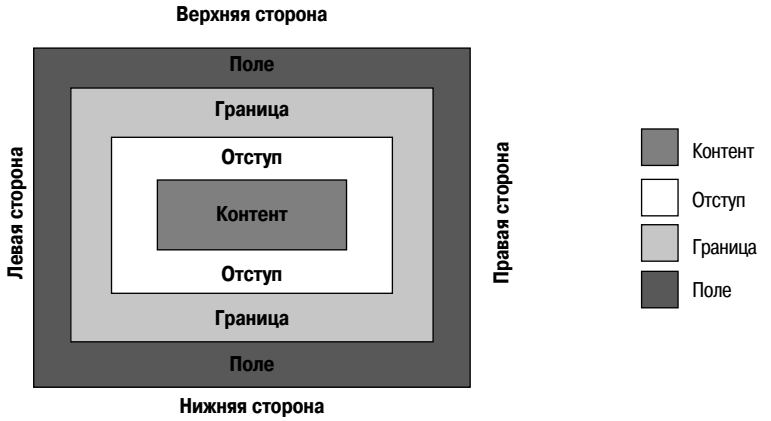


Рис. 6.1. Блочная модель элемента в CSS

Контент

Область ***контента*** состоит из текста и других элементов страницы, таких как изображения, абзацы, заголовки, списки и пр. ***Видимая ширина*** элемента на веб-странице — это сумма ширины содержания, ширины области отступа и ширины рамки. Однако свойство `width` задает только фактическую ширину контента страницы, оно не включает в себя отступы, границу или поля.

Отступ

Область ***отступа*** — это область между контентом и границей. По умолчанию ширина отступа равна 0. При конфигурировании фона элемента установленная конфигурация распространяется как на область содержимого, так и на область отступа. Используйте свойство `padding`, чтобы задать отступы элемента (см. главу 4).

Граница

Область ***границы*** (рамки) расположена между областью отступа (поле внутри элемента) и полем (вне элемента). По умолчанию ширина границы равна 0 и она не отображается. Используйте свойство `border`, чтобы задать границу элемента (см. главу 4).

Поле

Полем называется пустое пространство между соседними элементами. Сплошная линия на рис. 6.1, которая ограничивает область поля, не будет отображаться на веб-странице.

Свойство margin

Используйте свойство margin, чтобы конфигурировать поля со всех сторон элемента. Поле всегда прозрачное — в этой области виден цвет фона веб-страницы или родительского элемента. В самих браузерах по умолчанию устанавливаются определенные размеры поля для страницы документа и некоторых элементов, таких как абзацы, заголовки, формы и пр. Для замещения этой функции используйте свойство margin.

Чтобы задать размер поля, используйте числовое значение (в пикселах или единицах em). Чтобы удалить поле, установите его значение на 0 (не указывая единицу измерения). С помощью значения auto можно указать, что браузер должен рассчитать величину поля сам. В главах 3 и 4 вы использовали значения margin-left: auto; и margin-right: auto; чтобы конфигурировать макет веб-страницы, выровненный по центру. В таблице 6.1 показаны свойства CSS, задающие поля.

Таблица 6.1. Конфигурирование полей с помощью CSS

Свойство	Описание и распространенные значения
margin	Упрощенная нотация конфигурации полей элемента. Числовое значение (px или em) или величина в процентах; например, margin: 10px; Значение auto заставляет браузер автоматически рассчитывать поля элемента. Два числовых значения (px или em) или величи́ны в процентах: первое значение задает верхнее и нижнее поля, а второе — правое и левое, например: margin: 20px 10px; Три числовых значения (px или em) или величи́ны в процентах: первое значение задает верхнее поле, второе — правое и левое поля, а третье значение задает нижнее поле, например: margin: 10px 20px 5px; Четыре числовых значения (px или em) или величи́ны в процентах: значения конфигурируют поля в следующем порядке: margin-top, margin-right, margin-bottom, margin-left, например: margin: 10px 30px 20px 5px;
margin-bottom	Нижнее поле: числовое значение (px или em) или величина в процентах
margin-left	Левое поле: числовое значение (px или em) или величина в процентах
margin-right	Правое поле: числовое значение (px или em) или величина в процентах
margin-top	Верхнее поле: числовое значение (px или em) или величина в процентах

Блочная модель в действии

Веб-страница, показанная на рис. 6.2 (*Примеры\Глава_06\box.html* на прилагающемся к книге диске), демонстрирует блочную модель в действии с элементами h1 и div.

- Для элемента `h1` задан светло-голубой фон, отступ шириной 20 пикселов (пространство между контентом и границей), а также граница черного цвета шириной 1 пиксел.
- Свободное пространство, где проглядывает белый фон веб-страницы, — это поле. Когда стыкуются два вертикальных поля (к примеру, как между элементами `h1` и `div`), браузер не применяет оба значения полей, а устанавливает в качестве величины поля большее из значений.
- Для элемента `div` сконфигурирован синий фон, установленный браузером по умолчанию отступ (нулевой отступ) и граница черного цвета шириной 5 пикселов.

В этой главе вы потренируетесь использовать блочную модель.

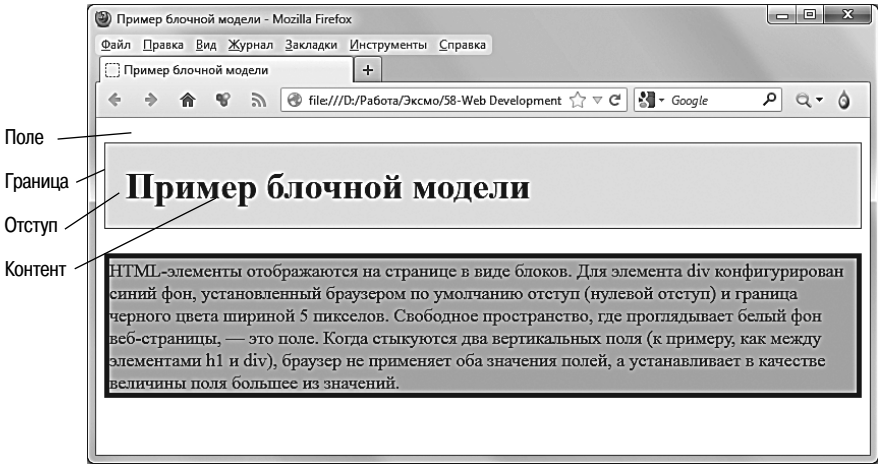


Рис. 6.2. Пример блочной модели

6.2. Нормальный поток

Браузеры считывают код вашей веб-страницы строка за строкой в том порядке, как он размечен в документе *.html*. Этот процесс называется **нормальным потоком.** Нормальный поток отображает элементы в том порядке, в котором они расположены в исходном коде.

На рис. 6.3 и 6.4 показаны элементы `div`, содержащие текст. Посмотрим на них внимательно. На рис. 6.3 показан снимок двух элементов `div`, размещенные на странице один за другим. На рис. 6.4 один элемент находится внутри другого. В обоих случаях браузер работает в приня-

том по умолчанию режиме нормального потока и отображает элементы в том порядке, в котором они расположены в исходном коде.

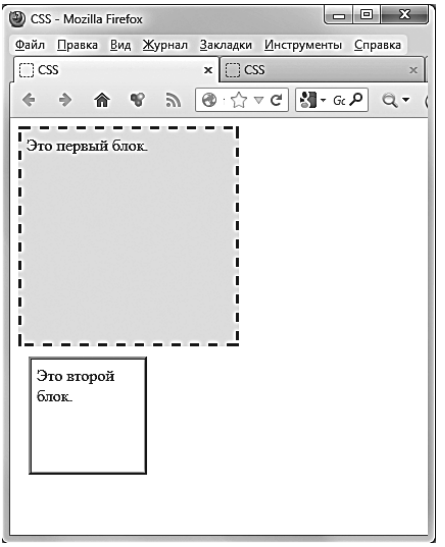


Рис. 6.3. Модель двух элементов `div`

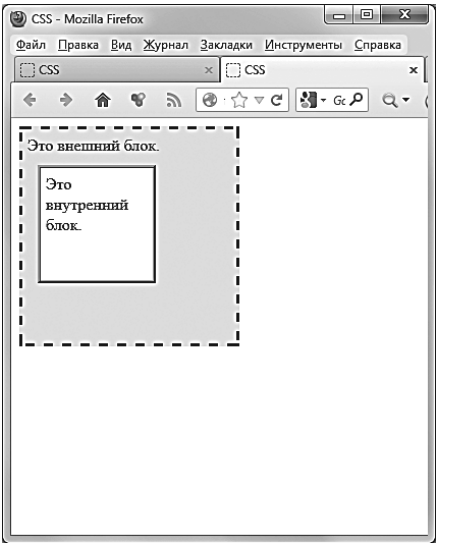


Рис. 6.4. Модель вложенных элементов `div`

На предыдущих практических заданиях вы создавали структуры, отображавшиеся браузером в режиме нормального потока. На нескольких последующих заданиях вы еще попрактикуетесь в этом, а также поэкспериментируете с позиционированием для организации потока (или положения) элемента на веб-странице.

Практическое задание 6.1

На этом практическом задании вы будете изучать блочную модель, создавая веб-страницы, показанные на рис. 6.3 и рис. 6.4.

Упражнение с нормальным потоком

Откройте файл *Примеры\Глава_02\template.html* в программе Блокнот (Notepad). Внесите изменения в код согласно листингу ниже. Сохраните файл с именем *box1.html*. Отредактируйте код веб-страницы, добавив следующие строки, конфигурирующие два элемента `div`.

```
<div class="div1">
Это первый блок.
</div>
```

```
<div class="div2">
Это второй блок.
</div>
```

Теперь добавим в раздел заголовка глобальные стили CSS, конфигурирующие «блоки». Добавьте новые правила стиля для класса с именем `div1`, которые задают синий фон, штриховую границу, ширину элемента 200 пикселов, высоту 200 пикселов и отступ 5 пикселов. Код выглядит следующим образом:

```
.div1 {width:200px;
height:200px;
background-color:#D1ECFF;
border:dashed; #000000
padding:5px; }
```

Создайте новые правила стиля для класса с именем `div2`, которые задают ширину и высоту элемента в 100 пикселов, белый фон, ребристую границу, поле шириной 10 пикселов и отступ 5 пикселов. Код выглядит следующим образом:

```
.div2 {width:100px;
height:100px;
background-color:#ffffff;
border: ridge; #000000
padding:5px;
margin:10px; }
```

Сохраните файл и проверьте его в браузере. Ваша страница должна выглядеть так, как показано на рис. 6.3. Пример находится на прилагающемся к книге диске, в папке *Примеры\Глава_06*.

Упражнение с нормальным потоком и вложенными элементами

Откройте файл *Примеры\Глава_06\box1.html* в программе Блокнот (Notepad). Внесите изменения в код согласно листингу ниже. Удалите содержимое раздела `body` веб-страницы. Добавьте следующие строки, конфигурирующие два вложенных элемента `div`.

```
<div class="div1">
Это внешний блок.
<div class="div2">
Это внутренний блок.
</div>
</div>
```

Сохраните файл с именем *box2.html*. Запустите браузер и проверьте в нем ваш файл. Ваша страница должна выглядеть так, как показано на рис. 6.6. Образец находится на диске, прилагающемся к книге, в папке *Примеры\Глава_06*.

Пример, рассмотренный на этом задании, содержал два элемента `div`, однако блочная модель применяется к различным типам элементов, а не только к элементам `div`. В этой главе вы продолжите практиковаться в применении блочной модели.

Свойства CSS-макета

Вы уже видели, что при нормальном потоке браузер отображает элементы страницы в том порядке, в котором они расположены в исходном HTML-коде. Когда вы применяете CSS для конфигурирования страницы, у вас не раз возникает желание задать элементам определенное положение на странице — абсолютные координаты в пикселах, или положение относительно точки, где он должен был бы находиться в режиме нормального потока, либо свободно перемещающийся (обтекаемый или плавающий). Свойства CSS, используемые для этих целей, будут обсуждаться далее.

6.3. Позиционирование с помощью CSS

Вы уже видели, что при нормальном потоке браузер отображает элементы страницы в том порядке, в котором они расположены в исходном HTML-коде. Когда вы применяете CSS для компоновки страницы, в некоторых ситуация вам потребуется больше контроля над расположением элементов. В данном разделе мы познакомим вас с относительным и абсолютным позиционированием.

Относительное позиционирование

Применяйте ***относительное позиционирование*** для незначительного изменения положения элемента по отношению к той точке, где он появился бы в нормальном потоке. Конфигурируйте относительное позиционирование с помощью свойства `position:relative` вместе с одним или несколькими из следующих свойств: `left`, `right`, `top` и `bottom`. В таблице 6.2 перечислены свойства CSS для позиционирования и смещения. На рис. 6.5 показана веб-страница (см. файл *Примеры\Глава_06\relative.html* на диске, прилагающемся к книге), использующая относительное позиционирование и свойство `left`, чтобы задать поло-

жение элемента относительно того, которое он имел бы в случае нормального потока. В этом случае элемент-контейнер — это элемент `body` веб-страницы.

Таблица 6.2. Свойства CSS для относительного и абсолютного позиционирования

Свойство	Значение	Цель
`position`	`static`	Установленное по умолчанию значение; элемент визуализируется в нормальном потоке
	`relative`	Задает положение элемента по отношению к той точке, где он находился бы, если бы позиционировался в нормальном потоке
	`absolute`	Задает положение элемента вне нормального потока точно в элементе-контейнере
	`fixed`	Задает положение элемента в окне просмотра браузера; элемент не смещается при прокручивании страницы
`left`	Числовое значение или величина в процентах	Положение элемента со смещением от левого края элемента-контейнера
`right`	Числовое значение или величина в процентах	Положение элемента со смещением от правого края элемента-контейнера
`top`	Числовое значение или величина в процентах	Положение элемента со смещением от верхнего края элемента-контейнера
`bottom`	Числовое значение или величина в процентах	Положение элемента со смещением от нижнего края элемента-контейнера

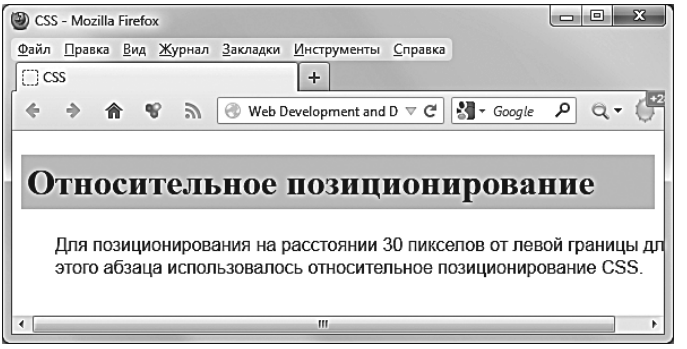


Рис. 6.5. Этот абзац сконфигурирован с использованием относительного позиционирования

В результате содержимое элемента расположено на расстоянии 30 пикселов от левой границы, а в случае нормального потока оно начиналось бы точно на левой границе браузера. Обратите внимание, как свойства `padding` и `background-color` конфигурируют элемент заголовка. Код CSS имеет следующий вид:

```
#myContent {position: relative;
left:30px;
font-family:Arial,sans-serif; }
h1 {background-color:#cccccc;
padding:5px;
color: #000000; }
```

а исходный HTML-код выглядит так:

```
<h1>Относительное позиционирование</h1>
<div id="myContent">
<p>Для позиционирования на расстоянии 30 пикселов
от левой границы для этого абзаца использовалось
относительное позиционирование CSS.</p>
</div>
```

Абсолютное позиционирование

Для размещения элемента в точно заданном положении используется ***абсолютное позиционирование***. Задайте абсолютное позиционирование с помощью свойства `position:absolute` вместе с одним или несколькими из следующих свойств смещения: `left`, `right`, `top` и `bottom`. На рис. 6.6 показана веб-страница (см. файл *Примеры\Глава_06\absolute.html* на диске, прилагающемся к книге), использующая абсолютное позиционирование элемента `div`, чтобы отобразить контент со смещением на 200 пикселов вправо от левого края и на 100 пикселов вниз от верхнего края элемента-контейнера, который является телом документа.

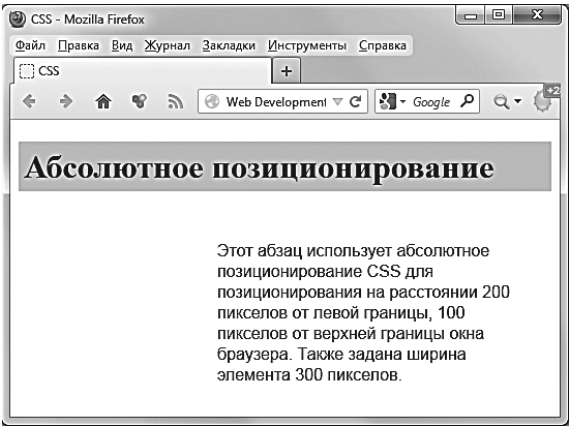


Рис. 6.6. Элемент `div` сконфигурирован с использованием функции абсолютного позиционирования

Свойства `padding` и `background-color` используются для конфигурирования заголовка. Код CSS имеет следующий вид:

```
#content {position: absolute;
left:200px;
top:100px;
font-family:Arial,sans-serif;
width:300px; }
h1 { background-color:#cccccc;
padding:5px;
color: #000000; }
```

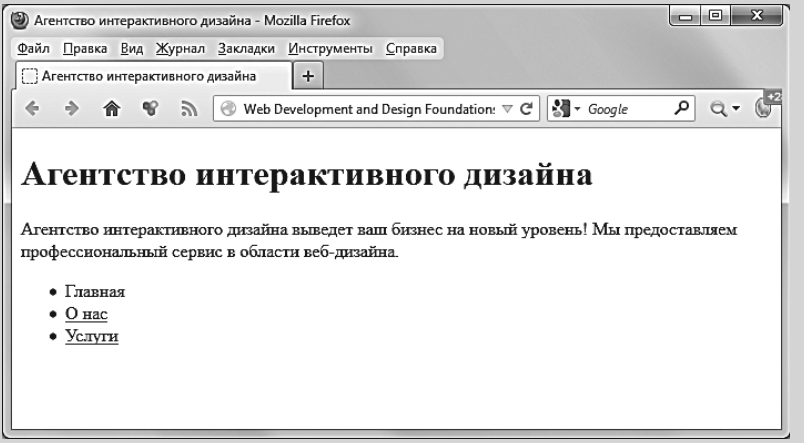
а HTML-код выглядит так:

```
<h1>Абсолютное позиционирование</h1>
<div id="content">
<p>Этот абзац использует абсолютное
позиционирование CSS для позиционирования на
расстоянии 200 пикселов от левой границы, 100
пикселов от верхней границы окна браузера. Также
задана ширина элемента 300 пикселов.</p>
</div>
```

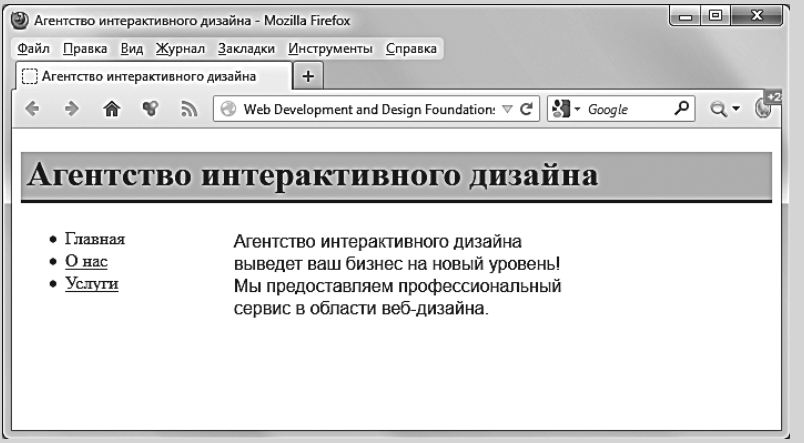
При работе с абсолютным позиционированием необходимо иметь в виду, что элементы, для которых не используется абсолютное позиционирование, отображаются браузером в соответствии с ходом нормального потока. При этом элементы с абсолютным позиционированием отображаются вне нормального потока. Вы изучите это на следующем практическом задании.

Практическое задание 6.2

На рис. 6.7 приведены снимки двух веб-страниц с похожим HTML-контентом. Страница, показанная на верхнем снимке, не использует таблицу CSS. На показанной на нижнем рисунке веб-странице с помощью правил CSS конфигурируется текст, цвет и задается абсолютное положение абзаца. Откройте файл *Примеры\Глава_06\starter2.html* в программе Блокнот (Notepad). Внесите изменения в код согласно листингу ниже. Запустите браузер и проверьте страницу. Браузер будет отображать страницу в режиме нормального потока и HTML-элементы будут отображаться в том порядке, в котором они расположены: `h1`, `div`, `p` и `u1`.



Без таблицы CSS



С таблицей CSS

Рис. 6.7. Нижняя веб-страница использует абсолютное позиционирование

Теперь добавим к странице правила CSS, чтобы сделать ее более стильной и похожей на отображенную на рис. 6.9 (внизу). Сохраните файл с именем *trillium.html*. В текстовом редакторе измените код следующим образом:

1. Эта страница использует глобальные стили. Добавьте открывающий и закрывающий теги `style` в разделе заголовка.

   ```
   <style>
   </style>
   ```

2. Создайте правила стиля для селектора `h1`. Задайте цвет фона `#B0C4DE`, цвет текста `#000080`, сплошную границу шириной 3 пиксела внизу с цветом `#000080` и отступ в 5 пикселов от нижней и левой границ элемента.

```
h1 {border-bottom: 3px solid #000080;
color: #000080;
background-color:#B0C4DE;
padding: 0 0 5px 5px; }
```

3. Создайте правила стилей для класса с именем `content`. Задайте абсолютное позиционирование: расстояние от левой границы 200 пикселов, расстояние от верхней границы 75 пикселов, ширина 300 пикселов и шрифт Arial или любой другой без засечек.

```
.content { position: absolute;
left: 200px;
top: 75px;
font-family: Arial,sans-serif;
width:300px; }
```

4. Назначьте абзацу класс `content`, для чего добавьте запись `class="content"` к открывающему тегу абзаца в разделе тела веб-страницы.

Сохраните файл и проверьте его в браузере. Ваша страница должна выглядеть так, как показано на рис. 6.7. Пример файла *trillium.html* вы найдете на прилагающемся к книге диске в папке *Примеры\Глава_06*. Обратите внимание, что хотя в коде неупорядоченный список расположен после абзаца, он отображается сразу после заголовка. Это является следствием того, что абзацу приписано абсолютное положение (`position: absolute`), и браузер отображает его вне нормального потока.

Примечание: для простоты редактирования страницы на этом практическом задании использовалась глобальная таблица стилей CSS. Однако если сайт содержит несколько страниц, целесообразно использовать внешние таблицы стилей CSS. Чтобы освежить свои знания по применению внешних таблиц стилей, перейдите к главе 3. Далее в этой главе вы будете использовать внешние таблицы CSS.

6.4. Свойство `float`

Элементы, которые могут перемещаться («плавать») в левую или в правую части окна браузера или другого элемента, часто конфигурируются с помощью ***свойства*** **`float`**. Браузер отображает такие элементы в нормальном потоке, а затем смещает их вправо или влево, насколько это возможно в окне браузера или в элементе, охватывающем данный элемент (обычно `div`):

- примените свойство `float: right;`, чтобы переместить элемент вправо в элементе-контейнере;
- примените свойство `float: left;`, чтобы переместить элемент влево в элементе-контейнере;
- ширину обтекаемого элемента требуется задавать, если только его ширина не задана неявно, как, например, ширина элемента `img`.

Остальные элементы и содержимое страницы будут обтекать плавающий элемент.

На рис. 6.8 представлен вид веб-страницы с изображением, смещение которого вправо в окне браузера задано с помощью свойства `float-right` (см. прилагающийся к книге диск *Примеры\Глава_06 \ float.html*). Для обтекаемых изображений полезно задать размеры пустого пространства между изображением и текстом на странице с помощью свойства `margin`.

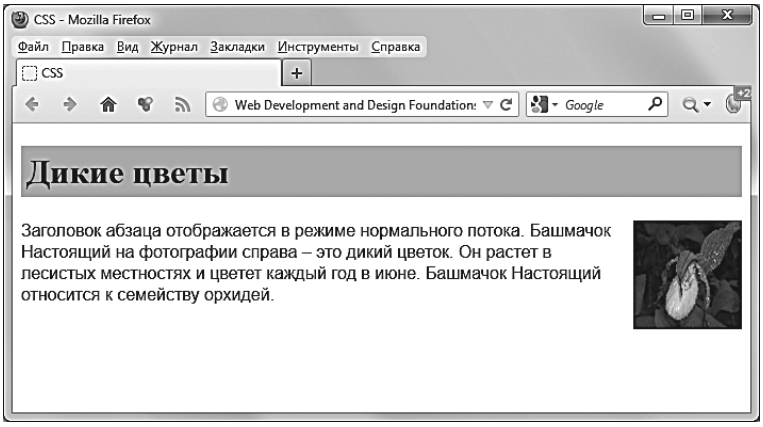


Рис. 6.8. Это изображение сконфигурировано как плавающее

Посмотрите на рис. 6.8 и обратите внимание, что изображение расположено у правой границы окна браузера. Для этого был создан идентификатор с именем `yls` и ему назначены свойства `float`, `margin` и `border`. Элементу `image` был присвоен атрибут `id="yls"`. Код CSS имеет вид:

```
h1 {     background-color:#A8C682;
padding:5px;
color: #000000; }
p {      font-family:Arial,sans-serif; }
```

```
#yls {float:right;
margin: 0 0 5px 5px;
border: 1px solid #000000; }
```

А HTML-код выглядит так:

```
<h1>Дикие цветы</h1>
<img id="yls" src="yls.jpg" alt="Башмачок
настоящий" height="100" width="100">
<p>Заголовок абзаца отображается в режиме
нормального потока. Башмачок настоящий на фотографии
справа - это дикий цветок. Он растет в лесистых
местностях и цветет каждый год в июне. Башмачок
настоящий относится к семейству орхидей.</p>
```

Практическое задание 6.3

На этом практическом задании вы будете тренироваться применять свойство CSS `float` для конфигурирования веб-страницы, показанной на рис. 6.9.

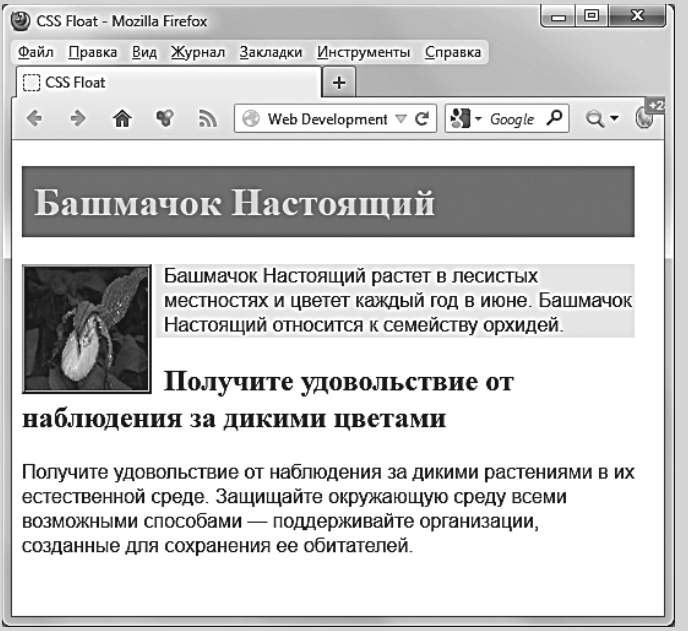


Рис. 6.9. Свойство CSS `float` выравнивает изображение по левому краю

Создайте папку с именем *ch6float*. Скопируйте файлы *starter3.html* и *yls.jpg* из папки *Примеры\Глава_06* в каталог *ch6float*. Откройте файл *starter3.html* в программе Блокнот (Notepad). Внесите изменения в код согласно листингу ниже. Посмотрите, в каком порядке отображаются рисунок и абзац. Обратите внимание на то, что для конфигурирования обтекаемого изображения здесь не были использованы CSS. Просмотрите файл *starter3.html* в браузере. Браузер работает в режиме нормального потока и отображает элементы страницы в том порядке, в котором они были включены в HTML-код.

Теперь добавим таблицу CSS для назначения элементу свойства `float`. Сохраните файл с именем *floatyls.html*. Отредактируйте файл *floatyls.html* в текстовом редакторе следующим образом:

1. Добавьте правила стиля классу с именем `float`, которые будут конфигурировать свойства `float`, `margin` и `border`.

```
.float {float:left;
margin-right:10px;
border: 3px ridge #000000; }
```

2. Присвойте элементу изображения класс с именем `float` (используйте запись `class="float"`).

Сохраните файл. Запустите браузер и проверьте вашу страницу. Она должна выглядеть так, как показано на рис. 6.9. Файл примера вы найдете на прилагающемся к книге диске, в папке *Примеры\Глава_06\floatyls.html*.

Обтекаемый элемент и нормальный поток

Уделите немного времени проверке вашего файла в браузере (см. рис. 6.9) и разберитесь, как браузер отображает страницу. Чтобы продемонстрировать, как обтекаемые элементы отображаются вне нормального потока, элемент `div` сконфигурирован со светлым фоном. Обратите внимание на то, что изображение и первый абзац находятся внутри элемента `div`.

Элемент `h2` указывается после `div`. Если бы все элементы отображались в режиме нормального потока, то область со светлым фоном содержала бы оба вложенных элемента `div`: рисунок и первый абзац. Кроме этого элемент `h2` был бы размещен на собственной строке ниже элемента `div`. Однако вследствие того, что изображение является обтекаемым и отображается вне нормального потока, светлый фон не появляется за первым абзацем, а элемент `h2` следует сразу за первым абзацем и рядом с обтекаемым рисунком. В следующем разделе вы будете изучать свойства, которые «отменят» свойство `float` и улучшат внешний вид страницы.

6.5. CSS: Отмена свойства `float`

Свойство `clear`

Свойство **`clear`** часто используется для отмены или прекращения обтекания. В зависимости от того, какой вариант обтекания требуется отменить, это свойство может принимать значения `left`, `right` или и то и другое. Вернемся к рис. 6.9 и примеру кода в файле *Примеры\Глава_06\floatyls.html*. Обратите внимание, что хотя элемент `div` содержит и изображение, и первый абзац, светлый фон, назначенный элементу `div`, отображается только в области, занятой первым абзацем — фон оканчивается немного раньше, чем мы ожидали. Отмена свойства `float` позволит устранить этот недостаток.

Удаление обтекаемого элемента с помощью разрыва строки

Обычно отмена свойства `float` внутри элемента-контейнера производится посредством добавления элемента разрыва строки (`br`) и свойства `clear`. Посмотрите пример в файле *Примеры\Глава_06\floatylsclear1.html*. Обратите внимание, что для отмены свойства `float: left` создан класс CSS:

```
.clearleft {clear:left; }
```

Кроме этого перед закрывающим тегом `</div>` добавлен элемент разрыва строки `<br>` для класса `clearleft`. Код элемента `div` имеет следующий вид:

```
<div>
<img class="float" src="yls.jpg" alt="Башмачок
настоящий" height="100" width="100">
<p>Башмачок настоящий растет в лесистых местностях
и цветет каждый год в июне. Башмачок настоящий
относится к семейству орхидей.</p>
<br class="clearleft">
</div>
```

На рис. 6.10 показан скриншот этой страницы. Рассмотрите изображение и обратите внимание на два изменения (по сравнению с рис. 6.9): светлый фон элемента `div` теперь распространяется дальше вниз страницы и текст элемента `h2` расположен на собственной строке ниже рисунка.

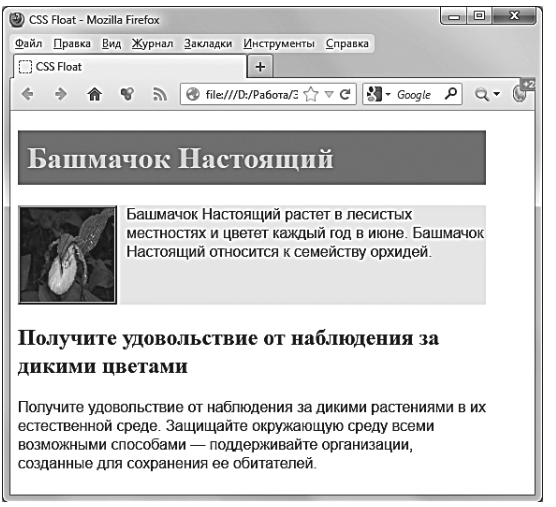


Рис. 6.10. К элементу разрыва строки применено свойство `clear`

Другой способ удалить обтекаемый элемент

Если вас не очень беспокоит отображение светлого фона, то можно использовать другой способ, не требующий включения элемента `br` — это применение класса `clearleft` к элементу `h2`. Этот подход не влияет на отображение светлого фона, но вынуждает элемент `h2` располагаться на собственной строке, что показано на рис. 6.11 (см. файл *Примеры\Глава_06\floatylsclear2.html* на прилагающемся к книге диске).

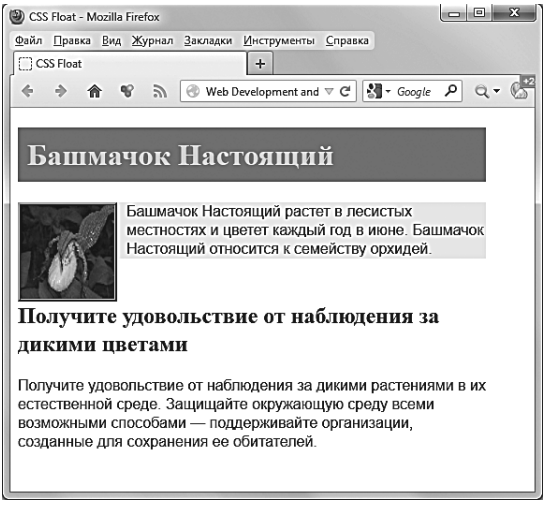


Рис. 6.11. Свойство `clear` применено к элементу `h2`

Свойство `overflow`

Свойство **`overflow`** часто используется для отмены обтекания, хотя его прямое предназначение — это конфигурирование отображения содержимого в случае, когда оно слишком велико для заданной области. В таблице 6.3 приведен перечень распространенных значений свойства **`overflow`**.

Таблица 6.3. Свойство **`overflow`**

Свойство	Цель
`visible`	Установленное по умолчанию значение; контент отображается и он слишком велик, контент выйдет за пределы отведенной ему области
`hidden`	Контент обрезан в соответствии с размером области, отведенной элементу в окне браузера
`auto`	Контент заполняет отведенную ему область и, если необходимо, появляются полосы прокрутки, обеспечивающие доступ к остальному контенту
`scroll`	Содержимое отображается в выделенной области и появляется полоса прокрутки

Удаление обтекаемого элемента

Рассмотрите рис. 6.9 и файл *Примеры\Глава_06\floatyls.html*. Найдите элемент `div`, содержащий обтекаемый рисунок и первый абзац страницы. Обратите внимание, что хотя элемент `div` содержит и изображение, и первый абзац, светлый фон, назначенный элементу `div`, отображается только в области, занятой первым абзацем. Для устранения этого недостатка изображения и отмены свойства `clear: left` можно использовать свойство `overflow`, применив его к элементу-контейнеру. В данном случае мы применим свойства `overflow` и `width` к селектору `div`. Код CSS, конфигурирующий указанным образом селектор `div`, имеет вид:

```
div {background-color:#F3F1BF;

overflow:auto;

width:100%; }
```

Данная запись в коде CSS — это все, что необходимо для запрета обтекания и придания странице вида, показанного на рис. 6.12 (см. файл *Примеры\Глава_06\floatylsoverflow.html*).

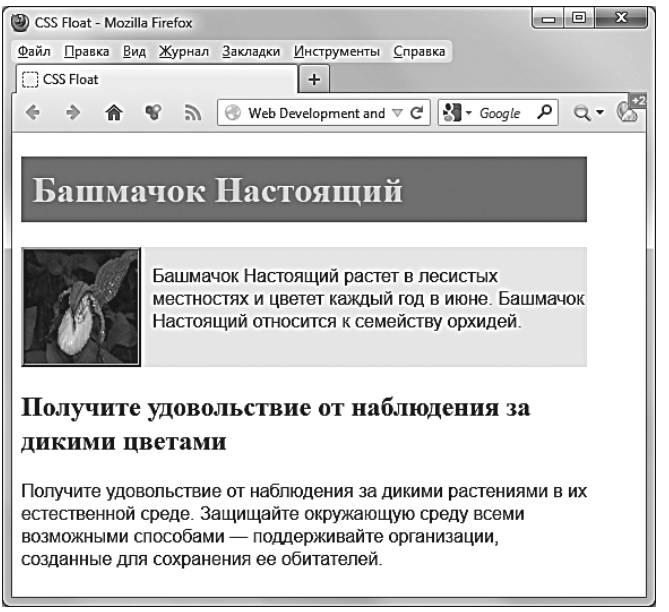


Рис. 6.12. Свойство `overflow` применено к элементу `div`

Обратите внимание, что при использовании свойства `overflow` (см. рис. 6.12) и применении свойства `clear` к элементу разрыва строки (см. рис. 6.10) отобразившиеся в результате веб-страницы внешне похожи. Вы можете задать вопрос, какие из свойств CSS (`clear` или `overflow`) лучше использовать для запрета обтекания. Свойство `clear` используется чаще, однако в этом примере более эффективным окажется применение к элементу-контейнеру свойства `overflow` (например, к контейнеру `div`). В этом случае свойство `overflow` запретит обтекание, будет устранена необходимость введения элемента `br` и будет обеспечено увеличение размера элемента-контейнера для отображения всего обтекаемого элемента.

Работая с этой книгой, вы продолжите практиковаться в использовании свойств `float`, `clear` и `overflow`. Обтекаемые элементы являются ключевыми компонентами при проектировании структур страниц с несколькими колонками с помощью CSS.

Конфигурирование полосы прокрутки

На рис. 6.13 показано использование свойства `overflow: auto;` для автоматического отображения полосы прокрутки, если содержимое

страницы выходит за границы отведенной ему области. В этом случае элемент `div`, содержащий абзац, и обтекаемое изображение были заданы шириной 300 и высотой 100 пикселов.

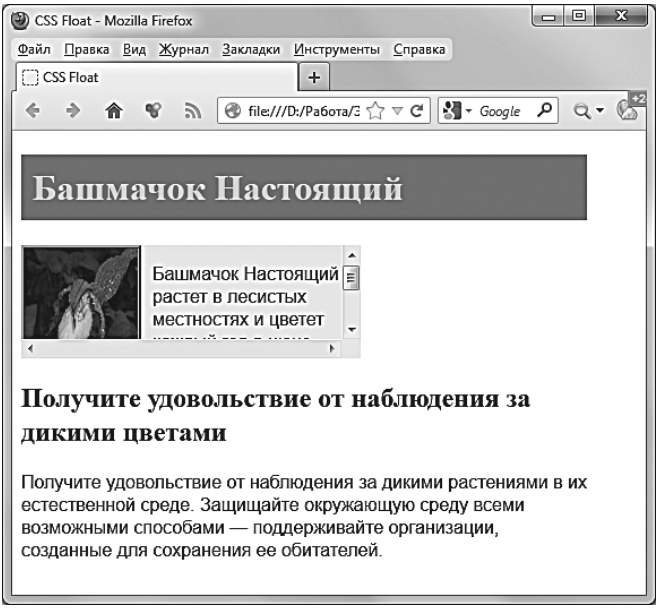


Рис. 6.13. Браузер отображает полосы прокрутки

Взгляните на образец веб-страницы (*Примеры\Глава_06\floatylsscroll.html* на прилагающемся к книге диске). Код CSS для элемента `div` приведен ниже:

```
div { background-color: #F3F1BF;
overflow: scroll;
width: 300px;
height: 100px; }
```

6.6. CSS: Макеты страниц в две колонки

Наиболее часто встречающийся дизайн веб-страницы — это компоновка в две колонки. Этого можно добиться с помощью CSS, сконфигурировав на веб-странице одну обтекаемую колонку. В этом разделе вы познакомитесь с двумя форматами макета страницы в две колонки.

Две колонки с навигацией слева

На рис. 6.14 представлен пример веб-страницы с двумя колонками. В левой колонке будут содержаться средства навигации. Образец HTML-кода для макета страницы приведен ниже:

```
<div id="wrapper">
<div id="leftcolumn">
</div>
<div id="rightcolumn">
<div id="header">
</div>
<div id="content">
</div>
<div id="footer">
</div>
</div>
</div>
```

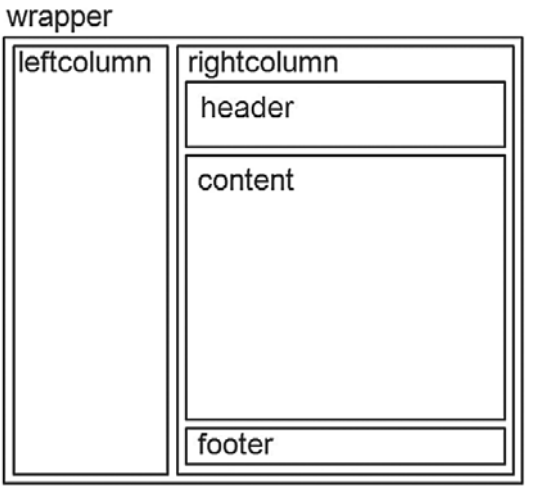


Рис. 6.14. Блок-схема макета в две колонки с навигацией слева

К веб-странице на рис. 6.15 применен макет в две колонки с навигацией слева (см. *Примеры\Глава_06\twocolumn1.html* на прилагающемся к книге диске). Ключевым моментом данного макета является то, что левая колонка сконфигурирована обтекаемой с помощью свойства

`float`. Браузер отображает остальной контент на странице в режиме нормального потока.

- Оболочка (область `wrapper`) выровнена по центру и занимает 80% ширины веб-страницы. Этой области назначен синий цвет фона, который будет отображаться за левой колонкой:

```
#wrapper { width: 80%;
margin-left: auto;
margin-right: auto;
background-color: #b3c7e6; }
```

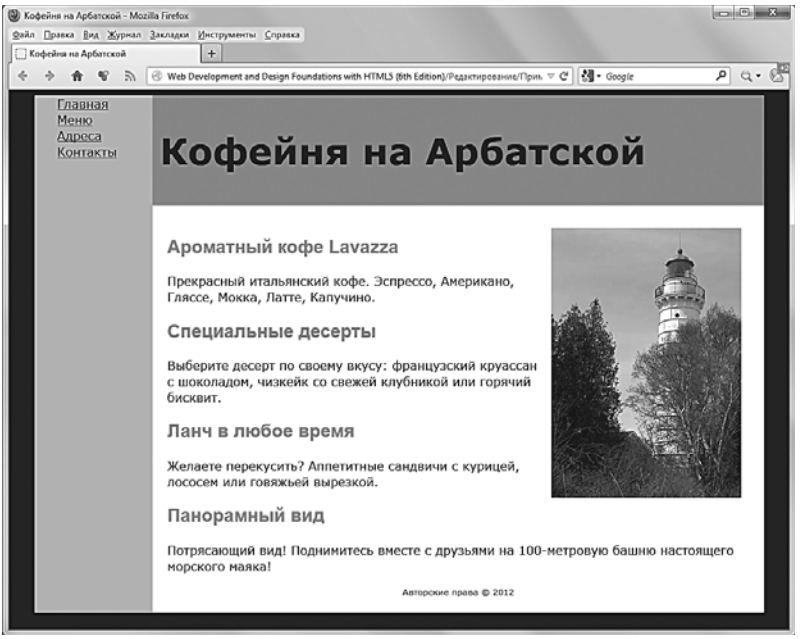


Рис. 6.15. Макет страницы в две колонки с навигацией слева

- Левой колонке задается фиксированная ширина, и она смещается к левому краю. Поскольку цвет фона не настроен, отображается цвет фона элемента-контейнера (элемент `div` оболочки):

```
#leftcolumn { float: left;
width: 150px; }
```

- Правой колонке задается поле слева, равное или превышающее ширину левой колонки. Благодаря этому полю создается видимость двух колонок (часто их называют искусственными колонками).

Правой колонке задается белый цвет фона, который перекрывает фон, конфигурированный в оболочке:

```
#rightcolumn { margin-left: 155px;
background-color: #ffffff; }
```

Две колонки с заголовком вверху и навигацией слева

На рис. 6.16 показана блок-схема веб-страницы, содержащая логотип вверху в разделе заголовка, имеющая две колонки и область навигации в левой колонке. HTML-шаблон для макета страницы следующий:

```
<div id="wrapper">
<div id="header">
</div>
<div id="leftcolumn">
</div>
<div id="rightcolumn">
<div id="content">
</div>
<div id="footer">
</div>
</div>
</div>
```

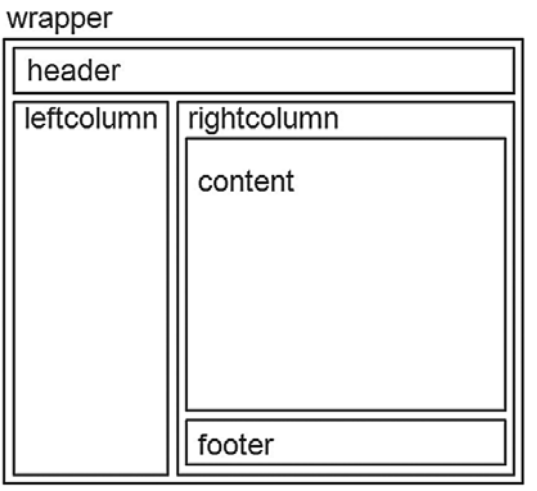


Рис. 6.16. Блок-схема макета в две колонки с областью логотипа вверху

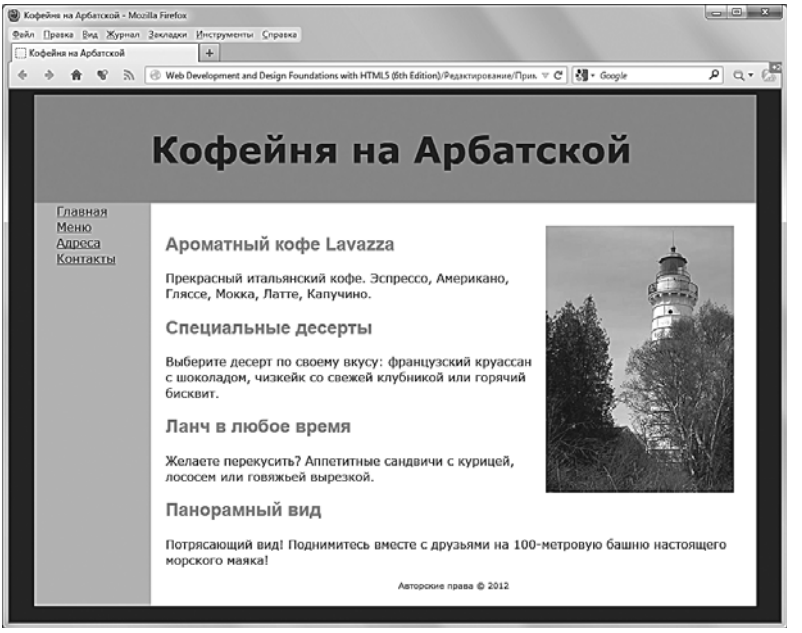


Рис. 6.17. Макет веб-страницы в две колонки с областью логотипа вверху и навигацией слева

К веб-странице на рис. 6.17 применен макет в две колонки с областью заголовка вверху (см. *Примеры\Глава_06\twocolumn2.html* на прилагающемся к книге диске). Код CSS, конфигурирующий области элементов `wrapper`, `leftcolumn` и `rightcolumn`, тот же самый, что и для веб-страницы, показанной на рис. 6.15. Однако обратите внимание, что расположение элемента `div`, которому присвоен идентификатор `header`, иное. Теперь он прописан в коде как первый элемент `div` в оболочке и отображается над левой и правой колонками.

ЧаВо

ОБЯЗАТЕЛЬНО ЛИ ДОБАВЛЯТЬ ОБОЛОЧКУ?

Нет, использовать оболочку или контейнер для макета веб-страницы не обязательно. Однако так намного проще добиться внешнего вида страницы с двумя колонками, поскольку цвет фона элемента `div` оболочки будет отображаться позади любого из его дочерних элементов, у которых не задан собственный цвет фона. Этот метод также предоставляет возможность конфигурировать на странице другой цвет фона или фоновое изображение (как показано на рис. 6.17), используя селектор элемента `body`.

Прежде чем макет веб-страницы в две колонки будет готов к демонстрации большой аудитории, нужно рассмотреть еще один аспект его дизайна. Область навигации — это список гиперссылок. Чтобы семантически наиболее близко описать область навигации, гиперссылки должны быть представлены в виде неупорядоченного списка. В следующем разделе вы узнаете, с помощью каких методов можно настроить гиперссылки горизонтальной и вертикальной панелей навигации как неупорядоченные списки.

6.7. Гиперссылки в виде неупорядоченного списка

Одним из достоинств CSS при его применении для компоновки вебстраниц является семантическая валидность кода. Написание семантически правильного кода означает, что используемые теги разметки максимально соответствуют назначению контента. Примерами семантически правильного кода являются использование заголовков разного уровня для заголовков и подзаголовков контента или размещение абзацев текста внутри элементов абзацев (а не использование переносов строки). Эти возможности программирования представляют собой шаг в направлении семантической Паутины. Ведущие разработчики вебсайтов, в том числе Эрик Мейер, Марк Ньюхаус, Джеффри Зелдман, поддерживают идею использования неупорядоченных списков для конфигурирования навигационного меню.

В конце концов, навигационное меню — это список гиперссылок. Конфигурирование меню с помощью списков повысит доступность сайтов, так как для программ экранного доступа упрощается навигация с клавиатуры и посредством звуковых сигналов по меню, организованным в виде списков, например, по номерам позиций в списке.

Конфигурирование маркеров списка с помощью CSS

Напомним, что по умолчанию в качестве маркера каждого элемента неупорядоченного списка отображается кружок.

В упорядоченном списке перед каждым элементом по умолчанию отображаются десятичные числа. При настройке ссылок навигации в виде неупорядоченного списка, вы не всегда захотите видеть эти маркеры.

Их легко задать с помощью CSS. Примените свойство `list-style-type`, чтобы задать маркер для неупорядоченного или упорядоченного списка. Основные значения параметров представлены в табл. 6.4.

Таблица 6.4. Свойства CSS для упорядоченного и неупорядоченного списков

Свойство	Описание	Значение	Вид маркера
`list-style-type`	Конфигурирует стиль маркера списка	`none`	Маркер не отображается
		`disc`	Кружок
		`circle`	Незаштрихованный кружок
		`square`	Квадрат
		`decimal`	Десятичное число
		`upper-alpha`	Заглавная буква
		`lower-alpha`	Строчная буква
		`lower-roman`	Маленькие римские числа
`list-style-image`	Замена маркера списка изображением		Изображение появляется перед каждым элементом списка
`list-style-position`	Задает местоположение маркера списка	`inside`	Маркеры размещаются с красной строки, текст идет под маркерами
		`outside`	Местоположение маркеров установлено по умолчанию

На рис. 6.18 показан неупорядоченный список с квадратными маркерами, сконфигурированными с помощью следующего кода CSS:

```
ul { list-style-type: square; }
```

- **Веб-дизайн**
- **Анимация**
- **Электронная коммерция**
- **Юзабилити**
- **Поисковая оптимизация**

Рис. 6.18. Неупорядоченный список с маркерами в виде квадратов

На рис. 6.19 показан упорядоченный список с заглавными буквами в качестве маркеров, сконфигурированными с помощью следующего кода CSS:

```
ol { list-style-type: upper-alpha; }
```

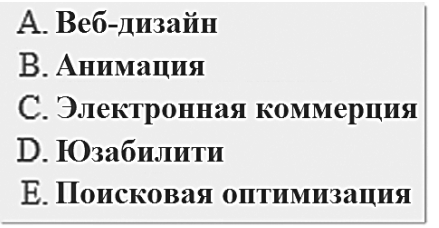


Рис. 6.19. Упорядоченный список с маркерами в виде заглавных букв

Конфигурирование изображения в качестве маркера списка

Если вы хотите, чтобы в качестве маркеров использовалась другое, выбранное вами изображение, используйте свойство `list-style-image`.

На рис. 6.20 для замены маркеров было взято изображение *trillium.gif*. Используется следующий код CSS:

```
ul {list-style-image: url(trillium.gif); }
```

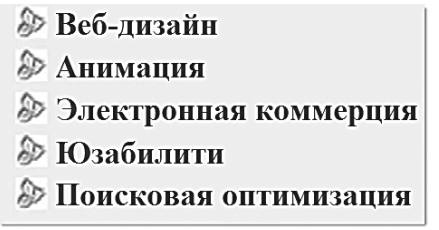


Рис. 6.20. Маркеры списка заменены изображениями

Вертикальная панель навигации с неупорядоченным списком

На рис. 6.21 показана область навигации веб-страницы (см. файл *Примеры\Глава_06\twocolumn3.html* на прилагающемся к книге диске), использующая неупорядоченный список для организации ссылок навигации. HTML-код следующий:

```
<ul>
<li><a href="index.html">Главная</a></li>
<li><a href="menu.html">Меню</a></li>
<li><a href="directions.html">Адреса</a></li>
<li><a href="contact.html">Контакты</a></li>
</ul>
```

Рис. 6.21. Навигация в виде неупорядоченного списка

Конфигурирование с помощью CSS

Теперь, когда наш код семантически верен, как можно улучшить внешний вид страницы? Применим CSS, чтобы избавиться от маркера списка. Также следует убедиться, что наши специальные стили применяются только к неупорядоченному списку в области навигации (внутри идентификатора `leftcolumn`), поэтому используем контекстный селектор. Код CSS для задания списка, отображенного на рис. 6.22, следующий:

```
#leftcolumn ul { list-style-type: none; }
```

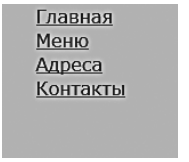


Рис. 6.22. Маркеры списка удалены с помощью CSS

Удаление подчеркивания с помощью CSS

Свойство `text-decoration` меняет вид текста в браузере. Оно чаще всего используется для устранения подчеркивания гиперссылок навигации (`text-decoration: none;`).

Код CSS для конфигурирования списка, показанного на рис. 6.23 (см. файл *Примеры\Глава_06\twocolumn4.html* на прилагающемся к книге диске), удаляющий подчеркивание гиперссылки в области навигации (внутри идентификатора `leftcolumn`), приведен ниже:

```
#leftcolumn a { text-decoration: none; }
```

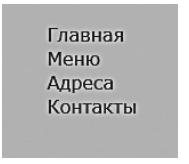


Рис. 6.23. Применено свойство CSS `text-decoration`

Горизонтальная панель навигации с неупорядоченным списком

Вы можете задать вопрос, как применить этот метод к горизонтальному навигационному меню. Это может быть сделано с помощью CSS! Элементы списка — это блочные элементы. Чтобы выстроить их по горизонтали, их следует сконфигурировать как строчные элементы. ***Свойство*** **`display`** задает, как браузер визуализирует или отображает элемент веб-страницы. Список наиболее распространенных значений представлен в табл. 6.5.

Таблица 6.5. Свойство `display`

Значение	Назначение
`none`	Элемент не отображается
`inline`	Элемент отображается как строчной на той же строке, что и окружающий его текст и/или элементы
`block`	Элемент отображается как блок с верхним и нижним полями

На рис. 6.24 показана область навигации веб-страницы (см. файл *Примеры\Глава_06\navigation.html* на прилагающемся к книге диске), в которой распложенные горизонтально ссылки навигации организованы как неупорядоченный список. HTML-код следующий:

```
<div id="nav">
<ul>
<li><a href="index.html">Главная</a></li>
<li><a href="menu.html">Меню</a></li>
<li><a href="directions.html">Адреса</a></li>
<li><a href="contact.html">Контакты</a></li>
</ul>
</div>
```

Конфигурирование с помощью CSS

В данном примере был применен следующий код CSS:

- чтобы избавиться от маркера списка, к селектору элемента `ul` применяется свойство `list-style-type: none;`:

```
#nav ul { list-style-type: none; }
```

- чтобы расположить элементы списка горизонтально, а не вертикально, примените свойство `display: inline`; к селектору элемента `li`:

```
#nav li { display: inline; }
```

- для устранения подчеркивание гиперссылок примените свойство `text-decoration: none;` к селектору элемента `a`. Кроме того, чтобы добавить пространства между гиперссылками, примените к селектору элемента `a` свойство `padding-right: 10px`;

```
#nav a { text-decoration: none; padding-right:
10px; }
```

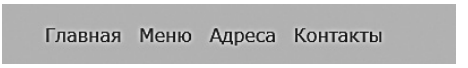


Рис. 6.24. Панель навигации в виде неупорядоченного списка

6.8. Интерактивность CSS с помощью псевдоклассов

Случалось ли вам, зайдя на сайт, обнаружить, что гиперссылки меняют свой цвет, когда вы устанавливаете на них указатель мыши? Это часто достигается с помощью специальных приемов с использованием ***псевдоклассов*** CSS, которые могут применяться для добавления к селектору специального эффекта. В таблице 6.6 приведены пять наименований псевдоклассов, которые могут применяться к элементам привязки.

Таблица 6.6. Наиболее часто используемые псевдоклассы CSS

Псевдокласс	Назначение
`link`	Состояние по умолчанию для непосещенной гиперссылки
`visited`	Состояние по умолчанию для посещенной гиперссылки
`focus`	Работает, когда гиперссылка является выделенной (например, после нажатия клавиши **Tab** на клавиатуре)
`hover`	Гиперссылка срабатывает при наведении указателя мыши
`active`	Определяет вид ссылки во время нажатия

Обратите внимание на порядок, в котором перечисляются псевдоклассы в табл. 6.6. Псевдоклассы для элементов привязки должны программироваться именно в этом порядке (хотя можно опускать один или

несколько из них). Если расположить псевдоклассы в другом порядке, они не будут работать должным образом. Обычно псевдоклассы `:hover`, `:focus` и `:active` конфигурируются одинаково.

Чтобы применить псевдокласс, укажите его в коде после селектора. Приведенный ниже образец кода конфигурирует текстовую гиперссылку, которая изначально имеет красный цвет. В коде также предусмотрен псевдокласс `hover`, изменяющий вид гиперссылки при наведении на нее указателя мыши таким образом, что исчезает подчеркивание и изменяется цвет.

```
a:link { color: #ff0000; }
a:hover { text-decoration: none;
color: #000066; }
```

На рис. 6.25 приведена часть веб-страницы, использующей подобную технологию. Обратите внимание на то, что когда указатель мыши находится на ссылке «Печать страницы», цвет текста изменяется и подчеркивание исчезает. Большинство современных браузеров поддерживает псевдоклассы CSS.

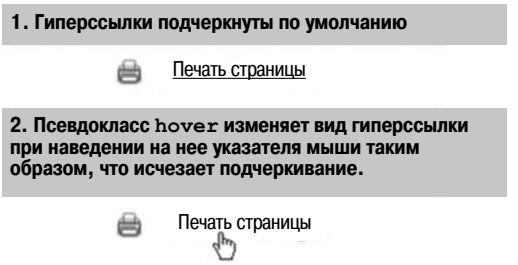


Рис. 6.25. Использование псевдокласса `hover`

Практическое задание 6.4

В этом практическом задании вы будете использовать псевдоклассы для создания интерактивных ссылок и конфигурировать несколько страниц, содержащих гиперссылки, выполненные в разных стилях.

Создайте папку с именем *hover*. Скопируйте в нее файлы *lighthouseisland.jpg, lighthouselogo.jpg* и *starter3.html* из папки *Глава_06* на прилагающемся к книге диске. Просмотрите веб-страницу в браузере. Она должно выглядеть как на рис. 6.26. Обратите внимание, что необходимо сконфигурировать область навигации.

Запустите текстовый редактор и откройте файл *starter3.html*. Сохраните его под именем *index.html* в папку *hover.*

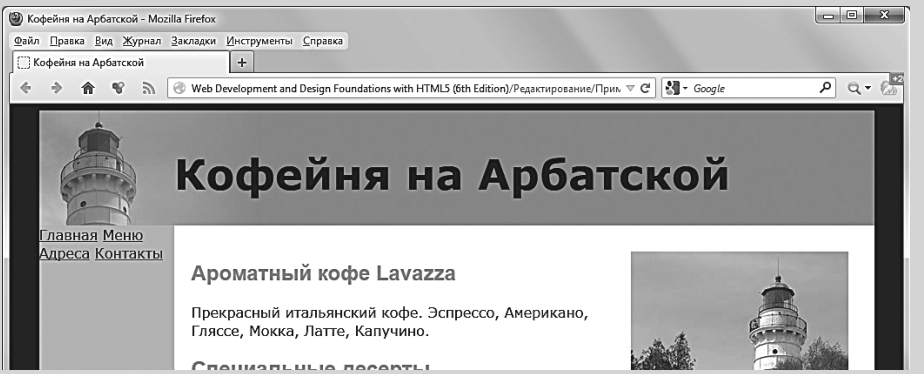


Рис. 6.26. В этом макете в две колонки необходимо создать область навигации

1. Просмотрите код этой страницы, использующей макет в две колонки. Изучите идентификатор `leftcolumn` и измените код, чтобы конфигурировать область навигации в виде неупорядоченного списка.

```
<ul>
<li> <a href="index.html"> Главная </a> </li>
<li> <a href="menu.html"> Меню </a> </li>
<li> <a href="directions.html">Адреса</a> </li>
<li> <a href="contact.html"> Контакты</a> </li>
</ul>
```

   Давайте добавим CSS к глобальным стилям, чтобы конфигурировать элементы неупорядоченного списка идентификатора `leftcolumn`. Удалите маркер списка и установите отступ в 10 пикселов.

```
#leftcolumn ul { list-style-type: none;
padding: 10px; }
```

2. Далее, добавьте основные элементы интерактивности с помощью псевдоклассов:

- задайте элементам привязки идентификатора `leftcolumn` отступ шириной в 10 пикселов, используйте полужирный шрифт и не отображайте подчеркивание:

```
#leftcolumn a { text-decoration: none;
padding: 10px;
font-weight: bold; }
```

- с помощью псевдоклассов конфигурируйте элементы привязки идентификатора `leftcolumn`, чтобы отображать белым (`#FFFFFF`) текстом непосещенные гиперссылки, светло-серым (`#eaeaea`) текстом посещенные гиперссылки и темно-синим (`#000066`) текст ссылки, на которую наведен указатель мыши:

```
#leftcolumn a:link { color: #ffffff; }
#leftcolumn a:visited { color: #eaeaea; }
#leftcolumn a:hover { color: #000066; }
```

Сохраните страницу и протестируйте ее в браузере. Наведите указатель мыши на панель навигации и заметьте, как меняется цвета текста. Ваша страница должна быть похожа на показанную на рис. 6.27 (см. файл *Примеры\Глава_06\hover\index.html* на прилагающемся к книге диске).

Рис. 6.27. Псевдоклассы CSS позволяют реализовать интерактивную панель навигации

6.9. Практическое задание с макетом CSS в две колонки

У вас уже есть опыт работы с макетом в две колонки, организации ссылок навигации и виде неупорядоченного списка и конфигурирования псевдоклассов CSS. Давайте закрепим эти навыки в ходе выполнения практического задания.

Практическое задание 6.5

В этом практическом задании вы создадите новую версию главной страницы веб-сайта кофейни с разделом заголовка, шириной в две колонки, колонкой контента (левая) и навигации (правая), а также нижним колонтитулом под колонками. Блок-схема показана на рис. 6.28. Вы будет конфигурировать CSS во внешней таблице стилей. Создайте новую папку под именем *practice*. Скопируйте в нее файлы *starter4.html*, *lighthouseisland.jpg* и *lighthouselogo.jpg* из папки *Глава_06* на прилагающемся к книге диске.

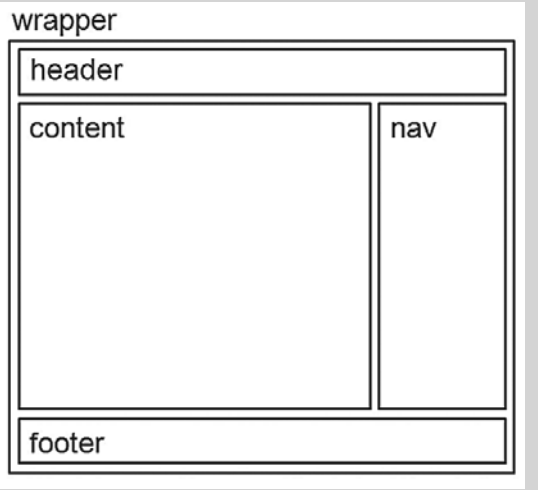


Рис. 6.28. Блок-схема макета в две колонки с областью заголовка

1. Запустите текстовый редактор и откройте файл *starter4.html*. Сохраните его с именем *index.html*. Добавьте в раздел заголовка веб-страницы элемент `link`, который свяжет этот файл с внешней таблицей стилей под именем *lighthouse.css*. Образец кода приведен ниже:

```
<link href="lighthouse.css" rel="stylesheet">
```

2. Сохраните файл *index.html*. Запустите текстовый редактор и создайте новый файл *lighthouse.css* в папке *practice*. Конфигурируйте CSS для разделов блок-схемы следующим образом:

- селектор элемента `body`: очень темный синий фон (`#00005D`) и шрифт без засечек Verdana, Arial или установленный по умолчанию:

```
body { background-color: #00005D;
       font-family: Verdana, Arial, sans-serif; }
```

- идентификатор `wrapper`: выравнивание по центру, ширина установлена на 80% окна просмотра браузера, минимальная ширина 960 пикселов, темно-синий цвет текста (`#000066`) и синий (`#B3C7E6`) цвет фона (*этот цвет будет отображаться в области навигации*):

```
#wrapper { margin: 0 auto;
width: 80%;
min-width: 960px;
background-color: #B3C7E6;
color: #000066; }
```

- идентификатор `header`: серо-синий (`#869DC7`) цвет фона, очень темный синий (`#00005D`) цвет текста, размер шрифта 150%, отступы сверху, справа и снизу шириной 10 пикселов, отступ слева шириной 155 пикселов и фоновое изображение *lighthouselogo.jpg*:

```
#header { background-color: #869DC7;
color: #00005D;
font-size: 150%;
padding: 10px 10px 10px 155px;
background-repeat: no-repeat;
background-image: url(lighthouselogo.jpg); }
```

- селектор элемента `h1`: конфигурируйте нижнее поле шириной 20 пикселов, чтобы избежать проблем при отображении страницы в браузере Internet Explorer:

```
h1 { margin-bottom: 20px; }
```

- идентификатор `nav`: обтекание по правому краю, ширина 150 пикселов, полужирный шрифт, расстояние между буквами 0,1 em:

```
#nav { float: right;
width: 150px;
font-weight: bold;
letter-spacing: 0.1em; }
```

- идентификатор `content`: белый цвет фона (`#FFFFFF`), черный цвет текста (`#000000`), отступы сверху и снизу шириной 10 пикселов, отступы слева и справа величиной 20 пикселов и свойство `overflow` со значением `auto`:

```
#content { background-color: #ffffff;
color: #000000;
padding: 10px 20px;
overflow: auto; }
```

- идентификатор `footer`: размер шрифта 70%, выравнивание текста по центру, 10 пикселов отступа, серо-голубой цвет фона (`#869DC7`), а также свойство `clear` со значением `both`:

```
#footer { font-size: 70%;
text-align: center;
padding: 10px;
background-color: #869DC7;
clear: both;}
```

Сохраните файл и просмотрите его в браузере. Страница должна быть похожа на показанную на рис. 6.29.

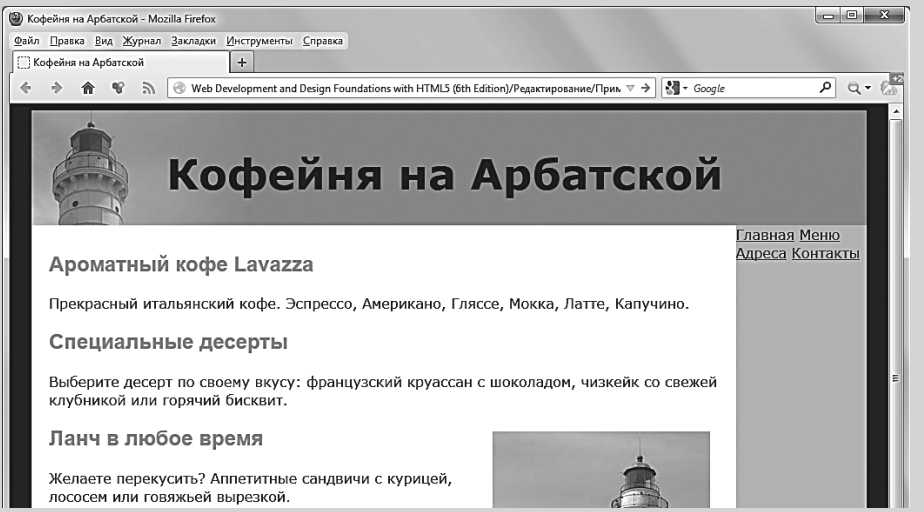


Рис. 6.29. Главная страница, основные разделы которой конфигурированы с помощью CSS

3. Продолжайте редактировать файл *lighthouse.css*, добавляя стили к селектору элемента `h2` и обтекаемого изображения. Конфигурируйте в селекторе элемента `h2` серо-голубой цвет текста (`#869DC7`) и шрифт Arial или другой шрифт без засечек. Задайте идентификатору `floatright` обтекание по правому краю с полем величиной 10 пикселов:

```
h2 { color: #869DC7;
font-family: Arial, sans-serif; }
#floatright { float: right;
margin: 10px; }
```

4. Продолжайте редактировать файл *lighthouse.css* и конфигурируйте область навигации:

- селектор элемента `ul`: удалите маркеры списка. Установите нулевые отступы и поля:

```
#nav ul { list-style-type: none; margin: 0; padding:
0; }
```

- селектор элемента `a`: без подчеркивания, отступ 20 пикселов, синий цвет фона (`#B3C7E6`) и сплошная нижняя граница белого цвета шириной 1 пиксел.

Примените свойство `display: block;`, чтобы посетитель вебстраницы, щелкнув в любом месте по «кнопке» привязки, мог активировать гиперссылку:

```
#nav a { text-decoration: none;
```

```
padding: 20px;
display: block;
background-color: #B3C7E6;
border-bottom: 1px solid #FFFFFF; }
```

Конфигурируйте псевдоклассы `:link`, `:visited` `:hover` следующим образом:

```
#nav a:link { color: #FFFFFF; }
#nav a:visited { color: #EAEAEA; }
#nav a:hover { color: #869DC7;
background-color: #EAEAEA; }
```

Сохраните ваши файлы. Откройте файл *index.html* в браузере. Наведите указатель мыши на область навигации и обратите внимание на интерактивность, как показано на рис. 6.30 (см. *Примеры\Глава_06\practice\index.html* на прилагающемся к книге диске).

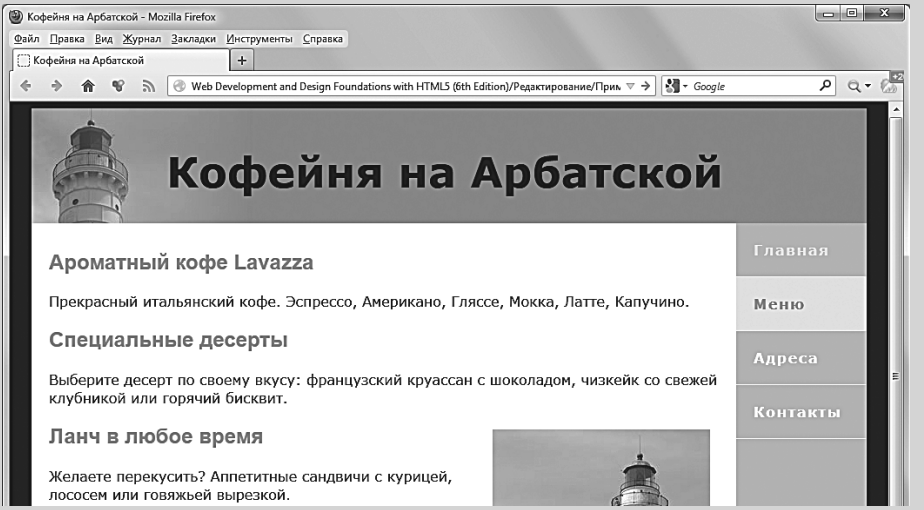


Рис. 6.30. Псевдоклассы CSS добавляют интерактивность на веб-страницу

6.10. Советы по отладке CSS

Использование CSS для компоновки веб-страниц требует терпения. Нужно время на то, чтобы к ним привыкнуть. Процесс устранения проблем в коде называется ***отладка***. Происхождение английской версии термина («debugging») связывают со временем, когда причиной сбоев в работе компьютера становились жуки (англ. «bug»), попавшие

внутрь устройства. Отладка может раздражать и занимать много времени. Одна из самых больших сложностей заключатся в том, что даже современные браузеры применяют CSS немного по-разному. Поддержка браузера меняется в зависимости от версии. Очень важно протестировать страницу в разных браузерах. Не стремитесь заставить страницу выглядеть в каждом их них совершенно одинаково, пусть она отображается немного по-разному. Учитывайте, что внешний вид страницы будет немного отличаться в различных браузерах. Несмотря на то, что поддержка стандартов CSS браузером Internet Explorer улучшается, все еще существуют различия в соответствии. Следующие проверки могут оказаться полезными, если код CSS не будет работать должным образом.

Проверьте синтаксические ошибки в HTML-коде

Проблемы в работе CSS могут возникнуть из-за неверного HTML-кода. Для проверки ошибок в синтаксисе HTML воспользуйтесь сервисом валидации разметки консорциума W3C: **validator.w3.org.**

Проверьте синтаксические ошибки в CSS

Иногда стили CSS не работают вследствие синтаксической ошибки. Для проверки синтаксиса CSS можно воспользоваться валидатором на сайте консорциума W3C **(jigsaw.w3.org/css-validator)**. Часто ошибка оказывается в строчках, находящихся над стилем, которые не работают так, как положено.

Создайте временные цвета фона

Бывает, что ваш код составлен правильно, но страница не отображается в браузере так, как вы предполагаете. Если временно задать хорошо различимые цвета фона, такие как красный или желтый, и снова протестировать страницу, будет легче увидеть, где кончается каждый блок.

Создайте временные рамки

Подобно заданию временных цветов фона, вы можете создать вокруг элемента красную рамку шириной в 3 пиксела — она будет бросаться в глаза и поможет быстрее найти ошибку.

Используйте комментарии для поиска незапланированных связей

Правила стилей и HTML-атрибуты, использованные далее в коде страницы, могут подавлять ранее использованные стили. Если ваши стили ведут себя не так, как вы ожидаете, попробуйте закомментировать некоторые стили (см. ниже) и проверить их небольшими группами. Затем для поиска места, где происходит сбой, добавляйте их один за другим в основной код. Наберитесь терпения и проверьте всю таблицу стилей таким образом. Обратите внимание но то, что комментарии игнорируются браузерами. Комментарии в таблицах стилей начинаются символами `/*`, а заканчиваются символами `*/`. Приведенный ниже комментарий — это пример документации, поясняющей назначение правила стилей:

```
/* Установите ширину полей 0*/
body {margin: 0
}
```

Комментарии могут занимать несколько строк. Комментарий на следующей странице начинается строкой выше описания класса `nav` и оканчивается строкой ниже этого описания. Эта запись заставляет браузер попускать класс `nav` при использовании таблиц стилей. Этот метод может быть полезным, когда вы экспериментируете с несколькими свойствами, и вам может потребоваться временно отключить правило стилей.

```
/* временный комментарий на период тестирования
.nav { text-decoration: none; }
*/
```

Распространенной ошибкой при использовании комментариев является размещение начальных символов `/*` без добавления соответствующих символов `*/` в конце комментария. В результате браузер воспринимает как комментарий *все*, что набрано после символа `/*`.

6.11. Веб-ресурсы, посвященные CSS

Эта глава познакомила вас с использованием таблиц стилей CSS для создания структуры страницы и положила начало изучению этой технологии. Вам может быть будет приятно узнать, что вы не одиноки

в стремлении изучить CSS. Существует много веб-ресурсов, содержащих документацию и уроки, а также обеспечивающих поддержку этой технологии. Методы компоновки веб-страниц, изложенные в этой книге — это только введение в большую область. Существует огромное количество веб-сайтов, на которых излагается другое видение проблемы и другие методы использования CSS для компоновки страниц. Вот некоторые из полезных ссылок:

- большая коллекция структур веб-страниц, созданных с помощью CSS, и ссылок на учебные материалы: **glish.com/css**;
- коллекция макетов страниц: **www.bluerobot.com/web/layouts**;
- вертикальные и горизонтальные списки с помощью CSS :**css.maxdesign.com.au/listamatic**;
- каскадные таблицы стилей консорциума W3C: **www.www.w3.org/Style/CSS**;
- экспериментальная таблица совместимости CSS: **westciv.com/wiki/Experimental_CSS_compatibility_table**;
- таблица поддержки браузерами HTML5 и CSS3**: www.findmebyip.com/litmus**;
- сайт Питера Пауля Коха, посвященный изучению и устранению несовместимости браузеров, связанной с CSS и JavaScript: **www.quirksmode.org/css/contents.html**;
- таблица CSS3: **www.impressivewebs.com/css3-click-chart**;
- справочник по CSS ресурса SitePoint: **reference.sitepoint.com/css**.

6.12. Структурные элементы HTML5

HTML5 вводит ряд семантических структурных элементов, которые можно использовать для конфигурирования областей веб-страницы. Эти новые блочные элементы не являются полноценной заменой элемента `div`, но предназначены для более конструктивного использования вместе с `div` и другими элементами структуры документов веб-страницы, указывающего цель структурных областей.

Обычной практикой при верстке на языке HTML4 и XHTML является использование на веб-странице только одного элемента `h1` и конфигурирование элементов заголовка как структуры страницы. Структура в HTML5 создается иначе. Вместо элементов заголовка структура также задается элементами разделов (такими как `section`, `article`, `nav` и `aside`) с разными уровнями заголовков для каждого раздела. По-

пробуйте применить инструмент, доступный на сайте **gsnedders.html5.org/outliner**.

В этом разделе мы рассмотрим элементы HTML5 — `header`, `hgroup`, `nav`, `footer`, `section`, `article`, `aside` и `time`. Давайте сначала поближе познакомимся с элементами `header`, `hgroup`, `nav` и `footer`.

Элемент `header`

В ***элементе*** **`header`** содержатся заголовки документа веб-страницы или области документа, к примеру, раздела или статьи. Этот элемент начинается с тега <header> и заканчивается тегом </header> . Элемент `header` — это блочный элемент и, как правило, содержит один или несколько уровней заголовков (с `h1` до `h6`), а при необходимости и элемент `hgroup`.

Элемент `hgroup`

Элемент **`hgroup`** группирует элементы заголовков разных уровней и полезен, если в области логотипа веб-страницы содержится не только название сайта, но и слоган (фраза, которая определяет и отражает суть бизнеса, например, слоган компании Nokia[1] «Ваш образ жизни. Ваш стиль работы. Ваш вкус»). Элемент `hgroup` — блочный. Он начинается с тега <hgroup> и заканчивается тегом </hgroup>. Когда в элементе `hgroup` несколько заголовков разных уровней, только первый заголовок помещается в структуру страницы.

Элемент `nav`

Элемент **`nav`** содержит раздел с гиперссылками навигации. Блочный элемент `nav` начинается с тега <nav> и заканчивается тегом </nav>.

Элемент `footer`

Элемент **`footer`** включает в себя содержимое нижнего колонтитула веб-страницы, раздел, статью, абзац или даже элемент цитирования. Блочный элемент `footer` начинается с тега <footer> и заканчивается

[1] www.nokia.com/ru-ru

тегом </footer>. Помните, что эти новые элементы HTML5 поддерживаются не всеми браузерами. Тем не менее вы можете начать практиковаться применять их уже сегодня.

Практическое задание 6.6

В этом практическом задании вы начнете с двухколоночной главной страницы кофейни, созданной в практическом задании 6.5 (см. рис. 6.30), и измените ее, чтобы применить структурные элементы HTML5. Вы также добавите слоган в область заголовка. Создайте новую папку с именем *structure*.

Скопируйте в нее файлы *index.html, lighthouse.css, lighthouseisland.jpg* и *lighthouselogo.jpg* из папки на прилагающемся к книге диске *Примеры\Глава_06*.

1. Запустите текстовый редактор и откройте файл *index.html.* Замените элемент div, которому присвоен идентификатор header, на элемент header языка HTML5, содержащий элемент hgroup. Кроме того, с помощью элемента h2 добавьте текст «лучший кофе в Москве». Новый код:

```
<header>
<hgroup>
<h1>Кофейня на Арбатской</h1>
<h2>лучший кофе в Москве</h2>
</hgroup>
</header>
```

2. Замените открывающий и закрывающий теги элемента div, которому присвоен идентификатор nav, на элемент nav HTML5.
3. Замените открывающий и закрывающий теги элемента div, которому присвоен идентификатор footer, на элемент footer HTML5.
4. Напомним, что в главе 4 рассматривались элементы HTML5 figure и figcaption. Добавьте в код элемент figure, содержащий изображение. Добавьте элемент figcaption, отображающий текст «Копия старого маяка 1870 года» в контейнере figure под изображением. Удаление из элемента изображения id="floatright". Вы конфигурируете обтекание всего элемента figure в шаге 9. Код показан ниже:

```
<figure>
<img src="lighthouseisland.jpg" width="250"
height="355"
alt="Маяк">
```

```
<figcaption>Копия старого маяка 1870 года</
figcaption>
</figure>
```

5. Сохраните файл *index.html*.
6. Откройте файл *lighthouse.css* в текстовом редакторе. Измените CSS, чтобы использовать селекторы HTML-элементов для элементов `header`, `nav` и `footer`. Замените селектор `#header` на селектор HTML-элемента `header`. Замените во всех случаях селектор `#nav` на селектор HTML-элемента `nav`. Замените селектор `#footer` на селектор HTML-элемента `footer`.
7. Конфигурируйте CSS для элементов `h1` и `h2` в разделе заголовка. Примените контекстные селекторы HTML. Установите величину нижнего поля элемента `h1`, равную 0. Для элемента `h2` установите отступ справа шириной 20 пикселов, никаких полей, а также начертание текста курсивом, размер текста .80 em, а цвет `#00005D`. Код CSS следующий:

```
header h1 { margin-bottom: 0; }
header h2 { margin: 0;
padding-right: 20px;
font-size: .80em;
font-style: italic;
text-align: right;
color: #00005D; }
```

8. Оптимизируйте оставшиеся правила CSS. Удалите правило стиля `#floatright` и правило стиля элемента `h1`.
9. Конфигурируйте правила стилей для селекторов элементов `figure` и `figcaption`. Селектор элемента `figure` установлен следующий: обтекание по правому краю, ширина 260 пикселов и поля 10 пикселов. Селектор элемента `figcaption` установлен на отображение мелкого текста, начертанного курсивом и выровненного по центру. Код CSS:

```
figure { float: right;
width: 260px;
margin: 10px; }
figcaption { text-align: center;
font-size: .80em;
font-style: italic; }
```

10. Актуальные версии браузеров Internet Explorer, Firefox, Safari, Chrome и Opera поддерживает элементы HTML5, прописанные в данном коде. Но как помочь более старым браузерам понять

код HTML5? Один из способов — применить CSS, чтобы конфигурировать селекторы элементов HTML5 как блочные. На момент написания данной книги популярные браузеры (за исключением Internet Explorer 8 и более ранних версий) правильно применяли это определение стилей. Добавьте следующий код CSS:

```
header, nav, footer, figure, figcaption { display:
block; }
```

11. Сохраните файл *lighthouse.css*.
12. Просмотрите страницу *index.html* в современном браузере. Она должна быть похожа на показанную на рис. 6.31 (см. файл *Примеры\Глава_06\structure\index.html* на прилагающемся к книге диске).

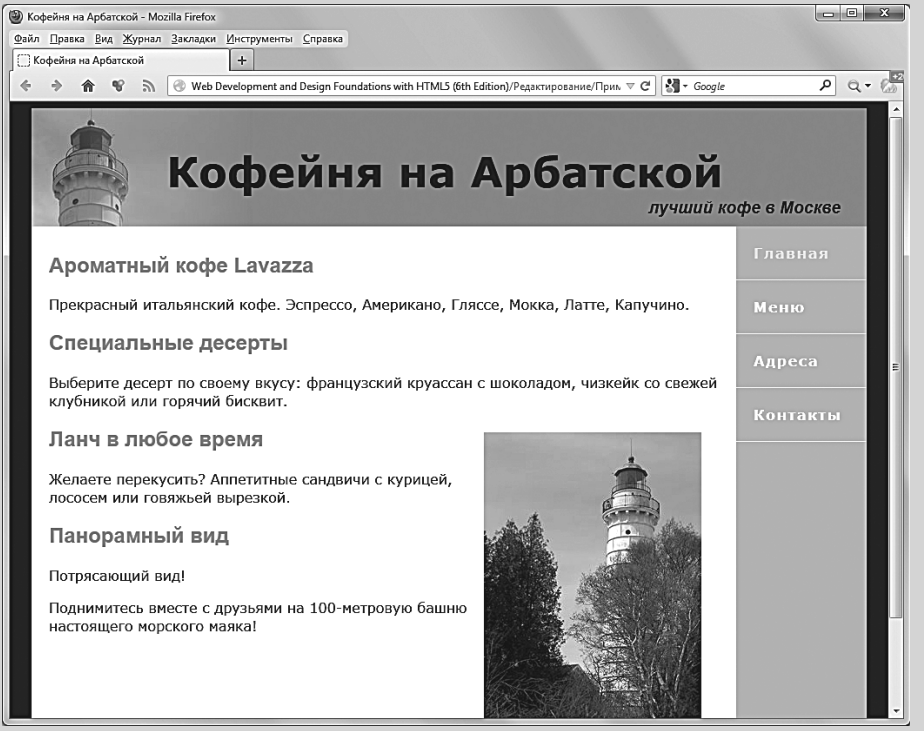


Рис. 6.31. На этой веб-странице были применены структурные элементы HTML5

Выполняя практическое задание, вы могли заметить, что некоторые имена идентификаторов, используемые для стандартных областей страницы, такие как `header`, `nav` и `footer`, также являются именами новых структурных элементов HTML5. Это отличный способ подготовиться к работе с HTML5, даже если вы все еще верстаете на XHTML! Привыкайте к новым именам элементов HTML5, задавая элементы `div`, где эти

имена будут использоваться как значения идентификатора или класса. Потом, позже, когда все будут верстать только на HTML5, у вас будет преимущество! На сайте **HTML5gallery.com** содержится множество примеров того, как HTML5 используется сегодня во Всемирной паутине. Далее мы рассмотрим элементы HTML5, `section`, `article`, `aside` и `time`.

Элемент `section`

Элемент `section` содержит «раздел» документа, к примеру, главу или тему. Элемент `section` — блочный. Он начинается с тега `<section>` и заканчивается тегом `</section>`. Элемент `section` может включать в себя элементы `header`, `footer`, `article`, `aside`, `div` и прочие, необходимые для отображения контента. В нем также могут содержаться другие элементы `section`.

Элемент `article`

Элемент `article` содержит независимый набор контента, например, сообщение в блоге, комментарий или статью электронного журнала, который может существовать сам по себе. Блочный элемент `article` начинается с тега `<article>` и заканчивается тегом `</article>`. Это блочный элемент, который может включать в себя элементы `header`, `footer`, `section`, `aside`, `div` и прочие, необходимые для отображения контента.

Элемент `aside`

Элемент `aside` — блочный и включает в себя боковую панель, заметки и прочий второстепенный контент. Элемент `aside` начинается с тега `<aside>` и заканчивается тегом `</aside>`.

Элемент `time`

Элемент `time` представляет собой дату или время, и может быть полезен при указании даты публикации статьи или сообщения в блоге. Строчный элемент `time` начинается с тега `<time>` и заканчивается тегом `</time>`. Необязательный атрибут даты можно использовать для определения календарной даты и/или времени в форме, понятном компьютеру. Указывайте дату в формате ГГГГ-ММ-ДД. Для указания времени применяйте 24-часовой формат или ЧЧ:ММ. Дополнительные варианты синтаксиса можно посмотреть на сайте консорциума W3C[1].

[1] www.w3.org/TR/HTML-markup/time.html

Практическое задание 6.7

В этом практическом задании вы начнете работать со страницей кофейни (рис. 6.31) и измените ее содержимое, применив формат блога, структурированный с помощью новых элементов HTML5 — `article`, `header`, `aside` и `time`. Создайте новую папку с именем *blog*. Скопируйте в нее файлы *index.html, lighthouseisland.jpg* и *lighthouselogo.jpg* из папки *Примеры\ Глава_06\structure* на прилагающемся к книге диске.

1. Запустите текстовый редактор и откройте файл *index.html*. Удалите текст и содержимое элемента `div`, которому присвоен идентификатор `content`. Замените их на элемент HTML5 `section`, присвойте ему идентификатор `content`, содержащий элемент `h1` с текстом «Блог за чашкой кофе» и две статьи блога. С помощью элементов `section`, `article`, `header`, `aside` и `time` создайте два сообщения блога на главной странице. Конфигурируйте новый раздел, как показано ниже и сохраните файл.

```
<section id="content">
<h1>Блог за чашкой кофе</h1>
<article>
<header><h1>День Св. Валентина</h1></header>
<time datetime="2011-02-01">14 февраля 2012</time>
<aside>Прекрасные скидки каждый праздник!</aside>
<p>Ко дню всех влюбленных капучино с сердечком дешевле на 30%!</p>
</article>
<article>
<header><h1>Новый вкус!</h1></header>
<time datetime="2011-01-11">25 декабря 2011</time>
<p>Самый вкусный и самый рождественский кофе-гляссе с нотками апельсина! Только в рождественские и новогодние дни 2011-2012 года!</p>
</article>
</section>
```

2. Откройте файл *lighthouse.css* в текстовом редакторе. Напомним, что элемент `aside` содержит контент, второстепенный по отношению к основному содержанию. Конфигурируйте CSS, чтобы отобразить элемент `aside` справа (примените свойство `float`) в светло-сером прямоугольнике примерно 120 пикселов в шири-

ну, с левым полем величиной 10 пикселов, отступами шириной 5 пикселов, размером шрифта 80% и с эффектом тени.

```
aside { float: right;
background-color: #eaeaea;
width: 120px;
padding: 5px;
margin-left: 10px;
font-size: 80%;
-webkit-box-shadow: 5px 5px 5px #828282;
-moz-box-shadow: 5px 5px 5px #828282;
box-shadow: 5px 5px 5px #828282; }
```

Сохраните файл *lighthouse.css*. Запустите современный браузер и протестируйте в нем страницу *index.html*. Вы, возможно, будете удивлены, увидев, как отображаются заголовки раздела и статьи. Нам нужно указать в коде определения CSS специально для заголовка статьи и для элемента `h1` в идентификаторе `content`.

Используйте контекстные селекторы. Запустите текстовый редактор и снова откройте файл *lighthouse.css*. Добавьте следующий код CSS:

```
article header { background-color: #FFFFFF;
background-image: none;
padding: 0;
font-size: 80%;}
#content h1 { color: #869dc7;
font-family: Arial, sans-serif;
font-size: 1.5em;}
```

3. Вспомните, что в практическом задании 6.6 использовалась техника CSS, позволяющая старым браузерам отображать элементы HTML5 должным образом. Добавьте в код CSS селекторы HTML-элементов `section`, `article` и `aside`. Новое правило стиля:

```
header, nav, footer, figure, figcaption, section,
article, aside { display: block; }
```

4. Сохраните файл *lighthouse.css.* Просмотрите страницу *index.html* в современном браузере. Она должна выглядеть так, как показано на рис. 6.32 (см. *Примеры\Глава_06\blog\index.html* на прилагающемся к книге диске).

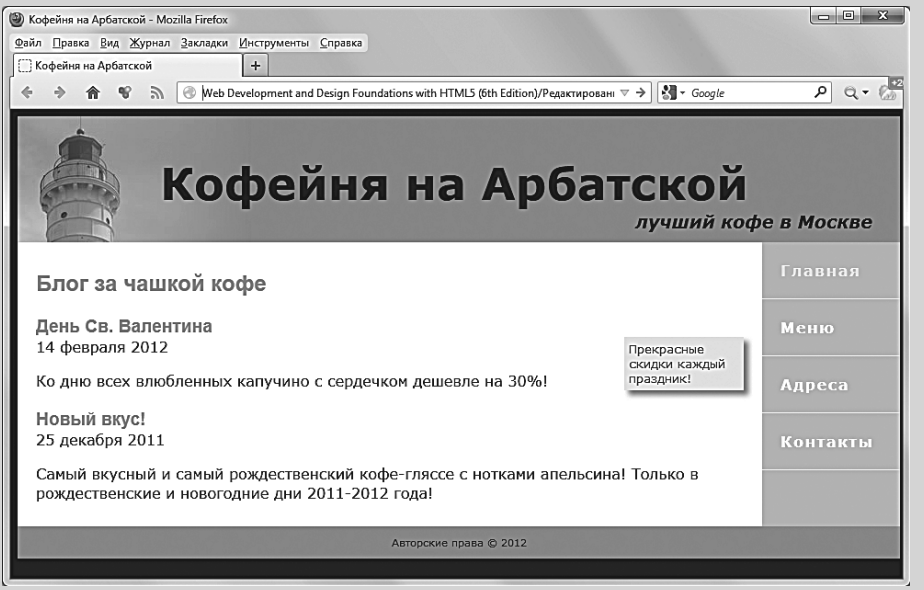


Рис. 6.32. На этой странице были использованы HTML5-элементы `section`, `article` и `aside`

HTML5 и современные браузеры

Internet Explorer 9 и текущие версии браузеров Safari, Chrome, Firefox и Opera предоставляют хорошую поддержку структурных элементов HTML5. Проблема в том, что многие люди до сих пор пользуются более ранними версиями браузеров. Если вы хотите начать работать с HTML5 уже сегодня, можно применить один из двух подходов: консервативный, прямой подход и прогрессивный подход для увлеченных, требующий больших усилий и навыков использования JavaScript (см. главу 11).

Консервативный подход к использованию HTML5

Чтобы добиться лучшей совместимости, пишите код с использованием синтаксиса HTML5 или XHTML и старайтесь избегать новых элементов HTML5. Вместо этого используйте названия новых элементов в качестве имен классов или идентификаторов. Так вы сможете познакомиться с новыми именами элементов и вам будет легче обновить страницы на HTML5 позже. Пример этой техники можно найти в файле *Примеры\Глава_06\blog\all.html*. Со временем старыми браузерами станут пользоваться реже, и все будет в порядке!

Прогрессивный подход к использованию HTML5

Для начала возьмите за образец практические задания 6.6 и 6.7: верстайте с использованием новых элементов HTML5 и включайте код CSS, задающий в старых, не поддерживающих HTML5, браузерах отображение элементов `header`, `figure`, `figcaption`, `footer`, `nav`, `section`, `article` и `aside` как блочных (примените свойство `display: block;`). Этот подход будет хорошо работать во всех браузерах, кроме Internet Explorer 8 и более ранних версий.

А что же делать с браузером Internet Explorer 8 и его более ранними версиями? Реми Шарп предлагает решение[1] для улучшения поддержки Internet Explorer 8 и более ранних версий. В этом методе применяются условные комментарии, которые поддерживаются только Internet Explorer и игнорируются другими браузерами. Условные комментарии инструктируют Internet Explorer интерпретировать определения JavaScript, позволяющие распознавать и обрабатывать CSS для нового селектора элемента HTML5. Шарп загрузил скрипт в виде проекта на сервис Google Code и сделал его доступным для всех. Добавьте следующий код в раздел заголовка веб-страницы после CSS, чтобы правильно отображать код HTML5 в браузере Internet Explorer 8 и его более ранних версиях:

```
<!--[if lt IE 9]>
<script src=" http://HTML5shim.googlecode.com/svn/
trunk/HTML5.js ">
</script>
<![endif]-->
```

В чем недостаток этого подхода? Знайте, что посетители вашей веб-страницы, пользующиеся браузером Internet Explorer 8 и более ранними версиями, могут видеть предупреждающее сообщение и, чтобы этот метод сработал, у них должно быть разрешено выполнение сценариев JavaScript. Откройте пример (см. *Примеры\Глава_06\blog\today.html* на прилагающемся к книге диске).

[1] remysharp.com/2009/01/07/HTML5-enabling-script

Глава 7
ДОПОЛНИТЕЛЬНЫЕ СВЕДЕНИЯ О ССЫЛКАХ, МАКЕТАХ И МОБИЛЬНЫХ УСТРОЙСТВАХ

Цели главы

В этой главе вы узнаете следующее:

- как создавать относительные гиперссылки на веб-страницы, расположенные во вложенных папках внутри каталога сайта;
- как создавать гиперссылки на именованный фрагмент, находящийся внутри страницы;
- как создавать изображения с помощью CSS-спрайтов;
- как задать макет в три колонки, используя CSS;
- как создавать CSS для подготовки страницы к печати;
- как создавать CSS для отображения страницы на мобильных устройствах;
- как применять медиазапросы CSS3 для оптимизации страниц для просмотра на мобильных устройствах.

Теперь, когда вы уже имеете некоторый опыт верстки кода HTML и CSS, вы готовы к изучению в этой главе еще нескольких методов, включающих использование относительных гиперссылок, гиперссылок на именованный фрагмент внутри страницы, CSS-спрайтов, макета страницы в три колонки, применение правил стилей для печати и отображения веб-страниц в мобильном веб-браузере, а также использование медиазапросов CSS3 для оптимизации страниц для просмотра на мобильных устройствах.

7.1. Другой подход к гиперссылкам

Гиперссылки связывают информацию во Всемирной паутине. В этой главе мы вернемся к теме гиперссылок, и вы научитесь программировать относительные ссылки, использовать атрибут `target`, позволя-

ющий открывать веб-страницу в новом окне браузера, а также создавать гиперссылки, работающие внутри страницы.

Дополнительная информация об относительных ссылках

Как уже говорилось в главе 2, относительные ссылки используются для указания на страницы внутри вашего сайта. Вы уже создавали относительные ссылки для отображения страниц, расположенных внутри одной и той же папки сайта.

Зачастую бывает необходимо сделать ссылку на файл, находящийся в другой папке сайта. Рассмотрим для примера веб-сайт, предоставляющий информацию по уходу за собаками и предлагающий как услуги, так и товары.

Для удобства разработчик этого веб-сайта создал отдельные папки для файлов, касающихся услуг, и файлов, относящихся к товарам. Структура папок этого сайта представлена на рис. 7.1.

Рис. 7.1. В структуру папок сайта включены папки *images*, *services* и *products*

Примеры относительных ссылок

Вспомните, что если ссылка указывает на файл, находящийся в той же папке, что и страница, на которой расположена ссылка, значением атрибута `href` является имя файла. Например, для создания ссылки на

файл страницы *contact.html* с главной страницы (*index.html*) необходимо сверстать элемент привязки следующим образом:

```
<a href="contact.html">Контакты</a>
```

Для создания относительной ссылки на файл, находящийся в папке, расположенной внутри папки текущей страницы, указывается имя папки и имя файла. Например, для того чтобы создать на главной странице (*index.html*) ссылку на файл страницы *collars.html*, находящийся в папке *products*, необходимо программировать элемент привязки следующим образом:

```
<a href="products/collars.html">Ошейники</a>
```

Из рис. 7.1 видно, что файл *collars.html* находится в папке нижележащего уровня относительно папки *groomer*, а файл главной страницы сайта *index.html* находится в папке *groomer*. При создании ссылки на файл, расположенный в папке более высокого уровня, используйте запись `../`. Например, код для создания ссылки на главную страницу (*index.html*) со страницы *collars.html* будет иметь вид:

```
<a href="../index.html">Главная</a>
```

Если в ссылке на страницу, находящуюся на том же уровне, что и текущая страница, для атрибута `href` используется запись `../`, то происходит переход к папке более высокого уровня, а затем вниз, к требующейся папке. Например, для создания ссылки на страницу *bathing.html*, расположенную в папке *services*, со страницы *collars.html*, расположенной в папке *products*, необходимо указать:

```
<a href="../services/bathing.html">Купание собак</a>
```

Не беспокойтесь о том, что использование записи `../` и создание ссылок на файлы, находящиеся в разных папках, кажется новым и непривычным, в большинстве упражнений в этой книге вы будете использовать либо абсолютные ссылки, либо ссылки на страницы, расположенные в текущей папке.

Идентификаторы фрагментов

Браузеры начинают отображать веб-страницу с верхней части документа. Однако иногда бывает необходимо обеспечить переход к определенной части веб-страницы. Это может быть сделано с по-

мощью гиперссылки на ***идентификатор фрагмента*** (иногда он называется именованный фрагмент или id фрагмента), который просто является HTML-элементом с назначенным ему идентификатором. Эта технология часто используется для списков часто задаваемых вопросов (FAQ).

В программировании идентификаторов фрагментов участвуют два компонента:

1. Элемент, определяющий статус ***именованного фрагмента*** веб-страницы. Этому элементу должен быть назначен идентификатор. Например, `<div id="content">`.
2. Элемент привязки, являющийся ссылкой на этот именованный фрагмент веб-страницы.

Ссылки на именованный фрагмент можно часто встретить на длинных веб-страницах. Это может быть гиперссылка «В начало», на которую посетитель нажимает, чтобы быстро вернуться к началу страницы и средствам навигации по сайту.

Другое применение идентификаторов фрагмента — обеспечение доступности. Чтобы разобраться, как использовать эти компоненты, предположим, что для обеспечения доступности веб-страница имеет идентификатор фрагмента, указывающий на действительное начало контента веб-страницы. Когда посетитель щелкает по гиперссылке «Перейти к содержимому», браузер переходит к идентификатору фрагмента, и в его окне отображается область контента. Такие ссылки «Перейти к содержимому» и «Перейти в область навигации» позволяют программам экранного доступа пропускать ненужные навигационные ссылки (рис. 7.2). Создание внутренних ссылок происходит в два этапа.

1. **Создание целевого фрагмента.** Создание идентификаторов фрагментов «Перейти к содержимому» путем конфигурирования элемента `div`, содержащего контент страницы с идентификатором, например:

   `<div id="content">`.

2. **Создание ссылки на целевой фрагмент.** В той позиции страницы, куда вы хотите поместить ссылку на контент, создайте элемент привязки. Добавьте атрибут `href` и символ `#` (иногда называемый «решетка») перед именем идентификатора фрагмента. Код ги-

перссылки на именованный фрагмент с именем "content" будет иметь вид:

```
<a href="#content">Перейти к содержимому></a>
```

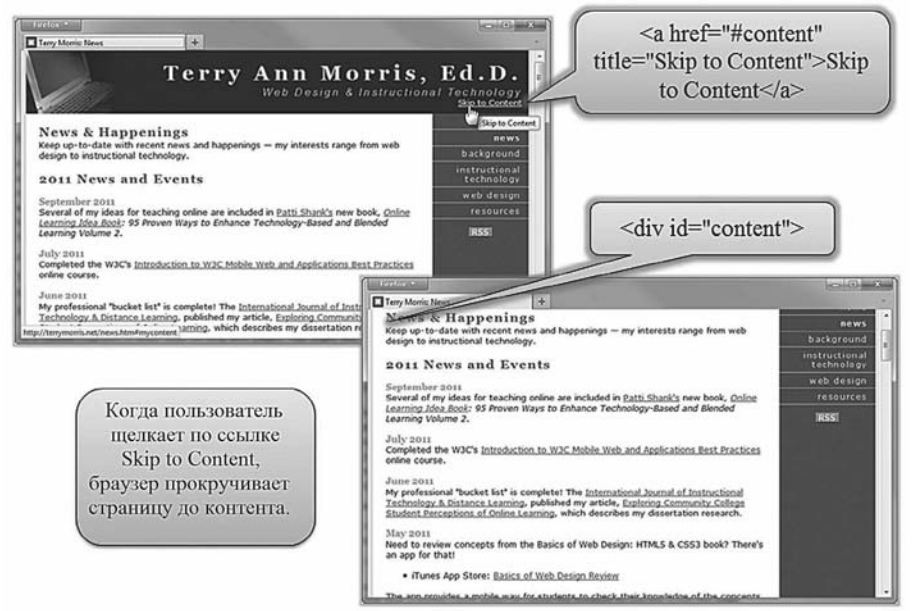


Рис. 7.2. Действие ссылки **Skip to Content**

Символ # означает, что браузер должен искать идентификатор на этой странице. Если вы забудете поставить этот знак, браузер вместо идентификатора будет на этой странице искать внешний файл.

Практическое задание 7.1

На этом практическом задании вы будете работать с идентификаторами фрагментов. Найдите на диске, прилагающемся к книге, файл *Примеры\Глава_07\starter1.html*. На рис. 7.3 представлен снимок части этой веб-страницы. Откройте файл *starter1.html* в программе Блокнот (Notepad) и сохраните под именем *favorites.html.* Рассмотрите код и обратите внимание, что в верхней части страницы расположен неупорядоченный список, состоящий из пунктов **Хобби**, **HTML**, **Веб-дизайн для мобильных устройств** и **Профсоюзы**, причем этот список соответствует тексту, содержащемуся в расположенных ниже элементах `h2`. После каждого элемента `h2` указан список от-

дельных подтем, касающихся данной темы, и URL-адрес для каждой категории. Такая структура может быть полезной посетителям сайта, поскольку, щелкнув по соответствующей ссылке, они могут сразу перейти к информации по данной теме. Это как раз тот случай, когда можно использовать ссылки на идентификатор фрагмента!

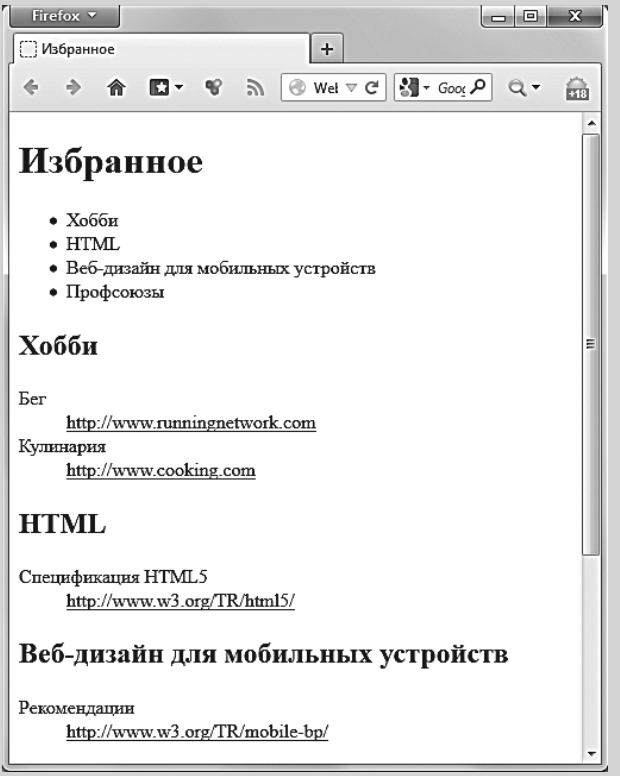


Рис. 7.3. Вы будете создавать гиперссылки для перехода на идентификатор фрагмента

Измените код страницы следующим образом:

1. В области определений создайте именованные фрагменты для каждого элемента `<h2>`. Пример:

   `<h2 id="hobbies">Хобби</h2>`.

2. Создайте гиперссылки для каждой позиции неупорядоченного списка таким образом, чтобы каждый элемент списка был связан с соответствующим элементом `<h2>`.
3. Добавьте именованные фрагменты в верхней части страницы.
4. В нижней части страницы *favorites.html* создайте ссылку для перехода в начало веб-страницы.

Сохраните файл и проверьте его в браузере. Сравните его с файлом *Примеры\Глава_07\favorites.html* на диске, прилагающемся к книге.

Иногда требуется создать ссылку на именованный фрагмент, находящийся на другой веб-странице. Для этого нужно указать символ `#` и значение `id` идентификатора фрагмента после имени файла в элементе привязки. Чтобы создать ссылку на фрагмент **Профсоюзы** (пусть имя этого именованного фрагмента `proof`) в коде любой другой страницы этого сайта, нужно указать следующее:

```
<a href="favorites.html#proof">Профсоюзы</a>
```

ЧаВо

ПОЧЕМУ НЕКОТОРЫЕ ИЗ МОИХ ССЫЛОК НА ИМЕНОВАННЫЕ ФРАГМЕНТЫ НЕ РАБОТАЮТ?

Веб-браузеры заполняют окно (окно просмотра) контентом веб-страницы и прокручивают содержимое окна, чтобы отобразить именованный объект в верхней части окна. Однако если именованный фрагмент находится рядом со ссылкой на него, перемещение не будет заметно. Попробуйте добавить пустые строки на странице (используйте элемент `br`). Сохраните файл и проверьте страницу еще раз.

Атрибут `target`

Вы, возможно, заметили в практическом задании 7.1, что когда пользователь щелкает мышью по гиперссылке, в том же окне браузера автоматически открывается новая веб-страница. ***Атрибут*** **`target`** может применяться к элементу привязки для открытия нового окна браузера. Например, код

```
<a href="http://yahoo.com" target="_blank">Yahoo!</a>
```

откроет главную страницу сайта Yahoo! в новом окне.

Обратите внимание, что вы не можете контролировать, откроется ли страница в новом окне или на новой вкладке. Это зависит от настроек браузера ваших посетителей.

Почему бы не создать и не протестировать пробную страницу? Наличие в записи атрибута `target` со значением `_blank` указывает на то, что страница будет открываться в новом окне браузера.

Практическое задание 7.2

На этом практическом задании вы будете работать с атрибутом `target`. Найдите на прилагающемся к книге диске документ *Примеры\Глава_07\favorites.* Откройте его в программе Блокнот (Notepad) и сохраните под именем *target.html*. Выберите одну из внешних гиперссылок, которую будете изменять. Затем добавьте атрибут `target="_blank"`, чтобы ссылка открывалась в новом окне или новой вкладке браузера. Код показан ниже:

```
<a href=" http://www.isoc.org" target="_blank"> http://
www.isoc.org </a>
```

Сохраните файл. Запустите браузер и протестируйте файл. Когда вы щелкнете по измененной гиперссылке, новая страница отобразится в новом окне браузера или вкладке. Вы можете сравнить вашу работу с примером на прилагающемся к книге диске (*Примеры\Глава_07\target.html)*.

Блочная привязка

Обычно чтобы конфигурировать фразы или даже отдельные слова как гиперссылки, используют элементы привязки. HTML5 предоставляет новые функции элемента привязки — блочная привязка (привязка блока).

С помощью блочной привязки можно сделать гиперссылкой один или несколько элементов (даже блочных, таких как `div`, `h1` или абзаца). См. пример на прилагающемся к книге диске (*Примеры\Глава_07\block.html*).

Гиперссылки голосовых вызовов и текстовых сообщений

Если на веб-странице отображается номер телефона, как было бы удобно человеку, использующему смартфон, иметь возможность нажать на него и позвонить или отправить SMS (текстовое сообщение). Можно очень легко создать такую гиперссылку.

В соответствии со спецификацией RFC 3966, можно настроить гиперссылку для голосового вызова с помощью следующей схемы: сначала укажите `tel:` в значении элемента `href`, а потом номер телефона. Вот так, например, можно настроить на веб-странице гиперссылку

для голосового вызова, чтобы мобильные браузеры могли ее использовать, — достаточно указать следующий код: `<a href="tel:888-555-5555">Позвоните 888-555-5555</a>`.

Спецификация RFC 5724 определяет, как создать ссылку для отправки текстовых сообщений. Ее можно реализовать, указав значение `sms:` элемента `href`, а затем номер телефона, как показано в следующем примере:

```
<a href="sms:888-555-5555">SMS 888-555-5555</a>.
```

Не все мобильные браузеры и устройства поддерживают описанные гиперссылки, но в будущем ожидается более широкое использование этой технологии. Вы получите возможность попрактиковаться в использовании схемы со значением `tel:` практических примерах к главе 7.

7.2. CSS-спрайты

Когда браузеры отображают веб-страницы, они выполняют отдельные запросы по протоколу HTTP каждого файла, используемого на странице, в том числе файлов *.css* и графических, например, с расширениями *.gif*, *.jpg* и *.png*. На каждый http-запрос тратится время и ресурсы. Как уже упоминалось в главе 4, спрайт — это графический файл, содержащий несколько небольших изображений. Использование правил CSS для конфигурирования небольших графических изображений, объединенных в спрайт, в качестве фоновых изображений для различных элементов веб-страницы называется методом ***CSS-спрайтов***. Он стал популярным благодаря Дэвиду Ши[1].

Метод CSS-спрайтов использует свойства CSS `background-image`, `background-repeat` и `background-position`, чтобы манипулировать расположением фонового изображения. Единый графический файл экономит время загрузки, так как браузеру необходимо сделать только один http-запрос на комбинированное изображение, а не несколько запросов на отдельные более мелкие изображения. На рис. 7.4 показан спрайт с двумя изображениями маяка на прозрачном фоне. Эти изображения настроены с помощью CSS как фоновые изображения ссылок навигации (рис. 7.5). Вы увидите спрайты в действии, когда будете выполнять следующее практическое задание.

[1] www.alistapart.com/articles/sprites

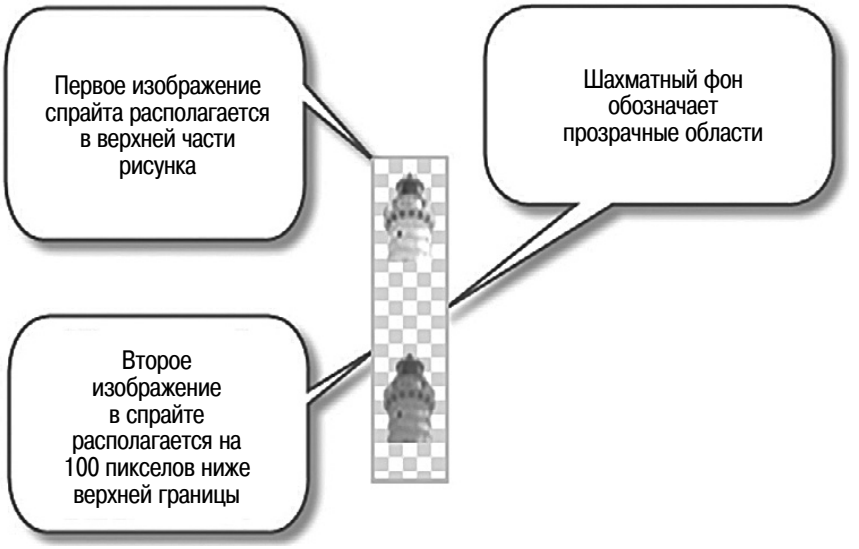


Рис. 7.4. Спрайт состоит из двух изображений в одном графическом файле

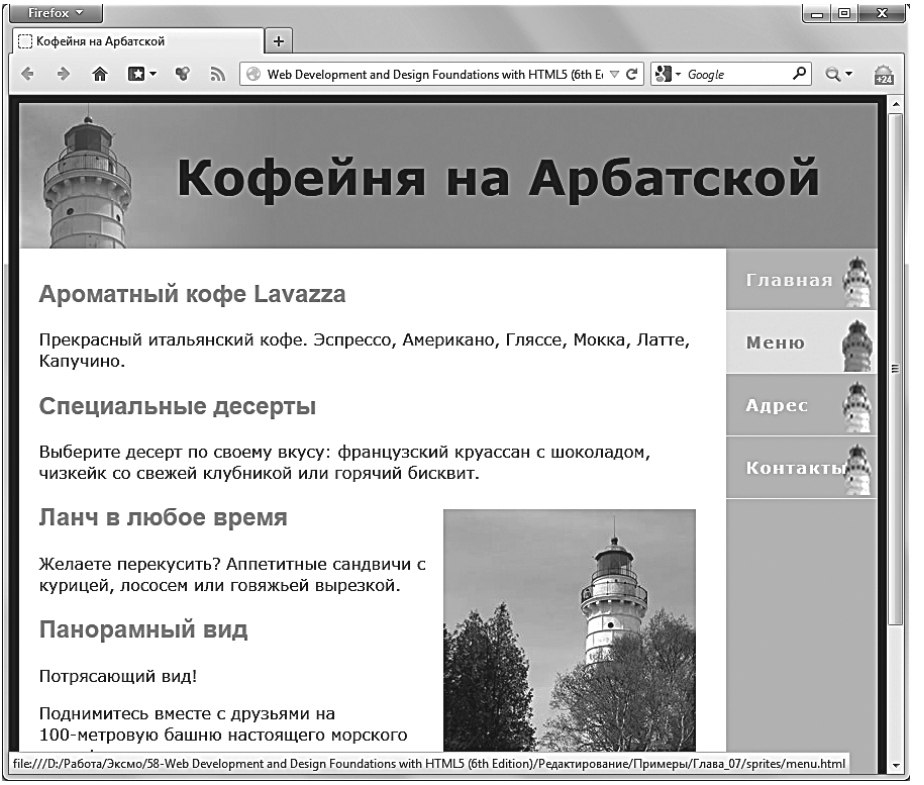


Рис. 7.5. Спрайты в действии

Практическое задание 7.3

На этом практическом задании вы будете работать с CSS-спрайтами, создавая веб-страницу, показанную на рис. 7.5. Создайте новую папку с именем *sprites*. Найдите на прилагающемся к книге диске документ *Примеры\Глава_07\starter2.html* и скопируйте его в папку *sprites*.

В эту же папку скопируйте следующие спрайты из папки *Примеры\Глава_07*: *lighthouseisland.jpg*, *lighthouselogo.jpg* и *sprites.gif*. Файл *sprites.gif*, как показано на рис. 7.4, содержит два изображения маяка.

Первое изображение располагается в верхней части графического файла. Второе — на 100 пикселов ниже от верхней части рисунка. Мы будем использовать значение 100, когда будем задавать отображение второго рисунка.

Откройте файл *starter2.html* в программе Блокнот (Notepad). Сохраните его под именем *index.html*. Вы будете редактировать глобальные стили, чтобы задать фоновое изображение для гиперссылок навигации.

1. Конфигурируйте фоновое изображение для гиперссылок навигации. Чтобы задать в качестве неповторяющегося фонового изображения файл *sprites.gif*, добавьте к селектору `#nav` следующие стили.

   Значение `right` свойства `background-position` конфигурирует появление изображения маяка справа от элементов навигации. Значение `0` свойства `background-position` задает отображение с нулевым смещением от верхнего края.

   ```
   #nav a { text-decoration: none;
   display: block;
   padding: 20px;
   background-color: #b3c7e6;
   border-bottom: 1px solid #ffffff;
   background-image: url(sprites.gif);
   background-repeat: no-repeat;
   background-position: right 0; }
   ```

2. Конфигурируйте второе изображение маяка, которое будет появляться при наведении указателя мыши на гиперссылку. Добавьте к селектору `#nav a:hover` следующие стили, чтобы отобразить второй маяк. Значение `right` свойства `background-position` конфигурирует появление изображения маяка справа от элемен-

тов навигации. Значение `-100px` свойства `background-position` задает отображение рисунка со смещением на 100 пикселов от верхнего края изображения.

```
#nav a:hover { background-color: #eaeaea;
color: #869dc7;
background-position: right -100px; }
```

Сохраните файл и протестируйте его в браузере. Ваша страница должна выглядеть так же, как показанная на рис. 7.5. Наведите указатель мыши на гиперссылки навигации, чтобы увидеть, как изменится фоновое изображение. Сравните свою работу с образцом на прилагающемся к книге диске (*Примеры\Глава_07\sprites\index.html*).

ЧаВо

КАК СОЗДАТЬ СОБСТВЕННЫЙ ГРАФИЧЕСКИЙ ФАЙЛ СПРАЙТА?

Большинство веб-разработчиков для редактирования изображений пользуются графическими редакторами, такими как Adobe Photoshop, Adobe Fireworks или GIMP, и сохраняют все в единый графический файл для использования в качестве спрайта. Или можно использовать веб-генератор спрайтов, к примеру, один из перечисленных ниже:

- генератор CSS-спрайтов: **csssprites.com**;
- генератор CSS-спрайтов: **spritegen.website-performance.org**;
- программа SpriteMe: **spriteme.org**.

Если у вас уже есть графический спрайт, попробуйте применить онлайн-инструмент **www.spritecow.com**, который поможет рассчитать для спрайта значения свойства `background-position` с точностью до пиксела.

7.3. Создание макетов веб-страниц с тремя колонками с помощью CSS

Часто веб-страница компонуется следующим образом: в верхней части страницы размещается раздел заголовка, а под ним три колонки — навигационный раздел, контент и боковая панель.

Если вы думаете, что в этом случае создается ряд блоков, то вы правы насчет использования CSS! На рис. 7.6 представлена блок-схема такой страницы, которую вы будете создавать в следующем практическом задании.

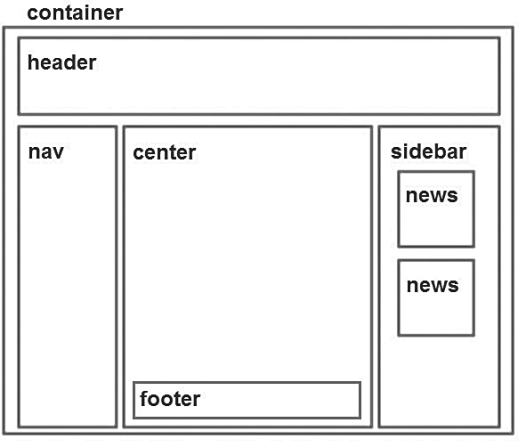


Рис. 7.6. Блок-схема страницы с тремя колонками

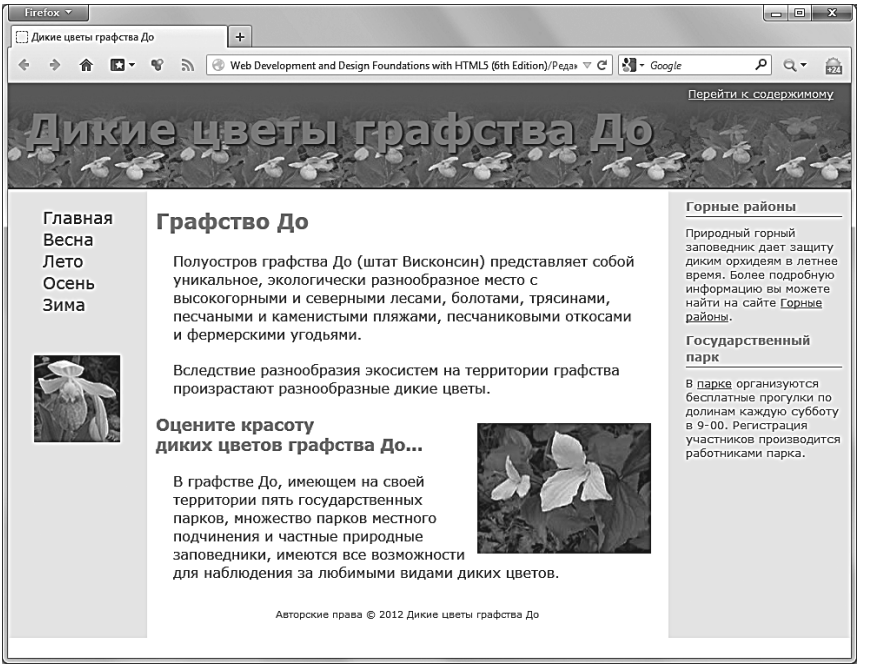


Рис. 7.7. Страница с тремя колонками создана с использованием CSS

Практическое задание 7.4

На этом практическом задании вы будете разрабатывать веб-страницу с тремя колонками, используя правила CSS. Будет применяться та же технология, что и при создании страницы с двумя колон-

ками — страница будет составлена из нескольких элементов или блоков. Пользуясь блок-схемой как подсказкой, конфигурируйте основную структуру на языке HTML. Затем примените CSS, чтобы задать области страницы, используйте идентификаторы или классы, где необходимо. Вспомните, что ключевым моментом конструирования страницы с двумя колонками было то, что левая колонка была создана обтекаемой справа (т.е. выровненной с левой стороны страницы). Ключевым моментом конструирования страницы с тремя колонками является то, что левая колонка будет обтекаемой справа (т.е. выровненной с левой стороны страницы) (`float:left`), а правая — будет обтекаемой слева (т.е. выровненной с правой стороны страницы) (`float:right`). Средняя колонка будет занимать середину страницы. Посмотрите на рис. 7.6 и 7.7, и вы увидите, как будет выглядеть страница по окончании работы.

Начало работы

Найдите на диске, прилагающемся к книге, в папке *Примеры\Глава_07\wildflowers* файлы *showybg.jpg*, *plsthumb.jpg* и *trillium.jpg*. Создайте новую папку с именем *wildflowers3*. Скопируйте указанные выше файлы в созданную папку.

Шаг 1. Верстка HTML-кода.

Вновь посмотрите на рис. 7.6 и 7.7 и выделите элементы страницы: область логотипа и фоновое изображение; левую колонку с областью навигации и рисунком; среднюю колонку с абзацами текста, заголовками и рисунком, смещенным вправо; правую колонку с двумя новостными сообщениями и колонтитул. Эти элементы будут конфигурироваться с применением идентификаторов и классов CSS, при этом будет задан ряд свойств, включающих `float`, `margin`, `border`, `font-family` и пр. Ссылки в навигационном меню будут конфигурироваться в виде неупорядоченного списка. В процессе верстки HTML-документа вы разместите его элементы на странице, присвоите значения идентификаторам и классам блоков, показанных на рис. 7.6. Создайте новый документ в программе Блокнот (Notepad) и укажите следующий код:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Дикие цветы графства До</title>
<meta charset="utf-8">
</head>
<body>
<div id="container">
<div id="header">
```

```
<span><a href="#center">Перейти к содержимому</a></span>
        <h1>Дикие цветы графства До</h1>
</div>
<div id="nav">
<ul>
<li><a href="index.html">Главная</a></li>
<li><a href="spring.html">Весна</a></li>
<li><a href="summer.html">Лето</a></li>
<li><a href="fall.html">Осень</a></li>
<li><a href="winter.html">Зима</a></li>
</ul>
<img src="plsthumb.jpg" width="100" height="100"
alt="Башмачок королевы">
</div>
<div id="sidebar">
<h3>Горные районы</h3>
<p class="news">Природный горный заповедник дает защиту
диким орхидеям в летнее время. Более подробную информа-
цию вы можете найти на сайте <a href="http://www.ridg-
esanctuary.org">Горные районы</a>.</p>
<h3>Государственный парк</h3>
<p class="news">В <a href="http://www.newportwilder-
nesssociety.org">парке</a> организуются бесплатные про-
гулки по долинам каждую субботу в 9-00. Регистрация
участников производится работниками парка.</p>
</div>
<div id="center">
<h2>Графство До</h2>
<p>Полуостров графства До (штат Висконсин) представля-
ет собой уникальное, экологически разнообразное место
с высокогорными и северными лесами, болотами, трясина-
ми, песчаными и каменистыми пляжами, песчаниковыми от-
косами и фермерскими угодьями.</p>
<p>Вследствие разнообразия экосистем на территории граф-
ства произрастают различные дикие цветы.</p>
<img src="trillium.jpg" width="200" height="150"
alt="Триллиум" id="floatright">
<h3>Оцените красоту <br>диких цветов графства До...</h3>
<p>В графстве До, имеющем на своей территории пять госу-
дарственных парков, множество парков местного подчинения
```

```
и частные природные заповедники, имеются все возможности
для наблюдения за любимыми видами диких цветов.</p>
<div id="footer"> Авторские права © 2012 Дикие цветы
графства До<br>
</div>
</div>
</div>
</body>
</HTML>
```

Сохраните страницу в файл *index.html* в папке *wildflowers3*. Протестируйте ее в браузере. Ваша страница не похожа на показанную на рис. 7.7, потому что вы еще не сконфигурировали CSS. Вид страницы должен быть таким, как на рис. 7.8.

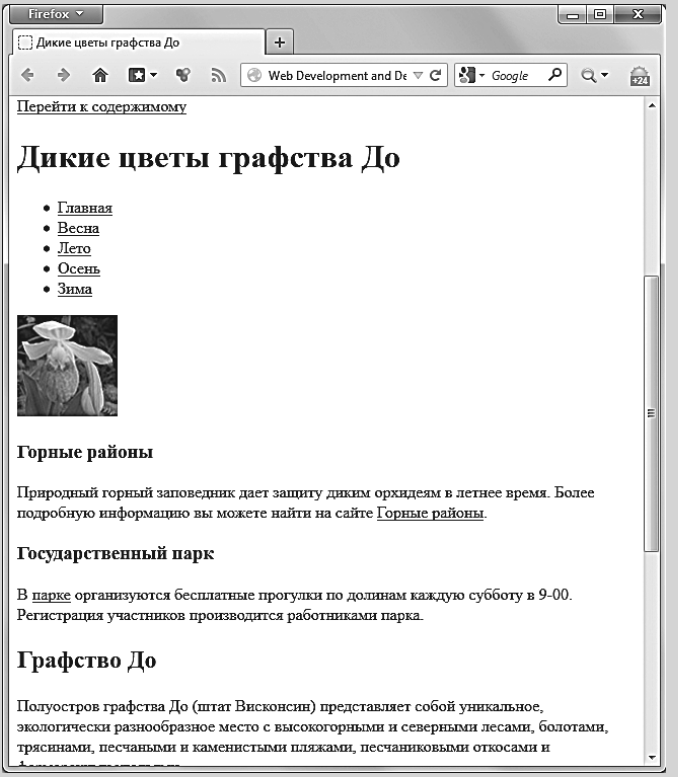


Рис. 7.8. Внешний вид страницы до применения стилей CSS

Шаг 2. Конфигурирование правил CSS.

Для упрощения работы на этом практическом задании вы будете верстать глобальную таблицу CSS в разделе заголовка веб-страницы.

Однако если бы вы создавали весь веб-сайт, вы, вероятно, применяли бы внешние таблицы стилей, как это было сделано на практическом задании 6.5.

В программе Блокнот (Notepad) откройте файл *index.html*. Рассмотрим, какие элементы содержит страница, представленная на рис. 7.6: логотип, левая, правая колонки, центральная колонка и колонтитул. Левая колонка содержит навигационную область и маленький рисунок. В центральной колонке находятся абзацы текста, заголовок и обтекаемый справа рисунок (т.е. выровненный с левой стороны страницы). В правой колонке расположен ряд заголовков и сообщений. Найдите положение этих областей на блок-схеме, данной на рис. 7.6. Обратите внимание, что на всей странице использован один и тот же тип шрифта, а страница начинается в правой части окна браузера.

В программе Блокнот (Notepad) измените раздел заголовка документа и добавьте тег `<style>`. Теперь займемся конфигурированием CSS. Сконфигурируйте в вашем документе стили CSS следующим образом:

1. **Селектор `body`.** Задайте ширину полей равной 0 и цвет фона `#ffffff`.

```
body {      margin:0;
background-color: #ffffff; }
```

2. **Контейнер.** Сконфигурируйте эту область с цветом фона `#eeeeee`, цветом текста `#006600`, минимальной шириной 960 пикселов и шрифтом Verdana, Arial или любым другим без засечек:

```
#container {      background-color: #eeeeee;
color: #006600;
min-width: 960px;
font-family: Verdana, Arial, sans-serif; }
```

3. **Заголовок.** Сконфигурируйте эту область с цветом фона (#636631) и фоновым рисунком (поместите файл *showybg.jpg* в нижнюю часть элемента, чтобы он повторялся по горизонтали). Высота области заголовка120 пикселов. Цвет шрифта должен быть `#cc66cc`, текст выровнен по правому краю, без отступов сверху и снизу. Задайте отступы справа и слева шириной 20 пикселов и границу шириной 2 пиксела внизу области:

```
#header { background-color: #636631;
background-image: url(showybg.jpg);
background-position: bottom;
background-repeat: repeat-x;
height: 120px;
```

```
color: #cc66cc;
text-align: right;
padding: 0 20px;
border-bottom: 2px solid #000000; }
```

4. **Левая колонка.** Важным моментом создаваемой структуры с тремя колонками является то, что левая колонка программируется обтекаемой справа, т.е. со смещением в левую часть окна браузера. Задайте ее ширину 150 пикселов:

```
#nav {      float: left;
width: 150px; }
```

5. **Правая колонка.** Для создаваемой структуры с тремя колонками существенно, чтобы правая колонка конфигурировалась обтекаемой слева, т.е. со смещением в правую часть окна браузера. Задайте ее ширину 200 пикселов:

```
#sidebar {  float: right;
width: 200px; }
```

6. **Центральная часть страницы.** Центральная колонка занимает место, оставшееся после создания обтекаемых левой и правой колонок. Для области контента особенно важно задать поля, так как правая и левая колонки созданы обтекаемыми каждая со своей стороны. Задайте ширину левого поля 160 пикселов, правого поля — 210 пикселов и оставшиеся поля — 0. Также задайте для этой области отступы. Задайте цвет фона `#ffffff`, цвет текста — `#006600`.

```
#center {   margin: 0 210px 0 160px;
padding: 1px 10px 20px 10px;
background-color: #ffffff;
color: #006600; }
```

7. **Колонтитул.** Текст колонтитула должен быть набран очень маленьким шрифтом и отцентрирован. Задайте цвет фона `#ffffff`, цвет текста — `#006600`. Верхний отступ должен быть 10 пикселов. Необходимо использовать свойство `clear:both` для запрета обтекания правой и левой колонками. Текст кода имеет следующий вид:

```
#footer {   font-size: .70em;
text-align: center;
color: #006600;
background-color: #ffffff;
padding-top: 10px;
clear:both; }
```

Теперь, когда вы сконфигурировали основные элементы структуры с тремя колонками, добавьте закрывающий тег `</style>`. Сохраните файл *threecolumns.html* в папке *wildflowers3*. Рекомендуется проверить страницу в браузере, чтобы убедиться, что вы движетесь в правильном направлении. Она должна выглядеть так, как показано на рис. 7.9. Обратите внимание, что над некоторыми деталями еще нужно поработать, однако вы на правильном пути!

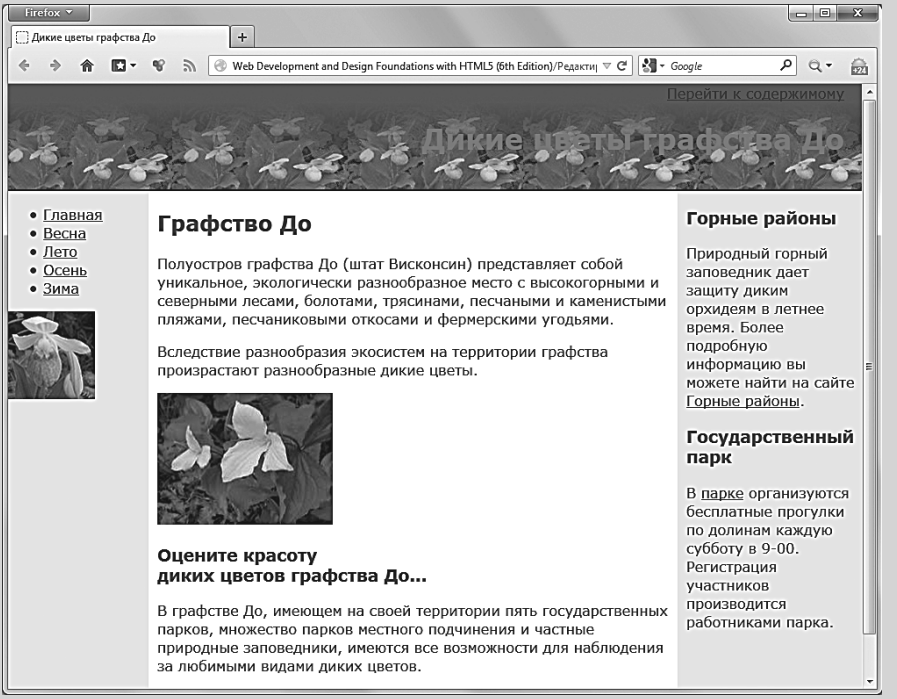


Рис. 7.9. Внешний вид страницы после конфигурирования основных элементов структуры с тремя колонками

Шаг 3. Продолжение верстки правил CSS.

Продолжим работу над конфигурированием стилей. Откройте файл *index.html* в программе Блокнот (Notepad) и установите курсор на пустую строку выше закрывающего тега `</style>`. Сначала мы сконфигурируем элементы левой колонки следующим образом:

1. Область логотипа.

- **Селектор элемента h1.** Обратите внимание на пустое пространство выше заголовка `h1` «Дикие цветы графства До», содержащегося в области логотипа. Вы можете убрать это пустое пространство путем назначения ширины 0 пикселов верхнему полю селектора `h1`. Также задайте для селектора `h1` размер шрифта 3 em:

```
h1 { margin-top: 0;
font-size: 3em;
text-align: left;
text-shadow: 2px 2px 2px #000000; }
```

- **«Перейти к содержимому».** Сконфигурируйте в разделе заголовка гиперссылку «Перейти к содержимому», задайте размер шрифта 0.80 em. Также конфигурируйте псевдоклассы `:link`, `:visited`, `:hover`, `:active` и `:focus`, установив цвет шрифта, как указано ниже:

```
#header a {font-size: 0.80em; }
#header a:link, #header a:visited { color: #ffffff; }
#header a:focus, #header a:hover { color: #eeeeee; }
```

2. Левая колонка с областью навигации.

- **Навигационное меню.** Сконфигурируйте неупорядоченный список без маркеров и с верхним полем, равным 20 пикселам.

```
#nav ul { margin-top: 20px;
list-style-type: none; }
```

Конфигурируйте навигационные ссылки без подчеркивания (`text-decoration: none`). Задайте размер шрифта 1,2 em. Должны быть сконфигурированы псевдоклассы `:link`, `:visited`, `:focus`, `:hover` и `:active` с разным цветом текста так, как указано ниже:

```
#nav a { text-decoration: none;
font-size: 1.2em; }
#nav a:link { color:#006600;}
#nav a:visited { color: #003300; }
#nav a:focus, #nav a:hover { color: #cc66cc; }
#nav a:active { color: #000000;}
```

- **Рисунок в левой колонке**. Конфигурируйте рисунки с идентификатором `nav` с полем шириной 30 пикселов следующим образом:

```
#nav img { margin: 30px;}
```

3. Центральная колонка.

- **Абзацы.** Задайте селектору `p` центральной области поле шириной 20 пикселов следующим образом:

```
#center p { margin: 20px; }
```

- **Заголовки**. Конфигурируйте селекторы элементов `h2` и `h3` в центральной области тем же цветом текста, что и логотип, и тем же цветом фона, что у основной части страницы.

```
#center h2, #center h3 { color: #cc66cc;
background-color: #ffffff; }
```

- **Обтекаемый слева рисунок.** Создайте идентификатор `floatright`, обтекаемый слева (т.е. выровненный с правой стороны страницы) (`float:right`) и с полем шириной 10 пикселов.

```
#floatright {margin: 10px;
float: right; }
```

4. Правая колонка.

- **Заголовки.** Сконфигурируйте элемент `h2` со сплошной границей шириной в 1 пиксел, с отступом внизу 2 пиксела, с полем в 10 пикселов, с текстом размером 0.90 em, того же цвета, что и текст заголовка в области логотипа:

```
#sidebar h3 { padding-bottom: 2px;
border-bottom: 1px solid #000000;
margin: 10px;
font-size: 0.90em;
color: #cc66cc;}
```

- **Колонка новостей.** Создайте класс с именем `news` с маленьким шрифтом и полем шириной 10 пикселов.

```
.news { font-size: 0.80em;
margin: 10px; }
```

Сохраните файл *index.html* в папку *windflowers3.*

Шаг 4. Проверка страницы.

Теперь, когда стили сконфигурированы, снова проверьте страницу *index.html*. Ее вид должен быть таким же, как показано на рис. 7.7. Помните, что браузер Internet Explorer не поддерживает свойство `text-shadow`. Если обнаружатся отличия, проверьте значения идентификаторов и классов в HTML-коде, а также синтаксис кода CSS. При проверке CSS вы можете воспользоваться валидатором на сайте консорциума W3C[1]. На диске, прилагающемся к книге, в папке *Примеры\Глава_07\wildflowers3*, записана копия файла *index.html*.

7.4. Подготовка страницы к печати с помощью CSS

Несмотря на то, что о достоинствах цифровых технологий говорится десятилетия, многие люди по-прежнему не могут обойтись без бумажных носителей и хотят, чтобы веб-страницы можно было распечатать. Таблицы CSS предоставляют возможность управлять процессом печати и внешним видом распечаток.

[1] jigsaw.w3.org/css-validator

Это проще всего делать с использованием внешних таблиц стилей. Создадим один вариант внешней таблицы стилей для отображения в окне браузера и второй вариант — со специальной конфигурацией для печати. Свяжем эту таблицу стилей с веб-страницей с помощью двух элементов `link`. Элементу `link` будет присвоен новый ***атрибут***, **`media`**, значения которого указаны в табл. 7.1.

Таблица 7.1. Наиболее часто используемые псевдоклассы CSS

Значение	Назначение
`screen`	Состояние по умолчанию, указывает, что используется таблица стилей для отображения в окне браузера на обычном цветном мониторе
`print`	Указывает на использование таблиц стилей, конфигурированных для печати
`handheld`	Предполагалось, что это значение, предложенное Консорциумом W3C, будет указывать на использование таблиц стилей, конфигурированных для мобильных устройств, однако на практике нет уверенности в его работе (подробнее об этом см. статью *Return of the Mobile Stylesheet*[1]). В следующем разделе мы рассмотрим другие способы разработки дизайна веб-страниц для мобильных устройств

[1] www.alistapart.com/articles/return-of-the-mobile-stylesheet/

Современные браузеры используют различные таблицы стилей в зависимости от того, требуется ли отобразить страницу на экране монитора или напечатать. Для отображения в окне браузера используется следующая конфигурация элемента `link`: `media:"screen"`. Для вывода на печать конфигурация элемента `link` выглядит следующим образом: `media: "print"`. Пример HTML-кода выглядит следующим образом:

```
<link rel="stylesheet" href="wildflower.css"
media="screen">
<link rel="stylesheet" href="wildflowerprint.css"
media="print">
```

Примеры стилей для печати

Возможно, вы задавались вопросом, чем таблица стилей для печати должна отличаться от CSS, используемой для отображения веб-страницы в браузере. Ниже перечислены популярные техники создания стилей страницы для печати.

Сокрытие несущественного контента

Чтобы при выводе на печать не печатались рекламные баннеры, элементы навигации и другие ненужные компоненты веб-страницы, часто используется свойство `display:none`.

Конфигурирование размера и цвета шрифта для печати

Другой распространенный прием — конфигурировать размер шрифта в пунктах — так легче управлять размером распечатки. Вы также можете использовать эти стили для конфигурирования черного цвета текста (`#000000`), если предполагаете, что посетители будут часто распечатывать страницы. Большинство браузеров по умолчанию настроены так, что не отображают при печати цвет фона и фоновые изображения, но запретить печать фонового изображения можно и в таблице стилей.

Управление разрывами страниц

Используйте свойства CSS **`page-break-before`** и **`page-break-after`** для управления разрывами страницы при печати. Отлично поддерживаемые браузерами значения этих свойств: `always` (разрыв страницы всегда будет происходить, как указано), `avoid` (если это возможно, разрыв страницы будет происходить до или после указанной позиции) и `auto` (по умолчанию). Например, чтобы задать разрыв страницы в определенном месте документа (в данном случае прямо перед элементом с присвоенным классом `newpage`), настройте CSS, как показано ниже:

```
.newpage { page-break-before: always; }
```

Печать URL-адресов гиперссылок

Подумайте, будет ли полезно, если при чтении распечатки вебстраницы человек увидит фактический адрес ресурса. Вы можете использовать правила CSS для отображения значения атрибута `href` прямо на странице с помощью двух методов: псевдоэлемента и свойства `content`. Целью ***псевдоэлемента*** CSS является применение к селектору определенного типа эффекта. В таблице 7.2 перечислены псевдоэлементы и их назначение.

Таблица 7.2. Псевдоэлементы CSS 2.1

Псевдоэлемент	Назначение
`:after`	Вставляет сгенерированный контент после селектора. Задайте этот контент, применив свойство `content`
`:before`	Вставляет сгенерированный контент перед селектором. Задайте этот контент, применив свойство `content`
`:first-letter`	Применяет стили к первой букве в тексте
`:first-line`	Применяет стили к первой строке текста

Для генерирования контента используйте ***свойство*** **`content`** вместе с псевдоэлементами `:after` и `:before`. Полезная особенность свойства `content` — функция `attr(x)`, которая возвращает значение указанного HTML-атрибута. Ее можно применить, чтобы распечатать URL-адрес из гиперссылки (значение атрибута `href`). Заключите в кавычки дополнительный текст или символы, например, круглые скобки. Приведенный ниже код CSS отобразит URL-адреса всех гиперссылок боковой панели в скобках после текста гиперссылки.

```
#sidebar a:after { content: " (" attr(href) ") "; }
```

На рис. 7.10 приведен вид в окне предварительного просмотра перед печатью страницы, созданной вами на практическом задании 7.4. Обратите внимание, что в окне предварительного просмотра отображается навигационная область.

На рис. 7.11 приводится альтернативный вариант страницы, на которой использованы возможности CSS для отмены вывода на печать области навигации, конфигурирования размера шрифтов в пунктах (`pt`) и печати URL-адресов гиперссылок в боковой панели. На следующем практическом задании вы будете осваивать эту технологию.

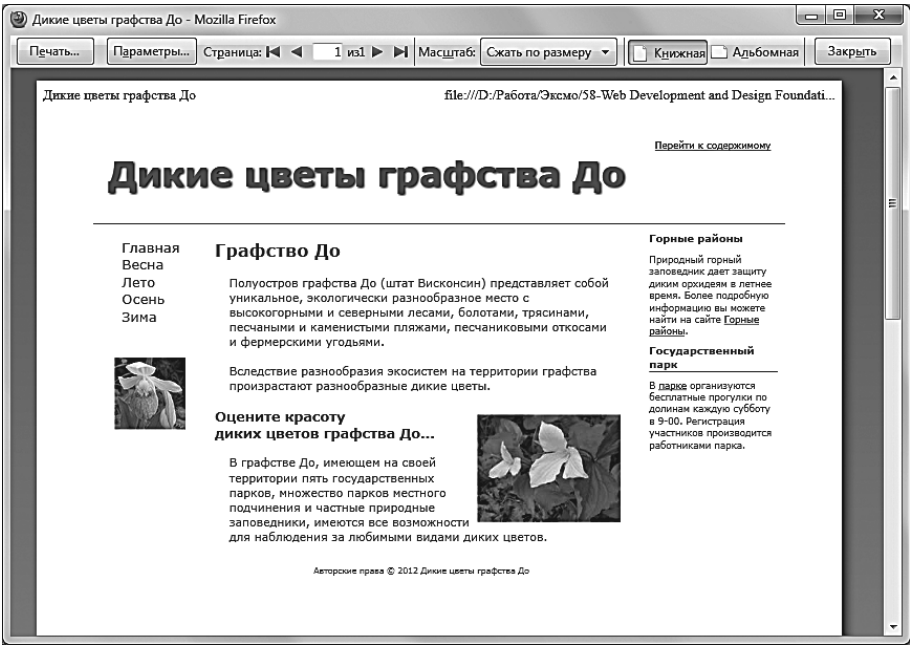


Рис. 7.10. Внешний вид страницы, приведенной на рис. 7.7, в окне предварительного просмотра

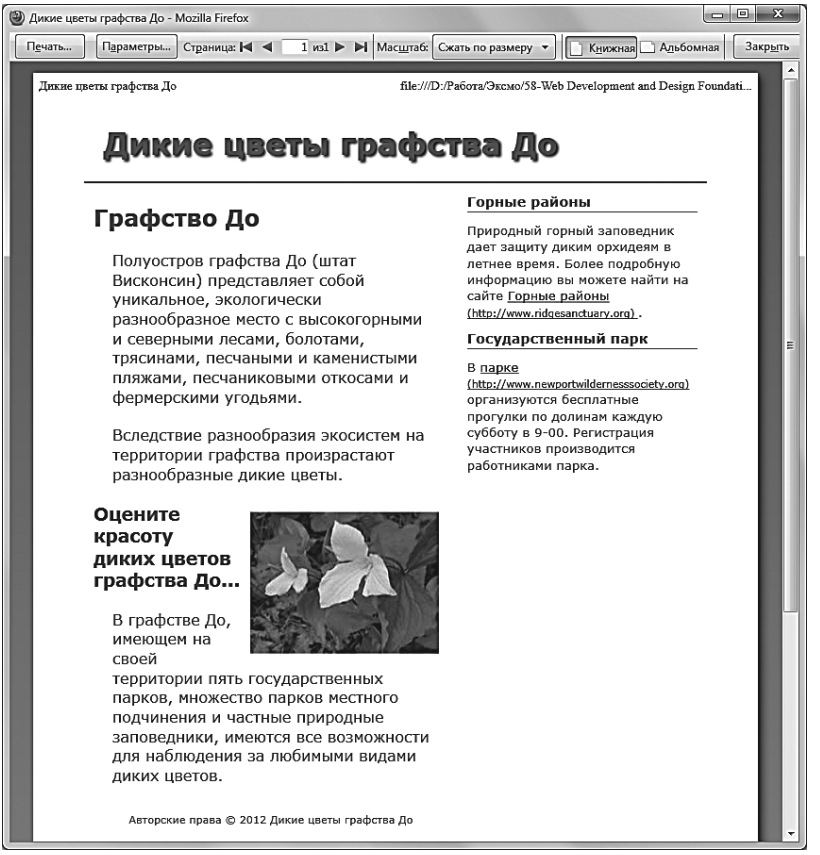


Рис. 7.11. Внешний вид страницы, к которой были применены CSS

Практическое задание 7.5

На этом практическом задании вы будете создавать специальные стили, используемые при печати веб-страниц. В качестве отправной точки мы будем использовать файл *index.html* сайта «Дикие цветы графства До», который вы создали на практическом задании 7.4. На рис. 7.11 представлен вид окна браузера с открытой страницей *index.html*. Вы создадите новую версию файла *index.html* и две внешние таблицы стилей: одну для отображения в браузере и одну для печати. По окончании работы распечатанная страница должна быть похожа на показанную на рис. 7.11.

Начало работы

Создайте новую папку с именем *print* и скопируйте в нее файлы *index.html, plsthumb.jpg, showybg.jpg и trillium.jpg* из папки на прилагающемся к книге диске *Примеры\Глава_07\wildflowers.*

Шаг 1. Создание таблицы стилей для отображения страницы на экране.

Запустите редактор Блокнот (Notepad) и откройте в нем файл *index.html*. Скопируйте все правила стилей между открывающим и закрывающим тегами `style`. Создайте новый файл в редакторе Блокнот (Notepad) и сохраните его под именем *wildflower.css* в папку *print.* Вставьте правила стилей в файл *wildflower.css* и сохраните его.

Шаг 2. Редактирование HTML-кода.

Отредактируйте файл *index.html* — удалите элемент `style`. Сконфигурируйте элемент `link`, чтобы связать веб-страницу с только что созданной внешней таблицей стилей *wildflower.css.* Добавьте атрибут `media` со значением `screen` к элементу `link` внешней таблицей стилей *wildflower.css*. Сконфигурируйте новый элемент `link` для внешней таблицы стилей с именем *wildflower.css*, предназначенный для печати страницы (`media: "print"`). HTML-код будет иметь следующий вид:

```
<link rel="stylesheet" href="wildflower.css"
media="screen">
<link rel="stylesheet" href="wildflowerprint.css"
media="print">
```

Сохраните страницу *index.html* в папке *print*.

Шаг 3. Создание новой таблицы CSS для печати.

Откройте файл *wildflower.css* в программе Блокнот (Notepad). Так как для печати вы будете использовать большинство стилей из этой таблицы, вы будете редактировать этот файл. Сохраните файл с именем *wildflowerprint.css* в папке *print*. Вы внесете изменения в нескольких областях этой таблицы стилей, конфигурирующих идентификатор `content`, селектор элемента `h1`, идентификаторы `header`, `nav`, `center` и гиперссылки в идентификаторах `header` и `sidebar`.

Удалите из идентификатора `container` определение стиля минимальной ширины (`min-width`).

1. Измените селектор элемента `h1` таким образом, чтобы принтер использовал шрифт с размером 24 пункта. Удалите свойство `text-align`.

   ```
   h1 { margin-top: 0;
   font-size: 24pt;
   text-shadow: 2px 2px 2px #000000; }
   ```

2. Измените правила стилей идентификатора `header`. Конфигурируйте белый цвет фона (`#ffffff`). Удалите определение фонового изображения. Свойство `height` также следует удалить,

поскольку у вас больше нет фонового изображения. Кроме того, удалите свойства `text alignment` и `padding`:

```
#header { color: #cc66cc;
background-color: #ffffff;
border-bottom: 2px solid #000000;
padding: 0 20px; }
```

3. Запретите отображение гиперссылки «Перейти к содержимому» в идентификаторе `header`. Замените все правила стилей, связанные с идентификатором `#header`, следующим кодом:

```
#header a { display: none; }
```

4. Запретите отображение области навигации. Удалите все правила стилей, связанные с идентификатором `#nav`, и добавьте следующий код:

```
#nav { display: none; }
```

5. Сконфигурируйте центральную область. Измените правила стилей идентификатора `#center`. Установите величину левого отступа равным 0, правого отступа — 40% и размер шрифта 12 пунктов.

```
#center { margin: 0 40% 0 0;
padding: 1px 10px 20px 10px;
font-size: 12pt;
background-color: #ffffff;
color: #006600; }
```

6. Задайте ширину боковой панели, равную 40%.

```
#sidebar { float: right;
width: 40%; }
```

7. Конфигурируйте класс `news`. Измените правило стиля, чтобы задать размер шрифта 10pt.

```
.news { font-size: 10pt;
margin: 10px; }
```

8. Конфигурируйте гиперссылки в боковой панели таким образом, чтобы URL-адреса распечатывались шрифтом черного цвета размером 8 пунктов. Добавьте следующий CSS-код:

```
#sidebar a:after { content " (" attr(href) ") ";
font-size: 8pt;
color: #000000; }
```

9. Просмотрите все правила стилей и задайте белый цвет фона (`#ffffff`) для всех селекторов элементов `h2` и `h3`.

Сохраните файл в папке *print*.

Шаг 4. Проверка работы.

Проверьте страницу *index.html* в браузере. Выберите команду меню **Файл ⇒ Предварительный просмотр** (File ⇒ Preview). Вид страницы должен быть таким же, как показано на рис. 7.11. Размер шрифта задан, а область навигации не отображается. URL-адреса указаны после текста гиперссылки в боковой панели. На диске, прилагающемся к книге, в папке *Примеры\Глава_07\print*, вы найдете готовые примеры.

Обратите внимание, что в синтаксисе CSS3 произошли изменения. Теперь перед каждым псевдоэлементом указываются два двоеточия, а не одно (например, в CSS3 используется вариант `::after` вместо `:after`). Однако мы пока продолжаем использовать псевдоэлементы и синтаксис CSS2, так как его поддерживает большее число браузеров.

7.5. Стили CSS для отображения веб-сайтов на мобильных устройствах

В главе 5 вы познакомились с тремя методами, которые могут быть использованы для предоставления доступа к веб-сайту посетителям, использующим мобильные устройства. Первый вариант заключается в разработке и публикации второго сайта с доменом высшего уровня .mobi TLD. Чтобы увидеть, как это выглядит на практике, посетите сайты **jcp.com** и **jcp.mobi**. Другой вариант заключается в разработке и публикации отдельного веб-сайта на собственном домене, оптимизированном для мобильного использования. Этот метод используется, к примеру, на веб-сайте **www.rambler.ru** (рис. 7.12). С мобильной версией сайта по адресу **m.rambler.ru** (рис. 7.13). Третий вариант — создание одного сайта с разными стилями для мобильных и настольных браузеров. Перед тем как уделить внимание коду, давайте рассмотрим методы веб-дизайна для мобильных устройств.

Рекомендации по дизайну веб-страниц для мобильных устройств

Мобильные пользователи обычно посещают веб-сайты в дороге, им необходимо быстро найти информацию и они часто отвлекаются. Веб-страница, оптимизированная для доступа с мобильных устройств, должна удовлетворять их потребностям. Уделите немного времени изучению рис. 7.12 и 7.13 и обратите внимание, как при дизайне мобильного веб-сайта были решены проблемы, упомянутые в главе 5:

- **Область заголовка уменьшена** так, чтобы она могла уместиться на маленьком экране.
- **Обратите внимание, что изображения**, присутствующие на рис. 7.12, не отображаются в версии веб-страницы для мобильных устройств.
- **Шрифты и цвета.** Применяются распространенные гарнитуры шрифтов. Также присутствует четкий контраст цветов фона и текста.
- **Сложность управления, ограниченная производительность процессора и малый объем памяти.** Мобильные веб-сайты реализуют макет страницы в одну колонку, который упрощает использование клавиатуры и легко управляется прикосновением на мобильном устройстве с сенсорным экраном. Страница содержит в основном текст, так как мобильный браузер может его быстро отобразить.
- **Функциональность.** Гиперссылки на популярные веб-сайты отображаются прямо под заголовком. Также предоставляется функция поиска.

Давайте подробнее поговорим о перечисленных рекомендациях по дизайну.

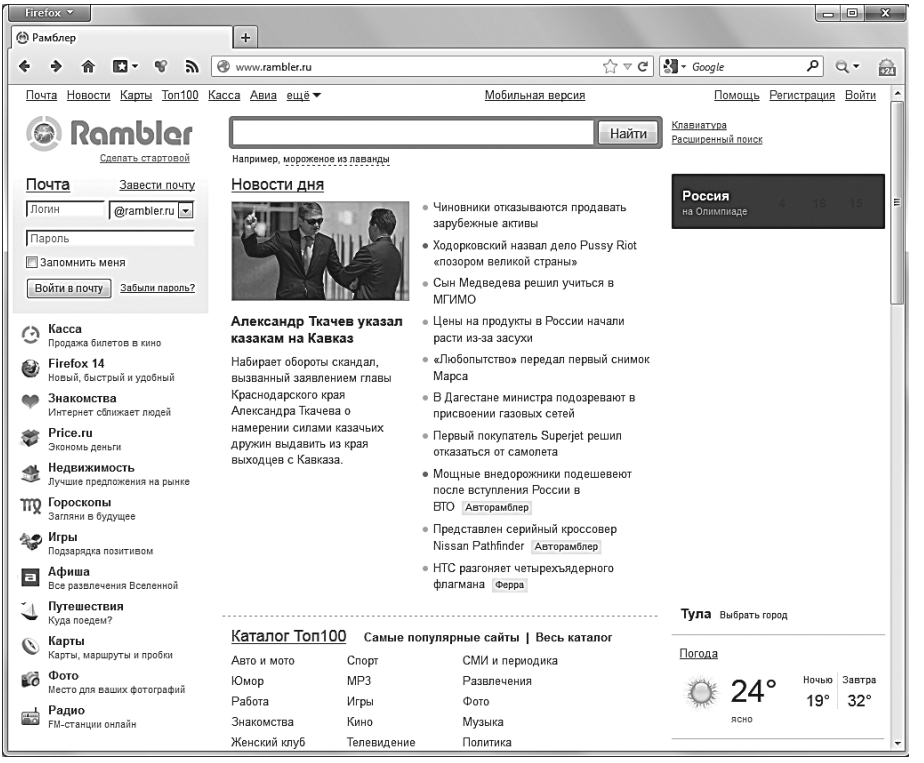


Рис. 7.12. Версия веб-сайта Rambler в браузере компьютера

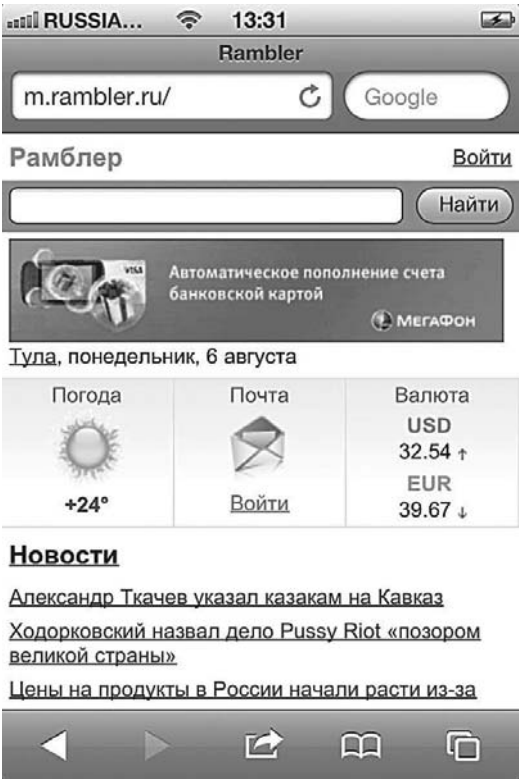


Рис. 7.13. Версия веб-сайта Rambler в браузере мобильного устройства

Оптимизация макета

Для отображения на мобильном устройстве хорошо подходит макет страницы в одну колонку (рис. 7.14) с небольшим заголовком, ссылками навигации, контентом и нижним колонтитулом. Разрешения экранов мобильных устройств сильно различаются (например, 320×240 [BlackBerry Pearl], 320×480 [Apple iPhone 3, HTC Dream], 360×640 [Nokia], 480×800 [HTC Desire, HTC Pro] и 960×640 [Apple iPhone 4\4S]). Рекомендации консорциума W3C включают следующее:

- ограничьте прокрутку в одном направлении;
- используйте элементы заголовка;
- примените списки для организации информации (например, неупорядоченные и упорядоченные списки, а также списки определений);
- избегайте использования таблиц (см. главу 8), потому что они обычно требуют горизонтальной или вертикальной прокрутки на мобильных устройствах;

- предоставьте ярлыки для средств управления формой (см. главу 9);
- не указывайте в таблицах стилей единицы измерения в пикселах;
- не используйте абсолютное позиционирование в таблицах стилей;
- скройте содержимое, не являющееся значимым для мобильного использования.

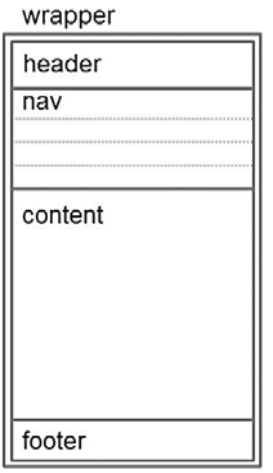


Рис. 7.14. Блок-схема типичного макета мобильной веб-страницы в одну колонку

Оптимизация навигации

Для мобильных устройств крайне важна простая навигация. Консорциум W3C рекомендует следующее:

- предоставьте минимальное количество необходимых средств навигации в верхней части страницы;
- соблюдайте последовательность навигации;
- избегайте гиперссылок, которые открывают файлы в новых окнах, а также всплывающих окон;
- постарайтесь сбалансировать число ссылок на странице и уровней вложенности, необходимых для доступа к информации.

Оптимизация изображений

Графика может помочь привлечь посетителей, но помните о следующих рекомендациях Консорциума W3C для мобильного использования:

- старайтесь не показывать изображений, которые шире, чем ширина экрана устройства (предположим, более 320 пикселов);

- используйте оптимизированные фоновые изображения малого размера;
- некоторые мобильные браузеры уменьшают разрешение всех изображений, поэтому текст на таких рисунках может стать нечитабельным;
- избегайте использования больших графических изображений;
- указывайте размер изображений;
- предоставьте замещающий текст для графики и других нетекстовых элементов.

Оптимизация текста

Текст на небольшом мобильном устройстве не всегда легко читается. Следующие рекомендации консорциума W3C весьма важны:

- задайте четкий контраст между цветами текста и фона;
- используйте распространенные гарнитуры шрифтов;
- задайте размер шрифта в единицах em или процентах;
- используйте короткое, описательное название страницы.

Консорциум W3C опубликовал список *Mobile Web Best Practices 1.0*, содержащий 60 лучших советов по дизайну для мобильных устройств, на сайте **www.w3.org/TR/mobile-bp**. Шпаргалки с основными советами из списка *Mobile Web Best Practices 1.0* можно найти на сайте **www.w3.org/2007/02/mwbp_flip_cards.html.**

Единая Всемирная паутина

Миссия консорциума W3C по созданию «Единой Всемирной паутины» опирается на концепцию обеспечения одного ресурса, настроенного для оптимального отображения на различных типах устройств. Это более эффективно, чем создание нескольких версий веб-документа. Помня о «Единой Всемирной паутине», мы рассмотрим использование метатега `viewport` и медиазапросов CSS для выявления и определения таблиц стилей, оптимизированных для отображения на мобильных устройствах.

Метатег `viewport`

Существует несколько применений для метатегов. Вы использовали мета-тег, начиная с главы 2, чтобы настроить кодировку на веб-странице.

Теперь мы изучим новый ***метатег*** **`viewport`**, который был создан компанией Apple как расширение, улучшающее отображение на мобильных устройствах, таких как iPhone и смартфоны на платформе Android, путем установления ширины и масштаба просмотра. На рис. 7.15 показан снимок веб-страницы, отображенной на смартфоне iPhone без метатега `viewport`.

Изучите рис. 7.15 и обратите внимание, что мобильные устройства отображают всю веб-страницу на крошечном экране в уменьшенном масштабе. Текст на сайте страницы трудно читать.

Рис. 7.15. Отображение веб-страницы на мобильном устройстве

На рис. 7.16 показана та же веб-страница после добавления метатега `viewport` в раздел заголовка документа. Код приведен ниже:

```
<meta name="viewport" content="width=device-width,
initial-scale=1.0">
```

Введите метатег `viewport` в HTML-коде, указав `name="viewport"` и атрибут `content`. Значение HTML-атрибута `content` может

содержать одну или несколько ***директив*** (также известных как свойства), таких как `device-width` (ширина устройства). Их цель — контролировать масштаб и его изменение. В таблице 7.3 перечислены директивы метатега `viewport` и их значения.

Рис. 7.16. Метатег `viewport` улучшает отображение веб-страницы на мобильном устройстве

Таблица 7.3. Директивы метатега `viewport`

Директива	Значение	Назначение
`width`	Числовое значение или значение **`device-width`**, указывающее фактическую ширину экрана мобильного устройства	Ширина окна просмотра в пикселах
`height`	Числовое значение или значение **`device-height`**, указывающее фактическую высоту экрана мобильного устройства	Высота окна просмотра в пикселах
`initial-scale`	Числовой коэффициент; установите значение в диапазоне от 1 до 100%	Исходный масштаб окна просмотра
`minimum-scale`	Числовой коэффициент; в мобильной версии браузера Safari по умолчанию установлено значение 0,25	Минимальный масштаб окна просмотра

Директива	Значение	Назначение
maximum-scale	Числовой коэффициент; в мобильной версии браузера Safari по умолчанию установлено значение 1,6	Максимальный масштаб окна просмотра
user-scalable	Значение Yes разрешает масштабирование, no — запрещает	Определяет, может ли пользователь менять масштаб страницы

Теперь, когда вы сделали страницу читабельной, как насчет того, чтобы с помощью стилей оптимизировать ее для использования на мобильном устройстве? И в этом нам помогут CSS.

Медиазапросы CSS

Ранее в этой главе вы использовали элементы link и media="print", чтобы настроить стили веб-страницы, оптимизирующие последнюю для вывода на печать. Значение атрибута media="handheld" первоначально предназначалось для той же цели, но он недостаточно широко поддерживается.

Так что же делать, если атрибут media="handheld" не поддерживается? Используйте **медиазапрос**[1] для определения характеристик мобильного устройства, таких как разрешение экрана и ориентация (книжная или альбомная), а также стилей браузеров, настроенных специально под эти параметры. На рис. 7.17 показана та же самая веб-страница, что и на рис. 7.15 и 7.16, но она выглядит совсем иначе потому, что элемент link, который включает в себя медиазапросы и связан с таблицей стилей, задан для отображения на мобильном устройстве. HTML-код приведен ниже:

```
<link href="lighthousemobile.css" rel="stylesheet"
media="only screen and (max-device-width: 480px)">
```

Данный пример кода задает оптимальное отображение на наиболее популярных смартфонах. Значение only — это ключевое слово, скрывающее медиазапрос от устаревших браузеров. Значение screen нацелено на устройства с экраном. Параметру max-device-width (см. табл. 7.4) задано значение 480px, что соответствует размерам экранов самых популярных смартфонов.

[1] www.w3.org/TR/css3-mediaqueries

Часто используемые ***типы устройств*** и ключевые слова включают следующее:

- `all` предназначен для всех устройств;
- `screen` для отображения на экране;
- `only` инструктирует старые браузеры, не поддерживающие медиазапросы, игнорировать их;
- `print` предназначен для подготовки страницы к выводу на печать.

Рис. 7.17. Медиазапросы CSS помогают конфигурировать веб-страницу для отображения на мобильном устройстве

Таблица 7.4. Часто используемые параметры медиазапросов

Параметры	Значения	Критерии
`max-device-height`	Числовое значение	Высота экрана устройства вывода в пикселах меньше или равна данному значению
`max-device-width`	Числовое значение	Ширина экрана устройства вывода в пикселах меньше или равна данному значению
`min-device-height`	Числовое значение	Высота экрана устройства вывода в пикселах больше или равна данному значению
`min-device-width`	Числовое значение	Ширина экрана устройства вывода в пикселах больше или равна данному значению

Параметры	Значения	Критерии
max-height	Числовое значение	Высота окна просмотра в пикселах меньше или равна данному значению (вычисляется заново при изменении размера экрана)
min-height	Числовое значение	Высота окна просмотра в пикселах больше или равна данному значению (вычисляется заново при изменении размера экрана)
max-width	Числовое значение	Ширина окна просмотра в пикселах меньше или равна данному значению (вычисляется заново при изменении размера экрана)
min-width	Числовое значение	Ширина окна просмотра в пикселах больше или равна данному значению (вычисляется заново при изменении размера экрана)
orientation	Книжная или альбомная	Ориентация экрана устройства

Тестирование отображения на мобильном устройстве

Лучший способ проверить отображение мобильной веб-страницы — опубликовать ее в Интернете и получить к ней доступ с мобильных устройств (см. Приложение Г для ознакомления с процессом публикации веб-сайта по протоколу FTP). Однако не у всех разработчиков есть смартфоны.

Ниже перечислены несколько вариантов эмуляции отображения на мобильном устройстве:

- **Opera Mobile Emulator**[1] (загружается и устанавливается только в операционной системе Windows). Поддерживает медиазапросы.
- **Mobilizer**[2] (загружается и устанавливается в операционных системах Mac и Windows). Поддерживает медиазапросы.
- **Онлайн-эмулятор Opera Mini**[3] (запускается в окне браузера). Отключает фоновые изображения, поддерживает медиазапросы.
- **iPhone Emulator**[4] (запускается в окне браузера). Не поддерживает медиазапросы.
- **iPhoney**[5] (загружается и устанавливается только в операционной системе OS X). Не поддерживает медиазапросы.

[1] www.opera.com/developer/tools/mobile
[2] www.springbox.com/mobilizer
[3] www.opera.com/developer/tools/mini/
[4] www.testiphone.com
[5] www.marketcircle.com/iphoney

Тестирование в браузере

Что делать, если у вас нет смартфона и не получается опубликовать свои файлы в Интернете? Вы можете посмотреть, как *приблизительно* будет отображаться ваш сайт на мобильном устройстве с помощью настольного браузера, используя следующую методику:

1. Измените элемент `link` так, чтобы он указывал на вашу мобильную таблицу стилей.
2. Откройте веб-страницу в браузере настольного компьютера, а затем уменьшите ширину и высоту окна просмотра, пока оно не приблизится к размеру экрана мобильного устройства (например, 320×480).

Данная методика продемонстрирована на рис. 7.18. Браузер Firefox используется для отображения того же сайта с использованием мобильных стилей (см. рис. 7.17).

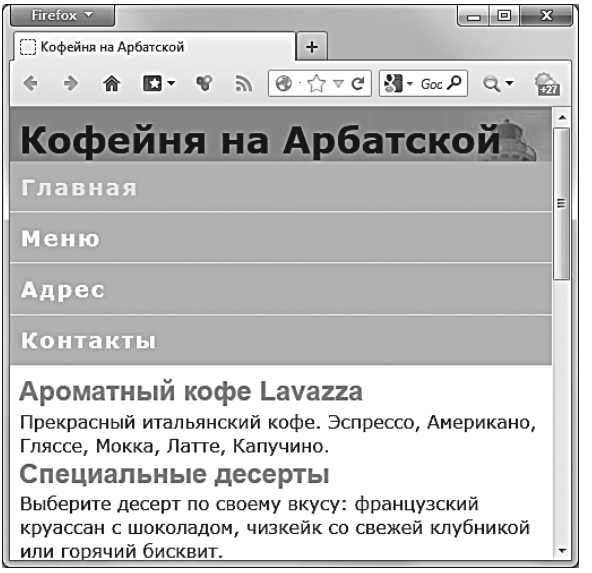


Рис. 7.18. Для просмотра мобильной веб-страницы используется браузер на компьютере

Только для профессиональных разработчиков

Если вы разрабатываете программное обеспечение или управляете им, возможно, вам стоит изучить пакеты SDK (комплект разработчика) для платформ iOS и Android. Каждый пакет SDK включает в себя эмулятор мобильных устройств:

- **iOS SDK** (только для операционной системы OS X)[1];
- **Android SDK**[2];

Теперь, когда вы усвоили некоторые советы по дизайну для мобильных устройств, узнали, как создать веб-страницу, ориентировать ее на мобильные устройства, а также какие варианты тестирования и эмуляции мобильных веб-сайтов использовать, давайте применим новые знания и конфигурируем страницу, показанную на рис. 7.17.

Практическое задание 7.6

Создайте новую папку с именем *mobile*. Скопируйте в нее файл *starter2.html* из папки *Примеры\Глава_07* прилагающегося к книге диска. Переименуйте файл *starter2.html* в *index.html*. Скопируйте в папку *mobile* следующие файлы из папки *Примеры\Глава_07*: *lighthouseisland.jpg, lighthouselogo.jpg* и *mobilelogo.jpg*. Запустите браузер и откройте в нем файл *starter2.html*, как показано на рис. 7.19. Вы будете изменять веб-страницу, используя метатег `viewport` и медиазапросы для нацеливания на использование на мобильных устройствах.

Шаг 1. Создание новой таблицы стилей.

Запустите программу Блокнот (Notepad) и откройте файл *index.html*. Скопируйте все правила стилей между открывающим и закрывающим тегами `style`. В программе Блокнот (Notepad) создайте новый файл с именем *lighthouse.css* в папке *mobile*.

Вставьте правила стилей в файл *lighthouse.css* и сохраните его.

Шаг 2. Измените HTML-код.

1. Отредактируйте файл *index.html* и удалите элемент `style`. Добавьте в код элемент `link`, чтобы связать веб-страницу с внешней таблицей стилей, которую вы только что создали (*lighthouse.css*). Добавьте к элементу `link` файла *lighthouse.css* атрибут `media` со значением `screen`. Добавьте второй элемент `link`, чтобы связать внешнюю таблицу стилей *lighthousemobile.css* для отображения на мобильном устройстве. HTML-код следующий:

```
<link href="lighthouse.css" rel="stylesheet"
media="screen">
<link href="lighthousemobile.css" rel="stylesheet"
media="only screen and (max-device-width: 480px)">
```

[1] developer.apple.com/programs/ios/develop.html
[2] developer.android.com/sdk/index.html

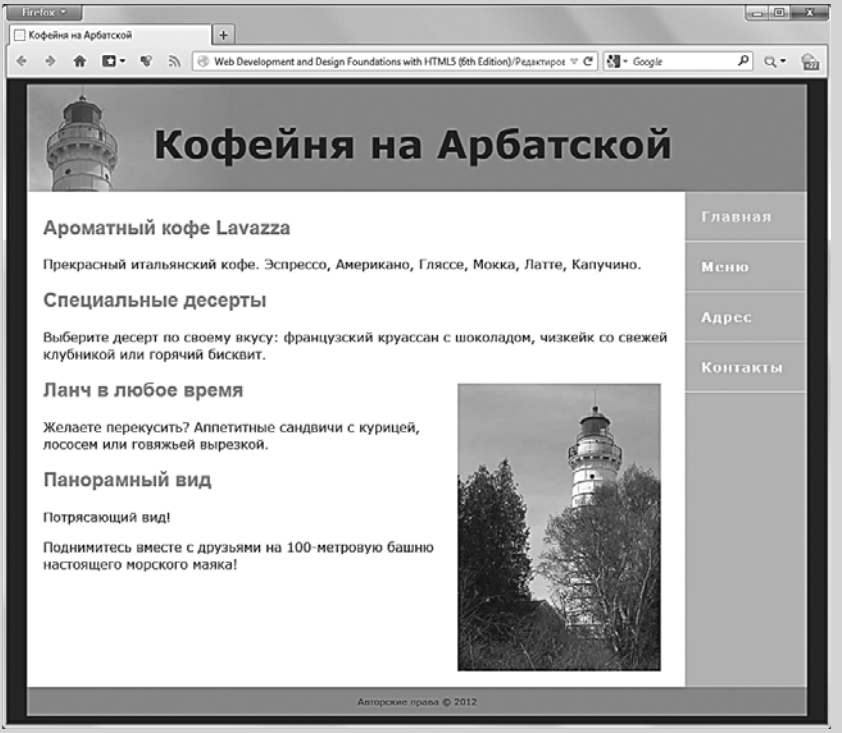


Рис. 7.19. Веб-страница, открытая в настольном браузере

2. Ниже элементов `link` добавьте метатег `viewport`. HTML-код следующий:

```
<meta name="viewport" content="width=device-width,
initial-scale=1.0">
```

3. Преимуществом отображения на мобильном устройстве станет ссылка «Вверх» в нижней части страницы. Добавьте элемент `div` с присвоенным ему идентификатором `backtotop` в нижней части области контента, которая содержит гиперссылку на `идентификатор #header`. HTML-код следующий:

```
<div id="backtotop"><a href="#header">Вверх</a></div>
```

Сохраните файл *index.html* в папку *mobile*.

Шаг 3. Укажите новый код CSS.

Откройте файл *lighthouse.css* в программе Блокнот (Notepad). Так как вам нужно сохранить большинство стилей для отображения на мобильном устройстве, начните с создания новой версии внешней таблицы стилей. Сохраните файл *lighthouse.css* под именем *lighhousemobile.css* в папку *mobile*. В этой таблице стилей вы измените несколько областей, в том числе селектор элемента `body`, иденти-

фикатор `wrapper`, селектор элемента `h1`, идентификаторы `header`, `nav` и `footer`, а также гиперссылки навигации.

1. Отредактируйте селектор элемента `body`. Добавьте правило стиля, задающее отсутствие полей.

```
body { font-family: Verdana, Arial, sans-serif;
background-color: #00005D;
margin: 0; }
```

2. Измените правила стилей идентификатора `wrapper` и задайте ширину 100%, без полей, без отступа и без минимальной ширины.

```
#wrapper {background-color: #b3c7e6;
color: #000066;
width: 100%;
margin: 0;
padding: 0;
min-width: 0; }
```

3. Отредактируйте правила стилей идентификатора `header`. Установите отступ, как показано ниже. Поменяйте размер шрифта на 100%. Задайте рисунок *mobilelogo.jpg* в качестве неповторяющегося фонового изображения. Установите положение фонового рисунка так, чтобы он отображался в правом верхнем углу.

```
header { background-color: #869dc7;
color: #00005D;
font-size: 100%;
padding: 0.5em 0 0 0.5em;
background-image: url(mobilelogo.jpg);
background-repeat: no-repeat;
background-position: right top; }
```

4. Удалите поля, сконфигурированные для селектора элемента `h1`.

```
h1 { margin: 0; }
```

5. Измените правила стилей идентификатора `nav`. Удалите обтекание, установите ширину 100%, а размер шрифта 120%.

```
#nav {      float: none;
width: 100%;
font-size: 120%;
letter-spacing: 0.1em;
font-weight: bold; }
```

6. Измените правила стилей идентификатора `content`. Установите отступ величиной 0.5 em.

```
#content { background-color: #ffffff;
color: #000000;
padding: 0.5em;
overflow: auto; }
```

7. Измените правила стилей идентификатора `footer`. Установите нулевой отступ.

```
#footer { font-size: 70%;
text-align: center;
padding: 0;
background-color: #869dc7;
clear: both; }
```

8. Удалите поля селектора элемента `h2`.

```
h2 { color: #869dc7;
font-family: Arial, sans-serif;
margin: 0; }
```

9. Конфигурируйте поле селектора элемента абзаца (`p`) величиной 0.05 em.

```
p {margin: 0.05em; }
```

10. Задайте класс `floatright` без обтекания и без полей.

```
#floatright { margin: 0;
float: none; }
```

11. Конфигурируйте новое правило стилей для элементов `li` идентификатора `nav`. Установите отступ, равный 0.

```
#nav li { padding: 0; }
```

12. Измените стили для гиперссылок идентификатора `nav`. Установите отступ, как показано ниже:

```
#nav a { text-decoration: none;
display: block;
padding: 0.5em;
background-color: #b3c7e6;
border-bottom: 1px solid #ffffff; }
```

13. Конфигурируйте новое правило стилей для идентификатора `backtotop`. Присвойте значение `block` атрибуту `display`.

```
#backtotop {display: block; }
```

Сохраните файл в папку *mobile*.

Шаг 4. Обновите таблицу CSS для компьютеров.

Откройте файл *lighthouse.css* в программе Блокнот (Notepad). Добавьте правило CSS для запрета отображения идентификатора `backtotop`.

```
#backtotop { display: none; }
```

Сохраните файл.

Шаг 5. Проверка работы.

Протестируйте файл *index.html* в браузере на компьютере. Страница должна выглядеть так, как показано на рис. 7.19. Затем просмотрите файл *index.html* в мобильном эмуляторе. Страница должна быть похожа на рис. 7.17 (смартфон на платформе iOS) или рис. 7.18 (приблизительный вариант в настольном браузере). Образец можно найти на прилагающемся к книге диске в папке *Примеры\Глава_07\mobile*.

Дополнительные сведения о медиазапросах

Современные веб-разработчики используют медиазапросы не только для того, чтобы ориентировать веб-сайт для смартфонов, но и с целью оптимизировать его отображение на других устройствах, в том числе планшетах и нетбуках.

На сайте Media Queries[1] представлена галерея сайтов, демонстрирующих ***адаптивный веб-дизайн***, который Этан Маркотт[2] описывает как прогрессивное улучшение веб-страницы для просмотра на различных устройствах (таких как смартфоны и планшеты) с помощью разнообразных методов, в том числе медиазапросов. Снимки страниц в галерее на сайте **mediaqueri.es** демонстрируют веб-страницы со следующей шириной экрана: 320 пикселов (экран смартфона), 768 пикселов (экран планшета), 1024 пикселов (экран нетбука) и 1600 пикселов (экран настольного компьютера/ноутбука). Для получения образца кода медиазапроса, ориентированного на различные устройства и типы экранов, посетите блог Hardboiled CSS3 Media Queries[3].

В примерах этой книги используются отдельные таблицы стилей для отображения на экранах компьютеров и мобильных устройств, однако можно включить все стили в одну таблицу стилей и указывать медиазапросы непосредственно в CSS с помощью ***правила*** `@media`. Ниже приведен пример кода, показывающий другое фоновое изображение специально для смартфона.

```
@media only screen and (max-device-width: 480px) {
#header { background-image: url(mobile.gif); }
```

[1] mediaqueri.es
[2] www.alistapart.com/articles/responsive-web-design
[3] www.stuffandnonsense.co.uk/blog/about/hardboiled_css3_media_queries

ЧаВо

СКАЗЫВАЕТСЯ ЛИ ИСПОЛЬЗОВАНИЕ МЕДИАЗАПРОСОВ НА ПРОПУСКНОЙ СПОСОБНОСТИ ИНТЕРНЕТ-КАНАЛА?

При использовании медиазапросов для нацеливания на мобильные устройства создается полезная и функциональная веб-страница для мобильных посетителей. Этот метод не улучшает пропускную способность интернет-канала. Несмотря на то, что медиазапросы по идее должны инициировать браузеры применять только определенные стили, современные браузеры по-прежнему загружают все таблицы стилей и ресурсы (например фоновые изображения), связанные с вашей веб-страницей, просто на всякий случай. Так, что рекомендуется оптимизировать все ваши изображения, чтобы уменьшить загруженность интернет-канала на мобильных устройствах. Будем надеяться, что последующие версии браузеров будут загружать только ресурсы, изначально необходимые для отображения страницы.

В этом разделе вы познакомились с мобильным веб-дизайном. Вы создали таблицу стилей, задающую для существующего веб-сайта макет мобильной веб-страницы в одну колонку. Если вы создаете новый веб-сайт, сначала примените альтернативный подход к разработке таблиц стилей для мобильных устройств, а затем разрабатывайте альтернативные таблицы стилей для планшетов и/или браузеров настольных ПК, которые постепенно улучшают дизайн, добавляя колонки и более крупные изображения. Вы можете узнать больше об этом подходе на сайте **www.lukew.com/ff/entry.asp?933**. Так как мобильные устройства развиваются, и доля их использования возрастает, несомненно, появятся новые разработки в этой области веб-дизайна.

Если вы хотите изучать эту тему дальше, следующие ресурсы будут вам полезны:

- Opera Mini: Руководство по созданию веб-контента: **dev.opera.com/articles/view/opera-mini-web-content-authoring-guidelines**
- Mobile Web Best Practices 1.0: **www.w3.org/TR/mobile-bp**
- Сайт Консорциума W3C: **http://www.w3.org/Mobile**
- Разработка страниц для мобильных устройств: **www.slideshare.net/estellevw/web-development-for-mobile**
- Первая помощь при серьезных проблемах: **www.lukew.com/ff/entry.asp?1117**

Глава 8
ТАБЛИЦЫ

Цели главы

В этой главе вы узнаете следующее:

- как создавать таблицы на веб-страницах;
- как применять атрибуты, форматирующие таблицы, строки таблицы и ячейки таблицы;
- как повышать доступность таблицы;
- как использовать каскадные таблицы стилей, чтобы конфигурировать HTML-таблицу.

Ранее таблицы часто использовались для форматирования макета веб-страницы, однако сегодня веб-разработчики выбирают CSS в качестве инструмента для создания макета. В этой главе вы познакомитесь с приемами верстки HTML-таблиц для организации информации на веб-странице.

8.1. Краткий обзор HTML-таблиц

Назначение таблицы — организация информации. В прошлом, до того, как браузеры стали поддерживать CSS, таблицы использовались для форматирования макета страницы. HTML-таблица состоит из рядов и столбцов, как электронная таблица. Каждая отдельная ячейка таблицы образована пересечением определенного ряда и столбца:

- каждая таблица начинается с тега `<table>` и заканчивается тегом `</table>`;
- каждая строка таблицы начинается с тега `<tr>` и заканчивается тегом `</tr>`;
- каждая ячейка (данные таблицы) начинается с тега `<td>` и заканчивается тегом `</td>`;
- ячейки таблицы могут содержать текст, изображения и другие HTML-элементы.

Рисунок 8.1 демонстрирует типовую таблицу с тремя строками, четырьмя столбцами и границей

Имя	Дата	Телефон
Вася	12.05	857-555-5555
Петя	28.11	303-555-5555

Рис. 8.1. Таблица с тремя строками, тремя столбцами и границей

Исходный HTML-код для таблицы, показанной на рис. 8.1, приведен ниже:

```
<table border="1">
<tr>
<td>Имя</td>
<td>Дата</td>
<td>Телефон</td>
</tr>
<tr>
<td>Вася</td>
<td>12.05</td>
<td>857-555-5555</td>
</tr>
<tr>
<td>Петя</td>
<td>28.11</td>
<td>303-555-5555</td>
</tr>
</table>
```

Обратите внимание, как таблица размечается в коде строка за строкой. Кроме того, каждая строка описывается последовательно ячейка за ячейкой. Пример можно найти на прилагающемся к книге диске (*Примеры\Глава_08\table1.html*).

Элемент таблицы

Элементы **`table`** — блочные и в них содержится информация о таблице. Таблица начинается с тега `<table>` и заканчивается тегом `</table>`. Основные атрибуты таблицы перечислены в табл. 8.1.

Таблица 8.1. Распространенные атрибуты элемента `table`

Атрибут	Значение	Назначение
`align`	`left` (по умолчанию), `right`, `center`	Выравнивание таблицы по горизонтали (устаревший атрибут, не используется в HTML5)
`bgcolor`	Допустимое значение цвета	Цвет фона таблицы (устаревший атрибут, не используется в HTML5)
`border`	0	Значение, установленное по умолчанию; нет видимой границы (устаревший атрибут, не используется в HTML5)
	1–100	Видимая граница, ширина которой указана в пикселах
`cellpadding`	Числовое значение	Определяет в пикселах величину отступа между содержимым ячейки таблицы и границей (устаревший атрибут, не используется в HTML5)
`cellspacing`	Числовое значение	Определяет в пикселах величину пробела между границами ячеек таблицы (устаревший атрибут, не используется в HTML5)
`summary`	Текстовое описание	Обеспечивает доступность путем предоставления текстового описания, содержащего обзор информации в таблице (на момент написания книги атрибут считался устаревшим в HTML5, но предпринимались усилия по возвращению к его использованию)
`title`	Текстовое описание	Краткое текстовое описание, предоставляющее обзор таблицы; в некоторых браузерах может отображаться как всплывающая подсказка
`width`	Числовое значение или величина в процентах	Определяет ширину таблицы (устаревший атрибут, не используется в HTML5)

Элемент описания таблицы

***Элемент* `caption`** часто используется с таблицей данных для описания ее содержимого. Он начинается с тега `<caption>` и заканчивается тегом `</caption>`. Текст, содержащийся в элементе `caption`, отображается на веб-странице над таблицей, но далее вы узнаете, что его положение можно задать с помощью CSS. В таблице, показанной на рис. 8.2, используются элементы `caption`, чтобы установить заглавие, «Голоса птиц». Заметьте, что элемент `caption` был помещен сразу после открывающего тега `table`. Пример вы можете найти на прилагающемся к книге диске (*Примеры\Глава_08\table2.html*).

Голоса птиц

Птица	Дата
Соловей	05.05.12
Ткачик	12.07.12

Рис. 8.2. Описание этой таблицы — «Голоса птиц»

HTML-код для таблицы, показанной на рис. 8.2, выглядит так:

```
<table border="1">
<caption>Голоса птиц</caption>
<tr>
<td>Птица</td>
<td>Дата</td>
</tr>
<tr>
<td>Соловей</td>
<td>05.05.12</td>
</tr>
<tr>
<td>Ткачик</td>
<td>12.07.12</td>
</tr>
</table>
```

8.2. Строки, ячейки и заголовки таблицы

Элемент строки таблицы

Элемент строки таблицы настраивает строку в таблице на веб-странице. Строка таблицы начинается с тега `<tr>` и заканчивается тегом `</tr>`. Часто используемые атрибуты элемента строки таблицы приведены в табл. 8.2.

Таблица 8.2. Часто используемые атрибуты элемента `tr`

Атрибут	Значение	Назначение
`align`	`left` (по умолчанию), `right`, `center`	Выравнивание таблицы по горизонтали (устаревший атрибут, не используется в HTML5)
`bgcolor`	Допустимое значение цвета	Цвет фона таблицы (устаревший атрибут, не используется в HTML5)

Элемент данных таблицы

Элемент данных таблицы задает ячейку в строке таблицы на веб-странице. Ячейка таблицы начинается с тега `<td>` и заканчивается тегом `</td>`. Часто используемые атрибуты элемента данных таблицы приведены в табл. 8.3.

Таблица 8.3. Часто используемые атрибуты элементов `td` и `th`

Атрибут	Значение	Назначение
`align`	`left` (по умолчанию), `right`, `center`	Выравнивание таблицы по горизонтали (устаревший атрибут, не используется в HTML5)
`bgcolor`	Допустимое значение цвета	Цвет фона таблицы (устаревший атрибут, не используется в HTML5)
`colspan`	Числовое значение	Число столбцов в ячейке
`headers`	Значение(я) идентификатора ячейки заголовка столбца или строки	Связывает ячейки данных таблицы с ячейками заголовка таблицы, может быть считан программами экранного доступа
`height`	Числовое значение или величина в процентах	Высота ячейки (устаревший атрибут, не используется в HTML5)
`rowspan`	Числовое значение	Число строк в ячейке
`scope`	`row`, `col`	Указывает на содержимое ячейки заголовка (заголовок для столбца или строки), может быть считан программами экранного доступа
`valign`	`top`, `middle` (по умолчанию) `bottom`	Выравнивание содержимого ячейки по вертикали (устаревший атрибут, не используется в HTML5)
`width`	Числовое значение или величина в процентах	Определяет ширину таблицы (устаревший атрибут, не используется в HTML5)

Элемент заголовка таблицы

Элемент заголовка таблицы похож на элемент данных и задает ячейку в строке таблицы на веб-странице. Его цель — конфигурирование столбцов и строк заголовков. Текст, отображаемый в элементе заголовка таблицы, отцентрирован и выделен полужирным. Элемент заголовка таблицы начинается с тега `<th>` и заканчивается тегом `</th>`. Часто используемые атрибуты элемента заголовки таблицы приведены в табл. 8.3. На рис. 8.3 показана таблица с заголовками столбцов, заданных с помощью элементов `th`. HTML-код для таблицы, показанной на рис. 8.3, приведен ниже (см. также файл *Примеры\Глава_08\table3.html* на прилагающемся к книге диске).

Имя	Дата	Телефон
Вася	12.05	857-555-5555
Петя	28.11	303-555-5555

Рис. 8.3. Использование элементов `th` для задания заголовков столбца

Обратите внимание, что для первой строки указан элемент `th` вместо `td`.

```
<table border="1">
<tr>
<th>Имя</th>
<th>Дата</th>
<th>Телефон</th>
</tr>
<tr>
<td>Вася</td>
<td>12.05</td>
<td>857-555-5555</td>
</tr>
<tr>
<td>Петя</td>
<td>28.11</td>
<td>303-555-5555</td>
</tr>
</table>
```

Практическое задание 8.1

На этом практическом задании вы создадите веб-страницу, похожую на отображенную на рис. 8.4 и описывающую историю обучения. Используйте описание «История обучения». В таблице будут три строки и три столбца. Сконфигурируйте первую строку с элементами `th` и заголовками «Учебное заведение», «Годы» и «Предмет». Вторую и третью строку заполните собственной информацией.

Для начала откройте файл *Примеры\Глава_02\template.html* в программе Блокнот (Notepad). Измените значение элемента `title`. Используйте элементы таблицы, строки, заголовка, данных и описания для настройки таблицы, как показано на рис. 8.4.

Советы:

- таблица состоит из трех строк и трех столбцов;
- чтобы задать границу, добавьте атрибут `border="1"` к элементу `table`;
- для устранения промежутка между границами ячеек используйте атрибут `cellspacing="0"` элемента `table`;
- отступ в каждой ячейке задайте с помощью атрибута `cellpadding="5"` элемента `table`;
- используйте элемент заголовка таблицы для ячеек в первой строке.

История обучения

Учебное заведение	Годы	Предмет
Средняя школа города Шаумбург	2006—2010	Подготовка к поступлению в колледж
Харпер-колледж	2010—2012	Интернет и веб-проектирование

Рис. 8.4. Таблица истории обучения

Пример файла вы найдете на диске, прилагающемся к книге (*Примеры\Глава_08\table4.html*). Код страницы не пройдет валидацию HTML5, так как вы используете устаревшие элементы, такие как `border`, `cellspacing` и `cellpadding`. В этой главе устаревшие атрибуты используются потому, что они допустимы в языке XHTML и все еще используются во Всемирной паутине. Позднее в этой главе вы научитесь конфигурировать таблицу посредством правил CSS.

8.3. Объединение строк и столбцов

Вы можете объединить ячейки, присвоив элементам `td` атрибуты `colspan` (объединение по горизонтали) и `rowspan` (объединение по вертикали). Поскольку вы начнете создавать более сложные конфигурации таблицы, то убедитесь, что сделали эскиз таблицы на бумаге прежде, чем начнете верстать HTML-код.

***Атрибут* `colspan`** определяет количество столбцов в строке, которое займет после объединения ячейка. Рисунок 8.5 демонстрирует строку, объединяющую два столбца

Объединение двух столбцов	
Столбец 1	Столбец 2

Рис. 8.5. Таблица со строкой, объединяющей два столбца

Исходный HTML-код для таблицы, показанной на рис. 8.5:

```
<table border="1">
<tr>
<td colspan="2">Объединение двух столбцов</td>
</tr>
<tr>
<td>Столбец 1</td>
<td>Столбец 2</td>
</tr>
</table>
```

***Атрибут* `rowspan`.** Этот атрибут задает количество строк, которое займет ячейка. Пример столбца, объединяющего две строки, показан на рис. 8.6.

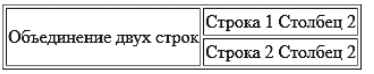


Рис. 8.6. Таблица со столбцом, объединяющим две строки

Исходный HTML-код для таблицы, показанной на рис. 8.6:

```
<table border="1">
<tr>
<td rowspan="2">Объединение двух строк</td>
<td>Строка 1 Столбец 2</td>
</tr>
<tr>
<td>Строка 2 Столбец 2</td>
</tr>
</table>
```

Примеры таблиц с рис. 8.5 и 8.6 можно найти на прилагающемся к книге диске, в папке *Примеры\Глава_08\table5.html*.

Практическое задание 8.2

На этом практическом задании вы потренируетесь работать с атрибутом `rowspan`. Чтобы создать веб-страницу, показанную на рис. 8.7,

запустите программу Блокнот (Notepad) и откройте страницу *Примеры\Глава_02\template.html* на прилагающемся к книге диске.

Маяк у кофейни	Оригинал построен: 1870
	Копия: 2011
	Высота: 100 метров

Рис. 8.7. Таблица со столбцом, объединяющим три строки

Сохраните файл с именем *myrowspan.html.* Измените значение элемента `title`. Конфигурируйте таблицу с помощью элементов `table`, `tr`, `th` и `td`.

1. Укажите в коде открывающий тег `<table>`. Конфигурируйте границу с помощью атрибута `border="1"`, уберите пространство между границами ячеек, применив атрибут `cellspacing="0"`, и задайте отступ внутри каждой ячейки, применив атрибут `cellpadding="5"`.
2. Начните первую строку с тега `<tr>`.
3. Ячейка данных таблицы с текстом «Маяк у кофейни» охватывает три строки. Укажите в коде элемент `td`. Используйте атрибут `rowspan="3"`.
4. Добавьте в код элемент `td`, содержащий текст «Оригинал построен: 1870».
5. Завершите первую строку тегом `</tr>`.
6. Начните вторую строку с тега `<tr>`. В этой строке будет всего один элемент `td` потому, что ячейка в первом столбце уже зарезервирована для текста «Маяк у кофейни».
7. Добавьте в код элемент `td`, содержащий текст «Копия: 2011».
8. Завершите вторую строку тегом `</tr>`.
9. Начните третью строку с тега `<tr>`. В этой строке будет всего один элемент `td` потому, что ячейка в первом столбце уже зарезервирована для текста «Маяк у кофейни».
10. Добавьте в код элемент `td`, содержащий текст «Высота: 100 метров».
11. Завершите вторую строку тегом `</tr>`.
12. Добавьте закрывающий тег `</table>`.

Сохраните файл и просмотрите его в браузере. Образец можно найти на прилагающемся к книге диске (*Примеры\Глава_08\table6.html*). Из-за использования атрибутов `border`, `cellpadding` и `cellspacing` ваша веб-страница не пройдет проверку кода HTML5. Тем не менее эти атрибуты еще действительны в синтаксисе XHTML.

Обратите внимание, как выровнен текст «Маяк у кофейни». Это заданное по умолчанию вертикальное выравнивание. Его вы можете изменить с помощью атрибута `valign` элемента `td`. Несмотря на то, что в данном примере используются атрибуты HTML, современный подход — это настройка таблицы с помощью CSS, что вы и изучите далее в этой главе.

8.4. Конфигурирование доступной таблицы

Таблицы бывают удобны при организации информации на странице, но что было бы, если бы вы не могли увидеть таблицу и были вынуждены прибегать к помощи вспомогательных средств, таких как программы экранного доступа, чтобы прочитать таблицу? Вы бы слушали, как программа читает контент таблицы в том порядке, как он указан в коде: строку за строкой, ячейку за ячейкой. Понять его было бы нелегко. В этой главе описаны методы верстки для улучшения доступности содержимого таблиц. Для простой информации, такой, как представленная на рис. 8.8, Консорциум W3C в рамках программы по обеспечению доступности веб-контента версии 2.0 (*Web Content Accessibility Guidelines 2.0 (WCAG 2.0)*) рекомендует следующее:

- использовать элементы `th` для обозначения заголовков строк или столбцов;
- использовать элемент `caption`, предназначенный для создания текстового заголовка или подписи к таблице.

Образец веб-страницы хранится на прилагающемся к книге диске (*Примеры\Глава_08\table7.html*).

Голоса птиц

Птица	Дата
Соловей	05.05.12
Ткачик	12.07.12

Рис. 8.8. В этой простой информационной таблице для улучшения доступности содержимого применены элементы `th` и `caption`

Исходный HTML-код следующий:

```
<table border="1">
<caption>Голоса птиц</caption>
<tr>
```

```
<th>Птица</th>
<th>Дата</th>
</tr>
<tr>
<td>Соловей</td>
<td>05.05.12</td>
</tr>
<tr>
<td>Ткачик</td>
<td>12.07.12</td>
</tr>
</table>
```

Однако для более сложных таблиц Консорциум W3C рекомендует специально связать значения ячеек данных таблицы с соответствующими заголовками. Предлагаемый метод предполагает использование атрибута `id` (как правило, в элементе `th`) для обозначения определенной ячейки заголовка и атрибут `headers` элемента `td`. Код для настройки таблицы, показанной на рис. 8.8, включает атрибуты `id` и `headers` и выглядит следующим образом (см. файл *Примеры\Глава_08\table8.html* на прилагающемся к книге диске):

```
<table border="1">
<caption>Голоса птиц</caption>
<tr>
<th id="name">Птица</th>
<th id="date">Дата</th>
</tr>
<tr>
<td headers="name">Соловей</td>
<td headers="date">05.05.12</td>
</tr>
<tr>
<td headers="name">Ткачик</td>
<td headers="date">12.07.12</td>
</tr>
</table>
```

ЧаВо

А КАК НАСЧЕТ АТРИБУТА SUMMARY?

На момент написания книги атрибут `summary` считался устаревшим, однако прилагаются усилия по его восстановлению. Этот атрибут использовался много лет и до сих пор действителен в XHTML. Атрибут `summary` позволяет кратко описать предназначение и структуру информации, содержащейся в таблице. Информация в атрибуте `summary` может быть доступна программам экранного доступа, но не отображается в окне браузера. Ниже приведен пример использования атрибута `summary` для таблицы, показанной на рис. 8.8:

```
<table border="1" summary="Список записанных голосов
певчих птиц. В первом столбце указывается название пти-
цы, а во втором - дата записи.">
```

Новым в спецификации HTML5 является то, что W3C предлагает следующие методы для обеспечения контекста к информации, содержащейся в таблице: конфигурировать описательный текст в элементе `caption`, добавить абзац с пояснением непосредственно на веб-страницу или упростить таблицу.

ЧаВо

ЧТО ТАКОЕ АТРИБУТ SCOPE?

Атрибут `scope` определяет связи ячеек таблицы с заголовками строк или столбцов таблицы. Он используется, чтобы указать, является ли ячейка таблицы заголовком для столбца (`scope="col"`) или строки (`scope="row"`). Пример того, как верстать таблицу на рис. 8.8, которая использует этот атрибут, показан ниже.

```
<table border="1">
<caption>Голоса птиц</caption>
<tr>
<th scope="col">Птица</th>
<th scope="col">Дата</th>
</tr>
<tr>
<td>Соловей</td>
<td>05.05.12</td>
</tr>
<tr>
<td>Ткачик</td>
```

```
<td>12.07.12</td>
</tr>
</table>
```

Если вы рассмотрели данный код, то, возможно, заметили, что использование атрибута `scope` с целью обеспечения доступности содержимого требует меньше кода, чем использование атрибутов `id` и `headers`. Однако из-за несовместимости поддержки с программами экранного доступа атрибута `scope` Инициативная группа по веб-доступности (WAI) Консорциума W3C рекомендует использовать при верстке стандарт WCAG 2.0 — атрибуты `headers` и `id`, а не `scope`.

8.5. Использование CSS для оформления таблиц

Раньше для оформления таблицы обычно использовались HTML-атрибуты. Более современный подход — применить правила CSS для оформления таблицы. В этом разделе вы научитесь использовать каскадные таблицы стилей для конфигурирования границы, отступа, выравнивания, ширины, высоты, вертикального выравнивания и фона элементов таблицы. Таблица 8.4 содержит список свойств CSS, соответствующих HTML-атрибутам, используемым для стилизации таблиц.

Таблица 8.4. Свойства CSS, используемые для стилей таблиц

HTML-атрибут	Свойство CSS

`align`	Чтобы выровнять таблицу, используйте свойства `width` и `margin` для селектора `table`.
	Например, чтобы отцентрировать таблицу:
	`table { width: 75%;`
	`margin: auto; }`
	Чтобы выровнять элементы внутри ячеек таблицы:
	`text-align`

`width`	`width`
`height`	`height`
`cellpadding`	`padding`
`cellspacing`	`border-spacing`; числовое значение в пикселах, единицах em или величина в процентах. Если установлено значение 0, единицу измерения можно опустить. Одно числовое значение с единицей измерения (px или em) задает одновременно интервалы по вертикали и горизонтали. Если используются два числовых значения с единицей измерения (px или em): первое задает интервал по горизонтали, а второе — по вертикали. `border-collapse` конфигурирует область границы. Его значения: `separate` (по умолчанию) и `collapse`. Для удаления лишнего пространства между таблицей и границей ячеек используйте свойство `border-collapse: collapse;`
`bgcolor`	`background-color`

HTML-атрибут	Свойство CSS
`valign`	`vertical-align`
`border`, `bordercolor`	`border`, `border-style`, `border-spacing`
нет	`background-image`
нет	`caption-side` — определяет местоположение подписи. Значения: `top` (по умолчанию) и `bottom`

Практическое задание 8.3

В этом практическом задании вы будете верстать код CSS, чтобы оформить информационную таблицу на веб-странице. Создайте новую папку с названием *ch8table*. Скопируйте в нее файл *starter.html* из каталога *Примеры\Глава_08* на прилагающемся к книге диске. Мы будем использовать глобальные стили для простоты редактирования и тестирования вашей страницы. Откройте файл *starter.html* в браузере; файл должен выглядеть так, показано как на рис. 8.9.

Кофейня на Арбатской - Специальное меню

Специальная цена	Описание	Цена
Лайт-латте	Прекрасный кофе Lavazza со вспененным молоком.	50 р.
Мокко-латте	Всем любителям шоколада - ароматный кофе с молоком и, на ваш выбор, темный, молочный или белый шоколад.	77 р.
Карамелька	Мокко-латте с добавлением ложки карамели в каждую чашку.	99 р.

Рис. 8.9. Эта таблица конфигурирована с помощью HTML

Запустите программу Блокнот (Notepad) или другой текстовый редактор и откройте файл *starter.html* из папки *ch8table.*

1. Просмотрите код веб-страницы и обратите внимание на атрибуты элемента `table` — `border`, `width`, `align`, `cellpadding` и `cellspacing`. Удалите эти атрибуты из элемента `table` таблицы. Вы будете использовать стили CSS, чтобы заменить функциональность этих атрибутов.
2. Сконфигурируйте селектор `table`. Задайте расположение глобальных стилей в разделе заголовка веб-страницы. Добавьте правило стиля для селектора `table` в области, которая конфигурирует таблицу, она должна быть отцентрирована, иметь границу и ширину 600 пикселов.

```
table { margin: auto;
border: 1px solid #5c743d;
width: 600px; }
```

Сохраните файл под именем *menu.html* и откройте свою страницу в браузере. Заметьте, что этот способ конфигурирует границу, окружающую всю таблицу, а не вокруг каждой ячейки таблицы.

3. Сконфигурируйте селекторы элементов `td` и `th`. Добавьте правило стиля, которое конфигурирует границу и отступ.

```
td, th { border: 1px solid #5c743d;
padding: 5px;
font-family: Arial, sans-serif; }
```

Сохраните файл и отобразите свою страницу в браузере. Каждая ячейка таблицы теперь обведена границей и текст в них набран шрифтом без засечек.

4. Обратите внимание на свободное пространство между границами ячеек таблицы. ***Свойство*** **`border-spacing`** может использоваться для устранения этого промежутка. Добавьте определение стиля `border-spacing: 0;` селектору `table`. Сохраните файл и откройте свою страницу в браузере.

5. Конфигурируйте подпись так, чтобы она отображалась шрифтом Verdana или другим установленным по умолчанию шрифтом без засечек и была отформатирована полужирным начертанием. Установите размер шрифта равным 1.2 em и нижний отступ в 5 пикселов. Задайте следующее правило стилей:

```
caption { font-family: Verdana, sans-serif;
font-weight: bold;
font-size: 1.2em;
padding-bottom: 5px; }
```

6. Теперь поэкспериментируем и зададим цвета фона для строк вместо границ ячейки. Измените правила стиля для селекторов `td` и `th` и удалите определение границы. Новое правило стиля для ячеек:

```
td, th { padding: 5px;
font-family: Arial, sans-serif; }
```

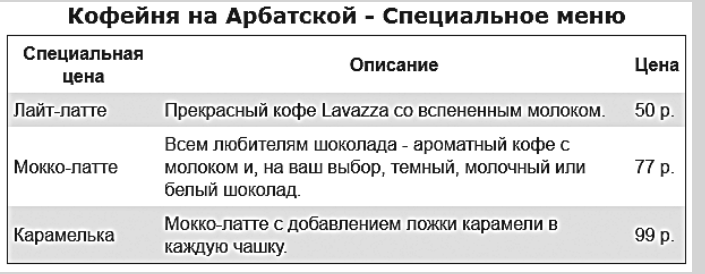
7. Создайте новый класс, `altrow`, который устанавливает цвет фона.

```
.altrow { background-color: #eaeaea; }
```

8. Измените элементы `tr` в HTML: назначьте второму и четвертому элементам `tr` класс `altrow`.

Сохраните страницу и отобразите ее в браузере. Область таблицы должна выглядеть подобной той, что показана на рис. 8.10.

Обратите внимание, как чередующийся фоновый цвет строк добавляет утонченную привлекательность веб-странице. Сравните свою страницу с файлом примера, расположенным в папке *Примеры\Глава_08\menu.html* на диске, прилагающемся к книге.



Кофейня на Арбатской - Специальное меню

Специальная цена	Описание	Цена
Лайт-латте	Прекрасный кофе Lavazza со вспененным молоком.	50 р.
Мокко-латте	Всем любителям шоколада - ароматный кофе с молоком и, на ваш выбор, темный, молочный или белый шоколад.	77 р.
Карамелька	Мокко-латте с добавлением ложки карамели в каждую чашку.	99 р.

Рис. 8.10. Цвет фона строк чередуется

В этом практическом задании вы настраивали внешний вид таблицы с помощью правил CSS. Эта техника верстки все чаще и чаще будет использоваться в будущем.

ЧаВо

ЕСТЬ ЛИ СПОСОБ СОЗДАТЬ ТАБЛИЧНЫЙ МАКЕТ СТРАНИЦЫ С ПОМОЩЬЮ CSS?

Да, если вы хотите использовать CSS для табличного дизайна веб-страниц, то изучите свойство `display`. Как вы помните из предыдущих глав, свойство `display` определяет, как должен отображаться элемент. Вы уже применяли свойства `display:none`, `display:block` и `display:inline`. Internet Explorer 8 был последним основным браузером, в который добавили поддержку значение свойства `display:table`. В своей статье «Everything You Know About CSS Is Wrong»[1] Рейчел Эндрю (Rachel Andrew) призывает разработчиков использовать значение свойства `display:table`.

Помните, что описываемая техника все еще весьма неполноценна. Например, нет возможности эмулировать атрибуты `rowspan` или `colspan` в HTML-таблицах. Однако эти возможности присутствуют в черновых рекомендациях CSS3[2], содержащих указания по работе с многоколоночными макетами и позиционированию по сетке.

8.6. Структурные псевдоклассы в CSS3

В предыдущем разделе вы конфигурировали правила CSS и применяли класс к каждой второй строке таблицы, чтобы задать чередующиеся цвета фона, часто называемые «полосатая разметка». Возможно, вам такой вариант показался неудобным и вас интересует, есть ли более

[1] 64.13.255.16/articles/everything_you_know_about_CSS_Is_wrong/
[2] www.w3.org/Style/CSS/current-work

эффективный метод. Да, есть! ***Селекторы структурных псевдоклассов*** в спецификации CSS3 позволяют выбирать и применять классы к элементам на основе их положения в структуре документа, к примеру, к каждой второй строке. Псевдоклассы CSS3 поддерживаются в текущих версиях браузеров Firefox, Opera, Chrome, Safari и Internet Explorer. Более ранние версии Internet Explorer (8 и ранее) такой поддержкой не обладают, поэтому попробуйте использовать этот прием только для улучшения веб-страниц. В таблице 8.5 перечислены распространенные селекторы структурных псевдоклассов CSS3 и их назначение.

Таблица 8.5. Распространенные структурные псевдоклассы CSS

Псевдокласс	Назначение
`:first-of-type`	Применяется к первому элементу указанного типа
`:first-child`	Применяется к первому дочернему элементу (селектор CSS2)
`:last-of-type`	Применяется к последнему элементу указанного типа
`:last-child`	Применяется к последнему дочернему элементу
`:nth-of-type(n)`	Применяется к элементу `"nth"` указанного типа. Значения: число, `odd` или `even`.

Чтобы применить псевдокласс, укажите его после селектора. В следующем примере кода задается отображение первого пункта неупорядоченного списка текстом красного цвета:

```
li:first-of-type { color: #FF0000; }
```

Практическое задание 8.4

На этом практическом задании вы будете дорабатывать таблицу, созданную в ходе практического задания 8.3, и примените селекторы структурных псевдоклассов, чтобы задать цвет.

1. Откройте файл *menu.html*, хранящийся в папке *ch8table*, в программе Блокнот (Notepad) (его также можно найти на прилагающемся к книге диске *Примеры\Глава_08\menu.html*). Сохраните его под именем *menu2.html.*
2. Просмотрите исходный код и обратите внимание, что второму и четвертому элементам `tr` присваивается класс `altrow`. Вам не потребуется этот класс, если вы используете селекторы структурных псевдоклассов CSS3. Удалите `class="altrow"` из элементов `tr`.
3. Проверьте правила CSS и найдите класс `altrow`. Измените селектор так, чтобы использовать структурный псевдокласс, кото-

рый будет применять стиль к четным строкам таблицы. Замените `.altrow` на `tr:nth-of-type (even)`, как показано в следующем определении CSS:

```
tr:nth-of-type(even) { background-color: #eaeaea; }
```

4. Сохраните файл и откройте страницу в браузере. Область таблицы должна быть похожа на показанную на рис. 8.10, если вы используете современный браузер, который поддерживает структурные псевдоклассы CSS3.
5. Давайте конфигурируем темно-серый цвет (# 666) первой строки и светло-серый текст (#eaeaea) с помощью структурного псевдокласса `:first-of-type`. Добавьте следующие правила CSS:

```
tr:first-of-type { background-color: #666;
color: #eaeaea; }
```

6. Сохраните файл и откройте страницу в браузере. Область таблицы должна быть похожа на показанную на рис. 8.11, если вы используете современный браузер, который поддерживает структурные псевдоклассы CSS3. Образец содержится на прилагающемся к книге диске (*Примеры\Глава_08\menucss3.html*).

Кофейня на Арбатской - Специальное меню

Специальная цена	Описание	Цена
Лайт-латте	Прекрасный кофе Lavazza со вспененным молоком.	50 р.
Мокко-латте	Всем любителям шоколада - ароматный кофе с молоком и, на ваш выбор, темный, молочный или белый шоколад.	77 р.
Карамелька	Мокко-латте с добавлением ложки карамели в каждую чашку.	99 р.

Рис. 8.11. Структурные псевдоклассы CSS3 задают стиль строк таблицы

Структурные псевдоклассы CSS удобны в использовании, но имейте в виду, что браузер Internet Explorer 8 и более ранние его версии не поддерживают эту технологию. Хотя поддержка браузеров со временем расширяется, на сегодняшний день лучше всего применять эти псевдоклассы лишь с целью прогрессивного улучшения.

8.7. Конфигурирование разделов таблицы

Существует несколько вариантов конфигурирования при верстке таблиц. Строки таблицы могут быть сгруппированы тремя путями: заголовок таблицы (`thead`), тело таблицы (`tbody`) и колонтитул таблицы (`tfoot`). Это может быть полезно, когда вам нужно по-разному сконфи-

гурировать области в таблице, используя или атрибуты, или каскадные таблицы стилей (смотрите раздел 8.3). Элемент `tbody` требуется для конфигурирования области `thead` или `tfoot`, хотя вы можете опустить или заголовок таблицы, или колонтитул таблицы, если вам так нравится. Когда вы группируете строки таблицы, то разделы `thead` и `tfoot` должны быть сверстаны *перед* разделом `tbody`, чтобы удовлетворить требованиям Консорциума W3C. В примере кода в этом разделе используется синтаксис HTML5, в котором раздел `tfoot` указан после `tbody`, что более интуитивно понятно. Следующий пример кода (см. файл *Примеры\Глава_08\tfoot.html)* конфигурирует таблицу, показанную на рис. 8.12, и демонстрирует использование таблицы CSS для конфигурирования заголовка, тела и колонтитула таблицы с различными стилями.

Расписание

День	Часов
Понедельник	4
Вторник	3
Среда	5
Четверг	3
Пятница	3
Всего	**18**

Рис. 8.12. Правила CSS конфигурируют селекторы `thead`, `tbody` и `tfoot`

Стили CSS задают отцентрированный заголовок таблицы шириной 200 пикселов, который представлен крупным полужирным шрифтом; раздел заголовка таблицы со светло-серым (`#eaeaea`) цветом фона; раздел тела таблицы, оформленный менее крупным (`.90em`) шрифтом Arial или любым другим без засечек; селекторы `td` устанавливают некоторый отступ слева и нижнюю пунктирную границу; раздел нижнего колонтитула таблицы отцентрирован, отформатирован полужирным шрифтом и светло-серым цветом фона (`#eaeaea`).

Исходный код таблицы CSS для таблицы, показанной на рис. 8.12, выглядит так:

```
table { width: 200px;
margin: auto; }
caption { font-size: 2em;
font-weight: bold; }
thead { background-color: #eaeaea; }
tbody { font-family: Arial, sans-serif;
```

```
font-size: .90em; }
tbody td { border-bottom: 1px #000033 dashed;
padding-left: 25px; }
tfoot { background-color: #eaeaea;
font-weight: bold;
text-align: center; }
```

Исходный HTML-код таблицы, показанной на рис. 8.12, выглядит следующим образом:

```
<table>
<caption>Расписание</caption>
<thead>
<tr>
<th id="day">День</th>
<th id="hours">Часов</th>
</tr>
</thead>
<tbody>
<tr>
<td headers="day">Понедельник</td>
<td headers="hours">4</td>
</tr>
<tr>
<td headers="day">Вторник</td>
<td headers="hours">3</td>
</tr>
<tr>
<td headers="day">Среда</td>
<td headers="hours">5</td>
</tr>
<tr>
<td headers="day">Четверг</td>
<td headers="hours">3</td>
</tr>
<tr>
```

```
<td headers="day">Пятница</td>
<td headers="hours">3</td>
</tr>
</tbody>
<tfoot>
<tr>
<td headers="day">Всего</td>
<td headers="hours">18</td>
</tr>
</tfoot>
</table>
```

Пример выше демонстрирует эффективность правил CSS в форматировании документов. Элементы `td` в пределах каждого селектора группы строк таблицы (`thead`, `tbody` и `tfoot`) унаследовали стили шрифта, конфигурируемые для их родительского селектора группы. Обратите внимание, как селектор-потомок конфигурирует отступ и границу только для элементов `td`, которые содержатся в пределах (фактически «дочерних элементов») элемента `tbody`. Файл с кодом, показанным выше, расположен на диске, прилагающемся к книге, в папке *Примеры\Глава_08\таблицы\tfoot.html*. Изучите исходный код и вид страницы в браузере.

Глава 9
ФОРМЫ

Цели главы

В этой главе вы узнаете следующее:

- общие приемы применения форм на веб-страницах;
- как создавать формы на веб-страницах, используя элементы `form`, `input`, `textarea` и `select`;
- как создавать формы, предоставляющие дополнительные специальные возможности, используя атрибуты `accesskey` и `tabindex`;
- как ассоциировать группы и элементы управления формы, используя элементы `label`, `fieldset` и `legend`;
- как создавать кнопки с графическими изображениями и использовать элемент `button`;
- как использовать таблицы CSS для стилизации форм;
- как конфигурировать новые элементы управления формы с помощью HTML5, в том числе поля ввода адреса электронной почты и URL-адреса, перечень данных, диапазон значений, счетчик, календари и палитру;
- описание возможностей и приемов обработки данных на стороне сервера;
- инициирование обработки данных формы на стороне сервера;
- бесплатные ресурсы, посвященные обработке данных на стороне сервера.

Формы применяются для различных целей по всей Всемирной паутине. Они используются поисковыми серверами, анализирующими ключевые слова, и интернет-магазинами для обработки покупок в электронных корзинах. Веб-сайты содержат формы, позволяющие выполнять разнообразные функции, — осуществлять обратную связь с посетителями, поощряя их к отправке новостей другу или коллеге, регистрировать адреса электронной почты для оформления подписки, принимать информацию о заказах и т. д. Эта глава рассматривает очень мощный инструмент веб-дизайнеров — формы, которые принимают данные от посетителей веб-страницы.

9.1. Краткий обзор форм

Каждый раз, когда вы пользуетесь поисковым сервером, размещаете заказ или подписываетесь на новостную рассылку, вы используете ***форму***. Форма — это HTML-элемент, содержащий и организующий объекты, называемые ***элементами управления формы*** — такие, как текстовые поля, флажки и кнопки, которые могут принять данные от посетителей веб-сайта.

Например, взгляните на веб-сайт Google, который осуществляет ввод поисковых запросов с помощью формы, показанной на рис. 9.1. Вы, возможно, использовали ее много раз, но никогда не думали, как она работает.

Форма весьма проста; она содержит только два элемента управления: текстовое поле, которое принимает ключевые слова и кнопку отправки данных **Поиск в Google**, с помощью которой можно начать поиск.

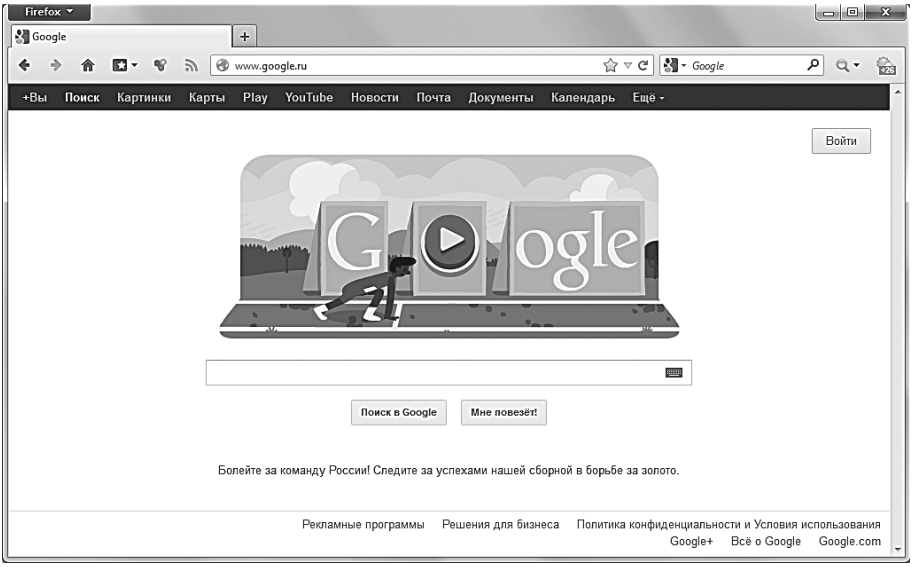


Рис. 9.1. Форма поиска на главной странице портала Google

Рисунок 9.2 демонстрирует более детализированную форму, используемую для оформления бланка банковского безналичного платежа на сайте **www.sbform.ru**. Эта форма содержит текстовые поля для ввода информации, например, **Наименование получателя** и **Адрес плательщика.**

Рис. 9.2. Информация в полях ввода формы используется для создания на сервере квитанции безналичного платежа

Раскрывающиеся списки (иногда называемые выпадающими списками) используются для ввода данных с ограниченным числом правильных значений, например, информацию о числе и месяце. Когда посетитель нажимает кнопку **Получить бланк**, информация, содержащаяся в форме, отправляется и начинает процесс оформления бланка. Если форма используется для поиска веб-страниц или публикации, одной формы недостаточно для завершения процесса обработки. Форма должна вызвать программу или скрипт на сервере, чтобы осуществить поиск в базе данных или сделать заказ. Обычно используются два компонента формы:

1. HTML-форма в виде пользовательского интерфейса веб-страницы.
2. Программное обеспечение для обработки данных на сервере, которое работает с данными формы и отправляет электронное письмо, осуществляет запись в текстовый файл, обновляет базу данных или выполняет какой-либо другой тип обработки на стороне сервера.

Элемент `form`

Теперь, когда вы освоили предназначение форм, сосредоточимся на HTML-коде, с помощью которого создается форма. ***Элемент*** **`form`** содержит форму веб-страницы. Тег `<form>` определяет начало кода формы на веб-странице. Закрывающий тег `</form>` определяет конец кода формы на веб-странице. На веб-странице может быть много форм, но они не могут быть вложены друг в друга. Элемент `form` может быть указан с атрибутами, которые определяют, как серверная программа или файл должны обработать форму, как данные формы будут отправлены на сервер, а также имя формы. Эти атрибуты перечислены в табл. 9.1.

Таблица 9.1. Атрибуты элемента `form`

Атрибут	Значение	Назначение
`action`	При вызове обработки на стороне сервера значение должно быть допустимым URL-адресом или именем файла/путем к файлу на веб-сервере	Обязателен; указывает, куда отправить информацию формы по ее завершении. Атрибут `mailto:e-mailaddress` запустит установленную по умолчанию у пользователя почтовую программу для отправки данных формы
`autocomplete`	`on`	Атрибут HTML5; значение, устанавливаемое по умолчанию. Браузер применит автозаполнение к полям формы
	`off`	Атрибут HTML5; браузер не применит автозаполнение к полям формы
`id`	Латинские буквы и цифры без пробелов. Значение должно быть уникальным и не использоваться для других `id` в том же самом документе HTML	Этот атрибут необязателен. Он предоставляет собой уникальный идентификатор для формы
`method`	`get`	Установлен по умолчанию; значение `get` применяется к данным формы, в результате чего они добавляются к URL-адресу и отправляются на веб-сервер
	`post`	Метод `post` более безопасен и передает данные формы в теле HTTP-ответа. Этот метод предпочтительнее по мнению Консорциума W3C
`name`	Латинские буквы и цифры, без пробелов, начинается с буквы. Выберите значение имени формы, которое является описательным, но коротким. Например, `OrderForm` лучше чем `Form1` или `WidgetsRUsOrderForm`.	Этот атрибут необязателен. Он присваивает форме имя так, чтобы к ней могли легко обращаться скриптовые языки на стороне клиента, например, JavaScript, чтобы редактировать и проверить информацию формы прежде, чем будет вызвана обработка на стороне сервера

Например, чтобы конфигурировать форму с именем `order`, используя метод `post` и вызвать скрипт *demo.php*, находящийся на вашем веб-сервере, исходный HTML-код должен быть следующим:

```
<form name="order" method="post" id="order"
action="cgi-bin/order.php">
...размещение элементов формы...
</form>
```

Элементы управления формы

Цель формы состоит в том, чтобы собрать информацию. Типы элементов управления формы включают текстовые поля, текстовые области с прокруткой, раскрывающиеся списки, переключатели, флажки и кнопки. HTML5 предлагает новые элементы управления формы, включая предназначенные для указания адресов электронной почты, URL-адресов, даты, времени, чисел и даже выбора даты. Соответствующие HTML-элементы будут рассмотрены в следующих разделах.

9.2. Элемент ввода данных

***Элемент* `input`** — автономный и используется для настройки нескольких различных типов элементов управления формы. В синтаксисе HTML5 он указывается в виде `<input>`, а в XHTML — как `<input />`. Используйте его для указания типа элемента управления формы, который браузер должен отобразить.

Текстовое поле

Тег `<input>` с атрибутом `type="text"` конфигурирует ***текстовое поле***. Этот элемент управления формы принимает текстовую или числовую информацию, например, имена, почтовые адреса, телефонные номера и другой текст. Общие атрибуты элемента `input` для текстового поля перечислены в табл. 9.2. Образец текстового поля показан на рис. 9.3.

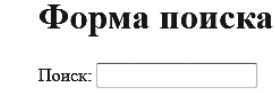


Рис. 9.3. Тег `input` с атрибутом `type="text"` конфигурирует текстовое поле формы

Таблица 9.2. Общие атрибуты элемента текстового поля

Атрибут	Значение	Назначение
`accesskey`	Символ клавиатуры	Задает клавишу быстрого вызова для элемента управления формы
`autocomplete`	`on`	Атрибут HTML5; установлен по умолчанию. Браузер выполнит автозаполнение элемента управления формы
	`off`	Атрибут HTML5. Браузер не применит автозаполнение элемента управления формы
`autofocus`	`autofocus`	Атрибут HTML5. Браузер размещает курсор в элементе управления формы и фокусируется на нем
`disabled`	`disabled`	Элемент управления формы отключен
`id`	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает уникальный идентификатор для элемента формы
`list`	Идентификатор элемента списка данных	Атрибут HTML5, связывает элемент управлении формы с элементом списка данных
`maxlength`	Числовое значение	Конфигурирует максимальную длину данных, допустимых для ввода в текстовое поле
`name`	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает имя элементу формы так, чтобы к нему могли легко обращаться скриптовые языки на стороне клиента (например JavaScript) или ПО при обработке на стороне сервера. Имя должно быть уникальным
`placeholder`	Латинские буквы или числа	Атрибут HTML5, краткая информация в помощь пользователю
`readonly`	`readonly`	Элемент управления формы только отображается, но его значение нельзя изменить
`required`	`required`	Атрибут HTML5, обязательное к заполнению поле — браузер проверяет наличие данных в нем прежде, чем отправить форму
`size`	Числовое значение	Конфигурирует ширину текстового поля для отображения браузером. Если размер не указан, то браузер отображает текстовое поле с собственным значением размера по умолчанию
`type`	`text`	Указывает, что используется текстовое поле
`value`	Латинские буквы или цифровые символы	Присваивает начальное значение, которое отображается браузером в поле. Принимает информацию, набранную в текстовом поле. К этому значению могут обращаться скриптовые языки на стороне клиента и ПО при обработке на стороне сервера

Некоторые атрибуты являются новыми в HTML5. Особенно интересен ***атрибут* `required`,** инициирующий проверку заполнения того или иного элемента формы. Браузеры, которые поддерживают этот атрибут, будут автоматически проверять, чтобы в текстовое поле была введена информация, и отображать сообщение об ошибке, если условие не выполняется. Исходный HTML-код:

```
E-mail: <input type="text" name="email" id="email"
required="required">
```

На рис. 9.4 показано сообщение об ошибке, автоматически генерирующееся браузером Firefox, которое отображается после того, как пользователь нажал на кнопку отправки данных, не введя информацию в обязательном текстовом поле формы.

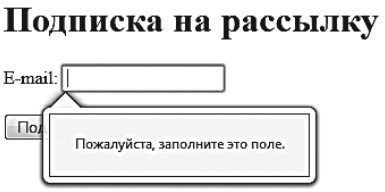


Рис. 9.4. Браузер Firefox отображает сообщение об ошибке

Браузеры, которые не поддерживают HTML5 или атрибут `required`, проигнорируют этот атрибут.

И хотя веб-дизайнеры в восторге от атрибута `required` и других новых функций обработки формы, предлагаемых HTML5, пройдет еще некоторое время, прежде чем все браузеры начнут поддерживать эти новые возможности. А пока помните, что проверку и валидацию информации формы рекомендуется производить на стороне клиента или с помощью скриптов на стороне сервера.

ЧаВо

ПОЧЕМУ АТРИБУТЫ `NAME` И `ID` ИСПОЛЬЗУЮТСЯ В ЭЛЕМЕНТАХ УПРАВЛЕНИЯ ФОРМЫ?

Атрибут `name` используется для присвоения имени элементу формы так, чтобы к нему могли легко обращаться скриптовые языки на стороне клиента, например JavaScript, или языки обработки на стороне сервера, например PHP. Значение, заданное атрибуту `name` для элемента формы, должно быть уникальным для этой формы.

Атрибут `id` используется в правилах CSS и в скриптах. Значение атрибута `id` должно быть уникальным для всей веб-страницы, которая содержит форму. Как правило, значения, присваиваемые атрибутам `name` и `id` конкретного элемента формы, идентичны.

Кнопка отправки данных

Этот элемент управления используется для отправки данных формы на веб-сервер. При нажатии ***кнопка отправки данных*** запускает метод действия этого элемента управления и браузер отправляет данные формы (пары имя и значение для каждого элемента управления). Веб-сервер вызовет программу обработки на стороне сервера или скрипт, перечисленный в свойстве действия формы. Кнопка отправки задается элементом `input` с атрибутом `type="submit"`. Например:

```
<input type="submit">
```

Кнопка сброса

Этот элемент управления формы используется для приведения полей формы к их начальным значениям. ***Кнопка сброса*** не отправляет форму. Она задается элементом `input` с атрибутом `type="reset"`. Например:

```
<input type="reset">
```

На рис. 9.5 показана форма с текстовым полем и двумя кнопками: отправки данных и сброса.

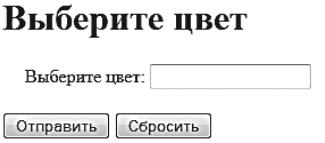


Рис. 9.5. Данная форма содержит текстовое поле и две кнопки: отправки данных и сброса

В данном примере значения по умолчанию, указываемые на кнопках, изменены следующим образом:

```
<input type="submit" value="Отправить"> <input
type="reset" value="Сбросить">
```

Общие атрибуты элементов кнопок отправки данных и сброса перечислены в табл. 9.3.

Таблица 9.3. Общие атрибуты кнопок отправки данных и сброса

Атрибут	Значение	Назначение
`accesskey`	Символ клавиатуры	Задает клавишу быстрого вызова для элемента управления формы
`id`	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает уникальный идентификатор для элемента формы
`name`	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает имя элементу формы так, чтобы к нему могли легко обращаться скриптовые языки на стороне клиента (например JavaScript) или ПО при обработке на стороне сервера. Имя должно быть уникальным
`tabindex`	Числовое значение	Задает порядок переключения элементов управления формы с помощью клавиш клавиатуры
`type`	`submit`	Конфигурирует кнопку отправки данных
	`reset`	Конфигурирует кнопку сброса
`value`	Текстовые или цифровые символы	Определяет текст, изображаемый на кнопке. По умолчанию отображается текст «Отправить запрос», а на кнопке сброса по умолчанию отображается текст «Сброс»[1]

[1] Текст на кнопке по умолчанию зависит от используемого браузера. В таблице указаны значения для браузера Firefox. — *Прим. пер.*

Практическое задание 9.1

На этом практическом задании вы создадите форму. Приступим. Запустите программу Блокнот (Notepad) и откройте файл *Примеры\Глава_02\ template.html,* хранящийся на прилагающемся к книге диске. Сохраните его с именем *form1.html*. Вы создадите веб-страницу с формой, похожей на пример, показанный на рис. 9.6.

1. Измените элемент `title`, указав текст «Пример формы».
2. Создайте элемент `h1` с текстом «Подписка на рассылку». Вы готовы сконфигурировать область формы. HTML-форма начинается с элемента `form`. Добавьте пустую строку под только что созданным заголовком и введите тег `<form>`, как показано ниже:

   ```
   <form method="get">
   ```

   В вашей первой форме вы будете использовать минимальное количество HTML-кода.

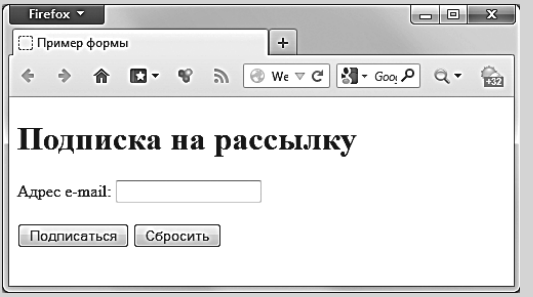


Рис. 9.6. Надпись на кнопке отправки данных: «Подписаться»

3. Чтобы создать элемент управления для ввода адреса электронной почты посетителя, надо ввести следующий HTML-код на пустой строке ниже элемента `form`:

   ```
   Адрес e-mail: <input type="text" name="email"
   id="email"> <br><br>
   ```

   Этот код помещает текст «Адрес e-mail:» перед текстовым полем, используемым для ввода электронного почтового адреса посетителя. Элементу `input` присвоен атрибут `type` со значением `text`, который инструктирует браузер отображать текстовое поле. Атрибут `name` присваивает имя `email` информации, введенной в текстовое поле (значение), и используется при обработке на стороне сервера. Атрибут `id` является уникальным идентификатором элемента на странице. Элементы `br` устанавливают разрывы строки.

4. Теперь вы можете добавить кнопку отправки формы следующей строкой. Добавьте атрибут `value` со значением «Подписаться»:

   ```
   <input type="submit" value="Подписаться">
   ```

   Этот исходный код инструктирует браузер отображать кнопку с надписью «Подписаться», вместо установленной по умолчанию кнопки «Отправить запрос»[1].

5. Оставьте пустое пространство после кнопки отправки данных и добавьте кнопку сброса:

   ```
   <input type="reset" value="Сбросить">
   ```

6. Наконец, вы готовы ввести закрывающий тег — `</form>`.

   Сохраните файл *form1.html*. Протестируйте свою страницу в браузере. Она должна выглядеть также, как показано на рис. 9.6. Пример файла вы найдете на диске, прилагающемся к книге, в папке *Примеры\Глава_09*.

Попытайтесь ввести информацию в свою форму. Попробуйте нажать кнопку отправки данных. Не беспокойтесь, если повторно

[1] Текст на кнопке по умолчанию зависит от используемого браузера. — *Прим. пер.*

отображается форма и ничего не происходит, когда вы нажимаете кнопку, так как вы не настроили эту форму для какой-либо обработки данных на стороне сервера. Подключение форм к обработке на стороне сервера рассматривается далее в этой главе. В следующем разделе вы детально рассмотрите другие элементы управления форм.

Флажок

Этот элемент управления формы позволяет пользователю выбирать один или несколько заранее определенных элементов. ***Флажок*** конфигурируется элементом `input` с атрибутом `type="checkbox"`. Общие атрибуты элемента флажка перечислены в табл. 9.4.

На рис. 9.7 показан пример с несколькими флажками. Обратите внимание, что пользователь может выбрать несколько флажков.

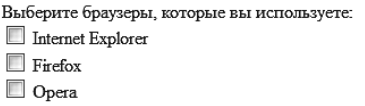


Рис. 9.7. Пример страницы с группой флажков

Исходный HTML-код выглядит так:

```
Выберите браузеры, которые вы используете: <br>
<input type="checkbox" name="IE" id="IE"
value="yes"> Internet Explorer<br>
<input type="checkbox" name="Firefox" id="Firefox"
value="yes"> Firefox<br>
<input type="checkbox" name="Opera" id="Opera"
value="yes"> Opera<br>
```

Таблица 9.4. Общие атрибуты элемента флажка

Атрибут	Значение	Назначение
`accesskey`	Символ клавиатуры	Задает клавишу быстрого вызова для элемента управления формы
`autofocus`	`autofocus`	Атрибут HTML5. Браузер размещает курсор в элементе управления формы и фокусируется на нем

Атрибут	Значение	Назначение
checked	checked	Определяет, установлен ли флажок по умолчанию при отображении страницы в браузере
disabled	disabled	Элемент управления формы отключен
id	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает уникальный идентификатор элементу формы
name	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает имя элементу формы так, чтобы к нему могли легко обращаться скриптовые языки на стороне клиента или ПО при обработке на стороне сервера. Имя каждого флажка должно быть уникальным
readonly	readonly	Элемент управления формы только отображается, но его значение нельзя изменить
required	required	Атрибут HTML5, обязательное к заполнению поле — браузер проверяет наличие данных в нем прежде, чем отправить форму
tabindex	Числовое значение	Задает порядок переключения элементов управления формы с помощью клавиш клавиатуры
type	checkbox	Указывает, что данный элемент формы — флажок
value	Латинские буквы или цифровые символы	Присваивает значение, которое выполняется при установке флажка. К этому значению могут обращаться скриптовые языки на стороне клиента и ПО при обработке на стороне сервера

Переключатель

Этот элемент управления формы позволяет пользователю выбирать один (и только один) вариант из группы заранее определенных элементов. Всем ***переключателям*** в группе присваивают одинаковое имя атрибута и уникальное значение. Поскольку имя одно и то же, элементы определены как часть группы, и только один из них может быть выбран. Переключатель конфигурируется элементом `input` с атрибутом `type="radio"`. Образец группы переключателей показан на рис. 9.8. Обратите внимание, что одновременно пользователем может быть выбран только один элемент. Общие атрибуты элемента переключателя перечислены в табл. 9.5.

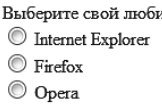


Рис. 9.8. Используйте переключатели, если только один вариант правильный

Исходный HTML-код для формы, показанной на рис. 9.8, выглядит так:

```
Выберите свой любимый браузер:<br>
<input type="radio" name="favbrowser" id="favIE"
value="IE"> Internet Explorer<br>
<input type="radio" name="favbrowser"
id="favFirefox"
value="Firefox"> Firefox<br>
<input type="radio" name="favbrowser" id="favOpera"
value="Opera"> Opera<br>
```

Обратите внимание, что у всех атрибутов `name` одинаковое значение — `favbrowser`. Таким образом создается группа. Каждый переключатель в той же самой группе может быть однозначно определен ее ***атрибутом*** **`value`**. Каждый переключатель в той же самой группе сконфигурирован с различным значением.

Таблица 9.5. Общие атрибуты элемента переключателя

Атрибут	Значение	Назначение
`accesskey`	Символ клавиатуры	Задает клавишу быстрого вызова для элемента управления формы
`autofocus`	`autofocus`	Атрибут HTML5. Браузер размещает курсор в элементе управления формы и фокусируется на нем
`checked`	`checked`	Определяет, что переключатель был установлен по умолчанию при отображении страницы в браузере
`disabled`	`disabled`	Элемент управления формы отключен
`id`	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает уникальный идентификатор для элемента формы
`name`	Латинские буквы и цифры, без пробелов, начинается с буквы	Обязательный параметр. У всех переключателей в группе должно быть то же самое имя. Этот атрибут также присваивает имя элементу формы так, чтобы к нему могли легко обращаться скриптовые языки на стороне клиента или ПО при обработке на стороне сервера
`readonly`	`readonly`	Элемент управления формы только отображается, но его значение нельзя изменить
`required`	`required`	Атрибут HTML5, обязательное к заполнению поле — браузер проверяет наличие данных в нем прежде, чем отправить форму

Атрибут	Значение	Назначение
tabindex	Числовое значение	Задает порядок переключения элементов управления формы с помощью клавиш клавиатуры
type	radio	Указывает, что используется переключатель
value	Латинские буквы или цифровые символы	Присваивает значение переключателю, которое выполняется, когда переключатель установлен. Это должно быть уникальное значение для каждого переключателя в группе. К этому значению могут обращаться скриптовые языки на стороне клиента и ПО при обработке на стороне сервера

Скрытый элемент формы

Этот элемент управления формы хранит текстовую или числовую информацию, но не отображается на веб-странице. ***Скрытые элементы формы*** могут быть доступны скриптам как на стороне клиента, так и на стороне сервера. Скрытый элемент формы конфигурируется элементом `input` с атрибутом `type="hidden"`. Общие атрибуты скрытого элемента формы перечислены в табл. 9.6. HTML-код для создания скрытого элемента управления формы с именем `sendto` и значением в виде адреса электронной почты следующий:

```
<input type="hidden" name="sendto" id="sendto"
value="order@site.com">
```

Таблица 9.6. Общие атрибуты скрытого элемента формы

Атрибут	Значение	Назначение
disabled	disabled	Элемент управления формы отключен
id	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает уникальный идентификатор для элемента формы
name	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает имя элементу формы так, чтобы к нему могли легко обращаться скриптовые языки на стороне клиента (например JavaScript) или ПО при обработке на стороне сервера. Имя должно быть уникальным
type	hidden	Указывает, что используется скрытый элемент формы
value	Латинские буквы или цифровые символы	Присваивает значение скрытому элементу управления формы. К этому значению могут обращаться скриптовые языки на стороне клиента и ПО при обработке на стороне сервера

Поле ввода пароля

Поле ввода пароля похоже на текстовое поле, но оно принимает информацию, которая должна быть скрыта во время ввода, например, символы пароля. Это поле конфигурируется элементом `input` с атрибутом `type="password"`. Когда пользователь вводит информацию в поле пароля, то вместо символов будут отображены звездочки (*), как показано на рис. 9.9.

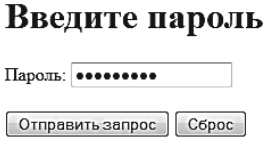


Рис. 9.9. Был напечатан текст secret999, но браузер скрыл введенные символы (*примечание*: ваш браузер может использовать другой значок, чтобы скрыть вводимые символы, например, стилизованный кружок)

Поле ввода пароля скрывает информацию от того, кто смотрит через плечо печатающего человека. Фактически введенные символы отправлены на сервер и информация на самом деле не является зашифрованной и скрытой (см. главу 12, чтобы узнать больше о шифровании и безопасности).

Поле ввода пароля — это специальное текстовое поле. Общие атрибуты элемента поля ввода пароля перечислены в табл. 9.2. Исходный HTML-код выглядит так:

```
Пароль: <input type="password" name="myPassword"
id="myPassword">
```

9.3. Текстовая область с прокруткой

Элемент `textarea`

Текстовая область с прокруткой используется для принятия в свободной форме замечаний, вопросов или комментариев. Данный элемент формы задается ***тегом*** **`textarea`**. Открывающий тег `<textarea>` обозначает начало ***текстовой области с прокруткой***. Закрывающий тег `</textarea>` обозначает ее конец. Текст, который вы помещаете между открывающим и закрывающим тегами, будет отображен в текстовой области с прокруткой. Пример показан на рис. 9.10.

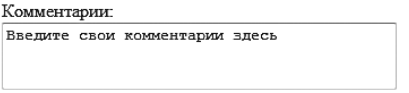


Рис. 9.10. Текстовая область с прокруткой

Исходный HTML-код приведен ниже:

```
Комментарии:<br>
<textarea name="comments" id="comments" cols="40"
rows="2"> Введите свои комментарии здесь</textarea>
```

Общие атрибуты текстовой области с прокруткой перечислены в табл. 9.7.

Таблица 9.7. Общие атрибуты текстовой области с прокруткой

Атрибут	Значение	Назначение
`accesskey`	Символ клавиатуры	Задает клавишу быстрого вызова для элемента управления формы
`autofocus`	`autofocus`	Атрибут HTML5. Браузер размещает курсор в элементе управления формы и фокусируется на нем
`cols`	Числовое значение	Обязательное. Определяет ширину текстовой области с прокруткой в символах. Если атрибут `cols` пропущен, то браузер отображает текстовую область с прокруткой с собственным значением ширины, установленным по умолчанию
`disabled`	`disabled`	Элемент управления формы отключен
`id`	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает уникальный идентификатор для элемента формы
`maxlength`	Числовое значение	Конфигурирует максимальную длину данных, допустимую для ввода в текстовое поле
`name`	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает имя элементу формы так, чтобы к нему могли легко обращаться скриптовые языки на стороне клиента (например JavaScript) или ПО при обработке на стороне сервера. Имя должно быть уникальным
`placeholder`	Латинские буквы или цифровые символы	Атрибут HTML5, краткая информация в помощь пользователю
`readonly`	`readonly`	Элемент управления формы только отображается, но его значение нельзя изменить

Атрибут	Значение	Назначение
`required`	`required`	Атрибут HTML5, обязательное к заполнению поле — браузер проверяет наличие данных в нем прежде, чем отправить форму
`rows`	Числовое значение	Обязательное. Определяет высоту текстовой области с прокруткой в строках. Если атрибут `rows` пропущен, то браузер отображает текстовую область с прокруткой с собственным значением высоты, установленным по умолчанию
`tabindex`	Числовое значение	Задает порядок переключения элементов управления формы с помощью клавиш клавиатуры
`wrap`	`hard` или `soft`	Атрибут HTML5, задает разрывы строк вводимой информации

Практическое задание 9.2

На этом практическом задании вы создадите форму (рис. 9.11) со следующими элементами управления: текстовые поля **Имя, Фамилия, Адрес e-mail** и текстовая область с прокруткой для комментариев. В качестве начальной точки вы будете использовать форму, которую создали на практическом задании 9.1 (см. рис. 9.6).

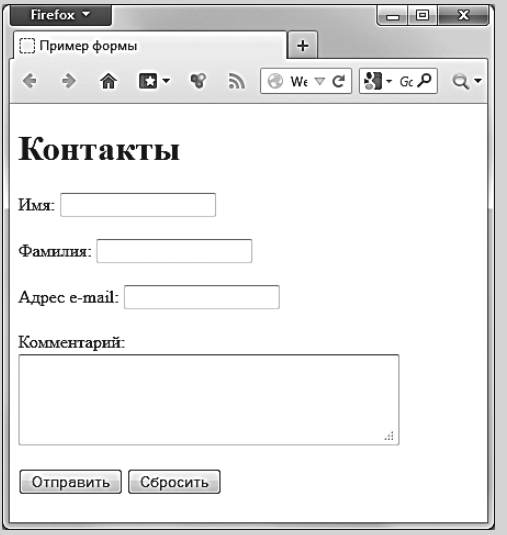


Рис. 9.11. Стандартная форма контакта

Запустите программу Блокнот (Notepad) и откройте страницу *Примеры\Глава_09\form1.html* на прилагающемся к книге диске. Сохраните ее под именем *form2.html*.

1. Измените элемент `title` так, чтобы он содержал текст «Пример формы».
2. Создайте элемент `h1` с текстом «Контакты».
3. Элемент управления формы для поля адреса электронной почты уже указан в коде. Взгляните на рис. 9.11 и обратите внимание, что над полем адреса электронной почты вам потребуется добавить текстовые поля **Имя** и **Фамилия**. На новой строке перед открывающим тегом `<form>` добавьте следующий код, чтобы принять имя пользователя, заполняющего форму.

```
Имя: <input type="text" name="fmail"
id="fmail"><br><br>

Фамилия: <input type="text" name="lmail"
id="lmail"><br><br>
```

4. Теперь вы готовы добавить в форму текстовую область с прокруткой с помощью элемента `textarea`, на новой строке под полем адреса электронной почты. HTML-код следующий:

```
Комментарий :<br>

<textarea name="comments" id="comments"></
textarea><br><br>
```

   Сохраните файл и откройте веб-страницу в браузере, чтобы посмотреть, как отображается текстовая область с прокруткой с настройками по умолчанию. Обратите внимание, что отображение по умолчанию будет различаться в зависимости от браузера. На момент написания этой книги браузер Internet Explorer изначально отображал вертикальную полосу прокрутки, а в браузере Firefox полоса прокрутки появлялась только после того, как в поле было введено достаточно текста, чтобы она стала необходима.

5. Сконфигурируйте атрибуты `rows` и `cols` для текстовой области с прокруткой. Измените тег `textarea` и установите значения атрибутов `rows="4"` и `cols="40"`, как показано в листинге ниже:

```
Комментарий:<br>

<textarea name="comments" id="comments" rows="4"
cols="40"></textarea><br><br>
```

6. Далее измените текст, отображаемый на кнопках отправки и сброса данных формы. Установите любые значения.

   Сохраните файл *form2.html* и протестируйте его в браузере. Он должен выглядеть так же, как показано на рис. 9.11. Также вы можете просмотреть одноименный файл в папке *Примеры\Глава_09* на прилагающемся к книге диске.

ЧаВо

КАК Я МОГУ ОТПРАВИТЬ ДАННЫЕ, ВВЕДЕННЫЕ В ФОРМУ, НА АДРЕС ЭЛЕКТРОННОЙ ПОЧТЫ?

Формы обычно должны провоцировать некоторый способ обработки на стороне сервера, чтобы выполнить нужные функции, например, отправку электронной почты, запись в текстовый файл, обновление базы данных и т. д. Другим вариантом является создание формы для отправки информации с помощью программы электронной почты, настроенной по умолчанию на компьютере посетителя веб-страницы. Для этого элементу `form` присваивается атрибут `action`, значением которого является адрес электронной почты и текст `mailto:`:

```
<form method="post" action="mailto:lsnblf@yahoo.com">
```

Когда форма будет использоваться этим способом, посетитель веб-сайта увидит предупреждающее сообщение. Предупреждающее сообщение вызывает тревогу посетителя и может отрицательно сказаться на репутации вашего веб-сайта или бизнеса.

Знайте, что таким способом передавать информацию небезопасно. Важная информация, например, номер банковской карточки, не должна передаваться с использованием электронной почты. См. главу 12, чтобы узнать о шифровании передаваемых данных.

Есть другие причины не использовать значение `mailto:`*адрес_электронной_почты*. Например, когда несколько пользователей совместно используют один и тот же компьютер, — они, возможно, не работают с приложением электронной почты, настроенным по умолчанию. В этом случае заполнение формы — это пустая трата времени. Даже если пользователь и использует приложение электронной почты по умолчанию, возможно, он или она не хотят разглашать этот определенный адрес электронной почты. Возможно, у них есть другой почтовый адрес, который используется для форм и новостных рассылок, и они не хотят напрасно тратить время, заполняя вашу форму. Так или иначе результат — недовольство посетителя веб-сайта. Использовать значение `mailto:`*адрес_электронной_почты* легко, но с его помощью не всегда удается создать наиболее удобные веб-формы для ваших посетителей. Что должен сделать веб-разработчик? Обрабатывайте данные форм на стороне сервера (см. практическое задание 9.5) вместо значения `mailto:`*адрес_электронной_почты*.

9.4. Раскрывающийся список

Элемент управления ***раскрывающийся список***, показанный на рис. 9.12 и 9.13, известен под разными именами: раскрывающийся спи-

сок, меню выбора, выпадающее меню, раскрывающееся окно, список опций и т. п. Раскрывающийся список задается с помощью одного элемента `select` и нескольких элементов `option`.

Элемент select

Элемент `select` содержит и задает раскрывающийся список. Открывающий тег `<select>` обозначает начало раскрывающегося списка, закрывающий тег `</select>` обозначает конец списка. Атрибуты задают, сколько вариантов будет отображаться, и указывают, можно ли выбрать несколько из них одновременно. Общие атрибуты элемента `select` перечислены в табл. 9.8.

Таблица 9.8. Общие атрибуты элемента `select`

Атрибут	Значение	Назначение
`disabled`	`disabled`	Элемент управления формы отключен
`id`	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает уникальный идентификатор для элемента формы
`multiple`	`multiple`	Конфигурирует раскрывающийся список, позволяющий выбрать более одного варианта. По умолчанию только один вариант может быть выбран из раскрывающего списка
`name`	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает имя элементу формы так, чтобы к нему могли легко обращаться скриптовые языки на стороне клиента (например JavaScript) или ПО при обработке на стороне сервера. Имя должно быть уникальным
`size`	Числовое значение	Настраивает количество вариантов, которые отобразит браузер. Если значение установлено 1, то элемент функционирует как раскрывающийся список (см. рис. 9.10). Полосы прокрутки автоматически добавляются в браузере, если число вариантов больше указанного размера списка
`tabindex`	Числовое значение	Задает порядок переключения элементов управления формы с помощью клавиш клавиатуры

Элемент `option`

***Элемент* `option`** содержит и задает пункт, отображаемый в раскрывающемся списке. Открывающий тег `<option>` обозначает начало элемента, закрывающий тег `</option>` обозначает его конец. Атрибуты задают значение элемента `option` и является ли он выбранным по умолчанию. Общие атрибуты элемента `option` перечислены в табл. 9.9.

Таблица 9.9. Общие атрибуты элемента `option`

Атрибут	Значение	Назначение
`disabled`	`disabled`	Элемент управления формы отключен
`selected`	`selected`	Определяет пункт, который будет выбран по умолчанию при отображении страницы в браузере
`value`	Латинские буквы или цифровые символы	Присваивает значение элементу `option`. К этому значению может обращаться ПО при обработке на стороне клиента и сервера

Исходный HTML-код формы, показанной на рис. 9.12, выглядит так:

```
<select size="1" name="favbrowser" id="favbrowser">
<option>Выберите любимый браузер</option>
<option value="Internet Explorer">Internet
Explorer</option>
<option value="Firefox">Firefox</option>
<option value="Opera">Opera</option>
</select>
```

Отображается один элемент

Рис. 9.12. Раскрывающийся список, значение атрибута `size` которого равно 1, функционирует как меню выбора

Исходный HTML-код формы, показанной на рис. 9.13, выглядит так:

```
<select size="4" name="jumpmenu" id="jumpmenu">
<option value="index.html">Главная</option>
<option value="products.html">Продукция</option>
<option value="services.html">Услуги</option>
<option value="about.html">О нас</option>
<option value="contact.html">Контакты</option>
</select>
```

Отображается 4 элемента

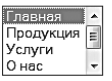


Рис. 9.13. Поскольку вариантов больше четырех, браузер отображает полосу прокрутки

ЧаВо

КАКИМ ОБРАЗОМ МЕНЮ, ПОКАЗАННОЕ НА РИС. 9.13, ОТОБРАЖАЕТ ВЫБРАННУЮ СТРАНИЦУ?

Вообще-то оно еще не работает. Необходим сценарий JavaScript (см. главу 14), чтобы проверить выбранный элемент и отправить браузеру инструкцию открыть файл. Поскольку для работы требуется JavaScript, этот тип меню не подойдет для главной области навигации, но может быть полезным для вторичной навигации.

9.5. Кнопки-изображения и элемент `button`

Поскольку вы работали с формами в этой главе, то, возможно, заметили, что стандартная кнопка отправки данных (см. рис. 9.11) весьма некрасива. Вы можете сделать эту кнопку более привлекательной и визуально интересной двумя способами:

1. Конфигурировать изображение, настроив элемент `input`.
2. Создать пользовательское изображение, заданное с помощью элемента **`button`.**

Кнопка-изображение

На рис. 9.14 показана форма с изображением, используемым вместо стандартной кнопки отправки данных. Это так называемая ***кнопка-изображение***. Когда ее нажимают или касаются на сенсорном экране, форма отправляется. Кнопка-изображение создана с использованием элемента `input` с атрибутом `type="image"` и атрибутом `src`, значение которого — имя графического файла. Например, чтобы использовать рисунок *login.gif* как кнопку-изображение, HTML-код будет следующим:

```
<input type="image" src="login.gif"
alt="Регистрация">
```

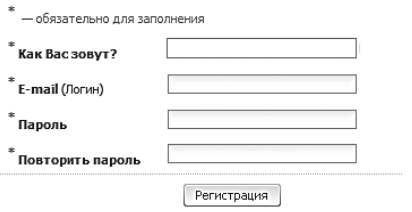


Рис. 9.14. Посетитель веб-страницы нажимает кнопку-изображение, чтобы отправить форму

Элемент `button`

Другой способ сделать форму интереснее — использовать ***элемент* `button`**, который можно применить для конфигурирования не только изображения, но и блоков текста, как активизирующуюся щелчком область, которая может отправить или сбросить форму. Любое содержимое веб-страницы, которое находится между тегами `<button>` и `</button>`, конфигурируется как часть кнопки. В таблице 9.10 перечислены общие атрибуты элемента `button`.

Таблица 9.10. Общие атрибуты элемента `button`

Атрибут	Значение	Назначение
`alt`	Краткое текстовое описание изображения	Обеспечивает доступность элемента для посетителей, неспособных рассмотреть изображение
`id`	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает уникальный идентификатор для элемента формы
`name`	Латинские буквы и цифры, без пробелов, начинается с буквы	Присваивает имя элементу формы так, чтобы к нему могли легко обращаться скриптовые языки на стороне клиента (например JavaScript) или ПО при обработке на стороне сервера. Имя должно быть уникальным
`type`	`submit`	Функционирует как кнопка отправки данных
	`reset`	Функционирует как кнопка сброса
	`button`	Функционирует как кнопка
`value`	Латинские буквы или цифровые символы	Значение, переданное элементу формы, отправляется обработчику формы

Рисунок 9.15 демонстрирует форму с изображением (*signup.gif*), сконфигурированным как кнопка отправки данных при помощи элемента `button`

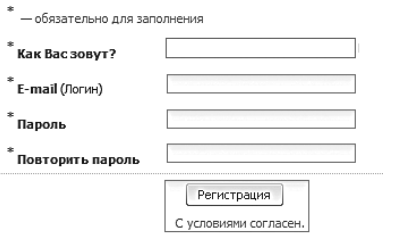


Рис. 9.15. Элемент `button` сконфигурирован как кнопка отправки данных

Исходный HTML-код формы, показанной на рис. 9.15, выглядит следующим образом:

```
<button type="submit">
<img src="signup.gif" width="80" height="28" alt="С
условиями согласен.">
</button>
```

Если вы просматриваете исходный код посещаемых веб-страниц, то заметите, что элемент `button` используется не так часто, как стандартная кнопка отправки данных или кнопка-изображение.

9.6. Доступность и формы

В этом разделе вы изучите методы повышения доступности элементов формы, таких как `label`, `fieldset` и `legend`, а также атрибутов `tabindex` и `accesskey`, которые облегчают использование ваших веб-страниц людьми с нарушениями зрения и двигательной активности. Применение элементов `label`, `fieldset` и `legend` может сделать веб-форму более удобочитаемой для всех пользователей и повысить ее юзабилити.

Элемент `label`

Элемент **`label`** — контейнерный тег, который используется, чтобы связать текстовое описание с элементом управления формы. Человеку, использующему программу экранного доступа, иногда бывает трудно связать текстовые описания на формах с их соответствующими элементами управления. Элемент `label` также приносит пользу людям с затруднениями в координации и моторике. Таким людям легче активировать элемент управления, щелкая или по элементу управления формы, или по его связанной текстовой метке. Есть два способа связать метку с элементом управления формы.

1. Первый метод — поместить контейнерный элемент `label` вокруг текстового описания и элемента HTML-формы. Код в этом случае следующий:

   ```
   <label>E-mail: <input type="text" name="email"
   id="email"></label>
   ```

2. Второй метод задействует атрибут `for`, чтобы связать метку с определенным элементом HTML-формы. Это более гибкий подход

и лучше подходит для форм, которые отформатированы таблицей. Пример кода выглядит так:

```
<label for="email">E-mail: </label>
<input type="text" name="email" id="email">
```

Обратите внимание, что значение ***атрибута*** **`for`** элемента `label` является тем же самым, что и значение атрибута `id` элемента `input`. Так создается ассоциация между элементом `label` и элементом управления формы. Элемент `input` использует оба атрибута, `name` и `id`, в различных целях. Атрибут `name` может использоваться на стороне клиента и на стороне сервера. Атрибут `id` создает идентификатор, который может использоваться элементами привязки, элементом `label` и селекторами CSS. Элемент `label` не отображается на веб-странице — он работает «за кадром», обеспечивая доступность.

Практическое задание 9.3

На этом практическом задании вы добавите элемент `label` в текстовое поле и текстовую область с прокруткой формы, созданной в ходе практического задания 9.2 (см. рис. 9.11). Эта форма используется как отправная точка. Запустите программу Блокнот (Notepad) и откройте страницу *Примеры\Глава_09\form2.html* на прилагающемся к книге диске. Сохраните ее под именем *form3.html*.

1. Найдите текстовое поле **Имя**. Добавьте элемент `label`, заключив в него элемент `input` следующим образом:

   ```
   <label>Имя: <input type="text" name="fname"
   id="fname"></label>
   ```

2. Применив показанный выше метод, добавьте элемент `label` полям **Фамилия** и **Адрес e-mail**.
3. Конфигурируйте элемент `label`, содержащий текст «Комментарий». Свяжите элемент `label` с текстовой областью с прокруткой. Образец кода:

   ```
   <label for="comments">Комментарий:</label><br>
   <textarea name="comments" id="comments" rows="4"
   cols="40"></textarea>
   ```

   Сохраните файл *form3.html* и протестируйте веб-страницу в браузере. Она должна быть похожа на показанную на рис. 9.11. Элементы `label` не изменяют внешний вид страницы, но посетителям с физическими недостатками должно быть проще использовать эту форму.

Вы можете сравнить вашу работу с примером, записанном на прилагающемся к книге диске (*Примеры\Глава_09\form3.html*).

Элементы fieldset и legend

Другой способ, который может использоваться, чтобы создать визуально привлекательную форму, позволяет группировать элементы посредством тега **fieldset**. Браузеры, которые поддерживают эту возможность, создадут границу вокруг элементов формы, сгруппированных тегом fieldset. Открывающий тег <fieldset> обозначает начало группы, закрывающий тег </fieldset> обозначает ее конец.

Элемент **legend** обеспечивает текстовое описание сгруппированных элементов. Открывающий тег <legend> обозначает начало текстового описания, закрывающий тег </legend> — конец. Исходный HTML-код для формы, показанной на рис. 9.16, приведен ниже:

```
<fieldset>
<legend>Адрес доставки</legend>
<label>Улица, дом, квартира: <input type="text"
name="street" id="street" size="55"></
label><br><br>
<label>Город: <input type="text" name="city"
id="city"></label>
<label>Область: <input type="text" name="reg"
id="reg" maxlength="20" size="20"></label>
<label>Индекс: <input type="text" name="zip"
id="zip" maxlength="7" size="7"></label>
</fieldset>
```

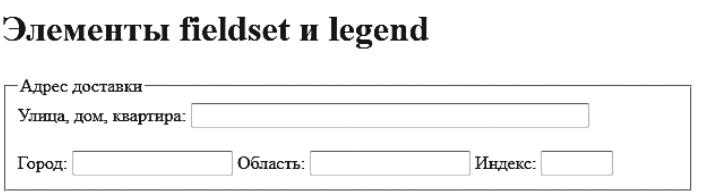


Рис. 9.16. Элементы формы, позволяющей указать адрес доставки

Визуальный эффект группировки элементов формы с помощью тега fieldset придает организованность и привлекательность веб-странице, содержащей форму. Использование тегов fieldset и legend улучшает доступность, организовывая элементы управления и визуально и семантически. Элементы fieldset> и <legend> могут быть доступны для

программ экранного доступа и являются полезным инструментом для настройки групп переключателей и флажков на веб-страницах.

Практическое задание 9.4

На этом практическом задании вы измените контактную форму (*form3.html*) с которой вы работали на практическом задании 9.3, используя элементы `fieldset` и `legend` (рис. 9.17).

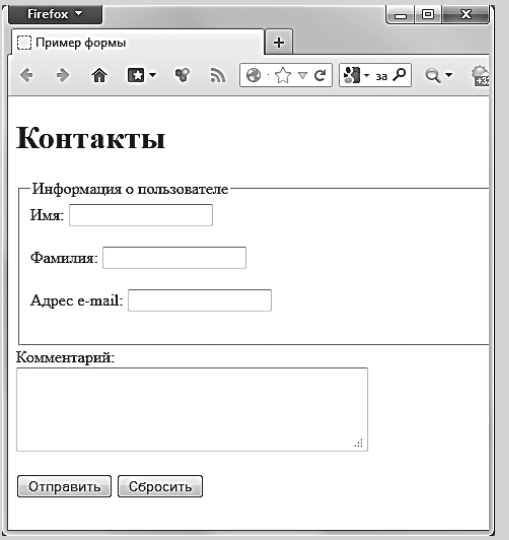


Рис. 9.17. Элементы `fieldset`, `legend` и `label`

Запустите программу Блокнот (Notepad) и откройте страницу *form3.html*, которую вы создали на практическом задании 9.3. Выполните следующие изменения:

1. Добавьте открывающий тег `<fieldset>` после открывающего тега `<form>`.
2. Сразу после `<fieldset>` укажите тег `<legend>`, содержащий следующий текст:

   «Информация о пользователе»
3. Укажите завершающий тег `<fieldset>` перед элементом `label` текстовой области с прокруткой для комментариев.
4. Сохраните файл с именем *form4.html* и протестируйте его в браузере. Он должен выглядеть так, как показано на рис. 9.17. Вы можете сравнить свою работу с одноименным файлом на прилагающемся к книге диске *Примеры\Глава_09\form4.html*. Как вы могли обратить внимание, когда вы нажимаете кнопку отправки данных,

то форма отображается повторно. Так происходит из-за того, что с элементом `form` не сопоставлено какое-либо действие. Этим вы займетесь в разделе 9.8.

5. Хотите быстренько взглянуть на форму, созданную с помощью стилей CSS? На рис. 9.17 и 9.18 показаны одни и те же элементы формы, но веб-форма на рис. 9.18 конфигурирована с помощью правил CSS, что делает ее не менее функциональной, но визуально более привлекательной.

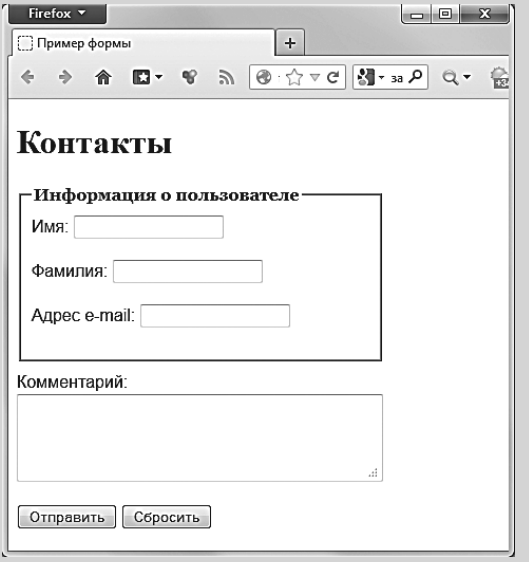


Рис. 9.18. Элементы `fieldset`, `legend` и `label` конфигурированы с использованием CSS

Откройте файл *form4.html* в программе Блокнот (Notepad) и добавьте глобальные стили в раздел заголовка, как указано ниже:

```
fieldset { width: 320px;
border: 2px ridge #ff0000;
padding: 10px;
margin-bottom: 10px; }
legend { font-family: Georgia, "Times New Roman", serif;
font-weight: bold; }
label { font-family: Arial, sans-serif; }
```

Сохраните файл под именем *form4CSS.html* и протестируйте веб-страницу в браузере. Она должна выглядеть так, как показано на рис. 9.18. Вы можете сравнить свою работу с образцом на прилагающемся к книге диске *(Примеры\Глава_09\form4CSS.html)*.

Атрибут `tabindex`

Некоторые из посетителей ваших веб-страниц могут испытать затруднения при использовании мыши и обращаться к вашей форме с клавиатуры. Они могут использовать клавишу **Tab**, чтобы переключаться от одного элемента управления формы к другому. Действие по умолчанию для клавиши **Tab** заключается в переходе к следующему элементу управления формы в последовательном порядке сверстанных в HTML элементов управления. Обычно это подходящий вариант. Однако если на странице последовательность элементов формы должна быть изменена, используйте атрибут `tabindex` для каждого элемента управления. Каждому элементу формы (`input`, `select`, `textarea`) можно присвоить ***атрибут*** **`tabindex`** с числовым значением, начиная с 1 и далее в порядке нумерации. HTML-код, определяющий текстовое поле электронной почты клиента как начальную позицию курсора, выглядит так:

```
<input type="text" name="Email" id="Email"
tabindex="1">
```

Если вы сконфигурируете элемент управления с `tabindex="0"`, то он будет выбран после всех других элементов управления формы, которым присвоен атрибут `tabindex`. Если вы случайно присвоите двум элементам управления формы то же самое значение `tabindex`, то тот элемент, который указан в HTML-коде сначала, будет выбран первым.

Подобным образом можно задать атрибут `tabindex` для элементов привязки. Действие по умолчанию, установленное для клавиши **Tab** и элементов привязки, — перемещение от гиперссылки к гиперссылке в том порядке, в каком они указаны на веб-странице. Если вам требуется изменить это поведение, используйте атрибут `tabindex`.

Атрибут `accesskey`

Другой метод, который может сделать вашу форму удобнее при управлении с клавиатуры, заключается в использовании ***атрибута*** **`accesskey`** применительно к элементам формы. Атрибуту `accesskey` можно задать значение одного из символов (латинская буква или число) на клавиатуре и создать сочетание клавиш, которое посетитель вашей веб-страницы может нажать, чтобы немедленно выбрать связанный элемент управления формы или гиперссылку. Сочетание клавиш

зависит от операционной системы. Пользователи Windows должны нажать клавишу **Alt** и назначенную клавишу. Пользователи OS X должны нажать клавишу **Ctrl** и назначенную клавишу. Например, если бы полю для ввода адреса электронной почты формы, показанной на рис. 9.11, был присвоен атрибут `accesskey="E"`, то посетитель веб-страницы, использующий операционную систему Windows, мог бы нажать сочетание клавиш **Alt+E**, чтобы немедленно выбрать текстовое поле адреса электронной почты. HTML-код в этом случае следующий:

```
<input type="text" name="Email" id="Email"
accesskey="E">
```

Помните, не следует надеяться, что браузер сам укажет, какая клавиша является «горячей». Вам придется вручную указать в коде информацию о «горячих» клавишах. Помогут визуальные подсказки, например, можно выделить полужирным шрифтом соответствующую букву или поместить сообщение, к примеру, `(Alt+E)` после элемента управления формы или гиперссылки, использующей это сочетание клавиш. Указывая значение атрибута `accesskey`, избегайте сочетаний, которые уже используются операционной системой (например, сочетание клавиш **Alt+F** открывает меню **Файл** (File)). Проверка работоспособности сочетаний клавиш крайне важна.

9.7. Стилизация форм

В этом разделе вы познакомитесь с тремя методами стилизации и конфигурирования внешнего вида формы — старомодным (используя таблицу HTML), переходным (таблица HTML и CSS) и современным (с помощью CSS).

Табличный метод

Веб-форма на рис. 9.11 (практическое задание 9.2) кажется немного «неаккуратной», и, наверное, вы задаетесь вопросом, как можно ее улучшить. Не так давно веб-дизайнеры использовали таблицы для настройки дизайна элементов формы, как правило, размещая текстовые метки и поля элементов формы в отдельных ячейках таблицы данных.

Пример использования этого метода показан на рис. 9.19, а соответствующий файл находится в папке *Примеры\Глава_09\formtable.html* на прилагающемся к книге диске.

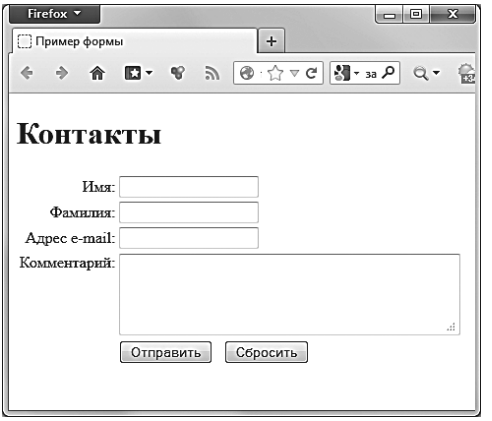


Рис. 9.19. Элементы управления формы хорошо выровнены

HTML-код следующий:

```
<form method="post">
<table border="0">
<tr>
<td align="right">Имя:</td>
<td><input type="text" name="fmail" id="fmail"></
td>
</tr>
<tr>
<td align="right">Фамилия:</td>
<td><input type="text" name="lmail" id="lmail"></
td>
</tr>
<tr>
<td align="right">Адрес e-mail:</td>
<td><input type="text" name="email" id="email"></
td>
</tr>
<tr>
<td align="right" valign="top">Комментарий:</td>
<td><textarea name="comments" id="comments"
rows="4" cols="40" ></textarea></td>
</tr>
```

```
<tr>
<td> </td>
<td><input type="submit" value="Отправить">  
<input type="reset" value="Сбросить"></td>
</tr>
</table>
</form>
```

Другой метод — организовать форму с использованием таблицы, сконфигурированной с помощью CSS, а не атрибутов HTML. Хотя таким образом не получится полностью безтабличный дизайн и таблица будет задействована для формирования небольшой части страницы, в определенных случаях можно рассмотреть использование CSS как альтернативный метод. Было бы лучше зарезервировать использование таблицы для полностью табличных данных, таких как прайс-листы и бюджеты. Однако цель этого примера состоит в том, чтобы показать, как CSS могут использоваться, чтобы упростить код HTML, необходимый для создания таблицы. Далее в этой главе вы узнаете, как конфигурировать макет и стили формы, применяя исключительно CSS и не используя таблицу HTML.

Переходный метод

Рисунок 9.20 демонстрирует веб-страницу (см. на диске, прилагающемся к книге, файл *Примеры\Глава_09\formtable2.html*) с формой, сверстанной с использованием таблицы, но стилизованной с помощью CSS.

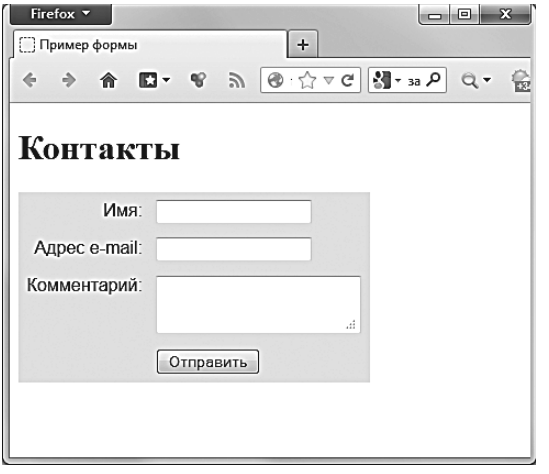


Рис. 9.20. Эта страница содержит таблицу, стилизованную с помощью CSS

Правила CSS, оформляющие внешний вид таблицы, задают селекторы элементов `table`, `th` и `td` со свойствами, которые были бы иначе определены HTML-атрибутами. Селектор `table` определяет светло-серый цвет, отсутствие границы, ширину 20 em и шрифт Arial или другой без засечек. Напомним, что элементы `th` по умолчанию выделяют текст полужирным и выравнивают его по центру. Селектор `th` задает обычный текст, выравнивание по правому краю и отступ шириной 5 пикселов. Код CSS следующий:

```
table { background-color: #eaeaea;
border-style: none;
width: 20em;
font-family: Arial, sans-serif; }
th { font-weight: normal;
text-align: right;
vertical-align: top; }
td, th { padding: 5px; }
```

Так как мы задаем таблицу с помощью CSS, HTML-атрибуты `border` и `align` больше не нужны. Другое изменение в HTML-коде состоит в том, что текстовое наименование каждого элемента формы теперь заключено в теги <th>, а не <td>, как было раньше. Код HTML выглядит так:

```
<form method="post">
<table>
<tr>
<th>Имя:</td>
<td><input type="text" name="Name" id="Name"></td>
</tr>
<tr>
<th>Адрес e-mail:</td>
<td><input type="text" name="myEmail"
id="myEmail"></td>
</tr>
<tr>
<th>Комментарий:</td>
<td><textarea name="myComments" id="myComments"
rows="2" cols="20"></textarea></td>
```

```
</tr>
<tr>
<td > </td>
<td><input type="submit" value="Отправить"></td>
</tr>
</table>
</form>
```

В результате получается более изящный, гибкий код. Дальнейшие изменения стилей можно легко произвести с помощью CSS, не затрагивая структуру формы. Однако самый гибкий вариант — не использовать таблицу совсем и конфигурировать всю форму, применяя только правила CSS. Далее вы изучите этот метод.

Современный метод

Многие веб-разработчики применяют правила CSS для создания макета веб-страницы, пока не сталкиваются с необходимостью конфигурировать форму. Как уже говорилось выше, для конфигурирования форм долгое время использовали таблицы.

В этом разделе мы продемонстрируем более современный подход, прибегающий к CSS для задания стилей макета формы без применения HTML-таблицы.

В этом методе блочная модель CSS используется для создания ряда полей, как показано на рис. 9.21. Внешнее поле определяет область формы. Другие поля обозначают элементы `label` и элементы управления формы. Таблица CSS используется, чтобы конфигурировать эти компоненты.

Рис. 9.21. Блок-схема формы

Рисунок 9.22 демонстрирует веб-страницу с формой, сконфигурированной таким способом (см. файл *Примеры\Глава_09\form2.html* на диске, прилагающемся к книге).

Рассматривая приведенный код CSS и HTML, обратите внимание, что селектор `label` конфигурирован следующим образом: блочное отображение, ширина 100 пикселов и обтекание с левой стороны формы, а остальные случаи обтекания слева отменены. У элементов `input` и `textarea` обозначено верхнее поле, и им также задано блочное отображение. Кнопке отправки данных присвоен идентификатор и поле слева. В результате получилась аккуратно выровненная форма.

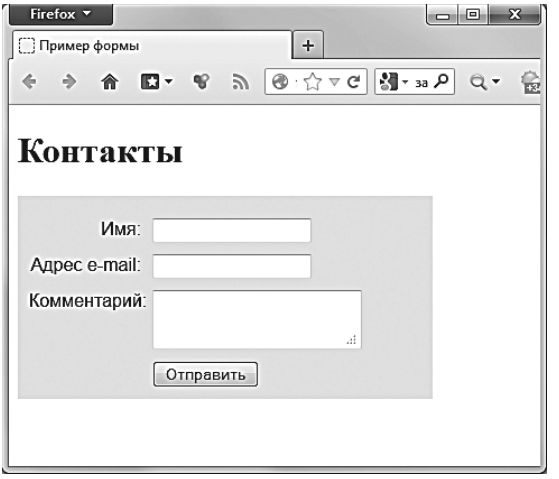


Рис. 9.22. Эта страница конфигурирована с помощью CSS

Код CSS следующий:

```
form { background-color:#eaeaea;
font-family: Arial, sans-serif;
width: 350px;
padding: 10px; }
label { float: left;
width: 100px;
display: block;
clear: left;
text-align: right;
padding-right: 10px;
```

```
margin-top: 10px; }
input, textarea { margin-top: 10px;
display: block;}
#mySubmit { margin-left: 110px; }
```

HTML-код следующий:

```
<form>
<label for="myName">Имя:</label>
<input type="text" name="myName" id="myName">
<label for="myEmail">Адрес e-mail:</label>
<input type="text" name="myEmail" id="myEmail">
<label for="myComments">Комментарий:</label>
<textarea name="myComments" id="myComments"
rows="2" cols="20"></textarea>
<input id="mySubmit" type="submit"
value="Отправить">
</form>
```

В этом разделе представлен метод стилизации формы с помощью CSS. Тестирование способа, которым различные браузеры передают форму, крайне важно.

Поскольку вы сверстали и отобразили формы в этой главе, то, возможно, заметили, что, когда вы нажимаете кнопку отправки данных, то форма только заново отображается и ничего не происходит. Причина — в теге `<form>` не задано какое-либо действие.

Следующий раздел сосредоточит ваше внимание на второй составляющей использования форм на веб-страницах — обработке на стороне сервера.

9.8. Обработка на стороне сервера

Ваш веб-браузер запрашивает веб-страницы и связанные с ними файлы с веб-сервера. Веб-сервер определяет местонахождение файлов и отправляет их вашему веб-браузеру. Затем веб-браузер визуализирует возвращаемые файлы и отображает запрашиваемые веб-страницы.

Рисунок 9.23 иллюстрирует связь между веб-браузером и веб-сервером.

Рис. 9.23. Веб-браузер (клиент) работает с веб-сервером

Иногда веб-сайт нуждается в большей функциональности, чем статические веб-страницы, — в поиске по сайту, формах заказов, списках рассылок, отображении баз данных, или другом типе интерактивной, динамической обработки. Это те случаи, когда необходима обработка на стороне сервера. Первые веб-серверы, чтобы предоставить эту функциональность, использовали протокол под названием ***общий шлюзовой интерфейс*** (Common Gateway Interface — CGI). CGI — это протокол, или стандартный метод, для веб-сервера, позволяющий передать запрос веб-страницы (который обычно инициализируется с помощью формы) пользователем к прикладной программе и принимать информацию для дальнейшей отправки пользователю. Веб-сервер обычно передает информацию формы небольшой прикладной программе, которая выполняется операционной системой и обрабатывает данные, и он обычно отсылает назад веб-страницу с подтверждением или сообщение. Perl и C — популярные языки программирования для приложений CGI.

Скрипт (сценарий) на стороне сервера — это технология, при которой на веб-сервере запускается серверный скрипт для динамической генерации веб-страниц. Примеры технологий скриптов на стороне сервера — PHP, Ruby on Rails, Microsoft Active Server Pages, Adobe ColdFusion, Oracle JavaServer Pages и Microsoft .NET. Скрипт на стороне сервера отличается от CGI ***прямым выполнением*** — он исполняется или веб-сервером непосредственно или модулем расширения для веб-сервера.

Веб-страница вызывает обработку на стороне сервера с помощью атрибута в коде формы или гиперссылки — используется URL-адрес скрипта. Любые данные формы, которые существуют, передаются скрип-

ту. Когда скрипт завершает обработку, то он может генерировать веб-страницу с подтверждением или ответ с запрашиваемой информацией.

Вызывая скрипт на стороне сервера, веб-разработчик и программист сервера должны передать информацию об ***атрибуте*** **`method`** (`get` или `post`) формы, ***атрибуте*** **`action`** (URL-адрес скрипта на стороне сервера), а также о любых специальных элементах управления формы, ожидаемых скриптом на стороне сервера.

Атрибут `method` используется в виде тега, чтобы указать, каким образом пары имя и значение должны быть переданы на сервер. Значение `get` атрибута `method` присоединяет данные формы к URL-адресу, который нагляден, что небезопасно.

Значение `post` атрибута `method` передает данные формы не в URL-адресе, а в теле запроса HTTP, что делает их более скрытыми. Консорциум W3C рекомендует использовать значение `post`.

Атрибут `action` присваивается элементу `form` с целью вызвать скрипт на стороне сервера. Атрибуты `name` и `value`, связанные с каждым элементом управления, передаются скрипту на стороне сервера. Значения атрибута `name` каждого элемента управления формы могут использоваться как имена переменных при обработке. В следующем практическом задании вы вызовете с помощью формы скрипт на стороне сервера.

Практическое задание 9.5

На этом практическом задании вы сконфигурируете форму для запуска скрипта на стороне сервера. Пожалуйста, проверьте, что ваш компьютер подключен к Интернету, когда вы начнете тестировать страницу. При использовании скрипта на стороне сервера вам потребуется некоторая информация, или документация, от человека или организации, предоставляющей скрипт. Вам нужно будет знать расположение скрипта, требует ли он присваивания каких-либо определенных имен элементам управления формы, или размещения скрытых элементов формы.

Информация о скрипте представлена ниже.

- Расположение скрипта: **http://webdevfoundations.net/scripts/demo.php**
- Метод отправки данных формы: `post`
- Назначение скрипта: этот скрипт примет ввод формы и отобразит имена элемента управления формы и значения на веб-странице. Это

пример скрипта для практических заданий. Он демонстрирует, что обработка на стороне сервера была инициирована. Скрипт, используемый фактическим веб-сайтом, выполнил бы функцию, например, отправку почтового сообщения или обновление базы данных.

Теперь вы добавите код, необходимый для взаимодействия файла *demo.php* на стороне сервера с вашей формой. Запустите программу Блокнот (Notepad) и откройте файл *formcss.html* (его можно найти в папке *Примеры\Глава_09* на диске, прилагающемся к книге). Измените тег `<form>`, добавив атрибут `method` со значением `"post"` и атрибут `action` со значением `"http://webdevfoundations.net/scripts/demo.php"`. HTML-код измененного элемента `form` приведен ниже:

```
<form method="post" action="http://webdevfoundations.
net/scripts/demo.php">
```

Сохраните страницу с именем *contact.html* и откройте ее в браузере. Страница в браузере должна выглядеть примерно так, как показано на рис. 9.22.

Теперь вы готовы протестировать свою форму. Компьютер должен быть подключен к Интернету, чтобы тестирование формы завершилось успешно. Введите информацию в элементы управления формы и нажмите кнопку отправки данных. Вы должны увидеть страницу с подтверждением, как показано на рис. 9.24.

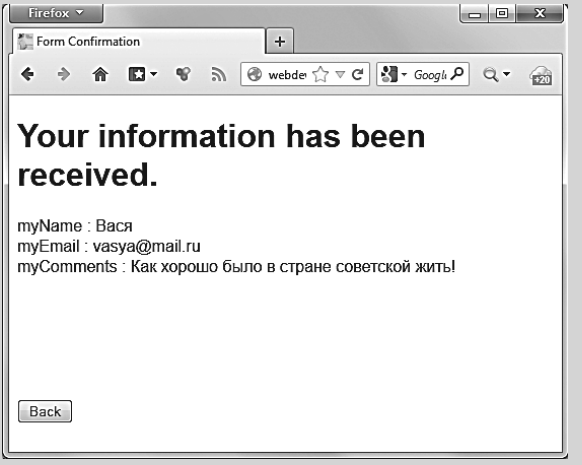


Рис. 9.24. Скрипт на стороне сервера сгенерировал эту страницу в ответ на форму

Скрипт *demo.php* создает веб-страницу, отображающую сообщение и информацию формы, которую вы ввели. Откуда появилась эта страница с подтверждением? Она была создана скриптом на стороне сервера с помощью атрибута `action` элемента `form`. Вы можете

задаться вопросом, каков исходный код файла *demo.php*. Однако написание скриптов для обработки на стороне сервера выходит за рамки этой книги. Но если вам любопытно, то откройте файл *Примеры\Глава_09\script.pdf* на диске, прилагающемся к книге, чтобы увидеть исходный код этого скрипта.

ЧаВо

ЧТО МНЕ ДЕЛАТЬ, ЕСЛИ НИЧЕГО НЕ ПРОИЗОШЛО, КОГДА Я ТЕСТИРОВАЛ МОЮ ФОРМУ?

Попробуйте устранить неполадки с помощью этих советов:

- проверьте, подключен ли ваш компьютер к Интернету;
- проверьте правильность написания имени файла и адреса скрипта в атрибуте `action`.

Очень важно внимание к деталям!

Конфиденциальность и формы

Вы только что узнали, как получить информацию от посетителей вашего веб-сайта. Вы считаете, что посетители, возможно, хотят знать, как вы планируете использовать информацию, которую собираете? Руководящие принципы обеспечения конфиденциальности информации своих посетителей называются ***политикой конфиденциальности***. Веб-сайты указывают эту политику непосредственно на странице с формой, либо создают отдельную страницу, которая описывает политику конфиденциальности (и другие политики компании). Например, страница на сайте **www.softkey.ru/about/privacy.php** указывает на следующее:

«Компания ЗАО "Софткей" никому не будет продавать ваши сведения личного характера или передавать иным способом».

Для более детализированного примера конфиденциальности веб-сайта посетите популярный сайт, например Livejournal.com или Rambler.ru; вы найдете ссылки на их политики конфиденциальности (иногда называемую конфиденциальностью) в нижней части страницы.

С политикой конфиденциальности компании Google можно ознакомиться на сайте **www.google.ru/intl/ru/privacy/**. Добавьте на свой сайт уведомление о конфиденциальности для информирования посетителей о том, как вы планируете использовать информацию, которую они вам предоставили.

Ресурсы, посвященные обработке на стороне сервера

Сервисы бесплатной обработки форм с удаленным управлением

Если провайдер не поддерживает обработку на стороне сервера для вашего веб-хоста, то одним из вариантов может стать удаленное хранение и использование ваших скриптов. Скрипт не размещается на вашем сервере и поэтому вам не нужно беспокоиться об его установке на веб-хосте и поддержке со стороны провайдера. Недостатком является то, что обычно отображается некоторое количество рекламы. Вот несколько сайтов, которые предлагают эту услугу:

- FormBuddy.com: **formbuddy.com**
- ExpressDB: **www.expressdb.com**
- FormMail: **www.formmail.com**
- Master.com: **www.master.com**

Источники бесплатных скриптов на стороне сервера

Чтобы использовать бесплатные скрипты, у вас должен быть доступ к веб-серверу, который поддерживает язык, используемый скриптом. Свяжитесь со своей хостинговой компанией, чтобы определить, какие языки поддерживаются. Учтите, что большинство бесплатных хостинговых компаний не поддерживают обработку на стороне сервера (за что платите, то и получаете!). Вот несколько сайтов, которые предлагают бесплатные скрипты и другие ресурсы:

- PHP Resource Index: **php.resourceindex.com**
- Code Guru: **www.codeguru.com**
- HotScripts: **www.hotscripts.com**
- Matt's Script Archive: **www.scriptarchive.com**
- ScriptSearch.com: **www.scriptsearch.com**

Изучение технологий обработки на стороне сервера

Различные технологии могут быть использованы для скриптов со стороны сервера, обработки формы и информационного совместного использования:

- PHP: **www.php.net**
- Oracle JavaServer Pages Technology: **www.oracle.com/technetwork/java/javaee/jsp**

- Active Server Pages: **msdn.microsoft.com/en-us/library/ms972337.aspx**
- Adobe ColdFusion: **www.adobe.com/products/coldfusion**
- Ruby on Rails: **www.rubyonrails.org**
- Microsoft .NET: **www.microsoft.com/net**

Любая из этих технологий прекрасно подойдет для будущего изучения. Веб-разработчики часто изучают сначала сторону клиента (HTML, CSS и JavaScript), а затем переходят к изучению скриптов со стороны сервера или языка программирования.

9.9. Элементы управления формы в HTML5

Язык HTML5 предлагает веб-разработчикам целый ряд новых элементов формы, которые обеспечивают повышенную юзабилити для посетителей веб-страницы, пользующихся современными браузерами. Например, некоторые новые элементы управления предлагают встроенные в браузер правки и проверки. В будущем веб-дизайнеры, вероятно, будут воспринимать эти функции как должное, однако вы сейчас стали свидетелями этого огромного шага вперед в верстке веб-страниц, так что теперь самое время ознакомиться с этими новыми элементами. Отображение и поддержка новых элементов формы HTML5 будет зависеть от браузера, но их можно использовать прямо сейчас! Браузеры, которые не поддерживают новые типы элемента `input`, будут отображать их как текстовые поля и игнорировать неподдерживаемые атрибуты и элементы. В данном разделе вы изучите новые элементы управления HTML5 для реализации на сайте поля ввода адреса электронной почты, URL-адреса, номера телефона, поля поиска, списка данных, ползунка, счетчика, календарей и палитры.

Поле ввода адреса электронной почты

Поле ввода адреса электронной почты напоминает текстовое поле. Его цель — принять информацию, которая должна быть указана в формате адреса электронной почты, например **DrMorris2010@gmail.com**. Управление вводом адреса электронной почты задается элементом `input` с атрибутом `type="email"`. Проверить правильность введенной информации смогут только браузеры, поддерживающие элемент `email` HTML5. Другие браузеры будут визуализировать этот элемент

управления формы как текстовое поле. Атрибуты, поддерживаемые элементом управления формы `email`, приведены в табл. 9.2.

На рис. 9.25 (см. файл *Примеры\Глава_09\email.html* на прилагающемся к книге диске) показано сообщение об ошибке, появляющееся в браузере Firefox, когда введен текст в формате, отличном от адреса электронной почты. Обратите внимание, что браузер не проверяет, существует ли в действительности такой адрес электронной почты, а только то, что текст введен в правильном формате. HTML-код следующий:

```
<form method="get">
<label for="email">Адрес e-mail:</label>
<input type="email" name="email" id="email">
<br><br>
<input type="submit" value="Подписаться"> <input
type="reset" value="Сбросить">
</form>
```

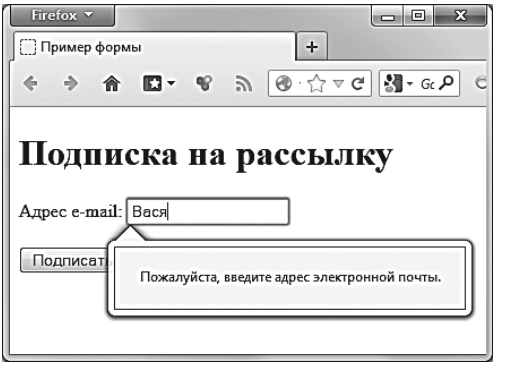


Рис. 9.25. Браузер Firefox отображает сообщение об ошибке

Поле ввода URL-адреса

Поле ввода URL-адреса напоминает текстовое поле. Его назначение — принять любой допустимый тип URL или URI, например, **http://webdevfoundations.net**. Элемент управления формой `url` конфигурируется элементом `input` с атрибутом `type="url"`. Проверить правильность введенной информации смогут только браузеры, поддерживающие HTML5-элемент `url`. Другие браузеры будут визуализировать этот элемент управления формы как текстовое поле. Атрибуты, поддерживаемые элементом управления формы `url`, приведены в табл. 9.2.

На рис. 9.26 (см. файл *Примеры\Глава_09\url.html* на прилагающемся к книге диске) показано сообщение об ошибке, появляющееся в браузере Firefox, когда введен другой текст, кроме URL-адреса. Обратите внимание, что браузер проверяет, существует ли в действительности такой URL-адрес, что текст введен в правильном формате. HTML-код следующий:

```
<form method="get">
<label for="myWebsite">Предложить веб-сайт:</label>
<input type="url" name="myWebsite" id="myWebsite" >
<br><br>
<input type="submit" value="Отправить"> <input
type="reset" value="Сбросить">
</form>
```

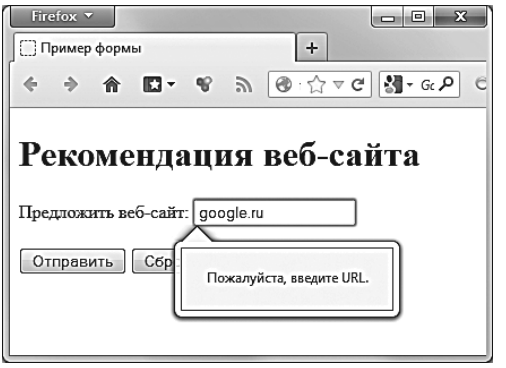


Рис. 9.26. Браузер Firefox отображает сообщение об ошибке

Поле ввода номера телефона

Поле ввода номера телефона напоминает текстовое поле. Его цель — принять телефонный номер. Элемент `tel` задается элементом `input` с атрибутом `type="tel"`. Пример можно найти на прилагающемся к книге диске (*Примеры\Глава_09\tel.html*). Атрибуты, поддерживаемые элементом управления формы `tel`, приведены в табл. 9.2. Браузеры, не поддерживающие атрибут `type="tel"`, будут визуализировать этот элемент управления формы как текстовое поле. HTML-код следующий:

```
<form method="get">
<label for="mobile">Номер телефона:</label>
<input type="tel" name="mobile" id="mobile" >
```

```
<br><br>
<input type="submit" value="Подписаться"> <input
type="reset" value="Сбросить">
</form>
```

Поле поиска

Поле поиска напоминает текстовое поле и используется для ввода запроса. Элемент `search` задается элементом `input` с атрибутом `type="search"`. Пример можно найти на прилагающемся к книге диске (*Примеры\Глава_09\search.html*). Атрибуты, поддерживаемые элементом управления формы `search`, приведены в табл. 9.2. Браузеры, не поддерживающие атрибут `type="search"`, будут визуализировать этот элемент управления формы как текстовое поле. HTML-код следующий:

```
<form method="get">
<label for="keyword">Поиск:</label>
<input type="search" name="keyword" id="keyword" >
<br><br>
<input type="submit" value="Перейти"> <input
type="reset" value="Сбросить">
</form>
```

ЧаВо

КАК УЗНАТЬ, КАКИЕ БРАУЗЕРЫ ПОДДЕРЖИВАЮТ НОВЫЕ ЭЛЕМЕНТЫ ФОРМЫ В HTML5?

Лучший способ — тестирование. Также ниже перечислены несколько ресурсов, которые содержат информацию о поддержке браузерами новых элементов HTML5:

- **caniuse.com**
- **findmebyip.com/litmus**
- **HTML5readiness.com**
- **HTML5test.com**
- **www.standardista.com/HTML5**

Список данных

На рис. 9.27 показан ***список данных***. Обратите внимание, что пользователю помимо списка выбора предлагается текстовое поле для ввода значения вручную.

Элемент `datalist` задается с помощью трех элементов: элемента `input`, элемента `datalist` и одного или нескольких атрибутов. Отобразить и обработать пункты элемента `datalist` смогут только браузеры, поддерживающие данный HTML5-элемент. Другие браузеры проигнорируют его и отобразят форму как текстовое поле.

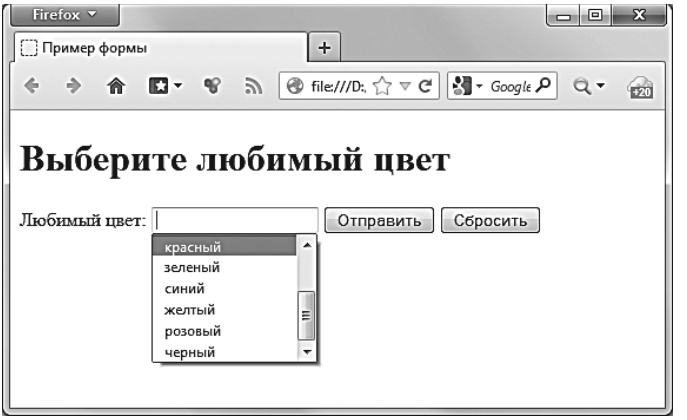


Рис. 9.27. Браузер Firefox отображает элемент управления формы `datalist`

Исходный код для элемента `datalist` доступен на прилагающемся к книге диске (см. файл *Примеры\Глава_09\list.html*).

HTML-код следующий:

```
<form method="get">
Любимый цвет: <input type="text" name="color"
id="color" list="colors">
<datalist id="colors">
<option value="красный">
<option value="зеленый">
<option value="синий">
<option value="желтый">
<option value="розовый">
<option value="черный">
</datalist>
<input type="submit" value="Отправить"> <input
type="reset" value="Сбросить">
</form>
```

Обратите внимание, что значение ***атрибута*** **`list`** тега `input` такое же, как и значение атрибута `id` элемента `datalist`. Это создает связь между текстовым полем и элементом формы `datalist`. Можно использовать один или несколько элементов `option`, чтобы предложить посетителям веб-страницы выбор из предопределенных вариантов. Атрибут `value` элемента `option` задает текст, отображаемый в каждой записи в списке. Посетитель веб-страницы может выбрать вариант из списка (см. рис. 9.27) или ввести текст непосредственно в поле, как показано на рис. 9.28.

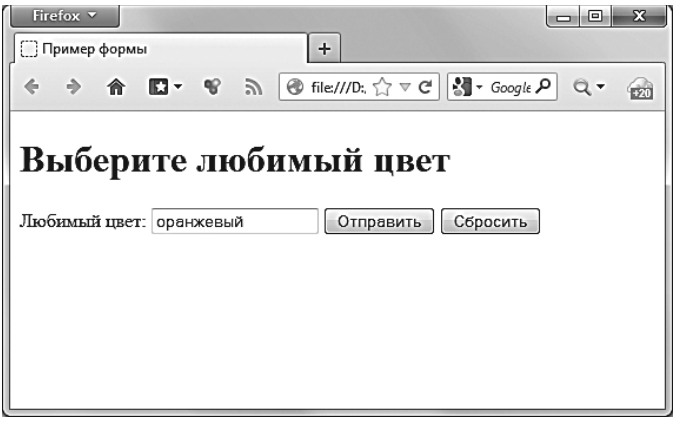


Рис. 9.28. Список исчез, когда пользователь начал вводить запись в текстовое поле

Элемент управления формы `datalist` — это удобный способ предложить выбор и в то же время обеспечить гибкость формы.

ЧаВо

ЗАЧЕМ МНЕ ЗНАТЬ О НОВЫХ ЭЛЕМЕНТАХ УПРАВЛЕНИЯ ФОРМЫ В HTML5, ЕСЛИ ОНИ ЕЩЕ НЕ ПОДДЕРЖИВАЮТСЯ ВСЕМИ БРАУЗЕРАМИ?

Новые элементы управления формы предлагают комфортные средства управления тем посетителям веб-страницы, которые пользуются современными браузерами. И они также совместимы с более старыми версиями браузеров. Браузеры, не поддерживающие новые объекты формы, будут отображать их как текстовые поля и игнорировать не поддерживающие атрибуты или элементы. На рис. 9.29 показана страница со списком данных в браузере Internet Explorer. Обратите внимание, что, в отличие от браузера Firefox, в Internet Explorer отображается не список, а только текстовое поле.

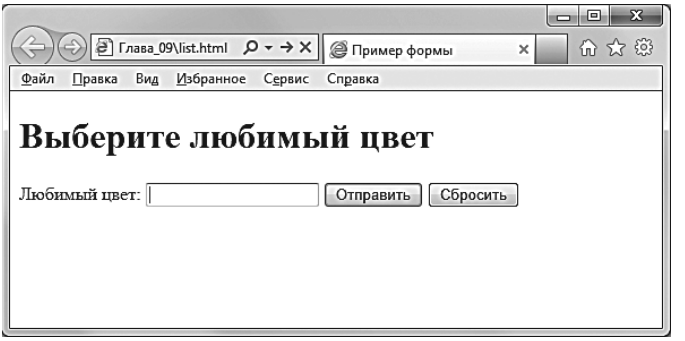


Рис. 9.29. Браузеры, не поддерживающие элемент `datalist`, отображают простое текстовое поле

Ползунок

Ползунок обеспечивает визуальный, интерактивный пользовательский интерфейс, который принимает числовую информацию. Он конфигурируется элементом `input` с атрибутом `type="range"`, позволяющим выбрать число в пределах указанного диапазона. По умолчанию установлен диапазон от 1 до 100. Только в браузерах, поддерживающих HTML5, значение `range` будет отображать интерактивный ползунок, как показано на рис. 9.30 (см. *Примеры\Глава_09\range.html* на прилагающемся к книге диске). Обратите внимание на положение ползунка на рис. 9.30: это означает, что выбрано значение 50. То, что установленное значение не отображается, может быть недостатком самого элемента. Не поддерживающие этот элемент браузеры визуализируют его в виде текстового поля, как показано на рис. 9.31.

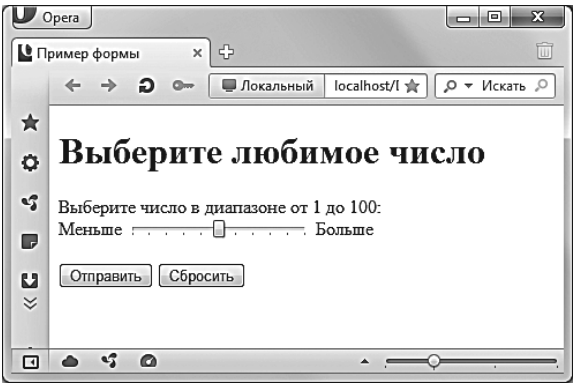


Рис. 9.30. Браузер Opera корректно отображает элемент управления формы `range`

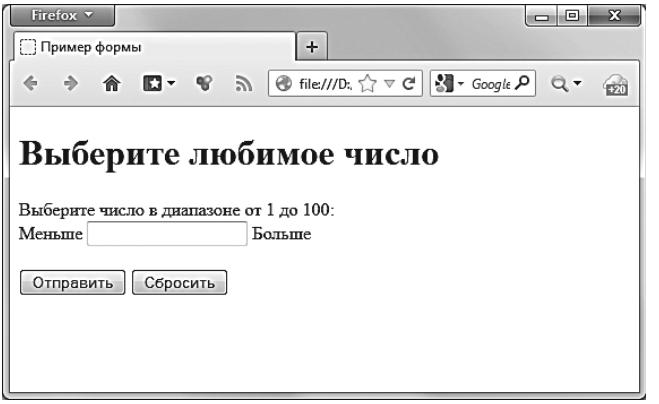


Рис. 9.31. Браузер Firefox 14 визуализирует элемент формы `range` в виде текстового поля

Ползунок принимает атрибуты, указанные в табл. 9.2 и 9.11. Новыми являются атрибуты `min`, `max` и `step`. Используйте ***атрибут*** **`min`** для настройки минимального значения диапазона. Для настройки максимального значения диапазона примените ***атрибут*** **`max`**. Числовые значения ползунка — это шаги с величиной 1 (или другим значением, указанным атрибутом `step`). Используйте ***атрибут*** **`step`** для настройки других значений шага, кроме 1. HTML-код для задания ползунка, показанного на рис. 9.30 и 9.31, приведен ниже:

```
<form method="get">

<label for="myChoice">Выберите число в диапазоне от
1 до 100:</label><br> Меньше <input type="range"
name="myChoice" id="myChoice"> Больше <br><br>

<input type="submit" value="Отправить"> <input
type="reset" value="Сбросить">

</form>
```

Таблица 9.11. Дополнительные атрибуты элементов `range`, `number` и `date`

Атрибут	Значение	Цель
`max`	Максимальное числовое значение	Атрибут HTML5 для элементов `range`, `number` и `date`, определяет максимальное значение
`min`	Минимальное числовое значение	Атрибут HTML5 для элементов `range`, `number` и `date`, определяет минимальное значение
`step`	Число, обозначающее шаг между значениями	Атрибут HTML5 для элементов `range`, `number` и `date`, определяет шаг между значениями

Счетчик

Счетчик отображает интерфейс, который принимает цифровую информацию и обеспечивает обратную связь с пользователем. Счетчик, в котором пользователь может ввести число в текстовое поле или выбрать значение в определенном диапазоне, конфигурируется элементом `input` с атрибутом `type="number"`. Только в браузерах, поддерживающих HTML5, значение `number` визуализирует интерактивное управление счетчика, как показано на рис. 9.32 (см. файл *Примеры\Глава_09\spinner.html* на диске, прилагающемся к книге). Другие браузеры отображают элемент `number` как текстовое поле. Стоит надеяться на улучшение поддержки браузеров в будущем.

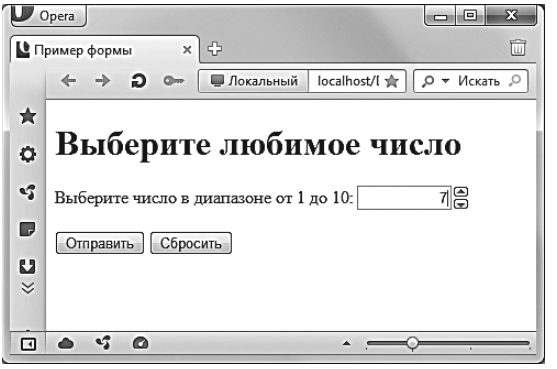


Рис. 9.32. Счетчик, отображенный в браузере Opera

Элемент управления формы `number` принимает атрибуты, перечисленные в табл. 9.2 и 9.11. Используйте атрибут `min` для настройки минимального значения. Для настройки максимального значения примените атрибут `max`. Используйте атрибут `step` для настройки значения шага между значениями, кроме 1. HTML-код для конфигурирования счетчика, показанного на рис. 9.32, следующий:

```
<form method="get">
<label for="myChoice">Выберите число в диапазоне от
1 до 10:</label>
<input type="number" name="myChoice" id="myChoice"
min="1" max="10"> <br><br>
<input type="submit" value="Отправить"> <input
type="reset" value="Сбросить">
</form>
```

Календари

Язык HTML5 предоставляет различные варианты ***календарей*** для указания даты и времени. Используйте элемент `input` и конфигурируйте атрибут `type`, чтобы определить элемент формы для отображения даты и времени. В таблице 9.12 перечислены варианты календарей в HTML5.

Таблица 9.12. Элементы календаря в HTML5

Значение атрибута `type`	Назначение	Формат
`date`	Дата	ГГГГ-ММ-ДД Пример: 2 января 2014 года будет представлено как «2014-01-02»
`datetime`	Дата и время с информацией о часовом поясе. Обратите внимание, что на часовой пояс указывает разница с всемирным координированным временем	ГГГГ-ММ-ДД ЧЧ:ММ UTC Пример: 2 января 2014 года, 9:58 утра будет представлено как «2014-01-02 09:58:00-06 UTC»
`datetime-local`	Дата и время без информации о часовом поясе	ГГГГ-ММ-ДД ЧЧ:ММ Пример: 2 января 2014 года, 9:58 утра будет представлено как «2014-01-02 09:58»
`time`	Время без информации о часовом поясе	ЧЧ:ММ Пример: 13:34 представлено как «13:34»
`month`	Год и месяц	ГГГГ-ММ Пример: Январь 2014 года представлен как «2014-01»
`week`	Год и неделя	ГГГГ-W##, где ## обозначает номер недели в году Пример: Третья неделя 2014 года представлена как «2014-W03»

В форме на рис. 9.33 (см. *Примеры\Глава_09\date.html* на прилагающемся к книге диске) для конфигурирования календаря, с помощью которого пользователь может выбрать дату, используется элемент `input` с атрибутом `type="date"`. HTML-код следующий:

```
<form method="get">
<label for="myDate">Выберите дату</label>
<input type="date" name="myDate" id="myDate">
<br><br>
```

```
<input type="submit" value="Далее">
</form>
```

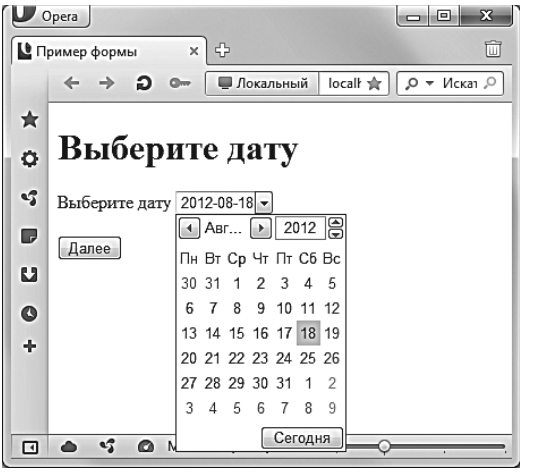


Рис. 9.33. Элемент формы, отображающий дату в браузере Opera

Элементы формы, позволяющие указать дату и время, используют атрибуты, перечисленные в табл. 9.2 и 9.11. На момент написания книги не все браузеры поддерживали данный элемент и отображали его в виде текстового поля.

Палитра

Данный элемент формы отображает интерфейс ***палитры***, который позволяет выбрать цвет. Элемент конфигурируется тегом `input` с атрибутом `type="color"`. Интерфейс цветовой палитры, показанный на рис. 9.34 (см. *Примеры\Глава_09\color.html* на прилагающемся к книге диске) отображается только в браузерах, поддерживающих HTML5. Другие браузеры отображают этот элемент формы как текстовое поле. HTML-код для цветовой палитры, показанной на рис. 9.34, следующий:

```
<form method="get">
<label for="myColor">Выберите цвет:</label>
<input type="color" name="myColor" id="myColor">
<br><br>
<input type="submit" value="Отправить"> <input
type="reset" value="Сбросить">
</form>
```

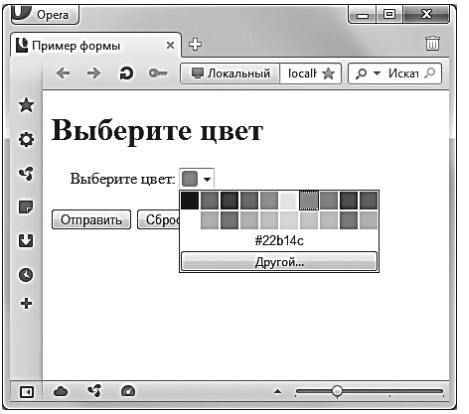


Рис. 9.34. Браузер Opera поддерживает элемент цветовой палитры формы

Далее в практическом задании вы потренируетесь работать с новыми элементами управления формы в HTML5.

Практическое задание 9.6

В этом практическом задании вы зададите элементы управления формы HTML5, когда будете конфигурировать форму, принимающую имя, адрес электронной почты, оценку и комментарии посетителей сайта. На рис. 9.35 показана форма, отображенная в браузере Opera, поддерживающем возможности HTML5, используемые на этом практическом задании. Форма на рис. 9.36 отображена в браузере Internet Explorer 9, который не поддерживает возможности HTML5. Обратите внимание, что форма в Opera комфортнее для посетителя, но функционирует в обоих браузерах, демонстрируя идею прогрессивного улучшения.

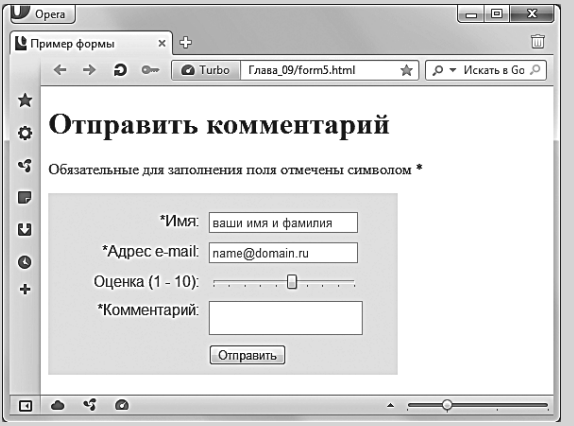


Рис. 9.35. Форма, отображенная в браузере Opera

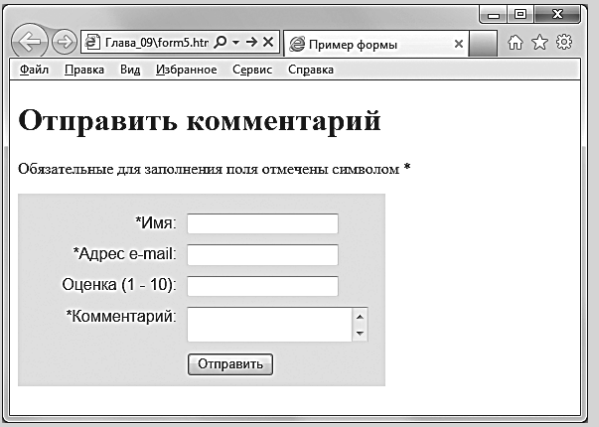


Рис. 9.36. Форма, отображенная в браузере Internet Explorer 9

Для начала запустите программу Блокнот (Notepad) и откройте страницу *Примеры\Глава_09\formcss.html* на прилагающемся к книге диске. Она показана на рис. 9.22. Сохраните файл с именем *form5.html*. Вы измените файл, чтобы создать веб-страницу, похожую на образцы, показанные на рис. 9.35 и 9.36.

1. Измените элемент `title`, указав текст «Пример формы». В элементе `h1` укажите текст «Отправить комментарий». Добавьте элемент `p`, чтобы с нового абзаца указать, что «Обязательные для заполнения поля отмечены символом *».
2. Измените глобальные стили:

- укажите значение ширины селектора `label` равным 150 пикселам;
- укажите величину левого поля `#mySubmit` равной 160 пикселам.

3. Измените элемент `form`, чтобы отправить информацию, содержащуюся в форме. Используйте метод `post` и отправьте ее скрипту на адрес **http://webdevbasics.net/scripts/demo.php**:

```
<form method="post" action="http://webdevbasics.net/
scripts/demo.php">
```

4. Модифицируйте элементы управления формы:

- конфигурируйте обязательные поля **Имя, Адрес e-mail** и **Комментарий**. Используйте звездочку, чтобы указать посетителям веб-страницы на обязательные поля;
- укажите в коде для поля адреса электронной почты элемент `type="email"` вместо `type="input"`;
- с помощью атрибута `placeholder` (см. табл. 9.2) добавьте подсказки пользователю относительно полей **Имя** и **Адрес e-mail**.

5. Добавьте ползунок (используйте атрибут `type="range"`), чтобы создать систему оценки от 1 до 10. HTML-код формы должен выглядеть следующим образом:

```
<form method="post" action="http://webdevbasics.net/
scripts/demo.php">
<label for="myName">*Имя:</label>
<input type="text" name="myName" id="myName"
required="required" placeholder="ваши имя
и фамилия">
<label for="myEmail">*Адрес e-mail:</label>
<input type="text" name="myEmail" id="myEmail"
required="required" placeholder="name@domain.ru">
<label for="myRating">Оценка (1 - 10):</label>
<input type="range" name="myRating" id="myRating"
min="1" max="10">
<label for="myComments">*Комментарий:</label>
<textarea name="myComments" id="myComments" rows="2"
cols="20" required="required"></textarea>
<input id="mySubmit" type="submit"
value="Отправить">
</form>
```

6. Сохраните файл *form5.html* и протестируйте веб-страницу в браузере. Если вы используете браузер, который поддерживает атрибуты HTML5, используемые в форме (например, Opera), ваша страница должна выглядеть примерно так, как показано на рис. 9.35. Если вы используете браузер, который не поддерживает атрибуты HTML5 (например, Internet Explorer 9), форма должна выглядеть как на рис. 9.36. Вид формы в других браузерах будет зависеть от уровня их поддержки HTML5. Список браузеров, поддерживающих HTML5 можно найти на сайте **www.standardista.com/HTML5/HTML5-web-forms**.

7. Попробуйте отправить форму без ввода информации. На рис. 9.37 показан результат в браузере Opera 11. Обратите внимание на сообщение об ошибке, которое указывает, что поле **Имя** является обязательным. Сравните свою работу с образцом на прилагающемся к книге диске (*Примеры\Глава_09\form5.html*).

   Как показало данное практическое задание, уровень поддержки атрибутов элементов формы в HTML5 различается. Пройдет некоторое время, прежде чем все браузеры начнут поддерживать эти новые возможности. Создавайте формы, помните о прогрес-

сивном улучшении и не забывайте о преимуществах и ограничениях при использовании новых возможностей HTML5.

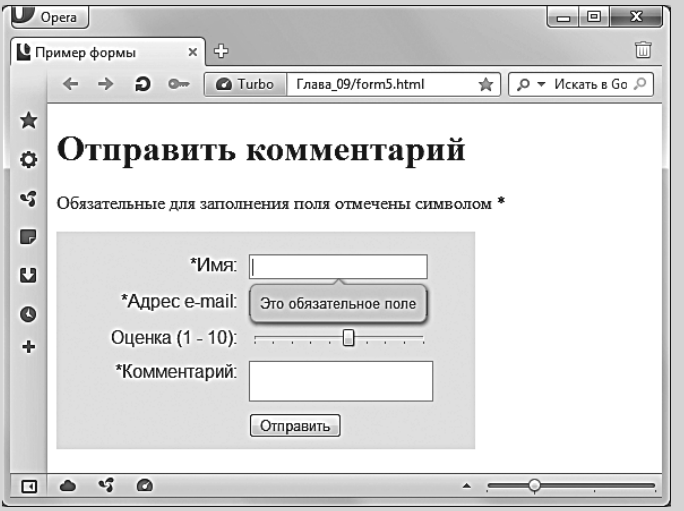


Рис. 9.37. Браузер Opera отображает сообщение об ошибке

HTML5 и прогрессивное улучшение

Используя элементы управления формы HTML5, помните о концепции прогрессивного улучшения. Браузеры, не поддерживающие HTML5, будут отображать текстовые поля вместо неподдерживаемых элементов формы. Браузеры, поддерживающие HTML5, будут отображать и обрабатывать новые элементы управления формы. Это прогрессивное улучшение в действии: все увидят удобную форму, а посетители, пользующиеся современными браузерами, только выиграют, благодаря улучшенным возможностям.

Глава 10
РАЗРАБОТКА ВЕБ-САЙТА

Цели главы

В этой главе вы узнаете следующее:

- как описать необходимые навыки, задачи и должностные обязанности, необходимые для успешной разработки веб-проекта;
- как определить стадии стандартного жизненного цикла программного обеспечения;
- другие распространенные методологии разработки программного обеспечения;
- как применять методологию жизненного цикла программного обеспечения при разработке веб-проектов;
- как определять возможности и устанавливать цели во время стадии Концептуализации;
- как определять информационные темы и требования к сайту во время стадии Анализа;
- как создавать карту сайта, макет страницы, прототип и документацию в рамках стадии Проектирования;
- как разрабатывать веб-страницы и связанные с ней файлы во время стадии Реализации;
- как проверять функциональность веб-сайта и использовать план тестирования во время стадии Тестирования;
- как получить одобрение клиента и запустить веб-сайт;
- как изменять и улучшать веб-сайт во время стадии Сопровождения;
- как сравнивать задачи, поставленные при разработке веб-сайта, и результаты полученной работы в рамках стадии Оценки;
- как найти лучшего хостинг-провайдера для вашего веб-сайта;
- как выбирать доменное имя для вашего веб-сайта.

В этой главе обсуждаются навыки, необходимые для успешной разработки широкомасштабного проекта, и происходит знакомство с общепринятыми приемами веб-разработки, выбора доменного имени и вариантов хостинга веб-сайта.

10.1. Успешная разработка широкомасштабного проекта

Широкомасштабные проекты не создаются одним или двумя людьми. Они создаются целой группой людей, которые работают вместе как одна команда. Для больших проектов обычно необходимы должности менеджера проектов, информационного дизайнера, уполномоченного представителя, копирайтера — автора рекламных текстов, редактора, графического дизайнера, администратора базы данных, администратора сети и веб-разработчика. В небольших компаниях и организациях каждый из участников проекта может выполнять обязанности нескольких должностей сразу и переключаться между своими обязанностями. Для мелкомасштабных проектов один из веб-разработчиков может сразу быть руководителем проекта, графическим дизайнером, администратором базы данных и/или информационным дизайнером. Важно понять, что каждый проект уникален, и у каждого свои потребности и требования. Его успех или провал напрямую зависит от выбора подходящих людей при создании команды для работы над проектом.

Роли участников проекта

Менеджер проектов

Менеджер проектов наблюдает за процессом разработки веб-сайта и координирует действия команды. Менеджер проектов создает план проекта и его расписание. Этот человек ответственен за промежуточные стадии разработки и конечный результат проекта. Он должен обладать отличными навыками организации, управления и коммуникации.

Информационный дизайнер

Информационный дизайнер проясняет основную цель и задачи сайта, помогает в определении функционала и организации сайта, его навигации и схемы. Веб-разработчик и/или менеджер проектов иногда берет эту роль на себя.

Уполномоченный представитель

Уполномоченный представитель занимается маркетинговым планом и целями организации. Уполномоченный представитель работает совместно с веб-дизайнерами, чтобы создать ***веб-представление***, или внешний вид и впечатление от сайта, которое соответствует маркетинговым целям организации. Кроме того, уполномоченный представитель помогает координировать работу веб-сайта с другими средствами массовой информации, используемыми для рекламы, такими как печатные издания, радио и телевидение.

Копирайтер и редактор

Автор текстов или ***копирайтер*** подготавливает и оценивает тексты. Прежде чем материал из существующих буклетов, информационных бюллетеней и технических описаний будет использован на веб-сайте, его нужно изменить для выполнения новых задач в рамках веб-сайта. ***Контент-менеджер*** или ***редактор*** могут провести совместную работу с копирайтером для проверки правописания и связанности текстов.

Контент-менеджер

Контент-менеджер принимает участие в стратегической и креативной разработке и модернизации веб-сайта. Он контролирует изменения, вносимые в содержимое сайта. Список навыков, необходимых успешному контент-менеджеру, включает в себя умения редактирования и написания текстов, маркетинга, знания специальной терминологии и навыков коммуникации. Человек с такой динамичной должностью должен участвовать в процессе внесения изменений в контент сайта.

Графический дизайнер

Графический дизайнер определяет цветовую схему сайта и графику, используемую на сайте, верстает блок-схемы и макеты страниц, создает логотипы и рисунки, а также оптимизирует изображения для размещения во Всемирной паутине.

Администратор базы данных

Администратор базы данных необходим, если сайт имеет доступ к информации, хранящейся в базах данных. Такие сотрудники создают базы данных, сопровождают базы данных (включая создание резерв-

ных копий и восстановление из них) и контролируют доступ к базам данных.

Администратор сети

Сетевой администратор настраивает и обслуживает ***веб-сервер***, устанавливает и обслуживает аппаратное и программное обеспечение, а также управляет средствами защиты доступа к нему.

Веб-разработчик/Веб-дизайнер

Должности веб-разработчика и веб-дизайнера часто взаимозаменяемы, однако веб-разработчик обычно больше занимается версткой HTML-кода и скриптов, а веб-дизайнер в основном занимается дизайном и графикой. ***Веб-дизайнер***, помимо некоторой работы с кодом, может выполнять работу, связанную с графическим дизайном, к примеру, подбор подходящей цветовой схемы, проектирование блок-схем и макетов веб-страниц, создание логотипов и рисунков, а также оптимизацию изображений для размещения во Всемирной паутине.

Веб-разработчик, которого еще называют ***специалистом по построению веб-приложений и веб-сайтов***, пишет HTML-код, CSS и скрипты, выполняющиеся на стороне клиента, такие как JavaScript. Некоторые веб-разработчики специализируются на написании скриптов, выполняющихся и на стороне сервера, с доступом к базе данных. Обычно для работы над большими проектами привлекают несколько веб-разработчиков, каждый из которых является экспертом в одной из областей своей профессии.

Критерии отбора персонала

Независимо от масштаба проекта, правильный подбор персонала является жизненно важным для его успешной реализации. При подборе людей на проект рассмотрите опыт каждого кандидата, его портфолио, полученное образование и отраслевые сертификаты.

Другим подходом к подбору персонала для веб-проекта (или разработке целого веб-сайта) станет привлечение внешних специалистов, иными словами — наем другой компании для выполнения вашей работы. Иногда внешние специалисты привлекаются для выполнения части проекта, такой как создание графики, мультимедийной анимации или

скриптов, выполняющихся на стороне сервера. При выборе этого варианта крайне важно общение между руководителем проекта и внешней организацией. Команда внешних специалистов должна четко понимать поставленные цели и сроки проекта.

Будь он маленький или большой, разрабатываемый своими силами или силами внешних специалистов, успех веб-проекта зависит от планирования и коммуникации. Формальная методология разработки веб-проекта используется для координации и облегчения планирования и коммуникации, необходимых для успешной реализации веб-проекта.

10.2. Процесс разработки

Большие корпоративные и коммерческие веб-сайты не возникают сами по себе. Они создаются постепенно, обычно следуя методологии разработки проекта. Методология разработки — это пошаговый план, который охватывает жизненный цикл проекта от начала до конца. Он состоит из ***стадий***, каждая из которых включает в себя свои особые операции и требования. Большинство современных методологий берет свои корни в ***жизненном цикле программного обеспечения*** (SDLC, System Development Life Cycle) — процессе, который использовали несколько десятилетий для создания масштабных информационных систем. Жизненный цикл ПО состоит из нескольких стадий, которые иногда называют этапами, ступенями или уровнями. Каждая стадия обычно завершается до того, как начинаются операции следующей стадии. Базовые стадии стандартного жизненного цикла ПО (рис. 10.1) — это системное исследование, системный анализ, системное проектирование, реализация системы и ее сопровождение.

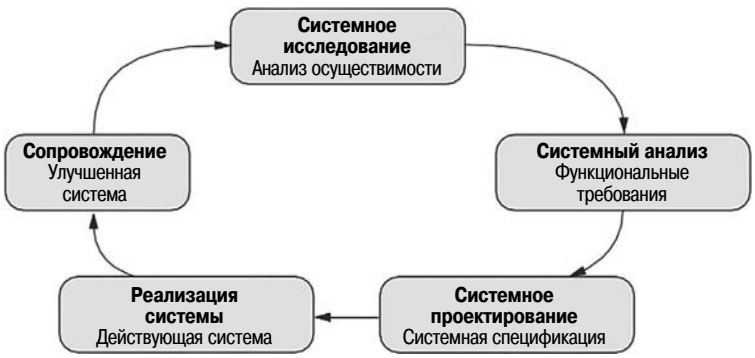


Рис. 10.1. Жизненный цикл программного обеспечения

Веб-сайты часто разрабатываются с использованием вариантов методологии жизненного цикла ПО, которые изменяют таким образом, чтобы они подходили к конкретному проекту. Большие компании и фирмы веб-дизайна обычно создают собственные методологии для использования в своих проектах. Цикл разработки веб-сайта — это путь к успешному управлению веб-проектом. В зависимости от объема работ и сложности конкретного проекта, некоторые стадии проекта могут быть завершены за одну-единственную встречу, другие стадии могут потребовать несколько недель или даже месяцев.

Цикл разработки веб-сайта, показанный на рис. 10.2, обычно состоит из следующих стадий: Концептуализация, Анализ, Проектирование, Реализация, Тестирование, Запуск, Сопровождение и Оценка.

Рис. 10.2. Цикл разработки веб-сайта

ЧаВо

А ЧТО НАСЧЕТ ДРУГИХ МЕТОДОЛОГИЙ РАЗРАБОТКИ ВЕБ-САЙТОВ?

Методология разработки, представленная в этой главе, — это версия традиционной методологии жизненного цикла ПО, измененная под задачи веб-разработки. Среди других методологий разработки можно найти следующие:

- **Макетирование.** Создается маленькая рабочая модель, которую показывают клиенту. Она постоянно улучшается разработчиком,

пока не будет соответствовать заявленным заказчиком требованиям. Этот метод можно легко включить в цикл разработки веб-сайта на стадии Проектирования.

- **Спиральная модель.** Эта методология прекрасно подходит для очень масштабных или поэтапных проектов, где очень важно снизить уровни риска. В спиральной модели небольшие части проекта завершаются одна за другой.
- **Совместная разработка приложений** (JAD, Joint Application Development). Этот тип разработки основан на групповых собраниях и сотрудничестве между пользователями и разработчиками веб-сайта или системы. Он обычно используется при разработке проекта силами самой организации (без привлечения внешних специалистов).
- **Гибкая методология разработки.** Эта методология разработки рассматривается как инновационная в том, что она обладает способностью гибко реагировать на возникающие нужды на основе создания и совместного обсуждения задач среди самой команды разработчиков и клиента. Философия этой методологии ставит код выше документации, и задачи проекта выполняются в виде множества небольших повторяющихся шагов.
- **Специальные методологии разработки отдельных организаций.** Большие компании и фирмы, занимающиеся веб-разработкой, часто создают свои собственные версии методологий по разработке сайтов, которые и используют в своих проектах.

Важным аспектом разработки веб-сайтов является то, что его никогда нельзя будет считать законченным — ваш сайт должен регулярно обновляться, кроме того, вы найдете ошибки и упущения, которые нужно будет исправить, а в дальнейшем понадобятся новые компоненты и страницы. Поэтому сначала нужно определить, какие цели должен преследовать сайт в первую очередь.

Концептуализация

Каким возможностям или целям сайт служит? Каков мотив его создания? Возможно, ваш клиент является хозяином розничного магазина и хочет продавать свои товары во Всемирной паутине.

Возможно, конкурент вашего клиента недавно создал веб-сайт, и вашему клиенту он тоже понадобился, просто чтобы не отставать. А может, у вас есть гениальная идея, которая станет в ряд с аукционом eBay!

Так как ваша главная задача — сделать сайт удобным в использовании и привлечь целевую аудиторию, вы должны определить ту аудиторию, которой сайт будет интересен. Крайне важно как можно больше знать о ваших посетителях и их предпочтениях.

Другая задача во время стадии ***концептуализации*** — это определить долгосрочные и краткосрочные задачи и цели. Возможно, краткрсрочная цель проекта — это публикация во Всемирной паутине простой домашней страницы. Возможно, долгосрочная цель проекта — это обработка 20% продаж компании-заказчика через ее веб-сайт. А может, вам просто нужно определенное количество посетителей ежемесячно. Какой бы ни была ваша цель, будет лучше, если ее успешность можно будет измерить. Подумайте, как определить уровень успешности (или неудачи) вашего веб-сайта.

Определение задач и целей сайта обычно проводится вместе с клиентом, руководителем проекта и информационным архитектором. В формальной среде проекта создается документ, детально описывающий результаты этой стадии, которая утверждается клиентом, прежде чем исполнитель приступает к самой разработке.

Анализ

Стадия анализа включает в себя деловые встречи и совещания с ключевым персоналом клиента. ***Анализ*** обычно проводится менеджером проектов, информационным дизайнером или другим аналитиком, а также уполномоченным представителем клиента и связанными с ним сотрудниками. Сетевой администратор и администратор базы данных тоже могут участвовать в совещаниях в зависимости от масштабов проекта. Во время стадии анализа обычно выполняются следующие задачи:

- **Определите информационные темы.** Упорядочьте представленную на сайте информацию в категории и создайте для них систему иерархии. Эти информационные темы позже будут использованы как стартовая точка для разработки навигации сайта.
- **Определите требования к функциональным возможностям.** Укажите, *что* сайт должен будет выполнять, а не то, *как* он будет это делать. Например, укажите «сайт будет принимать заказы клиентов по банковским картам», но не «сайт будет обрабатывать заказы клиентов, используя технологию ASP для просмотра информации о цене с НДС в базах данных Oracle, и будет использовать подтверждение

подлинности банковской карты в реальном времени, с помощью функций сайта somewebsite.com». Обратите внимание на разницу в уровне детализации этих ***требований к функциональным возможностям***.

- **Определите условия работы сайта.** Какие ***условия работы сайта*** потребуются его посетителям? Определите, какое аппаратное обеспечение, операционные системы, объемы памяти, разрешения экрана и пропускные способности будут использовать ваши посетители. Какой тип аппаратных и программных требований потребуется веб-серверу?

- **Определите требования к контенту.** Возможно, контент уже существует в другом формате — брошюрах, каталогах, технических описаниях? Определите, кто будет ответственен за создание и переназначение содержимого для сайта. Есть ли какие-то ***требования к контенту*** у компании-клиента или отдела маркетинга? Например, существует ли какой-то особый дизайн или предусматривается определенное «ощущение», которое должен испытать клиент, посетив сайт; а может, какой-то элемент бренда, который должен на нем присутствовать?

- **Сравните Старый Подход и Новый Подход.** Возможно, вы не создаете новый веб-сайт, а просто изменяете уже существующий. Какие выгоды или дополнительную ценность принесет новая версия?

- **Оцените сайты конкурентов.** Внимательная оценка веб-представлений ваших конкурентов поможет вам спроектировать сайт, который будет выделяться из ряда прочих и станет более востребованным среди потенциальных клиентов. Укажите хорошие и плохие стороны этих сайтов.

- **Оцените затраты.** Спрогнозируйте финансовые и временны́е затраты, которые понадобятся для создания сайта. На этой стадии часто создается или изменяется формальный план проекта. Довольно часто для оценки затрат и сроков выполнения проекта используют специальное приложение, такое как Microsoft Project.

- **Проведите анализ соотношения затрат и прибыли.** Создайте документ, который будет сравнивать затраты и прибыли сайта. Измеренная прибыль обычно наиболее привлекательно выглядит в глазах клиента. В формальной среде проекта документ, детально описывающий результаты этого **анализа соотношения затрат и прибыли**, должен быть утвержден клиентом, прежде чем команда разработчиков сможет продолжить работу.

Проектирование

Когда все уже знают, какой результат необходимо получить, настает момент определить, как его добиться. Стадия проектирования включает в себя встречи и совещания с ключевым персоналом компании-клиента. Задачи ***проектирования*** обычно выполняются менеджером проектов, информационным дизайнером или другим аналитиком, графическим дизайнером, старшим веб-разработчиком (или их командой) с одной стороны и уполномоченным представителем клиента и относящимся к нему персоналом с другой. В стадию проектирования обычно входят следующие задачи:

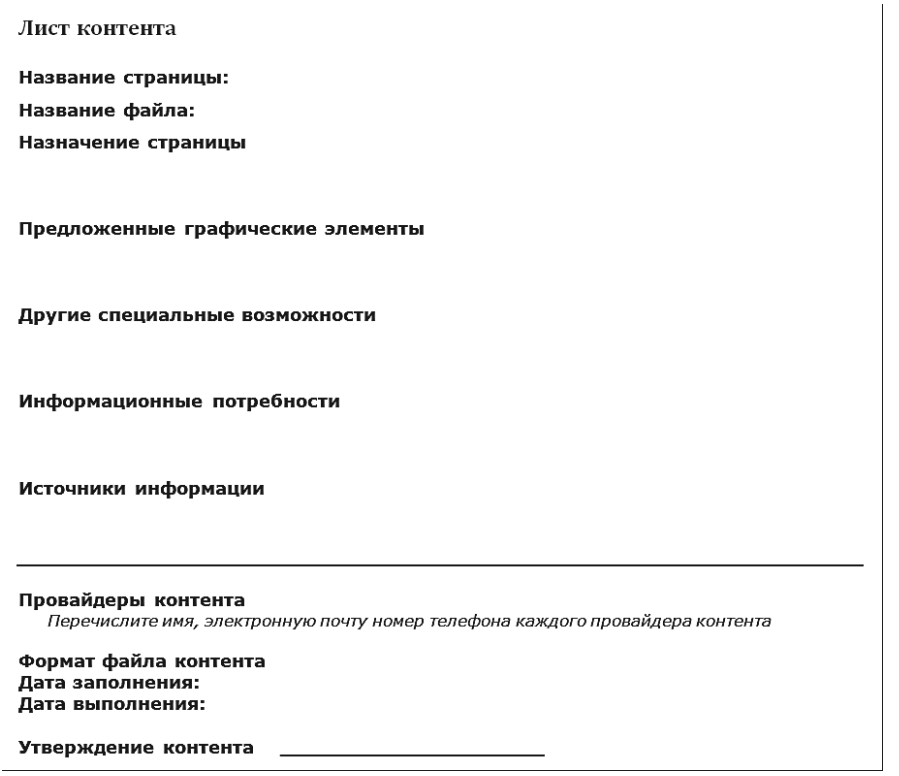
- **Выберите структуру сайта.** Как обсуждалось в разделе 5, обычными формами структур организации веб-сайта являются иерархическая, линейная и хаотичная. Определите, какая из них лучше всего подойдет для сайта проекта, и создайте карту сайта (которую иногда называют блок-схемой или раскадровкой).

- **Создайте прототип проекта.** Для начала сделайте набросок проекта на бумаге. Иногда полезно делать наброски в пустом окне браузера (см. файл *Примеры\Глава_10\Набросок.doc* на диске, прилагающемся к книге). Довольно часто для создания макета или блок-схемы используют графические приложения. Получившиеся макеты можно показать клиенту как прототип или как рабочую модель системы для ее одобрения заказчиком. Кроме того, их можно предоставить на рассмотрение фокус-группам для ***тестирования юзабилити***.

- **Разработайте проект макета страницы.** Общий макет, или представление сайта, тоже нужно спроектировать. Проект макета страницы используется как руководство по созданию макетов главной страницы и страницы с контентом. В нем должны быть утверждены такие элементы, как цветовая схема сайта, размер логотипа, изображения кнопок и текст. Используя проект макета страницы и карту сайта, создайте образцы макетов для главной страницы и страниц контента. Используйте графическое приложение для создания макетов этих страниц, чтобы получить представление того, как сайт будет функционировать. Если вы используете ваш инструмент по разработке веб-приложений на этой ранней стадии, вы рискуете вызвать у вашего руководителя или клиента мысль о том, что проект уже наполовину готов, в результате чего он может начать настаивать на его ранней сдаче.

- **Создайте документ каждой страницы .** Хотя это и кажется необязательным, недостаток контента является довольно частой причиной

задержки сдачи проектов веб-сайта. Подготовьте документ с содержимым для каждой страницы, такой как на рис. 10.3 (он доступен в файле *Примеры\Глава_10\Лист контента.doc* на диске, прилагающемся к книге), который описывает функциональные возможности страницы, текст и требования к графическому контенту, источники контента и графу утверждения контента.



Лист контента

Название страницы:
Название файла:
Назначение страницы

Предложенные графические элементы

Другие специальные возможности

Информационные потребности

Источники информации

Провайдеры контента
Перечислите имя, электронную почту номер телефона каждого провайдера контента

Формат файла контента
Дата заполнения:
Дата выполнения:

Утверждение контента _____

Рис. 10.3. Образец листа контента

Карта сайта и прототипы проектов страниц обычно утверждаются клиентом, прежде чем команда разработчиков сможет продолжить работу на стадии реализация.

Реализация

Во время стадии ***реализации*** вся ранее проделанная работа сводится вместе (в идеале) в удобный и эффективный веб-сайт. Во время стадии реализации веб-разработчики находятся на критическом пути — их

работа должна быть выполнена строго по плану, иначе задержится сдача всего проекта. Другие члены команды разработчиков при необходимости опрашиваются для разъяснения задач и утверждения проделанной работы. В стадию реализации обычно входят следующие задачи:

- **Выберите инструментарий для верстки веб-страниц.** Использование инструментария для верстки веб-страниц, такого как Adobe Dreamweaver или Microsoft Expression Web, может значительно увеличить продуктивность работы. К особым вспомогательным инструментам относятся заметки проектировщика, шаблоны страниц, управление задачами, а также проверка веб-страниц до и после сдачи, во избежание наложения обновлений страниц друг на друга. Использование инструментария для верстки веб-страниц также поможет при стандартизации HTML-кода, использованного в страницах проекта. Любые стандарты, относящиеся к форматированию абзацев, комментариев, и т. д., уже должны быть определены к этому времени.
- **Упорядочьте файлы вашего сайта.** Позаботьтесь о размещении изображений и файлов мультимедиа в их собственных папках. Поместите скрипты, выполняющиеся на стороне сервера, в отдельную папку. Определите условные обозначения имен для веб-страниц, изображений и файлов мультимедиа.
- **Разработайте компоненты и проведите их индивидуальное тестирование.** Во время выполнения этой задачи графические дизайнеры и веб-разработчики создают и отдельно тестируют те части сайта, за которые они отвечают. Сразу после создания изображений, веб-страниц и скриптов, выполняющихся на стороне сервера, они проходят отдельное тестирование. Это называется ***блочное тестирование***. В некоторых проектах старшие веб-разработчики или менеджер проектов проводят проверку компонентов на качество и соблюдение стандартов.

Когда все компоненты созданы, а блоки протестированы, настает время свести их вместе и приступить к стадии тестирования.

Тестирование

Компоненты следует опубликовать на тестовом веб-сервере. На этом веб-сервере должна быть установлена такая же операционная система и программное обеспечение, которое будет работать на реальном веб-

сервере сайта. Вот некоторые из наиболее часто выполняющихся задач ***тестирования***:

- **Протестируйте сайт в разных браузерах и версиях браузера.** Крайне важно протестировать вашу страницу во всех популярных браузерах и во всех версиях этих браузеров.
- **Протестируйте сайт при разном разрешении экрана .** Хотя вы, как веб-разработчик, можете использовать очень высокое разрешение экрана, не все пользователи используют разрешение 1920×1200 пикселов. Наиболее часто используемыми разрешениями экрана на момент написания этой книги являются 1024×768, 1280×800 и 1366×768 (**gs.statcounter.com**). Обязательно протестируйте ваши веб-страницы при разных разрешениях экрана — вы удивитесь полученным результатам.
- **Протестируйте сайт при разной пропускной способности.** В крупных городах пользователи зачастую обладают широкополосным доступом в Интернет. Тем не менее многие люди до сих пор используют коммутируемое подключение к Интернету с помощью телефонной линии. Важно протестировать ваш сайт как при быстром соединении, так и при медленном. Изображения, загрузку которых вы не замечаете при FTTx-подключении, могут крайне медленно загружаться при коммутируемом подключении со скоростью 56 Кбит/с.
- **Протестируйте сайт на другом компьютере.** Обязательно протестируйте ваш веб-сайт, используя другой компьютер, а не тот, на котором выполнялась его разработка, чтобы более точно симулировать обычного посетителя.
- **Протестируйте сайт на мобильных устройствах.** Посещение Всемирной паутины с помощью мобильных устройств становится все более популярным — протестируйте ваш сайт на одном или нескольких популярных смартфонах.
- **Тестируйте, тестируйте, тестируйте.** Не существует такого понятия, как чрезмерное тестирование. Люди совершают ошибки. Гораздо лучшим вариантом для вас и вашей команды будет найти эти ошибки самостоятельно, чем дать возможность клиенту указать на них при проверке сайта.

Звучит так, будто предстоит следить за многими вещами? Так и есть. Вот почему рекомендуется создать *план тестирования* — документ, описывающий, что будет протестировано на каждой странице веб-сайта.

Образец плана тестирования, показанный на рис. 10.4 (см. файл *Примеры\Глава_10\План тестирования.pdf* на диске, прилагающемся к книге), может помочь организовать тестирование, пока вы будете проверять свой сайт в разных браузерах и при разных разрешениях экрана. Раздел валидации документа затрагивает контент, ссылки и любые формы или скрипты, которые необходимы для корректной работы страницы. Метатеги для поисковой оптимизации рассматриваются в главе 13. Тем не менее уже на этой стадии вы должны убедиться, что заголовок страницы описательный и включает в себя название компании или организации. Тестирование вашей страницы при разной пропускной способности важно еще потому, что страницы, загрузка которых занимает длительное время, часто просто закрывают до окончания загрузки.

План тестирования документа веб-страницы

Имя файла:		Дата:	
Название страницы:		Тестер:	

Совместимость с браузером

	1024x768	1280x800	800x600	Другое	PC	Mac	Linux	с отключенными изображениями	с отключенными таблицами CSS	Другое	Заметки
Internet Explorer (версия #)											
Internet Explorer (версия #)											
Firefox (версия #)											
Safari (версия #)											
Opera (версия #)											
Chrome (версия #)											
JAWS Screen Reader											
Смартфон (название устройства)											
Другое											

Валидация документа

	Успех	Неудача	Заметки
Валидация XHTML			
Валидация CSS			
Проверка правописания			
Проверка связанного содержимого			
Проверка связанных изображений			
Проверка атрибутов alt			
Тестирование гиперссылок			
Тестирование доступности			
Обработка форм			
Работа сценариев/динамических эффектов			
Тестирование удобства			
Другое			

Оптимизация поисковых систем

	Заметки
Мета-теги (описание)	
Ключевые слова в названии страницы	
Ключевые слова в заголовках	
Keywords in content	
Other	

Проверка времени загрузки

	Время	Заметки
56.6 Кб/с		
128 Кб/с		
512 Кб/с		
T1/DS1 (1.544 Мб/с)		
Другая скорость		

Заметки

Рис. 10.4. Образец плана тестирования

Автоматические инструменты тестирования и валидаторы

В программах для верстки веб-страниц, которые вы используете при работе над проектом, скорее всего будут встроены инструменты отчетности и тестирования сайта. В такие приложения, как Adobe Dreamweaver и Microsoft Expression Web, заложены функции проверки орфографии и проверки ссылок, а также расчет времени загрузки страницы. Каждое приложение обладает своими уникальными особенностями. Программа Dreamweaver позволяет получить отчеты с результатами проверки ссылок, доступности и валидности кода. Существуют и другие ***автоматические инструменты тестирования*** и ***валидации***. Сервис проверки разметки Консорциума W3C (**validator.w3.org**) можно использовать

для валидации как HTML, так и XHTML-страниц. Протестируйте CSS на допустимость синтаксиса, используя сервис валидации CSS Консорциума W3C (**jigsaw.w3.org/css-validator**). Проанализируйте скорость загрузки вашей страницы, используя сервис **www.websiteoptimization.com/services/analyze/**. Корпорация Adobe также предлагает провести кросс-браузерное тестирование по адресу **browserlab.adobe.com/**. Инструменты, которые включают дополнительные возможности тестирования, такие как проверка орфографии, совместимость с браузерами, скорость загрузки страницы и проверка работоспособности ссылок, доступны по адресу **netmechanic.com** и на других сайтах. Посетите страницу **www.softwareqatest.com/qatweb1.html**, чтобы узнать о некоторых из них.

Тестирование на доступность

Доступными веб-страницами могут пользоваться все, включая людей с нарушениями зрения, слуха, а также двигательными и когнитивными расстройствами. На протяжении всей книги доступность была неотъемлемой частью дизайна и кода ваших веб-страниц, а не запоздалым размышлением. Вы конфигурировали заголовки и подзаголовки, навигацию в виде неупорядоченных списков, изображения с альтернативным текстом и связь между текстом и элементами управления формы. Все эти методы повышают доступность веб-страниц.

Стандарты доступности веб-сайта

В Российской Федерации требования доступности устанавливаются и регулируются Федеральным агентством по техническому регулированию и метрологии (**www.gost.ru**). Требования приведены в документе ГОСТ Р 52872-2007, электронную версию которого вы найдете на диске, прилагающемся к книге.

В руководстве по обеспечению доступности веб-контента версии 2.0 (Web Content Accessibility Guidelines (WCAG 2.0))[1], согласно которому доступной считается страница, которая легко воспринимаема, управляема и понятна людям с различными возможностями. Страница должна быть достаточно надежна, чтобы работать в разнообразных браузерах и других пользовательских программах, таких как вспомогательные технологии (к примеру, программы экранного доступа) и мобильные устройства. Руководящие принципы WCAG 2.0 следующие:

[1] www.w3.org/TR/WCAG20

1. Содержимое сайта должно быть ***воспринимаемым***.
2. Компоненты интерфейса содержимого должны быть ***управляемыми***.
3. Контент и управление должны быть ***понятны***.
4. Содержимое сайта должно быть достаточно ***надежным***, чтобы работать с пользовательскими устройствами, существующими в настоящее время, и с теми, которые появятся в будущем, включая вспомогательные технологии.

Проверьте соответствие вашего сайта этим требованиям, выполнив ***тестирование на доступность***. Существует множество сервисов такой проверки. Инструментарий Adobe Dreamweaver обладает встроенным инструментом проверки доступности. Два популярных бесплатных онлайн-инструмента оценки доступности: WebAIM Wave[1] и ATRC AChecker[2]. Кроме того, для оценки доступности можно использовать некоторые расширения браузеров, включая Web Developer[3], WAT-C Web Accessibility[4] и AIS Web Accessibility[5].

Помните, что нельзя полагаться только на автоматические тесты, необходимо просмотреть страницы самостоятельно. Например, автоматическое тестирование может проверить наличие атрибута `alt`, однако только человек способен критически осмыслить замещающий текст и решить, подойдет ли он как описание для человека, который не видит изображение. Компания WebAIM предлагает подробный список для проверки на веб-сайте **www.webaim.org/standards/wcag/checklist**, который подскажет вам, какие элементы просмотреть, чтобы убедиться в соответствии веб-страницы требованиям WCAG 2.0.

Тестирование юзабилити

Юзабилити — это степень удовлетворения, получаемого пользователем в результате взаимодействия с веб-сайтом. Смысл юзабилити в том, чтобы создать легко воспринимаемый, эффективный сайт, который нравится пользователям.

На сайте Usability.gov перечислены пять факторов, которые влияют на впечатление пользователей:

[1] wave.webaim.org
[2] www.achecker.ca/checker
[3] chrispederick.com/work/web-developer
[4] www.wat-c.org/tools
[5] www.visionaustralia.org.au/ais/toolbar

- **Простота освоения.** Насколько легко научиться пользоваться сайтом? Предлагается ли интуитивно понятная навигация? Считает ли новый посетитель, что научиться выполнять на веб-сайте простые действия легко, или он разражен?
- **Эффективность использования.** Как опытные пользователи воспринимают веб-сайт? Освоившись на сайте, могут ли они выполнять задачи быстро и эффективно, или они недовольны?
- **Запоминаемость.** Когда посетители возвращаются на сайт, помнят ли они достаточно, чтобы продуктивно его использовать, или им приходится осваивать все заново (и они недовольны)?
- **Частота и серьезность ошибок.** Делают ли посетители сайта ошибки при навигации или заполнении формы? Серьезные ли это ошибки? Легко ли их исправить?
- **Субъективное впечатление.** Пользователям понравился веб-сайт? Они удовлетворены? Почему?

Тестирование того, как настоящие посетители веб-сайта будут его использовать, и называется ***тестированием юзабилити***. Его можно провести на любой стадии разработки веб-сайта, и довольно часто эту процедуру проводят более одного раза. Тестирование юзабилити проводят, попросив пользователей выполнить несколько задач на веб-сайте, таких как оформление заказа, поиск телефона компании или поиск определенного продукта. Типы заданий варьируются в зависимости от того, какой именно сайт проходит тестирование. За пользователями наблюдают, пока они пытаются выполнить эти действия. Кроме того, их просят вслух выражать возникающие сомнения и трудности. Результаты записывают (часто на видео) и обсуждают с командой дизайнеров. Часто в навигацию и макеты страниц вносят изменения на основе этих тестирований.

Если тестирование юзабилити проводится на ранней стадии разработки веб-сайта, для него можно использовать макеты страницы на бумаге и карту сайта. Если команда веб-разработчиков столкнулась с проблемой в выборе дизайна, иногда тестирование юзабилити может помочь с определением лучшего варианта дизайна.

Когда тестирование юзабилити проводится на позднем этапе разработки веб-сайта, таком как стадия тестирования, тестируется уже сам веб-сайт. Это может дать уверенность в том, что сайт прост в использовании и правильно спроектирован, привести к изменениям веб-сайта

в последнюю минуту или к составлению плана его улучшения в ближайшем будущем.

Запуск

Ваш клиент — будь это другая компания или другой отдел в вашей организации — должен проверить и утвердить тестовый веб-сайт, прежде чем файлы будут опубликованы на реальном сайте. Иногда это утверждение получают на личной встрече, а иногда клиенту высылают ссылку на тестовый сайт, после чего тот утверждает его или просит о внесении изменений в проект через электронную почту.

Как только веб-сайт утвержден, он публикуется на рабочем веб-сайте (это называется ***запуск***). Если вы думаете, что на этом работа окончена — подумайте еще раз! Крайне важно протестировать все компоненты сайта после публикации, чтобы убедится, что сайт правильно функционирует в новых условиях. После этого обычно начинается работа по рекламированию и продвижению сайта (см. главу 13).

Сопровождение

Веб-сайт никогда не будет завершен. Всегда есть ошибки или упущения, которые пропустили во время процесса разработки. Клиенты обычно находят множество новых способов использования веб-сайта, когда он у них появляется, и требуют изменений, дополнений и добавления новых разделов (это называется ***сопровождением*** сайта). Поэтому на этой стадии команда проектировщиков выявляет новые возможности или улучшения и начинает новый цикл процесса разработки.

Другие типы обновлений относительно незначительны — возможно, появится неработающая ссылка, найдется ошибка в орфографии или нужно будет заменить графический элемент. Эти мелкие изменения обычно вносятся сразу, как только их замечают. Вопрос того, кто их вносит и кто утверждает, часто зависит от политики самой компании. Если вы внештатный веб-разработчик, ситуация будет куда ясней — вы вносите изменения, а ваш клиент их утверждает.

Оценка

Помните цели, поставленные для веб-сайта на стадии концептуализации? Во время стадии ***оценки*** настает время просмотреть их и опреде-

лить, соответствует ли ваш сайт поставленным целям. Если нет — подумайте о том, как вы можете улучшить сайт, и начните новый цикл процесса разработки.

10.3. Доменное имя

Важнейшей частью создания эффективного веб-представления является выбор ***доменного имени***; оно служит для определения расположения вашего сайта во Всемирной паутине. Если ваш бизнес или организация только создаются, хорошим решением станет выбор доменного имени одновременно с выбором имени для самой организации. Если ваша организация уже успешно учреждена, вам стоит выбрать доменное имя, которое относится к вашему текущему виду бизнеса. И хотя многие доменные имена давно выкуплены, в вашем распоряжении все еще остается огромный выбор.

Выбор доменного имени

- **Опишите ваш бизнес.** Хотя существует сложившаяся традиция выбирать «веселые» слова как доменные имена (например, **yahoo.com**, **google.com**, **bing.com**, **woofoo.com** и т. д.), хорошенько подумайте, прежде чем так делать. Доменное имя для традиционного бизнеса и организаций — это основа для представления организации во Всемирной паутине, и оно должно включать в себя название или цель бизнеса.
- **По возможности будьте лаконичны.** Хотя большинство людей находят новые сайты через поисковые системы, некоторые из ваших посетителей будут вводить ваше доменное имя в адресную строку браузера вручную. Короткое доменное имя предпочтительнее длинного — его будет легче запомнить вашим посетителям.
- **Избегайте дефисов (-).** Использование символа дефиса (который обычно называют тире) в доменных именах делает сложным произношение имени. Кроме того, тот, кто будет вручную вводить ваш адрес, может забыть поставить дефис, и в результате зайти на сайт вашего конкурента! Если можете — избегайте использования дефиса в доменных именах.
- **Существуют другие домены, кроме .com.** Хотя домен верхнего уровня .com наиболее популярен среди коммерческих и личных

веб-сайтов, подумайте о регистрации вашего доменного имени с другими доменами верхнего уровня, такими как .biz, .net, .ru, .mobi и т. д. Коммерческим организациям следует избегать регистрации в домене .org, который рекомендуется для некоммерческих организаций. Вам не нужно создавать веб-сайт для каждого доменного имени, которое вы регистрируете. Вы можете назначить для вашего доменного имени у своего регистратора (например, **www.webnames.ru**) «дополнительные» доменные имена, которые будут приводить ваших посетителей на то доменное имя, по которому расположен ваш веб-сайт. Это называется ***перенаправлением домена***.

- **Обдумайте потенциальные ключевые слова.** Подумайте о тех словах, которые ваши потенциальные посетители могут вводить в поисковую систему, когда будут искать организацию или бизнес вашего типа. Это отправная точка для вашего списка ***ключевых слов***. Если это возможно, включите одно или более ключевых слов в ваше доменное имя (но все же постарайтесь сделать его как можно короче).

- **Проверьте доменное имя.** Изучите, как ваше потенциальное доменное имя и ключевые слова уже используются во Всемирной паутине. Хорошим решением будет ввести ваше потенциальное доменное имя (и относящиеся к нему слова) в поисковую систему, чтобы посмотреть, какие сайты на их основе уже существуют.

- **Проверьте доступность имени.** Проверьте на сайте одного из множества ***регистраторов доменных имен*** доступность выбранных вами имен.

  Некоторые из сайтов, предоставляющих услуги регистрации доменного имени, перечислены ниже:

  — **www.webnames.ru**

  — **www.nic.ru**

  — **www.r01.ru**

Каждый из этих сайтов предоставляет возможность проверки любого доменного имени на доступность (в базе WHOIS), что дает вам возможность узнать, занято ли ваше доменное имя, и если занято, то кем. Довольно часто оказывается, что нужное вам доменное имя уже занято. В этом случае, система проверки предложит вам альтернативные решения, которые могут вам подойти. Не сдавайтесь — какое-то свободное доменное имя дожидается вашего бизнеса.

Регистрация доменного имени

Когда вы определитесь с идеальным доменным именем, не откладывайте его регистрацию. Цены за регистрацию доменного имени варьируются, но всегда остаются довольно умеренными. Средняя цена за регистрацию домена .com продолжительностью на один год в данный момент составляет около 500 рублей (кроме того, существуют системы скидок за предварительные платежи на несколько лет вперед или услуги совместного хостинга). Вполне нормально будет зарегистрировать доменное имя, даже если вы не готовы разместить ваш сайт немедленно. Существует множество компаний, которые предоставляют услуги регистрации доменов, как указано выше. Когда вы регистрируете доменное имя, ваша контактная информация (такая как имя, номер телефона, адрес фактической и электронной почты) будут занесены в базу WHOIS, и доступны для любого пользователя, если вы, конечно, не выберите параметр ***приватной регистрации***. Хотя за сокрытие ваших данных взимается небольшая ежегодная плата, она защищает вашу личную информацию от нежелательного спама и простого любопытства.

Приобретение доменного имени — это лишь первая часть на пути основания своего представления во Всемирной паутине; вам также понадобится где-то разместить свой веб-сайт. Следующий раздел познакомит вас с информацией, связанной с выбором веб-хостинга.

10.4. Веб-хостинг

Где найти подходящее место для «проживания» вашего веб-сайта? Выбор наиболее подходящей хостинговой компании, или ***хостинг-провайдера***, для вашего клиента или бизнеса может стать одним из самых важных принятых вами решений. Хороший сервис веб-хостинга предоставит вам надежный дом для вашего веб-сайта. Плохой веб-хостинг станет источником проблем и жалоб. Какой вы выберите?

Хостинг-провайдеры

Хостинг-провайдер — это организация, предлагающая услуги по хранению файлов вашего веб-сайта, а также предоставлению доступа к ним в сети Интернет. Ваше доменное имя, к примеру **vashsite.ru**, связано с IP-адресом, который указывает на ваш веб-сайт на веб-сервере хостинг-провайдера. Обычно хостинг-провайдеры взимают дополнительную пла-

ту за настройку хостинга в дополнение к ежемесячной оплате. Цены за оплату хостинга крайне широко различаются. Самый дешевый хостинг-провайдер не обязательно окажется тем, кто вам нужен. Никогда даже не думайте об использовании услуг «бесплатных» хостинг-провайдеров для коммерческих сайтов. Эти бесплатные сайты хороши для детей, студентов и людей, создавших сайт ради своего хобби, но они не являются профессиональными. А последнее, с чем захочет столкнуться ваш клиент, — это с непрофессиональным или несерьезным подходом к его бизнесу.

Когда вы будете выбирать хостинг-провайдера, обязательно проверьте отзывы о компаниях, которые вас заинтересовали. Также попробуйте связаться с ними по телефонам и электронной почте их службы поддержки, чтобы определить насколько быстро она работает на самом деле. Информация от знакомых, поисковые системы, местный телефонный справочник и списки компаний хостинга во Всемирной паутине, такие как **www.hostobzor.ru**, помогут вам определиться с выбором идеального для вас хостинг-провайдера.

Типы веб-хостинга

- **Виртуальный** хостинг или общий хостинг обычно выбирают маленькие веб-сайты (рис. 10.5). Сервер хостинг-провайдера разделен на определенное количество виртуальных доменов, и несколько веб-сайтов установлены на одном и том же компьютере. Вы можете обновлять файлы на пространстве своего веб-сайта, а хостинг-провайдер поддерживает работу веб-сервера и соединение с Интернетом.

Рис. 10.5. Виртуальный веб-хостинг

- **Выделенный веб-сервер** означает аренду и эксклюзивное использование компьютера и его соединения с Интернетом, который рас-

положен в офисе хостинг-провайдера, на его технической площадке. Выделенный сервер обычно нужен сайтам, которые используют большой объем трафика, например десять миллионов хитов в день. Сервер обычно можно конфигурировать и управлять им удаленно, из офиса клиента, или вы можете заплатить хостинг-провайдеру, чтобы он администрировал его вместо вас.

- ***Собственный сервер на технической площадке хостинг-провайдера*** (изредка такая ситуация называется колокацией) — это компьютер, который ваша организация сама приобрела и сконфигурировала. Ваш сервер хранится и подключается к Интернету на технической площадке хостинг-провайдера, но ваша организация администрирует этот компьютер.

ЧаВо

ПОЧЕМУ МЕНЯ ДОЛЖНА ВОЛНОВАТЬ ОПЕРАЦИОННАЯ СИСТЕМА, КОТОРУЮ ИСПОЛЬЗУЕТ МОЙ ХОСТИНГ-ПРОВАЙДЕР?

Знание того, какую операционную систему использует ваш провайдер, важно потому, что это может помочь вам справиться с проблемами, возникающими с вашим сайтом. Довольно часто сайты начинающих прекрасно работают на их собственных компьютерах (обычно в операционной системе Windows), но «падают» (ссылки перестают работать, а изображения загружаться) после размещения на бесплатном сервере, который использует другую операционную систему.

Некоторые операционные системы, такие как Windows, обрабатывают буквы в верхнем регистре и нижнем регистре абсолютно одинаково. Другие операционные системы, такие как UNIX и Linux, считают буквы в верхнем регистре и нижнем регистре разными. Это называется ***чувствительностью к регистру***. Например, когда веб-сервер, работающий на основе операционной системе Windows, получает запрос, сформированный элементом привязки, таким как `<a href="MyPage.html">Моя Страница</a>`, он вернет файл, который назван любой комбинацией этих букв в верхнем и нижнем регистре. Значения *MyPage.html*, *mypage.html*, *myPage.html* можно использовать с одинаковым результатом. Тем не менее, когда запрос, сгенерированный по тому же элементу привязки, получен веб-сервером, работающим в операционной системе UNIX (которая чувствительна к регистру), файл будет найден, только если он сохранен именно как *MyPage.html*. Если файл назван *mypage.html*, произойдет ошибка 404 (страница не найдена). Это хороший стимул быть последовательным при формировании имен файлов — подумайте о том, чтобы всегда использовать нижний регистр для названий файлов.

10.5. Выбор виртуального хостинга

Мы обсудили (или обсудим дальше) ряд важных факторов, влияющих на выбор виртуального хостинга, включая пропускную способность, объем дискового пространства, техническую поддержку и возможность подключения пакетов электронной коммерции. Список этих и других факторов, важных при выборе виртуального хостинга, можно увидеть в табл. 10.1.

Таблица 10.1. Контрольный лист выбора хостинга

Операционная система	UNIX Linux Windows	Некоторые веб-хостинги предлагают выбор этих платформ. Если вам нужно объединить ваш веб-сайт с вашими бизнес-системами, выберите одинаковую операционную систему в обоих случаях
Веб-сервер	Apache IIS	Эти два серверных приложения самые популярные. Apache обычно работает на операционных системах UNIX или Linux. IIS (Internet Information Services) предназначен для работы в некоторых операционных системах Windows корпорации Microsoft
Ширина канала	_____ Мб или Гб _____ Изменения при превышении порога	Некоторые хостинг-провайдеры внимательно следят за объемом передаваемых вашим сайтом данных, или *трафиком*, и взимают дополнительную плату за перерасход трафика. Хотя неограниченный объем трафика — замечательная вещь, он не всегда доступен. Типичный сайт с низким посещением обычно колеблется между 100 и 200 Мб трафика в месяц. Сайту со средним объемом трафика должно хватить 20 Гб в месяц
Техническая поддержка	Электронная почта Чат Форум Телефон	Прочитайте описание технической поддержки на сайте хостинг-провайдера. Доступна ли она круглосуточно и без выходных? Напишите в техподдержку письмо или позвоните по телефону, чтобы проверить это. Если организация не отвечает вам как возможному клиенту, задумайтесь о том, как к вам будут относиться, когда вы им станете
Соглашение об уровне услуг	Гарантия безотказной работы Автоматический мониторинг	Хостинг-провайдер, предлагающий Соглашение об уровне услуг (SLA) с гарантией безотказной работы, показывает, что они ценят уровень своих услуг и надежности. Использование функций автоматического мониторинга сообщит в техподдержку хостинг-провайдера о неисправностях на сервере, если таковые возникнут
Дисковое пространство	_____ Гб	Многие из виртуальных хостингов, как правило, предлагают несколько гигабайт дискового пространства. Если у вас маленький сайт, который почти не использует изображения, вам, возможно, будет достаточно и 100 Мб
Электронная почта	_____ Почтовых ящиков	Большинство виртуальных хостингов предлагают несколько ящиков электронной почты на каждый сайт. Эту возможность можно использовать, чтобы фильтровать сообщения — служба работы с клиентами, техническая поддержка, общие вопросы и т. д.

Загрузка файлов на сервер	FTP-доступ Файл-менеджер с веб-интерфейсом	Веб-хостинг, который предлагает доступ по протоколу FTP, предоставит вам наибольшую гибкость. Другие позволяют вносить обновления только через приложение менеджера файлов. Некоторые хостинги предлагают оба варианта
Стандартные скрипты	Обработка форм _____	Многие хостинги предоставляют готовые, заранее написанные, скрипты для обработки информации введенной в формы
Поддержка скриптов	PHP .NET _____ (другой)	Если вы планируете использовать скрипты, выполняющиеся на стороне сервера, определите, какие из них поддерживает ваш хостинг и поддерживает ли он их вообще
Поддержка баз данных	MySQL MS Access MS SQL	Если вы планируете использовать базу данных для ваших скриптов, определите, какой тип баз данных поддерживает ваш хостинг и поддерживает ли он их вообще
Пакеты электронной коммерции	_____	Если вы планируете заняться электронной коммерцией (см. главу 12), возможно будет легче, если ваш веб-хостинг предлагает пакет услуг, в который включена система электронной коммерции. Проверьте, так ли это
Расширяемость	Скрипты База данных Электронная коммерция	Скорее всего, вы выберете базовый (начальный) пакет услуг для вашего первого веб-сайта. Обратите внимание на расширяемость вашего хостинга — существуют ли другие пакеты с поддержкой скриптов, баз данных, пакетами электронной коммерции и дополнительной пропускной способностью или дисковым пространством, которые могут вам понадобиться с ростом сайта
Резервное копирование	Ежедневное Периодическое Не производится	Большинство хостингов будут регулярно создавать резервные копии ваших файлов. Проверьте, как часто это будет происходить, и доступны ли будут резервные копии вам. Не забывайте создавать резервные копии своего сайта самостоятельно
Статистика сайта	Необработанные файлы журнала событий Отчет журнала событий Нет доступа к журналам событий	Журнал событий сервера содержит много полезной информации о ваших посетителях, как они находят ваш сайт, и какие страницы их интересуют. Проверьте, доступен ли вам журнал событий. Некоторые хостинги предоставляют отчеты журнала событий (см. главу 13, чтобы узнать больше о журналах событий).
Имя домена	Требуется зарегистрировать с хостингом Можно зарегистрировать самостоятельно	Некоторые хостинги предлагают пакеты услуг, в которые включена регистрация вашего доменного имени. Вам лучше зарегистрироваться самостоятельно (см. **www.webnames.ru** или **www.r01.ru**), чтобы сохранить контроль над учетной записью своего доменного имени
Цена	_____ цена настройки веб-сервера _____ ежемесячная плата	Цена — последний пункт в этом списке, и не без причины. Не выбирайте хостинг для вашего сайта в зависимости от одной лишь цены — старая поговорка «за что платите, то и получите» здесь абсолютно правдива. Нет ничего необычного в единовременной оплате за настройку сервера для вашего сайта, и уже после этого — регулярной оплате — ежемесячной, ежеквартальной или ежегодной

Глава 11
МУЛЬТИМЕДИЙНЫЕ И ИНТЕРАКТИВНЫЕ ЭЛЕМЕНТЫ НА ВЕБ-СТРАНИЦАХ

Цели главы

В этой главе вы узнаете следующее:

- назначение плагинов, вспомогательных программ, медиаконтейнеров и кодеков;
- типы аудиофайлов, используемые во Всемирной паутине;
- как конфигурировать гиперссылки на мультимедийные файлы;
- как применить элемент `object` для отображения аудио- и видеофайлов;
- как применить элемент `object` для отображения на веб-странице Flash-роликов;
- как конфигурировать на веб-странице аудио и видео с помощью элементов HTML5;
- возможности и популярные способы использования роликов в формате Adobe Flash;
- как добавить Flash-анимацию на веб-страницу;
- возможности и популярные способы использования Java-апплетов;
- как добавить Java-апплеты на веб-страницу;
- как создать интерактивную галерею изображений с помощью CSS;
- как конфигурировать CSS3-свойства `transform` и `transition`;
- возможности и популярные способы использования JavaScript;
- назначение HTML5-элемента `canvas`;
- возможности и популярные способы использования DHTML;
- возможности и популярные способы использования Ajax;
- как найти ресурсы Flash-анимации, Java-апплеты, скрипты JavaScript и Ajax во Всемирной паутине.

Как гласит поговорка — лучше один раз увидеть, чем сто раз услышать. Вы уже знаете, что графические элементы помогают сделать веб-страницы по-настоящему неотразимыми. С другими типами мультимедийных файлов, такими как аудио и видео, мы познакомим вас в этой главе.

Правильное использование видео и звуков на вашей веб-странице сделают ее более интересной и информативной. Мы рассмотрим источники этих типов мультимедийных файлов, HTML-код, необходимый для их размещения на веб-странице, и предложим способы их использования.

Впервые вы столкнулись с интерактивными элементами в главе 6, когда использовали псевдоклассы CSS, чтобы гиперссылки реагировали на наведение на них указателя мыши. Вы расширите свои знания в области CSS, когда создадите интерактивную галерею изображений и изучите CSS3-свойства `transition` и `transform`.

Добавление определенной доли интерактивности на страницу может сделать ее привлекательной и притягательной для ваших посетителей.

Технологии, обычно использующиеся для добавления интерактивности веб-страницам, включают в себя Flash, Java-апплеты, JavaScript, DHTML и Ajax. Эта глава знакомит вас со всеми этими техниками, каждая из которых может стать отдельной темой для целой книги. По мере чтения и пробы приведенных примеров сосредоточьтесь на изучении свойств и возможностей каждой технологии, вместо того чтобы пытаться освоить детали.

11.1. Плагины, контейнеры и кодеки

Веб-браузеры спроектированы таким образом, чтобы отображать веб-страницы и изображения форматов GIF, JPG, PNG и т. д. Когда мультимедийный файл не относится к одному из этих типов, браузер ищет ***плагин*** или ***вспомогательное приложение***, способное отобразить файл такого типа.

Если он не может найти плагин или вспомогательное приложение (запускаемое в отдельном окне) на компьютере посетителя, веб-браузер предлагает посетителю сохранить файл на его компьютере.

К наиболее широко используемым плагинам относятся.

- **Adobe Flash Player**[1]. Плагин Flash Player отображает файлы формата *.swf*. Они содержат аудио, видео и анимацию, кроме того, являются интерактивными.
- **Adobe Shockwave Player**[2]. Плагин Shockwave Player отображает высококачественные мультимедийные файлы, созданные с помощью приложения Adobe Director.
- **Adobe Reader**[3]. Плагин Adobe Reader часто используется для обмена информацией, сохраненной в формате *.pdf*, такой как подготовленные к печати брошюры, документы и технические описания.
- **Java Runtime Environment**[4]. Плагин JRE используется, чтобы запускать приложения и апплеты, используя технологию Java.
- **Real Player**[5]. Плагин RealPlayer воспроизводит потоковое аудио, видео, анимацию и мультимедийные презентации во Всемирной паутине.
- **Windows Media Player**[6]. Плагин Windows Media воспроизводит потоковое аудио, видео, анимацию и мультимедийные презентации во Всемирной паутине.
- **Apple QuickTime**[7]. Плагин QuickTime отображает анимацию формата QuickTime, музыку, MIDI-файлы, аудио, видео, а также VR-панорамы и объекты прямо на странице.

Плагины и вспомогательные приложения, перечисленные выше, использовались во Всемирной паутине в течение многих лет. В HTML5 видео- и аудиоролики поддерживаются на уровне браузера, для них не нужны плагины. При работе с аудио- и видеороликами в HTML5 необходимо помнить о ***контейнере*** (который обозначается расширением файла) и ***кодеке*** (это алгоритм, используемый для сжатия информации).

Изучите табл. 11.1 и 11.2, в которых перечислены распространенные расширения мультимедийных файлов, указан тип файла-контейнера и предоставлено описание с информацией о кодеке (если он применим для HTML5).

---

[1] get.adobe.com/ru/flashplayer
[2] get.adobe.com/shockwave
[3] get.adobe.com/reader
[4] www.java.com/ru/download/manual.jsp
[5] real.com
[6] www.microsoft.com/windows/windowsmedia/download
[7] www.apple.com/ru/quicktime/download

Таблица 11.1. Распространенные типы аудиофайлов

Расширение	Контейнер	Описание
.wav	Wave	Был изначально разработан корпорацией Microsoft. Это стандарт для платформы Windows, но также поддерживается и компьютерами под управлением OS X
.aiff и *.aif*	Audio Interchange File Format	Популярный формат аудиофайлов на платформе OS X. Он также поддерживается и на платформе Windows
.mid	Musical Instrument Digital Interface — MIDI	Содержат инструкции для воссоздания музыкальных звуков, а не саму цифровую запись, ограниченное количество звуков, которые можно воспроизвести
.au	Sun UNIX Sound File	Более старый тип звукового файла, который, как правило, обладает худшим качеством звука, чем форматы новых типов
.mp3	MPEG-1 Audio Layer-3	Формат музыкальных файлов, ставший популярным благодаря кодеку MP3, который поддерживает два канала и использует улучшенный алгоритм сжатия
.ogg	Ogg Vorbis	Формат аудиофайлов с открытым исходным кодом, использующий кодек Vorbis
.m4a	MPEG 4 Audio	Этот MPEG-4 формат, предназначенный исключительно для хранения звука, использует кодек Advanced Audio Coding (AAC). Его поддерживают программы QuickTime, iTunes и мобильные устройства, такие как iPod и iPad

Таблица 11.2. Распространенные типы видеофайлов

Расширение	Контейнер	Описание
.3gp	3GPP Multimedia File	Кодек H.264 является стандартом для передачи мультимедиа по высокоскоростным беспроводным сетям 3-го поколения
.avi	Audio Video Interleaved	Изначально это был стандартный видеоформат для платформы Windows
.flv	Flash Video File	Flash-совместимый видеофайл поддерживает кодек H.264
.m4v и *.mp4*	MPEG-4	Кодеки MPEG-4 и H.264. Этот формат MPEG поддерживают программы QuickTime, iTunes и мобильные устройства, такие как iPod и iPad
.mov	QuickTime	Создан корпорацией Apple и изначально использовался на платформе Macintosh, также поддерживается операционной системой Windows
.mpg	MPEG	Разработан при поддержке экспертной группы по кинематографии (Motion Pictures Experts Group[1]). Этот формат поддерживается как платформой PC, так и платформой Mac
.ogv и *.ogg*	OGG	Формат видеофайлов с открытым исходным кодом, использующий кодек Theora[2]
.webm	WebM	Открытый формат мультимедийных файлов[3], разработанный при поддержке корпорации Google, использует видеокодек VP8 и аудиокодек Vorbis
.wmv	Windows Media Video	Это технология потокового видео, разработанная корпорацией Microsoft. Проигрыватель Windows Media Player поддерживает этот формат

[1] mpeg.chiariglione.org/mpeg
[2] www.theora.org
[3] www.webmproject.org

11.2. Начало работы с аудио- и видеофайлами

В процессе чтения этой главы вы изучите различные способы предоставления посетителям своего веб-сайта аудио и видеофайлов. К ним относятся предоставление гиперссылки, XHTML-решения с использованием элементов `object` и `param`, а также новые HTML5-элементы `audio` и `video`. Мы начнем с самого простого метода — создания гиперссылки.

Предоставление гиперссылки

Простейший способ предоставить посетителям вашей веб-страницы доступ к звуковому файлу — это создать на него гиперссылку. HTML-код, ведущий к звуковому файлу под названием *WDFpodcast.mp3*, выглядит так:

```
<a href="podcast.mp3" title="Подкаст; Эпизод
1">Эпизод 1</a>
```

Если посетитель вашего веб-сайта щелкнет по ссылке, плагин, который воспроизводит файлы формата *.mp3*, установленный на вашем компьютере (такой как QuickTime), отобразит встроенный проигрыватель в новом окне или вкладке браузера. После этого посетители вашей веб-страницы могут использовать плагин для прослушивания звука. Если посетитель щелкнет по ссылке правой кнопкой мыши, он сможет загрузить или сохранить мультимедийный файл.

Практическое задание 11.1

В этом практическом задании вы создадите копию веб-страницы, показанной на рис. 11.1, которая будет содержать элемент `h1` и гиперссылку на MP3-файл. Кроме того, на этой странице будет размещена гиперссылка для текстовой расшифровки файла для обеспечения доступности информации. Также рекомендуется указать на веб-странице тип файла (например MP3) и, по желанию, размер файла, к которому предоставляется доступ.

Скопируйте файлы *Podcast.mp3* и *podcast.txt* из папки *Примеры\Глава_11\starters* диска, прилагающегося к книге, и сохраните их в папку с именем *podcast*. Запустите Блокнот (Notepad) или другой текстовый редактор. Используйте в качестве образца файл *Приме-*

ры\Глава_02\template.html и создайте веб-страницу с заголовком «Подкаст», гиперссылкой к MP3-файлу и гиперссылкой к транскрипции подкаста.

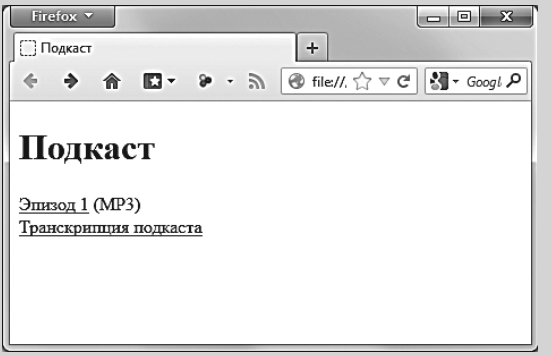


Рис. 11.1. Используемый по умолчанию MP3-проигрыватель запустится в браузере, когда посетитель щелкнет по ссылке **Эпизод 1**

Сохраните вашу веб-страницу с именем *podcast.html* и протестируйте в браузере. Попробуйте протестировать вашу страницу в разных браузерах и их версиях. Когда вы щелкните мышью по ссылке к MP3-файлу, аудиопроигрыватель (какой бы проигрыватель или плагин не был установлен для браузера по умолчанию) запустится для воспроизведения файла. Когда вы щелкните мышью по ссылке к расшифровке, в браузере отобразится текст. Сравните вашу работу с файлом *Примеры\Глава_11\podcast\podcast.html* на диске, прилагающемся к книге.

Элемент `object`

Еще одним способом добавить звук на вашу веб-страницу будет встроить в нее аудиофайл и, по желанию, разместить на ней панель управления звуком. При написании кода с использованием синтаксиса XHTML (см. главу 2) для этой цели применяется элемент `object`. В данном разделе используется HTML-код, однако элементы `object` и `param` также могут использоваться в HTML5.

***Элемент* `object`** является многоцелевым контейнерным тегом, используемым для добавления на страницу различных типов объектов. Он начинается с тега `<object>` и заканчивается тегом `</object>`. Дополнительные значения настроек, называемых параметрами, необходимо указывать в коде с помощью элемента `param`. Таблица 11.3 перечисляет стандартные атрибуты элемента `object`.

Таблица 11.3. Стандартные атрибуты элемента `object`

Атрибут	Значение	Использование

`classid`	Определяет программное обеспечение проигрывателя
	QuickTime:
	`classid="clsid:02BF25D5-8C17-4B23-BC80-D3488ABDDC6B"`
	Windows Player:
	`classid="clsid:6BF52A52-394A-11d3-B153-00C04F79FAA6"`
	Flash Shockwave Player:
	`classid="clsid:D27CDB6E-AE6D-11cf-96B8-44553540000"`
		Необязательный; используется в Windows. Атрибут `classid` определяет управляющий элемент ActiveX, который должен быть установлен на ПК посетителя

`codebase`	Указывает относительный путь к плагину
	QuickTime:
	`codebase="http://www.apple.com/qtactivex/qtplugin.cab"`
	Flash Shockwave Player:
	`codebase= "http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#Version=8,0,22,0 "`
		Указывает расположение плагина и загружает его по необходимости, не используется в Windows Media Player 7

`data`	Допустимое имя файла, имя аудиофайла	Обязательный; указывает имя воспроизводимого файла
`height`	Числовое значение в пикселах	Необязательный; указывает высоту панели управления мультимедийным файлом
`title`	Короткое текстовое описание	Необязательный; может отображаться браузером или вспомогательными технологиями
`type`	Допустимый тип MIME, такой как `audio/wav`, `audio/midi`, `audio/mpeg`, `video/quicktime` и т. д.	Необязательный; указывает MIME-тип мультимедийного файла, например, `audio/mpeg` или `video/quicktime`
`width`	Числовое значение в пикселах	Необязательный; указывает ширину панели управления мультимедийным файлом

Элемент `param`

Элемент **`param`** — это одиночный тег, не требующий закрывающей пары, имеющий два атрибута: `name` и `value`. Используйте разметку `<param />` в XHTML и `<param>` — в HTML5. Все элементы `param` следует помещать перед закрывающим тегом `</object>`. Документация проигрывателя подскажет, какие нужны параметры и какой формат вам следует использовать, при работе с мультимедийными файлами. Таблица 11.4 перечисляет значения атрибута элемента `param`.

Таблица 11.4. Значения атрибута элемента `param`

Имя параметра	Значение параметра	Использование
`autoplay`	`true`, `false`	Необязательный; определяет, будет ли мультимедийный файл воспроизводиться автоматически после загрузки страницы. Если не указывать это значение, файл не будет воспроизводиться автоматически
`controller`	`true`, `false` (поддерживается не всеми браузерами)	Необязательный; определяет, будет ли отображаться панель управления проигрывателем
`hidden`	`true` (поддерживается не всеми браузерами)	Необязательный; скрывает панель управления проигрывателя
`loop`	Числовое значение или `true`, для постоянного воспроизведения (поддерживается не всеми браузерами)	Необязательный; указывает, сколько раз мультимедийный файл следует воспроизвести повторно
`src`	Допустимое имя файла, имя аудиофайла	Обязательный; указывает имя воспроизводимого файла

Добавление аудиофайлов на веб-страницу

Так выглядит XHTML-код, позволяющий использовать тег `<object>`, чтобы встроить повторяющуюся звуковую дорожку в веб-страницу:

```
<object data="soundloop.mp3" height="50"
width="100" type="audio/mpeg" title="Музыкальный
цикл">
<param name="src" value="soundloop.mp3" />
<param name="controller" value="true" />
<param name="autoplay" value="false" />
</object>
```

Образец страницы, использующей элемент `object`, можно найти на диске, прилагающемся к книге, в файле *Примеры\Глава_11\musicbasic.html*. Взгляните на рис. 11.2, чтобы увидеть снимок этой страницы из браузеров Google Chrome (слева) и Internet Explorer (справа). Если вы видите предупреждение или сообщение об ошибке, проконсультируйтесь с вашим сетевым администратором или службой технической поддержки, чтобы узнать о рекомендованных параметрах безопасности и/или установке плагина.

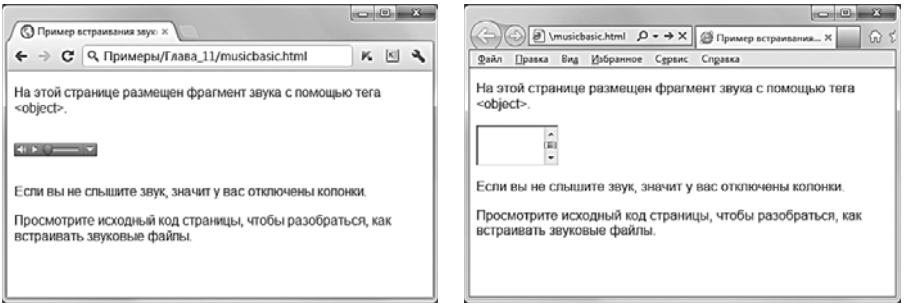


Рис. 11.2. Браузер Google Chrome (слева) правильно отображает мультимедийный объект, а браузер Internet Explorer (справа) — неправильно

Взгляните на рис. 11.2. Обратите внимание, что браузер Google Chrome (слева) правильно обрабатывает элемент `object` и отображает элементы управления для воспроизведения MP3-файла. Тем не менее, даже несмотря на то, что HTML-код правильный и соответствует рекомендациям Консорциума W3C, Internet Explorer не может правильно отобразить объект. Не волнуйтесь, для этой проблемы существует решение — сконфигурируйте еще один элемент `object`, который сможет обработать только Internet Explorer. Internet Explorer требует наличие атрибута `classid` (чтобы отобразить элементы управления ActiveX проигрывателя) и ассоциированного с ним атрибута `codebase`, чтобы правильно обработать элемент `object`, сконфигурированный для аудио- или видеофайлов. Код, воспроизводящий аудиофайл с помощью проигрывателя QuickTime в браузере Internet Explorer, показан ниже:

```
<object data="soundloop.mp3" height="50"
width="100" type="audio/mpeg"
classid="clsid:02BF25D5-8C17-4B23-BC80-
D3488ABDDC6B" codebase="http://www.apple.com/
qtactivex/qtplugin.cab">
<param name="src" value="soundloop.mp3" />
<param name="controller" value="true" />
<param name="autoplay" value="false" />
</object>
```

Описываемый прием использует *оба* элемента `object` вместе с условными комментариями (которые обрабатывает только Internet Explorer), направляющими остальные браузеры к стандартному коду. Это решение (см на диске, прилагающемся к книге, файл *Примеры\Глава_11\music.html*) приведено ниже:

```
<object data="soundloop.mp3" height="50"
width="100"
type="audio/mpeg" classid="clsid:02BF25D5-8C17-
4B23-BC80-D3488ABDDC6B" codebase="http://www.apple.
com/qtactivex/qtplugin.cab">
<param name="src" value="soundloop.mp3" />
<param name="controller" value="true" />
<param name="autoplay" value="false" />
<!--[if !IE]>-->
<object data="soundloop.mp3" height="50"
width="100" type="audio/mpeg">
<param name="src" value="soundloop.mp3" />
<param name="controller" value="true" />
<param name="autoplay" value="false" />
</object>
<!--<![endif]-->
</object>
```

Браузеры обрабатывают код сверху вниз, строка за строкой. Только браузер Internet Explorer обрабатывает условные комментарии. В этом случае условный комментарий будет означать, что Internet Explorer должен игнорировать код внутри комментария. Условный комментарий начинается со строки `<!--[if !IE]>-->` и заканчивается строкой `<!--<![endif]-->`. Таким образом, Internet Explorer обработает первый элемент `object` и пропустит второй, тогда как другие браузеры обработают их один за другим (в той же области окна браузера) и реализуют запрос, используя код из второго элемента `object`. Если вы думаете что все это довольно сложно — так и есть! Жизнь была бы намного проще, если бы все браузеры работали по более простым схемам. Вы немного наберетесь опыта использования элемента `object` в следующем практическом задании.

Практическое задание 11.2

В этом практическом задании вы создадите веб-страницу, которая будет отображать элементы управления воспроизведением звука (рис. 11.3). Скопируйте звуковой файл *music1.mp3* или *music2.mp3* из папки *Примеры\Глава_11* диска, прилагающегося к книге, и сохраните его в папку *music*. Запустите Блокнот (Notepad) или другой

текстовый редактор, и откройте шаблон XHTML-веб-страницы на прилагающемся к книге диске (*Примеры\Глава_02\templatex.html*).

1. Задайте подходящий заголовок веб-страницы.
2. Добавьте в код элемент `h1` с текстом «Пример встраивания звуков».
3. Используйте теги `<object>` и `<param />` для отображения элементов управления, которые позволят посетителю веб-страницы управлять аудиофайлом. Используйте в помощь код из примера выше и список атрибутов и значений из табл. 11.3 и 11.4. Сохраните веб-страницу под именем *objectaudio.html* в папке *music* и протестируйте ее в браузере.
4. Поэкспериментируйте с атрибутом `width` элемента `object` — попробуйте значения 25, 50, 100 и 110 и посмотрите, как изменится вид панели с элементами управления. Протестируйте значение `autoplay` элемента `param`, чтобы разобраться, как устанавливать автоматическое воспроизведение звука при загрузке страницы. Используйте значение `loop` элемента `param`, чтобы зациклить воспроизведение звука. Протестируйте веб-страницу в различных браузерах и их версиях. Сравните вашу работу с файлом *Примеры\Глава_11\object.html* на диске, прилагающемся к книге.

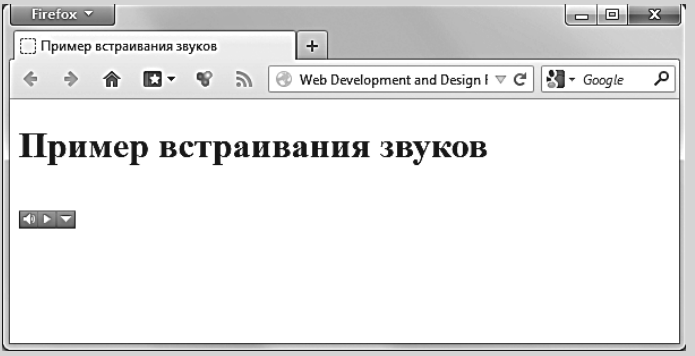


Рис. 11.3. На этой странице используется элемент `object` с атрибутом `width="50"`

ЧаВо

КАК СОЗДАТЬ ПОДКАСТ?

Подкасты — это аудиофайлы во Всемирной паутине, имеющие вид аудиоблога, радиошоу или интервью. Существуют три шага в публикации подкаста:

1. **Записать подкаст.** В операционных системах Windows и OS X имеются программы для записи аудио. Программа Apple Quick-

Time Pro[1] (доступна для операционных систем Windows и OS X) — это недорогое приложение, которое можно использовать для записи аудио. Если вы работаете в операционной системе OS X, еще одним вариантом может стать предустановленное музыкальное приложение GarageBand, предлагающее целый ряд вариантов записи и редактирования звука. Audacity[2] — бесплатный кроссплатформенный цифровой аудиоредактор (для операционных систем Windows и OS X). С помощью Audacity можно записать свой голос для подкастов и создавать музыкальные миксы и циклы, чтобы сделать подкаст интереснее. Созданный WAV-файл можно преобразовать в формат MP3 с помощью кодировщика LAME[3] или другого аналогичного приложения.

2. **Загрузить подкаст.** Загрузите MP3-файл на ваш веб-сайт. Если ваш веб-хост не разрешает использование MP3-файлов, альтернативным способом может стать размещение файла на сайте, который разрешает бесплатно использовать MP3, например, интернет-архив[4] или OurMedia[5].

3. **Сделать подкаст доступным.** Самым простым способом будет поместить код гиперссылки к аудиофайлу на веб-страницу. Гиперссылка позволит посетителям веб-сайта получить доступ к MP3-файлам подкаста, но не сделает подкаст доступным для подписки. Необходимо создать RSS-ленту, чтобы ваши посетители могли подписаться на ваши текущие и будущие подкасты. ***RSS-лента*** подкаста — это XML-файл, который перечисляет информацию о ваших подкастах. С помощью небольшой толики терпения вы можете создать свою RSS-ленту, используя текстовый редактор[6,7]. Кроме того, ряд веб-сайтов, к примеру, FeedBurner[8] и IceRocket[9], предоставляет сервис, который создает и поддерживает RSS-ленты пользователей. После того как RSS создана и загружена во Всемирную паутину (созданная лично вами или же специальным сервисом), укажите в коде страницы ссылку на файл. Компания Apple предоставляет инструкции для подписки вашего подкаста на iTunes на своем веб-сайте[10]. Посетители, использующие такое программное обеспечение, как iTunes компании Apple или бесплатный сервис для чтения RSS-лент[11], могут найти и автоматически загрузить ваш подкаст.

[1] apple.com/quicktime/pro
[2] audacity.sourceforge.net
[3] lame.sourceforge.net
[4] www.archive.org
[5] ourmedia.org
[6] www.downes.ca/cgi-bin/page.cgi?post=56
[7] www.masternewmedia.org/news/2006/03/09/how_to_create_a_rss.htm
[8] feedburner.google.com
[9] rss.icerocket.com
[10] apple.com/itunes/whatson/podcasts/specs.html
[11] rssreader.ru

Добавление видеофайлов на веб-страницу

Процесс добавления видеофайлов на веб-страницу с помощью элементов `object` и `param` мало чем отличается от добавления аудиофайлов. Просмотрите предыдущий раздел, чтобы вспомнить эти элементы. В данном разделе в коде используется синтаксис XHTML. Помните, что элементы `object` и `param` также могут применяться и в HTML5, несмотря на то, что в HTML5 вводятся новые элементы `audio` и `video`, которые вы изучите в следующем разделе. Страницу-пример, использующую элемент `object` для отображения видео, можно найти на диске, прилагающемся к книге, в файле *Примеры\Глава_11\video.html*, кроме того, она показана на рис. 11.4.

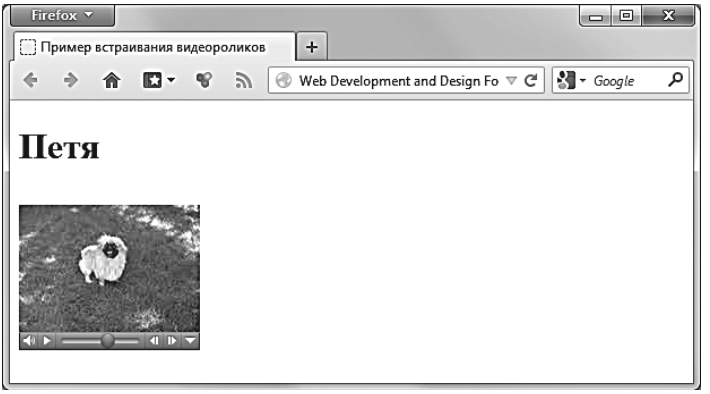


Рис. 11.4. Воспроизведение видеофайла

Страница была создана согласно технике[1] по созданию двух элементов `object` и использовании условных комментариев Internet Explorer. Код, воспроизводящий видеофайл *sparky.mov*, приведен ниже:

```
<object data="sparky.mov" height="150" width="160"
type="video/quicktime" classid="clsid:02BF25D5вЂ"8C
17вЂ"4B23-BC80-D3488ABDDC6B" codebase="http://www.
apple.com/qtactivex/qtplugin.cab" title="Видеоролик
про гавкающую собаку">
<param name="src" value="sparky.mov" />
<param name="controller" value="true" />
<param name="autoplay" value="false" />
```

[1] www.alistapart.com/articles/byebyeembed

```
<!--[if !IE]>-->
<object data="sparky.mov" height="150" width="160"
type="video/quicktime" title="Видеоролик про
гавкающую собаку">
<param name="src" value="sparky.mov" />
<param name="controller" value="true" />
<param name="autoplay" value="false" />
<p>Видеоролик про гавкающую собаку.</p>
</object>
<!--<![endif]-->
</object>
```

В зависимости от плагинов вашего браузера, видео может не отобразится на этой странице, если использовать элемент `object`. Страницы-примеры были протестированы при использовании плагина QuickTime для файлов с расширением *.mov*. Вопрос наличия плагина может стать проблемой для видеокомпонентов веб-сайта. Его тестирование при учете возможностей вашей целевой аудитории, а также подсказки по установке и использованию самых необходимых плагинов для вашей аудитории будут весьма полезны.

Что случится если браузер или другая программа пользователя не сможет отобразить видео? Аккуратно проверьте код и убедитесь, что перед двумя закрывающими тегами `</object>` указано описание. Эта фраза отобразится на веб-странице, если объект (в нашем случае видеопроигрыватель) невозможно будет обработать. Кроме того, для обеспечения доступности атрибуту `title` присвоили короткое текстовое описание видеоролика. Это описание можно будет прочитать некоторыми вспомогательными устройствами, такими как программы экранного доступа.

Практическое задание 11.3

В этом практическом задании вы создадите веб-страницу, показанную на рис. 11.5 и использующую элементы `object` и `param` для воспроизведения видеоклипа. Скопируйте файл *lighthouse.mov* из папки *Примеры\Глава_11* диска, прилагающегося к книге, и сохраните его в папку *movie* на ваш компьютер. Запустите Блокнот (Notepad) или другой текстовый редактор и откройте шаблон XHTML-веб-страницы на прилагающемся к книге диске (*Примеры\Глава_02\templatex.html*).

1. Задайте подходящий заголовок веб-страницы.
2. Добавьте в код элемент `h1` с текстом «Круиз по графству До».
3. Используйте теги `<object>` и `<param />` для отображения элементов управления, которые позволят посетителю веб-страницы управлять видеофайлом. Используйте в помощь код из примера выше и список атрибутов и значений из табл. 11.3 и 11.4. Сохраните веб-страницу под именем *objectvideo.html* в папке *movie* и протестируйте ее в браузере.
4. Поэкспериментируйте с атрибутами `height` и `width` элемента `object`. Примените атрибуты `autoplay` и `loop` элемента `param`. Протестируйте веб-страницу в разных браузерах и их различных версиях. Сравните вашу работу с файлом *Примеры\Глава_11\lighthouse.html* на диске, прилагающемся к книге.

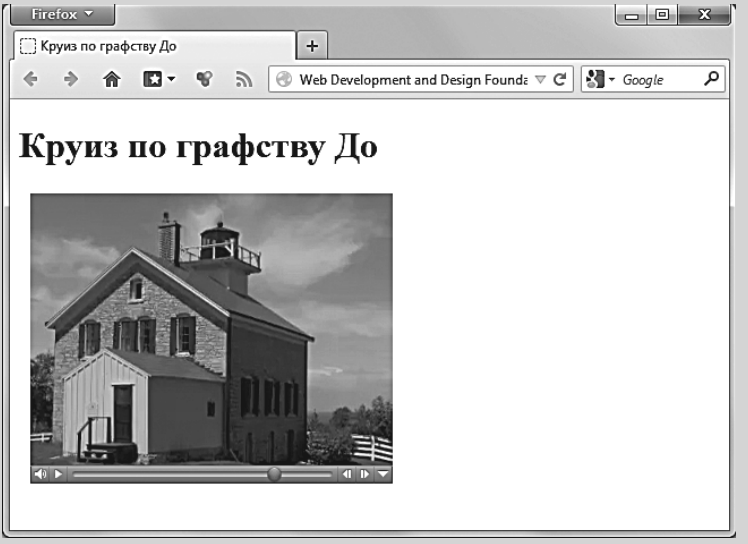


Рис. 11.5. Использование элемента `object` для встраивания видео

Мультимедийные файлы во Всемирной паутине

Дополнительные сведения об аудиофайлах

Существует несколько путей, которыми вы можете получить аудиофайлы. Вы можете записать собственные звуки, скачать сэмплы и музыку с бесплатных сайтов, записать музыкальные композиции с CD-диска или приобрести сборник звуков на диске. Существуют этические вопросы, касающиеся использования звуков и музыки, созданных другими людьми. Вы можете опубликовать только те звуки или музыку, которые

вы создали самостоятельно, или те, на которые вы приобрели права на опубликование (иногда называют лицензией).

Звуки и музыку с купленного в магазине CD-диска можно копировать только для личного использования, но не для публикации во Всемирной паутине. Свяжитесь с обладателем авторских прав, чтобы получить разрешение на использование музыки.

Во Всемирной паутине существует множество ресурсов — источников аудиофайлов. Некоторые из них предлагают бесплатные файлы, такие как сайт Clip Art and Media[1] корпорации Microsoft, сайт Loopasonic[2], сайт FreeAudioClips.com[3]. Другие же, такие как SoundRangers[4], хоть и могут предложить некоторое количество бесплатных звуков, в основном занимаются продажей звуковых дорожек и CD-дисков. Интересный ресурс бесплатных звуков можно найти по адресу **www.flashkit.com**; щелкните по ссылке **Sound Loops** в меню навигации, которое находится в правом верхнем углу сайта. Хотя этот сайт предназначен для разработчиков Adobe Flash, звуки можно использовать и без технологии Flash.

Аудиофайлы могут быть довольно большими, поэтому важно помнить о времени, необходимом на их загрузку для воспроизведения. Если вы решите использовать аудиофайл на веб-странице, постарайтесь максимально уменьшить его размер. Если вы записываете собственные аудиофайлы, помните, что частота дискретизации и разрядность повлияют на размер файла. ***Частота дискретизации*** — это величина, относящаяся к количеству сэмплов цифровых звуков на одну секунду при записи. Она измеряется в Килогерцах (кГц). Наиболее часто используемые частоты дискретизации варьируются от 8 кГц (качество звука радио коротковолновой частоты и звуковых эффектов) до 44.1 кГц (качество звука музыкального CD-диска). Как можно догадаться, звук, записанный с частотой 44.1 кГц, будет занимать гораздо больше дискового пространства, чем звук, записанный с частотой 8 кГц. Битовая глубина или ***разрядность*** — это еще один фактор, влияющий на размер аудиофайла. Файл, записанный с 8-битной разрядностью (подходящей для голоса или других простых звуков) будет занимать меньше дискового пространства, чем файл, записанный с 16-битной разрядностью (качество музыкального CD-диска).

[1] office.microsoft.com/ru-ru/images/
[2] www.loopasonic.com
[3] www.freeaudioclips.com
[4] www.soundrangers.com

Дополнительная информация о видеофайлах

Как и с аудиофайлами, есть несколько способов получить видеофайлы, включая запись вашего собственного видео, его загрузки из Всемирной паутины, приобретение CD, который содержит видео, или поиска видеофайлов во Всемирной паутине. Существуют этические вопросы, касающиеся использования видео, созданного другими людьми. Прежде чем опубликовать на своем сайте видео, созданное другими людьми, вы должны приобрести права или лицензию на его опубликование.

Многие цифровые фотокамеры обладают возможностью снимать как фотографии, так и короткие видеоролики. Это неплохой способ получить короткий видеоклип. Цифровые видеокамеры и веб-камеры записывают цифровое видео. Когда вы создадите видео, для монтажа вашего шедевра вам понадобится специальное приложение, такое как Adobe Premiere[1], Apple QuickTime Pro[2], Apple iMovie[3] или Киностудия Windows Live[4]. Многие цифровые фотокамеры и смартфоны позволяют записывать видеоролики и немедленно загружать их на сервис YouTube. С сервисом YouTube вы будете работать в главе 13.

Мультимедийные файлы и доступность

Предоставьте альтернативное содержимое для мультимедийных файлов, которые вы используете на вашем веб-сайте в виде транскрипции, субтитров или в формате PDF. Обеспечьте транскрипции для аудиофайлов, таких как подкасты. Часто за основу транскрипций можно взять сценарий подкаста, затем создать файл в формате PDF и загрузить на веб-сайт. Добавьте субтитры к видеофайлам. Такие приложения, как Media Access Generator (MAGpie)[5], позволяют добавлять титры к видео. Приложение QuickTime Pro компании Apple включает в себя возможности субтитрования — посмотрите субтитрованную версию видеоролика *Примеры\Глава_11\starters\sparkycaptioned.mov* на диске, прилагающемся к книге.

Когда вы загружаете видео на сервис YouTube, вы также можете добавить титры и субтитры[6]. Чтобы узнать подробнее о создании субтитров к видео, посетите сайт WebAIM[7].

[1] www.adobe.com/ru/products/premiere/
[2] www.apple.com/ru/quicktime/extending/
[3] www.apple.com/ru/ilife/imovie/
[4] windows.microsoft.com/ru-RU/windows-live/movie-maker-get-started
[5] ncam.wgbh.org/invent_build/web_multimedia/tools-guidelines/magpie
[6] www.google.com/support/youtube/bin/answer.py?answer=100077
[7] webaim.org/techniques/captions

Проблемы совместимости браузеров

При выполнении практического задания вы, возможно, сталкивались с проблемами воспроизведения видео в различных браузерах. Возможность воспроизведения аудио- и видеофайлов во Всемирной паутине зависит от плагинов, установленных в веб-браузере посетителя. Страница, которая отлично функционирует на вашем домашнем компьютере, может работать не у всех посетителей, и это зависит от конфигурации компьютера. У некоторых посетителей могут быть неправильно установлены плагины. У других типы файлов могут быть связаны с неверными или неверно установленными плагинами. У третьих может оказаться низкая пропускная способность и придется ждать загрузки вашего мультимедийного файла слишком долго. Замечаете закономерность? Иногда размещение мультимедийных файлов во Всемирной паутине может оказаться проблематичным.

В ответ на проблемы совместимости плагинов браузеров и чтобы уменьшить зависимость от собственной технологии, такой как Adobe Flash, HTML5 вводит новые элементы `audio` и `video`. Однако так как HTML5 пока не поддерживается распространенными версиями браузеров (в частности, старыми версиями Internet Explorer), веб-дизайнерам по-прежнему необходимо обеспечить запасной вариант, например, предоставление гиперссылки на мультимедийный файл и отображения Flash-версии мультимедийного файла. Вы будете работать с элементами HTML5 `audio` и `video` далее в этой главе, но сначала давайте рассмотрим Adobe Flash.

11.4. Технология Adobe Flash

Flash — это популярное мультимедийное приложение, часто использующееся для создания анимационных и мультимедийных эффектов на веб-страницах. Flash-анимация хранится в файле с расширением *.swf*. В отличие от других мультимедийных типов, файлы *.swf* воспроизводятся в процессе загрузки, поэтому дают ощущение быстрого отображения сложной графической анимации. Анимация может быть такой же простой, как эффект Flash, изображенный на рис. 11.6 (см. файл *Примеры\Глава_11\flash1.html* на диске, прилагающемся к книге). Технологию Flash также можно использовать для воспроизведения аудио- и видеофайлов и создания гораздо более сложных эффектов, включая полноэкранную анимацию, рекламные баннеры и интерактивную навигацию сайта, использующую встроенные аудиоклипы.

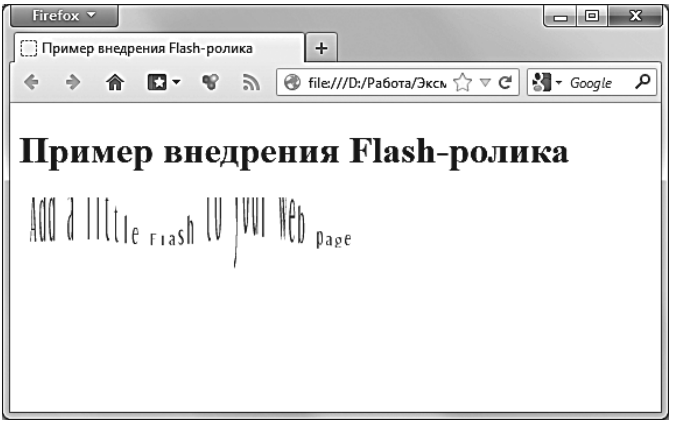


Рис. 11.6. Пример веб-страницы с Flash-роликом

Файл Flash-анимации может быть интерактивным; для него можно написать скрипт на языке ActionScript, чтобы файл реагировал на щелчки мыши, принимал информацию в текстовые поля, запускал CGI или другие скрипты, выполняющиеся на стороне сервера.

Технология Flash требует наличия плагина в браузере, который бесплатен и доступен для загрузки с сайта корпорации Adobe. По информации корпорации Adobe плагин Flash установлен на 99% настольных компьютеров, имеющих подключение к Интернету[1]. Вспомните, что воспроизведение аудио- и видеофайлов стандартных форматов на веб-странице крайне зависимо от плагинов для браузеров, установленных на компьютерах ее посетителей. Технологию Adobe Flash часто использовали для воспроизведения видео- и аудиофайлов на веб-страницах, пока не появились устройства iPhone, iPod и iPad корпорации Apple без поддержки Flash. Однако Flash поддерживается на других мобильных устройствах, таких как планшеты и смартфоны на платформе Android и Blackberry.

Технологию Flash можно использовать для создания интерактивной рекламы на веб-страницах, которая сможет реагировать на движения мыши пользователя со звуком и анимацией. Результаты исследований сайта DoubleClick о ценности мультимедийной рекламы показали[2], что мультимедийная реклама (такая как видеоролики, основанные на технологии Flash) увеличивает узнавание бренда, его популярность, и чаще вызывает

[1] adobe.com/products/player_census/flashplayer

[2] static.googleusercontent.com/external_content/untrusted_dlcp/www.google.com/en/us/doubleclick/pdfs/DoubleClick-06-2009-The-Brand-Value-of-Rich-Media-and-Video-Ads.pdf

у покупателей намерение совершить покупку, в отличие от других типов рекламы, таких как статическое изображение или простые Flash-баннеры.

Технологию Flash можно использовать и для создания целых веб-сайтов, включая навигацию, содержимое и формы. Прекрасным примером может стать сайт 2Advanced Studios[1]. Вся интерактивность сайта — навигация, анимация и содержимое — создана с помощью технологии Flash в файле с расширением *.swf*. Корпорация Adobe лицензирует файловый формат Flash для сторонних разработчиков. Это значит, что вы можете использовать и другие приложения, а не только Adobe Flash для создания эффектов технологии Flash (*.swf*). Приложение Camtasia[2] компании TechSmith и приложение Swish[3] — это лишь два примера приложений, которые можно использовать для создания мультимедийных проектов в формате *.swf*.

Сегодня веб-разработчик должен знать, как добавлять Flash-файлы формата *.swf* на веб-страницы. Если вы работаете над большим проектом, графический дизайнер может создать эффект и передать его вам для размещения на странице. Если вы работаете над маленьким проектом, вам, возможно, придется создать Flash-файл самостоятельно. Корпорация Adobe предлагает бесплатно загрузить пробную версию приложения Flash, включающую несколько руководств и уроков по использованию технологии Flash.

Добавление Flash-анимации на веб-страницу

Сегодня все современные браузеры поддерживают отображение мультимедийных файлов Flash с помощью элементов `object` и `param`. Как обсуждалось ранее, элемент `object` является многоцелевым тегом, служащим для добавления различных типов объектов на веб-страницу. Атрибуты элемента `object` различаются в зависимости от типа используемого объекта. Минимальный набор атрибутов, необходимый при работе с Flash-файлами, описан в табл. 11.5.

Flash-объект использует специальные значения, которые называют параметрами, для настройки имени файла *.swf*, качества изображения и цвета фона соответствующих областей страницы. Они настраиваются с помощью тегов `<param>`. Параметры, использующиеся для Flash-файлов, показаны в табл. 11.6.

[1] 2advanced.com
[2] www.techsmith.com
[3] www.swishzone.com

Таблица 11.5. Минимальный набор атрибутов для Flash-файлов

Атрибут	Описание и значение
`accesskey`	Необязательный; определяет «горячую» клавишу для доступа с клавиатуры. Пользователям операционной системы Windows следует нажимать клавишу **Crtl** и назначенную клавишу одновременно
`data`	Имя Flash-файла (файла с расширением *.swf*)
`height`	Указывает высоту области объекта в пикселах
`tabindex`	Необязательный; числовое значение, которое указывает порядок табуляции во Flash-файле
`type`	MIME-тип объекта. Используйте значение `type="application/x-shockwave-flash"`
`title`	Необязательный; указывает короткое текстовое описание, которое может быть отображено браузером или вспомогательными технологиями
`width`	Указывает ширину области объекта в пикселах

Таблица 11.6. Параметры Flash-файлов

Параметр	Значение параметра
`bgcolor`	Необязательный; фоновый цвет областей Flash-файлов; допустимо шестнадцатеричное цветовое значение
`loop`	Необязательный; указывает, повторяется ли воспроизведение файла *.swf*; допустимые значения — `"true"` и `"false"`
`movie`	Имя Flash-файла (файл формата *.swf*)
`quality`	Необязательный; описывает качество мультимедийного файла; обычно используется значение `high`
`wmode`	Необязательный; настраивает прозрачный фон Flash-файла в браузерах, которые поддерживают это свойство; допустимое значение — `"transparent"`

Следующий HTML-код конфигурирует SWF-файл, показанный на рис. 11.6:

```
<object type="application/x-shockwave-flash"
data="flashlogo.swf" width="300" height="70"
title="Пример внедрения Flash-ролика">
<param name="movie" value="flashlogo.swf">
<param name="bgcolor" value="#ffffff">
<param name="quality" value="high">
<p>Украсьте свой сайт</p>
</object>
```

Обратите внимание на код, размещенный перед закрывающим тегом `</object>` в приведенном выше примере. Он отобразится, если браузер

не поддерживает мультимедийный объект. По необходимости добавьте ссылку на веб-страницу, содержащую альтернативное текстовое содержимое. Хотя разработчики вспомогательных технологий, таких как программы экранного доступа, работают над поддержкой Flash-файлов, пока что технология Flash все еще не универсальна.

Практическое задание 11.4

В этом практическом задании вы создадите веб-страницу, которая будет отображать слайд-шоу фотографий на основе технологии Flash. Ваша страница будет выглядеть также, как показанная на рис. 11.7.

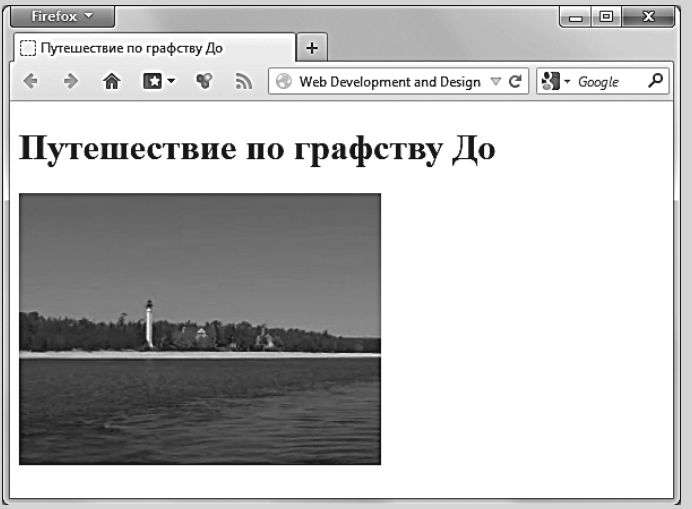


Рис. 11.7. Слайд-шоу изображений с использованием технологии Flash

Приступим. Создайте на вашем компьютере папку *lighthouse*. Скопируйте файл *lighthouse.swf* из папки *Примеры\Глава_11\starters* диска, прилагающегося к книге, и сохраните его в вашу папку *lighthouse*. Используйте в качестве отправной точки файл *Примеры\Глава_02\template.html* и создайте веб-страницу с заголовком «Путешествие по графству До», а также соответствующие элементы `object` и `param` для отображения Flash-файла *lighthouse.swf,* ширина которого 320 пикселов, а высота 240 пикселов.

HTML-код следующий:

```
<object type="application/x-shockwave-flash"
data="lighthouse.swf" width="320" height="240"
title="Путешествие по графству До">
```

```
<param name="movie" value="lighthouse.swf">
<param name="bgcolor" value="#ffffff">
<param name="quality" value="high">
<p>Путешествие по графству До</p>
</object>
```

Сохраните вашу страницу как *lighthouse2.html* и протестируйте ее в браузере. Сравните вашу работу с файлом *Примеры\Глава_11\lighthouse\lighthouse.html* на диске, прилагающемся к книге.

Элемент `embed`

Консорциум W3C официально не рекомендовал элемент `embed` до появления HTML5, несмотря на то, что данный элемент использовался в течение многих лет для настройки мультимедийных файлов и Flash-файлов на веб-страницах. Одним из принципов построения HTML5 является «сглаживание» процесса использования методов, которые, хотя и поддерживаются браузерами, но не являются частью официального стандарта W3C. На рис. 11.8 (см. *Примеры\Глава_11\flashembed.html* на прилагающемся к книге диске) показана веб-страница, на которой используется элемент `embed` для отображения файлов Flash в формате SWF.

Рис. 11.8. Элемент embed используется для размещения Flash-анимации

`embed` — контейнерный элемент, который дает возможность добавлять на веб-страницу контент, для воспроизведения которого необходим плагин или проигрыватель. Элемент `embed` можно использовать для отображения файлов SWF на веб-странице. Его атрибуты, обычно используемые с мультимедийными файлами Flash, приведены в табл. 11.7.

Таблица 11.7. Атрибуты элемента `embed`

Атрибут	Описание и значение
`bgcolor`	Необязательный; определяет фоновый цвет областей Flash-файлов; использует шестнадцатеричное цветовое значение
`height`	Указывает высоту области объекта в пикселах
`quality`	Необязательный; описывает качество мультимедийного файла; обычно используется значение `high`
`scr`	Имя файла Flash (файл SWF)
`title`	Необязательный; указывает короткое текстовое описание, которое может быть отображено браузером или вспомогательными технологиями
`type`	MIME-тип объекта. Используйте значение `type="application/x-shockwave-flash"`
`width`	Указывает ширину области объекта в пикселах
`wmode`	Необязательный; настраивает прозрачный фон Flash-файла в браузерах, которые поддерживают это свойство; допустимое значение — `"transparent"`

Следующий HTML-код конфигурирует файл SWF, показанный на рис. 11.7:

```
<embed type="application/x-shockwave-flash"
src="fall5.swf" width="640" height="100"
quality="high" title="Золотая осень">
```

Обратите внимание на значение атрибута `title` в приведенном выше коде. Данный текст может быть доступен вспомогательным технологиям, таким, как программы экранного доступа.

Практическое задание 11.5

В этом практическом задании вы запустите программу Блокнот (Notepad) и создадите веб-страницу, которая будет отображать слайд-шоу фотографий на основе технологии Flash. Ваша страница будет выглядеть также, как показанная на рис. 11.9.

Создайте папку и назовите ее *embed*. Скопируйте файл *lighthouse.swf* из папки *Примеры\Глава_11* диска, прилагающегося к книге, и сохраните его в вашу папку *embed*. Используйте в качестве отправной точки файл *Примеры\Глава_02\template.html* и создайте веб-страницу с заголовком «Путешествие по графству До» и элементом `embed` для отображения Flash-файла *lighthouse.swf,* ширина которого 320 пикселов, а высота 240 пикселов. HTML-код следующий:

```
<embed type="application/x-shockwave-flash"
```

```
src="lighthouse.swf"
width="320" height="240"
title="Путешествие по графству До">
```

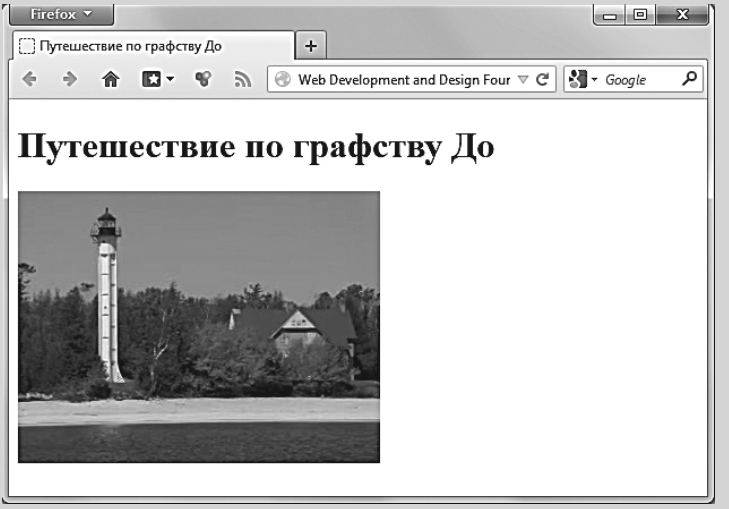


Рис. 11.9. Слайд-шоу изображений с использованием технологии Flash и элемента `embed`

Сохраните вашу страницу с именем *index.html* в папку *embed* и протестируйте ее в браузере. Сравните вашу работу с файлом *Примеры\Глава_11\lighthouse\embed.html* на диске, прилагающемся к книге.

Ресурсы с материалами по технологии Flash

Существует множество источников бесплатных Flash-анимаций и обучающих пособий по этой технологии во Всемирной паутине. В дополнение к ресурсам на сайте корпорации Adobe[1] следующие сайты содержат обучающие пособия и новости о технологии Flash:

- Flash Kit: **flashkit.com**
- ActionScript: **www.actionscript.org**
- ScriptOcean: **www.scriptocean.com/flashn.html**
- Kirupa: **www.kirupa.com/developer/flash/index.htm**

Во время посещения этих и других ресурсов, посвященных технологии Flash, помните, что некоторые мультимедийные материалы защи-

[1] adobe.com/ru/

щены авторским правом. Получите разрешение от создателя мультимедийных файлов, прежде чем использовать их на своем сайте, и следуйте всем инструкциям об указании источника такого материала. Некоторые сайты позволяют личное использование их Flash-файлов в некоммерческих целях, но требуют приобретения лицензии для коммерческого использования.

Корпорация Adobe стремится повысить доступность объектов Flash. Последние версии Flash совместимы со вспомогательными технологиями, такими как программа экранного доступа Window-Eyes, и дают возможность более широкой аудитории воспринимать многофункциональный контент веб-страниц. Flash поддерживает компоненты Microsoft Active Accessibility (MSAA), таким образом, не только предоставляя стандартный метод взаимодействия программ клиентов со вспомогательными технологиями, но и технику, которую могут использовать веб-разработчики, чтобы создаваемое ими в соответствии с этим стандартом клиентское программное обеспечение могло включать в себя поддержку Adobe Flash. Посетите веб-сайт компании Adobe[1], чтобы узнать новейшую информацию по вопросу доступности Flash.

ЧаВо

ЧТО ТАКОЕ MICROSOFT SILVERLIGHT?

Silverlight[2] — это плагин для передачи мультимедийных данных и многофункциональных веб-приложений во Всемирной паутине. Microsoft Expression Blend — это приложение, которое создает интерактивные мультимедийные объекты, отображаемые плагином Silverlight.

ЧаВо

ЧТО СЛУЧИТСЯ, ЕСЛИ БРАУЗЕР ПОСЕТИТЕЛЯ МОЕЙ ВЕБ-СТРАНИЦЫ НЕ ПОДДЕРЖИВАЕТ ТЕХНОЛОГИЮ FLASH?

Если вы использовали код из этого раздела для отображения Flash-файла на веб-странице, а браузер посетителя не поддерживает технологию Flash, в большинстве случаев браузер отобразит сообщение о необходимости установки недостающего плагина. HTML-код в этом разделе проходит валидацию на сайте Консорциума W3C и является минимальным кодом, необходимым для отображения Flash-файла на веб-странице. Если вы хотите добавить больше возможностей, на-

[1] www.adobe.com/accessibility/products/flash
[2] www.silverlight.net

пример, предложить быструю установку последней версии плагина Flash Player, вам следует узнать больше о методе *SWFObject*[1], который использует скрипт JavaScript для встраивания Flash-содержимого и совместим с HTML-стандартами Консорциума W3C.

На момент написания книги не существовало поддержки Flash для iPhone, iPhone Touch или iPad, однако эти устройства поддерживают новые HTML5-элементы `audio` и `video`, с которыми вы познакомитесь в следующем разделе.

11.5. Элементы HTML5 для добавления аудио- и видеофайлов

Новые HTML5-элементы `audio` и `video` позволяют браузерам воспроизводить мультимедийные файлы без плагинов. При работе с элементами `audio` и `video` важно учитывать ***контейнер*** (на него указывает расширение файла) и ***кодек*** (это алгоритм, используемый для сжатия информации). Обратитесь к табл. 11.1 и 11.2, содержащим перечень распространенных расширений мультимедийных файлов, типы файлов-контейнеров, а также описание с информацией о кодеке (если это применимо для HTML5). В идеальном мире все браузеры поддерживали бы все мультимедийные кодеки. В реальном это не так. Например, требуется платное лицензирование кодека H.264, и он не поддерживается браузерами Firefox и Опера, которые поддерживают кодеки OGG и WebM. Браузеры Internet Explorer и Safari поддерживают видео с кодеком H.264, но не поддерживают OGG и WebM. На момент написания книги браузер Chrome поддерживал кодеки H.264, OGG и WebM, но планирует отказаться от поддержки H.264 в будущем[2].

При работе с элементами `audio` и `video` вам необходимо подключать несколько версий каждого мультимедийного файла в различных контейнерах/кодеках. Давайте начнем с элемента `audio`.

Элемент `audio`

Новый HTML5-***элемент*** **`audio`** поддерживает воспроизведение аудиофайлов в браузере без установки дополнительных плагинов или проигрывателей. Он начинается с тега `<audio>` и заканчивается тегом `</audio>`. Атрибуты элемента `audio` перечислены в табл. 11.8.

[1] code.google.com/p/swfobject/wiki/documentation
[2] www.ibm.com/developerworks/ru/library/wa-HTML5video/index.html

Таблица 11.8. Атрибуты элемента `audio`

Атрибут	Значение	Применение
`autoplay`	`autoplay`	Необязателен; указывает, должно ли аудио воспроизводиться автоматически; применяйте с осторожностью
`controls`	`controls`	Необязателен; указывает, должна ли отображаться панель управления; рекомендуется
`scr`	Имя файла	Необязателен; имя аудиофайла
`type`	Тип MIME	Необязателен; тип MIME аудиофайла, например `audio/mpeg` или `audio/ogg`
`loop`	`loop`	Необязателен; указывает, будет ли аудиофайл проигрываться зациклено
`preload`	`none`, `metadata`, `auto`	Необязателен; значения: `none` (файл не загружается при загрузке страницы), `metadata` (загружает только метаданные мультимедийного файла), `auto` (загружает файл)
`title`	Текстовое описание	Необязателен; содержит краткое текстовое описание, которое будет отображаться браузерами или вспомогательными технологиями

Из-за того что браузеры поддерживают различные кодеки, вам нужно предоставить несколько версий аудиофайла. Планируйте предоставлять аудиофайлы, по крайней мере, в двух различных контейнерах, включая OGG и MP3. Атрибуты `scr` и `type` обычно не используют с элементом `audio`, а вместо этого конфигурируют несколько версий аудиофайла с элементом `source`.

Элемент `source`

Элемент `source` — контейнерный тег, который указывает на мультимедийный файл и тип MIME. Атрибут `src` определяет имя мультимедийного файла. Атрибут `type` указывает на MIME-тип файла.

Для файла MP3 добавьте код `type="audio/mpeg"`, а для аудиофайла, использующего кодек Vorbis, — код `type="audio/ogg"`. Конфигурируйте элемент `source` для каждой версии аудиофайла. Поместите его перед закрывающим тегом элемента `audio`.

Элемент `audio` на веб-странице

Следующий код конфигурирует веб-страницу, показанную на рис. 11.10 (см. файл *Примеры\Глава-11\audio* на диске, прилагающемся к книге), чтобы отобразить панель управления для аудиофайла:

```
<audio controls="controls">
<source src="soundloop.mp3" type="audio/mpeg">
<source src="soundloop.ogg" type="audio/ogg">
<a href="soundloop.mp3">Загрузить аудио-файл</a> (MP3)
</audio>
```

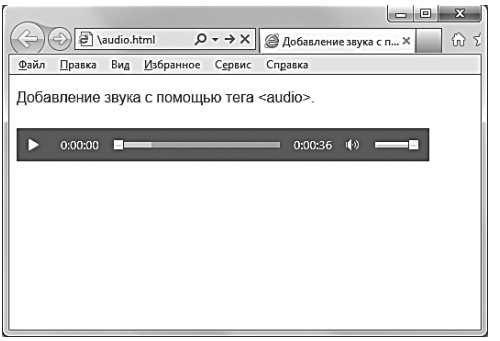


Рис. 11.10. Браузер Internet Explorer 9 поддерживает элемент `audio`

Текущие версии браузеров Safari, Chrome, Firefox, и Opera также поддерживают элемент `audio`. Браузер Internet Explorer 9 также поддерживает этот элемент, однако ранние версии Internet Explorer его не поддерживают. Каждый браузер отображает панель управления аудиофайла по-своему. Просмотрите предыдущий код и обратите внимание на ссылки, размещенные между вторым элементом `source` и закрывающим тегом `</audio>`. Любые HTML-элементы или текст, помещенный в этой области, визуализируются браузерами, не поддерживающими элемент `audio`. Это так называемый ***запасной контент***, и если элемент `audio` не поддерживается, MP3-версия файла доступна для скачивания. На рис. 11.11 показан снимок экрана в Internet Explorer 8, отображающий веб-страницу.

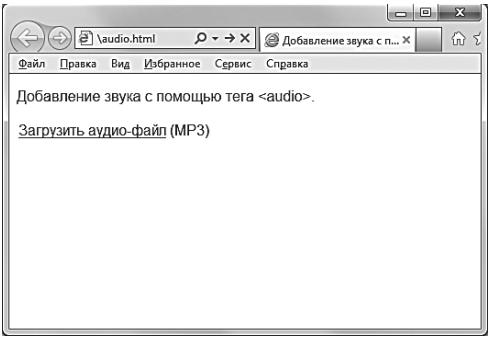


Рис. 11.11. Internet Explorer 8 не распознает элемент `audio`

Практическое задание 11.6

В этом практическом задании вы запустите программу Блокнот (Notepad) и создадите веб-страницу (рис. 11.12), на которой отображена панель управления аудиофайлом для воспроизведения подкаста.

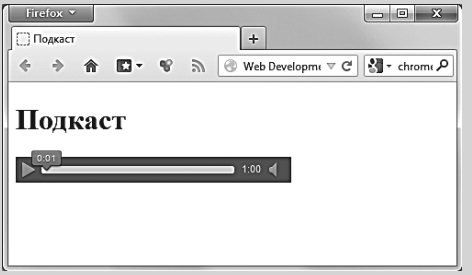


Рис. 11.12. Использование элемента `audio` для обеспечения доступа к подкасту

Скопируйте файлы *podcast.mp3, podcast.ogg* и *podcast.txt* из папки *Примеры\Глава_11\starters* на прилагающемся к книге диске и сохраните их в папку *audio5*. Используйте файл *Примеры\Глава_02\template.html* в качестве отправной точки и создайте веб-страницу с заголовком «Подкаст», панель управления звуком (используйте элемент `audio` и два элемента `source`) и гиперссылку на текст транскрипции. Конфигурируйте гиперссылку на MP3-файл в качестве запасного контента. Код для элемента `audio`:

```
<audio controls="controls">
<source src="podcast.mp3" type="audio/mpeg">
<source src="podcast.ogg" type="audio/ogg">
<a href="podcast.mp3">Загрузить подкаст</a> (MP3)
</audio>
```

Сохраните страницу с именем *index.html* в папке *audio5* и откройте ее в браузере. Протестируйте веб-страницу в различных браузерах, с использованием различных версий. Напомним, что браузер Internet Explorer (кроме версии 9) не поддерживает элемент `audio`, однако отображает запасной контент. При щелчке мышью по ссылке на текст транскрипции текст будет отображаться в браузере. Сравните свою работу с образцом (*Примеры\Глава-11\podcast\podcast5.html*).

ЧаВо

КАК КОНВЕРТИРОВАТЬ АУДИОФАЙЛЫ В ФОРМАТ OGG VORBIS?

Кодек Ogg Vorbis поддерживается приложением с открытым исходным кодом Audacity. Информацию о том, как его скачать, можно

найти на сайте **audacity.sourceforge.net**. Если вы ищете бесплатный веб-конвертер, можно загрузить иаудиофайлы в общий интернет-архив[1], и файл в формат OGG сгенерируется автоматически.

Элемент `video`

Новый HTML5-***элемент*** **`video`** поддерживает воспроизведение видеофайлов в браузере без установки дополнительных плагинов или проигрывателей. Он начинается с тега `<video>` и заканчивается тегом `</video>`. Атрибуты элемента `video` перечислены в табл. 11.9.

Таблица 11.9. Атрибуты элемента `video`

Атрибут	Значение	Применение
`autoplay`	`autoplay`	Необязателен; указывает, должно ли видео воспроизводиться автоматически; применяйте с осторожностью
`controls`	`controls`	Необязателен; указывает, должна ли отображаться панель управления; рекомендуется
`height`	Числовое значение	Необязателен; высота окна видео в пикселах
`loop`	`loop`	Необязателен; указывает, будет ли видео проигрываться зациклено
`poster`	Имя файла	Необязателен; определяет, какое изображение будет показано, если браузер не сможет воспроизвести видео
`preload`	`none`, `metadata`, `auto`	Необязателен; значения: `none` (видео не загружается при загрузке страницы), `metadata` (загружает только метаданные мультимедийного файла), `auto` (загружает мультимедийный файл)
`scr`	Имя файла	Необязателен; имя видеофайла
`title`	Текстовое описание	Необязателен; содержит краткое текстовое описание, которое будет отображаться браузерами или вспомогательными технологиями
`type`	Тип MIME	Необязателен; тип MIME видеофайла, например `video/mpeg` или `video/ogg`
`width`	Числовое значение	Необязателен; ширина окна видеоизображения в пикселах

Из-за того что браузеры поддерживают различные кодеки, вам нужно предоставить несколько версий видеофайла. Планируйте предоставлять видеофайлы, по крайней мере, в двух различных контейнерах, включая MP4 и OGG (или OGV). Атрибуты `scr` и `type` обычно не используют с элементом `video`, а вместо этого конфигурируют несколько версий видеофайла с элементом `source`.

[1] www.archive.org

Элемент `source`

Из предыдущего одноименного раздела вы помните, что ***элемент* `source`** — контейнерный тег, который определяет мультимедийный файл и тип MIME. Атрибут `src` определяет имя мультимедийного файла. Атрибут `type` указывает на MIME-тип файла. Для видеофайлов, использующих кодек MP4, добавьте код `type="audio/mpeg"`, а для видеофайла, использующего кодек Vorbis, — код `type="video/mp4"`. Конфигурируйте элемент `source` для каждой версии видеофайла. Поместите его перед закрывающим тегом `</video>`.

Элемент `video` на веб-странице

Следующий код конфигурирует веб-страницу, показанную на рис. 11.13 (см. *Примеры\Глава_11\sparky2.html* на диске, прилагающемся к книге), с заданной HTML5 панелью управления для отображения и воспроизведения видеофайла:

```
<video controls="controls" poster="sparky.jpg"
width="160" height="150">
<source src="sparky.m4v" type="video/mp4">
<source src="sparky.ogv" type="video/ogg">
<a href="sparky.mov">Петя - это собачка</a> (.mov)
</video>
```

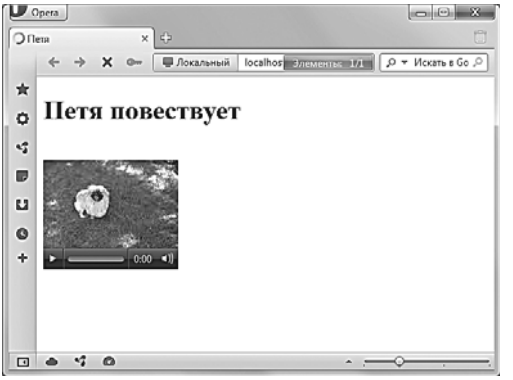


Рис. 11.13. Браузер Opera

Текущие версии браузеров Safari, Chrome, Firefox и Opera поддерживают элемент `video`. Браузер Internet Explorer 9 также поддерживает этот элемент, однако ранние версии Internet Explorer его не поддерживают.

Каждый браузер отображает панель управления видеофайла по-своему. Просмотрите предыдущий код и обратите внимание на элемент привязки между вторым элементом `source` и закрывающим тегом `</video>`.

Любые HTML-элементы или текст, помещенный в этой области, визуализируются браузерами, не поддерживающими элемент `video`. Это так называемый ***запасной контент***. Он предлагает пользователю закачать версию файла в формате .MOV (QuickTime). Другой запасной вариант — конфигурировать элемент `embed` для воспроизведения Flash-версии видеофайла в формате SWF. На рис. 11.14 показан снимок экрана браузера Internet Explorer 8.

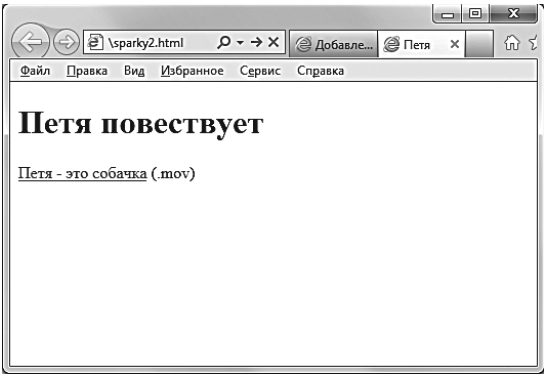


Рис. 11.14. Веб-страница, отображенная в браузере Internet Explorer 8

Практическое задание 11.7

В этом практическом задании вы запустите программу Блокнот (Notepad) и создадите веб-страницу (рис. 11.15), на которой отображена панель управления видеофайлом для воспроизведения видеоролика. Скопируйте файлы *lighthouse.m4v, lighthouse.ogv, lighthouse.swf и lighthouse.jpg* из папки *Примеры\Глава_11\starters* на диске, прилагающемся к книге, и сохраните их в папку *video*. Откройте файл *Примеры\Глава_02\template.html* и создайте веб-страницу с заголовком «Путешествие по графству До», панель управления видео (используйте элемент `video` и два элемента `source`). Конечно, можно было бы задать в качестве запасного вариант гиперссылку на видеофайл, однако в этом примере мы сконфигурируем элемент `embed`, чтобы отобразить мультимедийный файл Flash (*lighthouse.swf*), который станет запасным вариантом. Задайте файл *lighthouse.jpg*, чтобы он появился, если браузер поддерживает элемент `video`, но не может воспроизвести видеофайл.

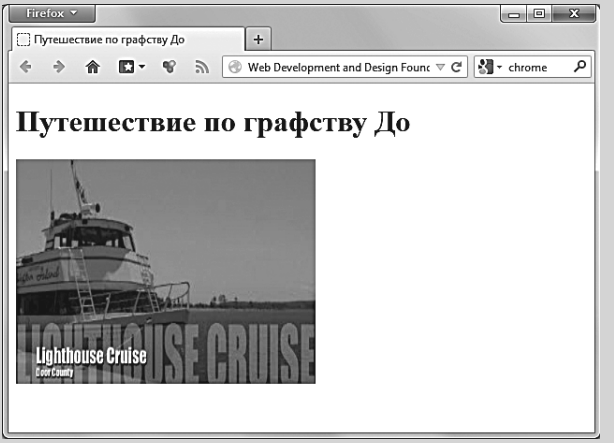


Рис. 11.15. Элемент `video`

Код для элемента `video`:

```
<video controls="controls" poster="lighthouse.jpg"
width="320" height="240">
<source src="lighthouse.m4v" type="video/mp4">
<source src="lighthouse.ogv" type="video/ogg">
<embed type="application/x-shockwave-flash"
src="lighthouse.swf" width="320" height="240"
title="Путешествие по графству До">
</video>
```

Сохраните страницу с именем *index.html* в папке *video* и откройте ее в браузере. Протестируйте веб-страницу в различных браузерах. Сравните свою работу с образцом (*Примеры\Глава-11\podcast\video.html*).

ЧаВо

КАК КОНВЕРТИРОВАТЬ ВИДЕОФАЙЛ С ПОМОЩЬЮ НОВЫХ КОДЕКОВ?

Для конвертации видеофайла с помощью кодека Ogg Theora можно использовать плагин Firefogg[1]. Сервис Online-Convert предлагает бесплатную конвертацию в формат WebM[2]. Бесплатное приложение с открытым исходным кодом MiroVideoConverter[3] может конвертировать большинство видеофайлов в форматы MP4, WebM и OGG.

[1] firefogg.org
[2] video.online-convert.com/convert-to-webm
[3] www.mirovideoconverter.com

11.6. Вопросы авторского права и мультимедийные файлы

Очень легко скопировать и загрузить изображение, аудио- или видеофайл с веб-сайта. Это может быть очень заманчивым — поместить чужой файл в ваш собственный проект, но это будет как неэтично, так и незаконно. Публикуйте только те веб-страницы, изображения и другие мультимедийные файлы, которые создали вы лично или на использование которых вы приобрели права или лицензию. Если другой человек создал изображение, звук, видеоролик или документ, который, как вы думаете, будет полезен на вашем собственном веб-сайте, запросите разрешение использовать этот материал, вместо того чтобы просто взять его без спроса. Любая работа (веб-страницы, изображения, звуки, видео и т. д.) защищена авторским правом — даже если на ней нет символа авторского права.

Впрочем, помните, что есть случаи, когда могут использоваться фрагменты чужой работы, и это не будет являться нарушением авторского права. Это называется ***законное использование***. Законное использование может осуществляться в целях критики, отчетности, преподавания, подготовки научной работы или исследования. Критерии для определения законности использования приведены ниже:

- использование должно быть скорее в образовательных целях, чем коммерческих;
- содержание скопированной работы должно быть скорее фактическим, нежели творческим;
- размер скопированной части материала должен быть как можно меньше;
- копирование не должно влиять на реализуемость оригинала.

Чтобы узнать больше о вопросах авторского права, посетите страницу **www.4uth.gov.ua/usa/russian/information/rl30495.htm**.

Некоторые авторы могут захотеть сохранить право собственности на их работу, но упростить ее использование или адаптацию другими. Сайт Creative Commons[1] предоставляет бесплатный сервис, позволяющий авторам регистрировать тип лицензии авторского права, который называется ***лицензия Creative Commons***. Есть несколько типов лицензий,

[1] creativecommons.org

из которых можно выбрать нужную вам — в зависимости от прав, которые вы, как автор, хотите предоставить любому желающему. Лицензия Creative Commons информирует других людей о допустимых действиях с авторской работой.

11.7. CSS3 и интерактивность

Создание галереи изображений с помощью CSS

Напомним, в главе 6 встречался CSS-псевдокласс `:hover`, позволяющий настроить стили, которые будут отображаться, когда посетитель веб-страницы наводит указатель мыши на элемент. Вы используете эти базовые элементы интерактивности наряду с позиционированием и свойствами отображения, чтобы настроить интерактивную галерею изображений с помощью CSS и HTML. На рис. 11.16 показана галерея в действии (см. файл *Примеры\Глава_11\gallery\gallery.html* на диске, прилагающемся к книге). Если вы поместите указатель мыши на миниатюру изображения, появится увеличенное изображение вместе с заголовком.

Если щелкнуть по миниатюре мышью, изображение откроется в отдельном окне браузера.

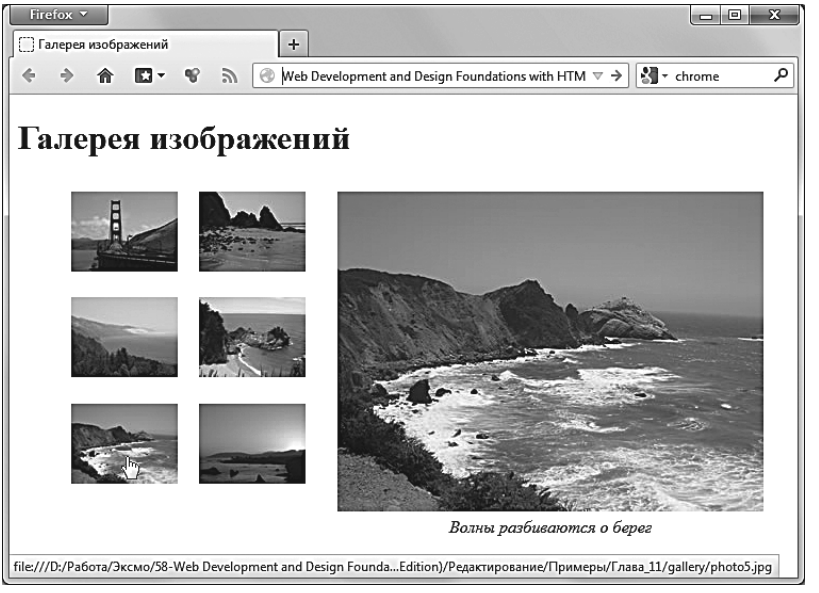


Рис. 11.16. Интерактивная галерея изображений, созданная с помощью CSS

Практическое задание 11.8

В этом практическом задании вы создадите веб-страницу с галереей изображений, показанную на рис. 11.16. Скопируйте следующие изображения из папки *Примеры\Глава_11*\starters на диске, прилагающемся к книге: *photo1.jpg, photo2.jpg, photo3.jpg, photo4.jpg, photo5.jpg, photo6.jpg, photo1thumb.jpg, photo2thumb.jpg, photo3-thumb.jpg, photo4thumb.jpg, photo5thumb.jpg* и *photo6thumb.jpg.* Сохраните их в папку с именем *gallery*.

Запустите программу Блокнот (Notepad) и измените файл *Примеры\Глава_02\template.html*, как показано ниже, чтобы конфигурировать веб-страницу:

1. Задайте в элементе `h1` текст «Галерея изображений».
2. Добавьте в код элемент `div`, которому назначен идентификатор с именем `gallery`. Этот `div` будет содержать эскизы.
3. Конфигурируйте в элементе `div` неупорядоченный список. Задайте шесть элементов `li` по одному для каждой миниатюры изображения. Миниатюры будут работать как графические ссылки с псевдоклассом `:hover`, который будет отображать на странице более крупное изображение. Мы заставим все это работать, задав элемент `href`, содержащий миниатюру изображения и элемент `span`, включающий в себя более крупное изображение вместе с описанием. Вот пример первого элемента `li`:

```
<li><a href="photo1.jpg"><img src="photo1thumb.jpg"
width="100" height="75" alt="Мост Золотые врата">
<span><img src="photo1.jpg" width="400" height="300"
alt="Мост Золотые врата"><br>Мост Золотые врата</
span></a>
</li>
```

4. Конфигурируйте подобным образом все шесть элементов `li`. Вместо фактического имени каждого файла в коде введите значения атрибутов `href` и `src`. Впишите собственное текстовое описание для каждого изображения. Для второго элемента `li` используйте файлы *photo2.jpg* и *photo2thumb.jpg*, для третьего элемента `li` — *photo3.jpg* и *photo3thumb.jpg* и т. д. Сохраните файл с именем *index.html* в папку *gallery*. Откройте веб-страницу в браузере. Вы увидите неупорядоченный список с миниатюрами изображений, более крупные иллюстрации и текстовые описания. На рис. 11.17 показан фрагмент экрана.
5. Теперь добавим глобальные правила CSS. Откройте файл в программе Блокнот (Notepad) и добавьте в раздел заголовка элемент `style`. Для идентификатора `gallery` будет использоваться относительное позиционирование.

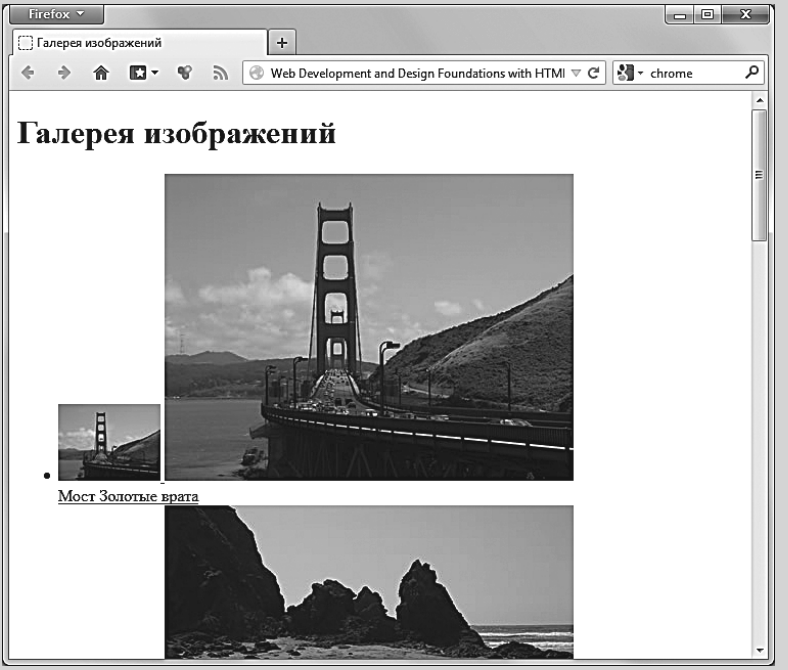


Рис. 11.17. Вид веб-страницы до применения CSS

Это не изменит расположения галереи, но позволит применить абсолютное позиционирование элемента `span` по отношению к его контейнеру (`#gallery`), а не по отношению ко всему документу. Это не будет иметь большого значения в нашем очень простом примере, но могло бы оказаться очень полезным, если бы галерея была частью более сложной веб-страницы. Добавьте в код глобальные правила CSS, чтобы сконфигурировать галерею с относительным позиционированием; неупорядоченный список без маркеров шириной 300 пикселов; элементы списка отображаются в строку, с обтеканием по левому краю и отступом 10 пикселов; изображения без границы; теги привязки без подчеркивания, имеют темно-серый цвет (# 333), и шрифт набран курсивом; и установлено по умолчанию значение `none` для тега `span`. Код CSS следующий:

```
#gallery {position: relative; }
#gallery ul {list-style-type: none;
width: 300px; }
#gallery li { display: inline;
float: left;
padding: 10px; }
```

```
#gallery img {border-style: none; }
#gallery a { text-decoration: none;
font-style: italic;
color: #333; }
#gallery span {display: none;
```

Далее присвойте элементу `span` значение `display: only;`, когда посетитель наводит указатель на графическую ссылку в виде миниатюры. Установите местоположение тега `span`, чтобы применить абсолютное позиционирование. Задайте промежуток 10 пикселов от верхней границы и 300 пикселов с левой стороны. Отцентрируйте текст, содержащийся в теге `span`:

```
#gallery a:hover span {display: block;
position: absolute;
top: 10px;
left: 300px;
text-align: center; }
```

Сохраните веб-страницу и откройте ее в браузере. Ваша галерея должна хорошо работать в современных браузерах. Следует отметить, что устаревший браузер Internet Explorer 6 не поддерживает динамическое отображение более крупного изображения, но вместо этого показывает неупорядоченный список и обрабатывает графические ссылки-миниатюры.

Сравните свою работу с рис. 11.16 и образцом на диске, прилагающемся к книге, *Примеры\Глава_11\gallery\gallery.html*.

Свойство `transform`

Спецификация CSS3 предоставляет способ изменить или преобразовать отображения элементов. ***Свойство*** **`transform`** позволяет вращать, масштабировать, наклонять или перемещать элемент. Возможно как двумерное (2D), так и трехмерное (3D) преобразование.

Преобразования часто используются в сочетании со свойством `transition` (описывается в следующем разделе).

Разработчики движков браузеров, таких как WebKit (используется в браузерах Safari и Google Chrome) и Gecko (используется в Firefox и других браузерах корпорации Mozilla), создали собственные свойства для реализации свойства `transform`. Итак, чтобы задать преобразование, вам нужно указать в коде несколько определений стиля:

- `-webkit-transform` (для браузеров на движке Webkit);
- `-moz-transform` (для браузеров на движке Gecko);
- `-o-transform` (для браузеров Opera);
- `-ms-transform` (для браузера Internet Explorer 9);
- `transform` (черновой синтаксис консорциума W3C).

В конце концов, все браузеры начнут поддерживать CSS3 и свойство `transform`, поэтому укажите в коде это свойство последним. Веб-страница, показанная на рис. 11.18 (см. файл *Примеры\Глава_11\transform\transform.html* на диске, прилагающемся к книге) демонстрирует использование CSS3-свойства `transform`, примененного, чтобы слегка повернуть фигуру. Правила CSS также используются для настройки границы, текстовой подписи и тени блока в элементе `div`, содержащем изображение.

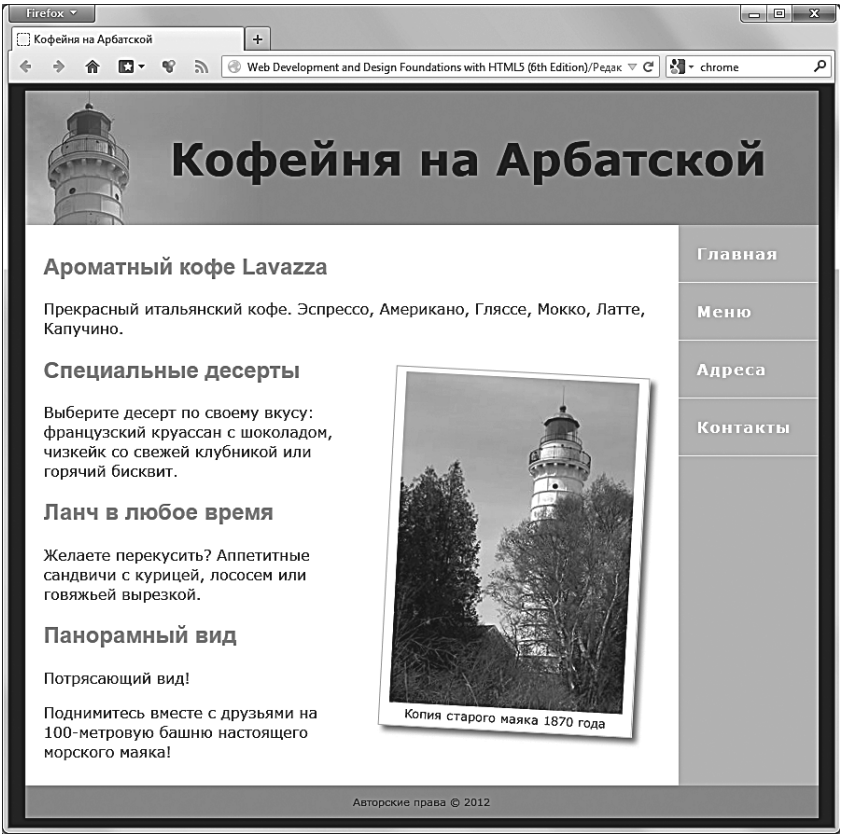


Рис. 11.18. Свойство `transform` в действии

Настройка вращения с помощью CSS3

Для преобразования `rotate()` значения указываются в градусах (как угол в геометрии), положительные значения для вращения вправо и отрицательные значения для вращения влево. Вращение происходит вокруг исходной точки, которой, по умолчанию, является середина элемента. Приведенный ниже CSS-код сначала задает для класса `figure` обтекание справа, ширину 260 пикселов, поля величиной 20 пикселов, отступ шириной 5 пикселов, белый фон, сплошную границу серого цвета шириной 1 пиксел и отцентрированный текст, размер которого 0.80 em. Следующий за ним CSS3-код задает вращение на 3 градуса и применяет тень блока. Код приведен ниже:

```
.figure {float: right;
width: 260px;
margin: 20px;
background-color:#FFF;
text-align: center;
font-size: .80em;
padding: 5px;
border: 1px solid #CCC;
-webkit-transform: rotate(3deg); ;
-moz-transform: rotate(3deg);
-o-transform: rotate(3deg);
-ms-transform: rotate(3deg);
transform: rotate(3deg);
-webkit-box-shadow: 5px 5px 5px #828282;
-moz-box-shadow: 5px 5px 5px #828282;
box-shadow: 5px 5px 5px #828282;
```

Напомним, что собственный синтаксис CSS браузеров, приведенный в этом разделе, нестандартный. Ваш код CSS не пройдет валидацию W3C при использовании этих свойств.

В этом разделе представлен обзор одного типа преобразования: вращение элемента. Посетите сайт Westciv[1], содержащий инструменты и ресурсы для веб-профессионалов, чтобы сгенерировать CSS-код для преобразований вращения, масштабирования, перемещения и наклона.

[1] www.westciv.com/tools/transforms/index.html

Для получения дополнительной информации о синтаксисе, используемом в преобразованиях, посетите соответствующие веб-сайты:

- Webkit: **www.webkit.org/blog/130/css-transforms**
- Mozilla: **developer.mozilla.org/en/CSS/-moz-transform**
- Opera: **dev.opera.com/articles/view/css3-transitions-and-2-dtransforms/#transforms**
- Internet Explorer: **msdn.microsoft.com/en-us/ie/ff468705.aspx#_CSS3_2D_Transforms**
- W3C: **www.w3.org/TR/css3-2d-transforms** и **www.w3.org/TR/css3-3d-transforms**

Свойство `transition`

***Свойство* `transition`** в спецификации CSS3 (переход) обеспечивает плавные изменения в значениях свойств в течение определенного времени. Со свойством `transition` можно использовать четыре различных параметра: `transition-property`, `transition-duration`, `transition-timing-function` и `transition-delay`. Эти параметры можно объединить в одно свойство `transition`. В таблице 11.10 перечислены свойства группы `transition` и их назначение.

Таблица 11.10. Свойства группы `transition`

Свойство	Описание
`transition`	Общее свойство. Перечислите свойства `transition-property`, `transition-duration`, `transition-timing-function` и `transition-delay`, разделив их пробелами. Установленные по умолчанию значения можно опустить, однако первая единица времени будет относиться к свойству `transition-duration`
`transition-delay`	Определяет начало перехода; по умолчанию установлено значение `0` (без задержки) или вы можете ввести числовое значение, чтобы указать время (как правило, в секундах)
`transition-duration`	Указывает период времени, на который применяется переход. Значение по умолчанию 0 задает немедленный переход, в других случаях для определения времени используйте числовые значения (как правило, указывается в секундах)
`transition-property`	Указывает свойство CSS, к которому применяется переход; перечень соответствующих свойств можно найти на сайте **www.w3.org/TR/css3-transitions**
`transition-timing-function`	Задает изменения скорости перехода, описывая, как рассчитываются непосредственные значения свойства; распространенные значения: `ease` (по умолчанию), `linear`, `ease-in`, `ease-out`, `ease-in-out`

Префиксы перед свойством браузера нужны для конфигурирования переходов. Переходы поддерживаются текущими версиями большинства современных браузеров. Тем не менее в настоящее время они не поддерживаются браузером Internet Explorer. Чтобы задать переход, укажите в коде четыре определения стиля:

- `-webkit-transition` (для браузеров на движке Webkit);
- `-moz-transition` (для браузеров на движке Gecko);
- `-o-transition` (для браузеров Opera);
- `transition` (черновой синтаксис W3C).

На веб-странице, показанной на рис. 11.19 (см. файл *Примеры\Глава_11\gallery\transition.html* на диске, прилагающемся к книге), свойство `transition` применяется, чтобы выровнять фотографию маяка, когда посетитель наводит на него указатель мыши. Сравните положение фотографии маяка на рис. 11.18 и 11.19, чтобы увидеть действие свойства `transition`.

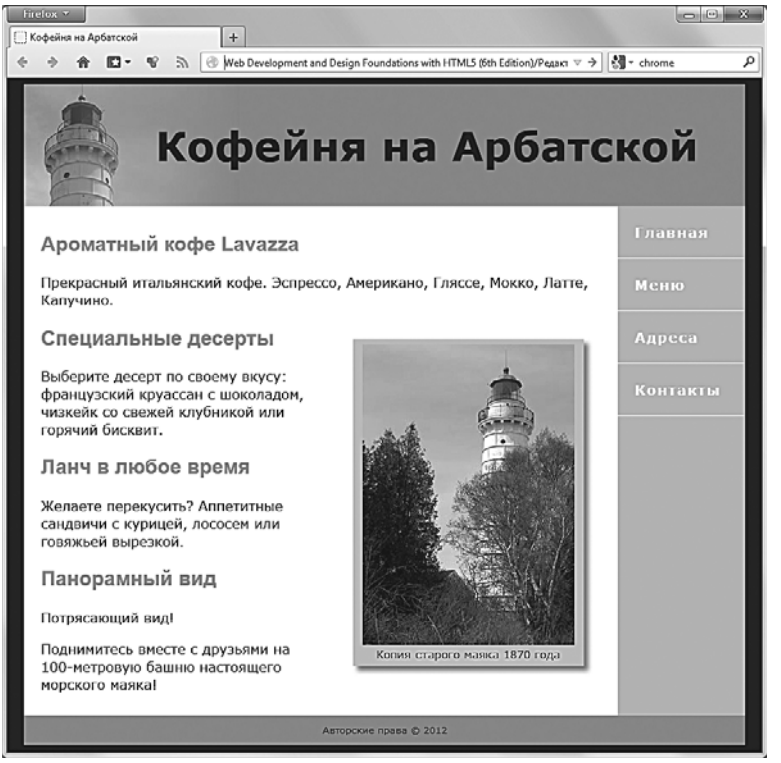


Рис. 11.19. Интерактивный эффект перехода с использованием CSS3

Настройка свойства `transition`

Свойство `transition` было применено к свойству `transform`. Для поворота обратно до 0 градусов за ½ секунды, плавного и с одинаковой скоростью (`linear`) используется общее свойство `transition`. Ключевой момент здесь — задание правила стиля для псевдокласса `.figure:hover` с определением стиля, в котором после определений стиля свойства `transition` указывается значение свойства `transform`.

В новом коде CSS много повторений из-за префиксов с указанием свойства браузера перед свойствами CSS3:

```
.figure:hover { -webkit-transform: rotate(0deg);
-moz-transform: rotate(0deg);
-o-transform: rotate(0deg);
transform: rotate(0deg);
-webkit-transition: -webkit-transform .5s linear;
-moz-transition: -moz-transform .5s linear;
-o-transition: -o-transform .5s linear;
transition: transform .5s linear; }
```

Как показано в коде выше, синтаксис для конфигурирования переходов отличается в зависимости от движка браузера. Ожидается, что в конечном итоге все браузеры будут поддерживать синтаксис перехода, предложенный консорциумом W3C, поэтому укажите это определение последним в списке. Напомним, что собственный синтаксис CSS браузера, рассматриваемый в этом разделе, нестандартный. Ваш код CSS не пройдет проверку валидатора W3C при использовании этих свойств.

Практическое задание 11.9

В этом практическом задании вы модифицируете веб-страницу, показанную на рис. 11.19, и сделаете так, чтобы цвет фона класса `figure` менялся с белого на светло-серый при установке указателя мыши на фотографию маяка. Создайте папку с именем *transition* и скопируйте в нее файлы *lighthouseisland.jpg* и *lighthouselogo.jpg* из папки *Примеры\Глава_11\starters* на диске, прилагающемся к книге. Запустите программу Блокнот (Notepad) и откройте файл *transition.html*, который находится в папке *Примеры\Глава_11\transform* на диске, прилагающемся к книге. Наведите указатель мыши на изображение *lighthouseisland.jpg*. Если ваш браузер поддерживает

свойство `transition`, странице должна быть похожа на изображенную на рис. 11.19.

Измените глобальные правила CSS в файле и модифицируйте селектор `.figure:hover` так, чтобы изменить цвет фона на `#ccc` с длительностью перехода в течение 1 секунды. Обратите внимание, как можно настроить переходы для нескольких свойств, разделив их запятой.

```
.figure:hover { background-color: #ccc;
-webkit-transform: rotate(0deg);
-moz-transform: rotate(0deg);
-o-transform: rotate(0deg);
transform: rotate(0deg);
-webkit-transition: -webkit-transform .5s linear, background-color 1s ease-in;
-moz-transition: -moz-transform .5s linear, background-color 1s ease-in;
-o-transition: -o-transform .5s linear, background-color 1s ease-in;
transition: transform .5s linear, background-color 1s ease-in; }
```

Сохраните файл под именем *index.html* в папке *transition* и протестируйте его в браузере, поддерживающем переходы. При наведении курсора на фотографию маяка вы должны увидеть, как она повернется и изменится цвет фона в области фотографии. Образец можно найти на прилагающемся к книге диске (*Примеры\Глава_11\transition2.html*).

Дополнительные примеры использования свойств CSS `transform` и `transition` представлены на следующих ресурсах:

- переходы CSS 101: **www.webdesignerdepot.com/2010/01/css-transitions-101**;
- использование свойств CSS `transform` и `transition`: **return-true.com/2010/06/using-css-transforms-and-transitions**;
- переходы CSS3: **net.tutsplus.com/tutorials/HTML-css-techniques/css-fundametals-css-3-transitions**.

11.8. Технология Java

Java — это объектно-ориентированный язык программирования (object-oriented programming, OOP), разработанный компанией Sun Mi-

crosystems (ныне Oracle). Объектно-ориентированная программа состоит из группы совместно действующих объектов, которые обмениваются сообщениями с целью достижения общей задачи. Java не является тем же языком, что и JavaScript. Он более мощный и гораздо более гибкий, чем JavaScript. Язык Java можно использовать для разработки как автономных приложений, так и вспомогательных программ, или ***апплетов***, которые будут запускаться веб-страницами.

Java-апплеты независимы от платформы; это значит, что их можно написать и запустить с любой платформы — Mac, UNIX, Linux и Windows. Java-апплеты компилируются (переводятся с подобных английскому языку предложений на языке Java в форму кода) и сохраняются как файлы с расширением *.class*, которые содержат байтовый код. Байтовый код интерпретируется приложением ***виртуальная машина Java*** (JVM, Java Virtual Machine) в веб-браузере в соответствующий для конкретной операционной системы машинный язык. После этого апплет выполняется и появляется на веб-странице. См. диаграмму на рис. 11.20, изображающую этот процесс. Когда Java-апплет загружается, область, зарезервированная для него на веб-странице, отображается как пустая прямоугольная область до того момента, как апплет начнет выполняться.

Рис. 11.20. Виртуальная машина Java интерпретирует байтовый код в машинный язык

Java-апплеты могут обрабатывать интерактивные панели навигации на веб-страницах. Посетите ресурсы Java Boutique[1] и Apycom[2], на которых можно найти множество готовых Java-апплетов меню и навигации.

[1] javaboutique.internet.com/navigation/menu.html
[2] apycom.com

Еще одним популярным способом использования Java-апплетов является создание игр для веб-страниц. Посетите сайт Java on the Brain[1], на котором вы увидите примеры игр в виде Java-апплетов.

Java-апплеты можно также использовать для создания графических и текстовых эффектов, таких как пример на рис. 11.21 (см. файлы *Примеры\Глава_11\java1.html* на диске, прилагающемся к книге).

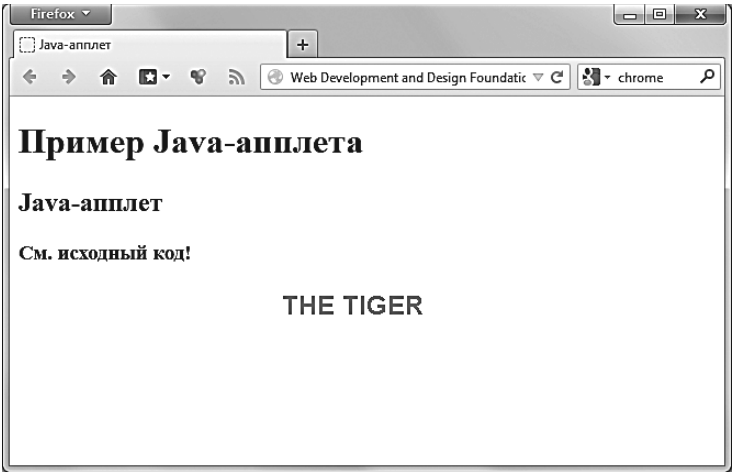


Рис. 11.21. Добавление Java-апплета на веб-страницу

Хотя эффекты изображений и игры — это интересный способ использования языка Java, использование Java-апплетов в бизнес-приложениях встречается все чаще благодаря таким функциям, как финансовые вычисления и визуализация.

Сайт JARS[2] предоставляет обзор и описание Java-апплетов, полезных в сфере бизнеса, таких как NetCharts[3]. Такой тип апплетов часто объединяют с базами данных на веб-сервере, благодаря чему они могут стать мощными инструментами для визуального отображения оперативных данных.

Java-апплеты могут выполнять самые разнообразные функции на веб-страницах. В ваши обычные обязанности как веб-разработчика не будут входить обязанности программиста Java — это значит, что никто не должен ожидать от вас создания Java-апплетов. Тем не менее вас могут

[1] www.javaonthebrain.com/brain.html
[2] www.jars.com
[3] visualmining.com

попросить о совместной работе с программистом Java, чтобы поместить апплеты на веб-страницу. Независимо от того, получите вы апплет от своего сотрудника или найдете его на бесплатном сайте, вам нужно создать соответствующий HTML-код для отображения апплета на странице.

Некоторое время назад для конфигурирования Java-апплета на веб-странице использовался ныне устаревший элемент `applet` вместе с элементом `param`. Консорциум W3C рекомендует использовать для помещения Java-апплета на веб-страницу многоцелевой элемент `object`. Тег `<object>` указывает на начало области апплета в разделе тела страницы. Закрывающий тег `</object>` указывает окончание кода апплета в теле веб-страницы. Элементу `object>` могут быть присвоены определенные атрибуты, описанные в табл. 11.11.

Таблица 11.11. Атрибуты элемента `object`

Атрибут	Значение
`height`	Определяет высоту области апплета в пикселах
`title`	Необязательный; короткое текстовое описание, которое может отображаться браузером или вспомогательными технологиями
`type`	Допустимый тип MIME Java-апплета, такой как `application/x-java-applet`
`width`	Определяет ширину области апплета в пикселах

Значения и имена параметров, необходимых конкретному Java-апплету, определяет программист, который его создает. Тем не менее будьте готовы к тому, что каждый апплет будет требовать разные параметры. Один параметр может служить для установки цвета фона, а другой содержать в себе имя какой-либо персоны. Параметры настраиваются с помощью элемента `param`. Параметр `code` указывает на имя Java-апплета.

Практическое задание 11.10

В этом практическом задании вы запустите программу Блокнот (Notepad) и создадите веб-страницу, которая будет содержать Java-апплет. В этом примере мы используем апплет Fader26 (созданный Йоханнесом Шеленном). Этот апплет по очереди отображает текстовые сообщения. Список текстовых сообщений извлекается из текстового файла (с расширением *.txt*), который вы создадите. Рабочий пример этого апплета вы можете увидеть, открыв файл *Примеры\Глава_11\java1.html* на диске, прилагающемся к книге.

Приступим. Создайте новую папку на вашем компьютере и назовите ее *testapplet*. Скопируйте файл апплета (*fader26.class*) из папки *Примеры\Глава_11* диска, прилагающегося к книге, и поместите его в папку *testapplet*. Не изменяйте имя апплета.

Вне зависимости от того, получили вы апплет от сотрудника или достали его на бесплатном сайте, у каждого апплета должна быть сопровождающая документация, которая определяет, какие параметры ему нужны. Документация апплета Fader26 указана в табл. 11.12.

Таблица 11.12. Документация для апплета Fader26

Имя параметра	Значение параметра
`code`	*fader26.class*
`AppletHome`	**http://www.crosswinds.net/~fader/**
`Data`	Имя текстового файла, содержащего сообщения, которые будет отображать апплет (***примечание***: каждая новая строка текстового файла должна начинаться с `text=`)
`bgColor`	Это цвет фона области Java-апплета; использует шестнадцатеричную цветовую схему

Используйте файл *Примеры\Глава_02\template.html* в качестве отправной точки. Создайте веб-страницу с заголовком «Java-апплет» и соответствующими элементами `object` и `param` для отображения Java-апплета *fader26.class* в области шириной 610 пикселов, высотой 30 пикселов, с фоном белого цвета (параметр `bgColor`) и имеющим доступ к текстовому файлу *message.txt* (параметр `data`). Также добавьте описательный заголовок и присвойте значение `http://www.crosswinds.net/~fader` параметру `AppletHome`. HTML-код приведен ниже:

```
<object type="application/x-java-applet" width="610"
height="30" title="Этот Java-апплет демонструет тесто-
вое сообщение.">
<param name="code" value="fader26.class">
<param name="AppletHome" value="http://www.crosswinds.
net/~fader/">
<param name="Data" value="message.txt">
<param name="bgColor" value="#FFFFFF">
Java-апплеты могут быть использованы для отображения
текста, изображений, воспроизведения игр и пр.
Посетите веб-сайт компании <a href="http://download.
oracle.com/javase/tutorial/">Oracle</a> для дополни-
тельной информации.
</object>
```

Сохраните файл в папку *testapplet* с именем *java.html*. Вы пока не готовы протестировать страницу — нужно создать и отредактировать текстовый файл, необходимый для работы апплета. Для работы апплета необходимо, чтобы каждая строка текстового файла начиналась со значения `text=`.

Рисунок 11.22 показывает пример текстового файла, содержащего оригинал стихотворения Уильяма Блейка «Тигр», созданного с помощью программы Блокнот (Notepad)

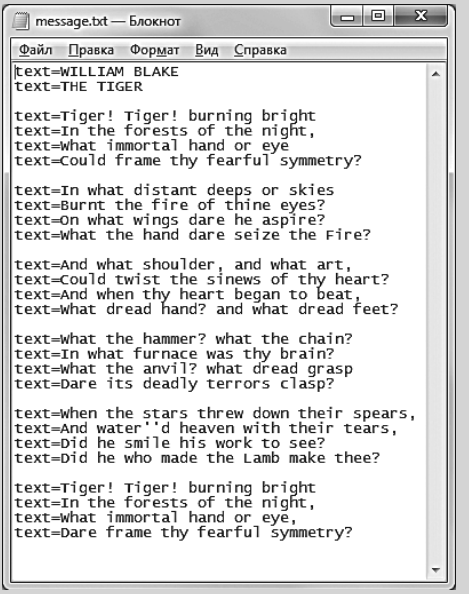


Рис. 11.22. Текстовый файл, необходимый для работы Java-апплета fader26

Используйте этот рисунок как пример при создании вашего текстового файла. Сохраните текстовый файл с именем *mymessage.txt* в папке *testapplet*. Имя текстового файла должно совпадать со значением параметра `Data` в HTML-коде. Теперь откройте вашу страницу в браузере. Апплет должен отображать текст по одной строке за раз (ваш браузер может показать предупреждение о том, что апплет fader26 был создан при использовании более ранней версии Java — просто нажмите кнопку **ОК** или **Продолжить** (Continue)).

Помните, что неудобство использования Java-апплетов состоит в задержке между временем первоначальной загрузки самой веб-страницы и временем, когда апплет начинает выполняться. Посетитель вашей веб-страницы будет видеть пустую область на веб-странице, зарезервированную для апплета, до того как он начнет выполняться.

ЧаВо

ПОЧЕМУ НЕ РАБОТАЕТ МОЙ JAVA-АППЛЕТ?

Если ваш апплет работает некорректно, проверьте следующие пункты:

- Включена ли обработка Java-апплетов в вашем браузере?
- Сохранен ли апплет в папке *testapplet*?
- Сохранен ли апплет с именем *fader26.class* (все буквы должны быть в нижнем регистре)?
- Сохранены ли файлы *java.html* и *mymessage.txt* в папке *testapplet*?

Ресурсы Java-апплетов

Существует много ресурсов как бесплатных, так и коммерческих Java-апплетов во Всемирной паутине. Вот несколько полезных сайтов:

- **www.javaonthebrain.com**
- **www.echoecho.com/freeapplets.htm**

При посещении этих и других ресурсов, посвященных Java, помните, что некоторые Java-апплеты защищены авторским правом. Обязательно получите разрешение на использование апплета от создателя, прежде чем использовать его на своем сайте. Могут быть определенные требования к указанию авторства, к примеру, имени разработчика или ссылки на веб-сайт. Следуйте инструкциям, которые предоставляются вместе с апплетом. Некоторые апплеты бесплатны для использования на личных веб-сайтах, но требуют покупки лицензии при использовании на коммерческом веб-сайте.

ЧаВо

МОЖЕТЕ ПРИВЕСТИ ПРИМЕР ИСПОЛЬЗОВАНИЯ УСТАРЕВШЕГО ЭЛЕМЕНТА `APPLET`?

Конечно! Хотя использование элемента `applet` и не рекомендуется, вам, возможно, встретятся веб-страницы, на которых он еще присутствует. Пример использования тега `<applet>` для отображения Java-апплета приведен ниже:

```
<applet code="fader26.class" width="610" height="30"
alt="Этот Java-апплет отображает сообщение, которое
описывает, для чего можно использовать Java-апплеты.">
```

```
<param name="AppletHome" value="http://www.crosswinds.
net/~fader/" />
<param name="Data" value="message.txt" />
<param name="bgColor" value="#FFFFFF" />
Java-апплеты можно использовать для отображения текста,
управления графикой, игр и многого другого.
<a    href="http://www.oracle.com/ru/technologies/java/
index.html">сайт компании Oracle</a>.
</applet>
```

11.9. Технология JavaScript

JavaScript — это объектно-ориентированный язык сценариев. В JavaScript вы работаете с объектами, ассоциированными с документом веб-страницы: окном, документом, а также с элементами, такими как формы, изображения и ссылки. JavaScript, разработанный компанией Netscape, изначально назывался LiveScript. Когда Netscape объединилась с Sun Microsystems, для внесения изменений в язык его переименовали в JavaScript. JavaScript не такой же, как язык программирования Java. В отличие от Java, JavaScript нельзя использовать для создания самостоятельных программ, запускаемых вне веб-браузера. Код JavaScript можно разместить в отдельных файлах (с расширением *.js*), к которым будет доступ у веб-браузера, но гораздо чаще их встраивают в код веб-страницы напрямую вместе с HTML-кодом. В обоих случаях код JavaScript интерпретируют веб-браузеры. JavaScript считается языком сценариев, выполняющихся на стороне клиента — он выполняется веб-клиентом (браузером), а не веб-сервером. Заметьте, что хотя некоторые веб-серверы (такие как Sun Java System Web Server) могут обрабатывать JavaScript на стороне сервера, язык чаще всего используется для создания скриптов, выполняющихся на стороне клиента.

ЧаВо

РАЗВЕ НЕ ВСЕ БРАУЗЕРЫ ПОДДЕРЖИВАЮТ JAVASCRIPT?

Большинство современных браузеров поддерживают JavaScript. Тем не менее у них есть также возможность отключения JavaScript. Некоторые вспомогательные технологии, такие как программы экранного доступа, могут не поддерживать JavaScript. Вы не може-

те быть уверены в том, что у каждого посетителя вашего веб-сайта разрешен JavaScript. Рекомендуется предложить посетителям альтернативу (простые текстовые ссылки, телефонный номер справочной службы и т. д.), если возможности вашего веб-сайта зависимы от JavaScript.

JavaScript обычно используют для реакции на события, такие как движение мыши, нажатие кнопки мыши и загрузка веб-страницы. Эта технология также часто используется для редактирования и подтверждения данных в элементах HTML-форм, таких как текстовые поля, флажки и переключатели. JavaScript можно использовать для создания всплывающих окон, отображения текущей даты, выполнения вычислений и т. д. На рис. 11.23 показан веб-сайт, использующий JavaScript, чтобы задать отображение текущей даты.

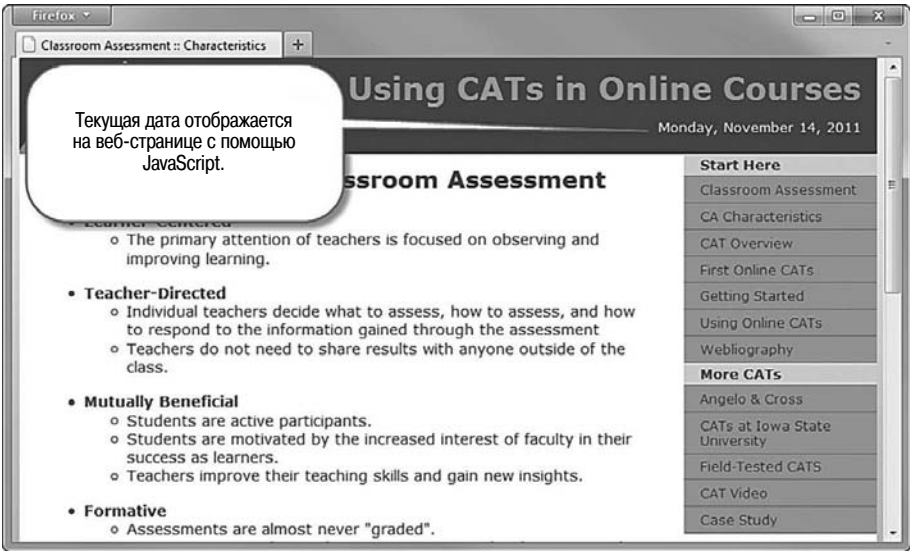


Рис. 11.23. Текущая дата отображается на веб-странице с помощью JavaScript

Вы можете ознакомиться с введением в JavaScript в главе 14. Важной частью работы с JavaScript является управление **объектной моделью документа (DOM).** DOM определяет каждый объект и элемент веб-страницы. Ее иерархическую структуру можно использовать для доступа к элементам страницы и применению стилей. Часть основной модели DOM, встречающейся в большинстве браузеров, показана на рисунке 11.24.

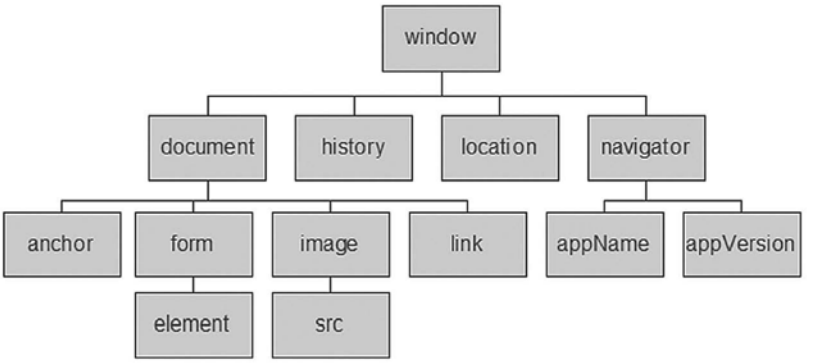


Рис. 11.24. Объектная модель документа (DOM)

Бесплатные ресурсы по технологии JavaScript

Изучению JavaScript можно уделить много времени, но для любого желающего во Всемирной паутине существует множество бесплатных ресурсов готовых скриптов и обучающих руководств JavaScript.

Вот некоторые из сайтов, предлагающих бесплатные скрипты и обучающие руководства:

- **javascript.ru/**
- **www.webmasterwiki.ru/JavaScript**

Когда вы хорошо освоите HTML и CSS, язык JavaScript станет подходящей технологией для последующего обучения. Попробуйте посетить некоторые ресурсы из предложенных выше и приступайте к изучению. Прочитайте главу 14, чтобы больше узнать о JavaScript. Следующий раздел познакомит вас с Ajax, технологией, использующей JavaScript.

11.10. Технология Ajax

Ajax является не отдельной технологией, а комбинацией сразу нескольких. Термин ***Ajax*** расшифровывается как асинхронный JavaScript и XML (Asynchronous JavaScript and XML). Эти технологии сами по себе не новы, но в последнее время их стали использовать вместе для повышения юзабилити сайта и создания интерактивных веб-приложений. Джесс Джеймс Гаррет на сайте Adaptive Path[1] указывается как родона-

[1] www.adaptivepath.com/publications/essays/archives/000385.php

чальник термина «Ajax». Технологии, которые использует Ajax, перечислены ниже:

- стандартно-ориентированный HTML и CSS;
- объектная модель документа (DOM);
- XML (и связанная с ней технология XSLT);
- асинхронное получение данных через использование функции XMLHttpRequest;
- JavaScript.

Вам могут быть неизвестны некоторые из этих технологий. Это вполне нормально на текущем этапе вашей карьеры как веб-разработчика. Пока вы получаете фундаментальные знания об HTML и CSS, и позже можете продолжить обучение, изучив дополнительные веб-технологии. В данный момент вам достаточно знать, что эти технологии существуют, и для чего они могут быть использованы.

Ajax — это часть ***Веб 2.0*** — перехода Всемирной паутины от изолированных статических веб-сайтов к платформам, использующим современные технологии для обеспечения многофункциональных интерфейсов и возможностей социальных сетей для пользователей. Прочитайте статью[1] о Веб 2.0 Тима О'Райли, который был одним из основателей самого этого термина.

Ajax — это техника веб-разработки для создания интерактивных веб-приложений. Вспомните о модели «клиент-сервер», которую мы рассматривали в главах 1 и 9. Браузер делает запрос у сервера (часто вызванный нажатием на ссылку или кнопку отправки данных), а сервер возвращает целую новую веб-страницу для отображения браузером. Ajax передает большую часть обрабатываемых операций клиенту (браузеру), используя JavaScript и XML, и часто использует «фоновые» асинхронные запросы к серверу, чтобы обновить часть отображаемой в браузере страницы, вместо того чтобы заменять всю веб-страницу целиком. Главное то, что при использовании технологии Ajax код JavaScript (который работает в браузере на компьютере клиента) может напрямую общаться с сервером, обмениваясь данными и изменяя части отображаемой веб-страницы, не перезагружая ее целиком. Например, как только посетитель веб-сайта введет почтовый индекс в форму, запустится поиск значения в базе почтовых индексов, и с помощью Ajax автоматически будет выведена информация о городе/области — и все это делается, пока

[1] Перевод части статьи опубликован на странице **www.computerra.ru/think/234100/**

пользователь вводит информацию в форму, еще до того как он нажимает кнопку отправки данных. В результате пользователь воспринимает веб-страницу как более гибко реагирующую и получает впечатление большей интерактивности. Взгляните на страницу обзора свойств CSS[1], чтобы увидеть хороший пример технологии Ajax в действии. Как показано на рис. 11.25, пока вы вводите имя свойства CSS, вам предоставляются подсказки — без обновления самой страницы.

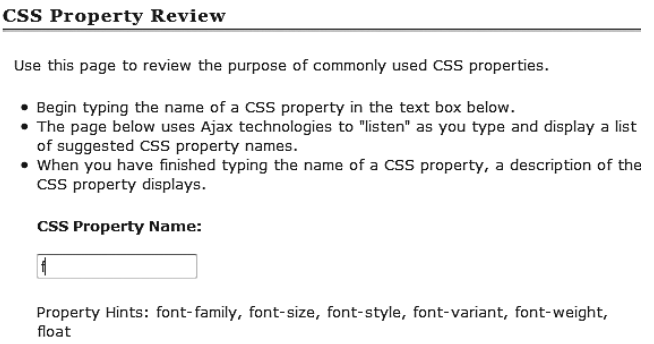


Рис. 11.25. Технологии Ajax используются для обновления страницы, пока пользователь вводит текст

Разработчики используют Ajax для поддержки веб-приложений, которые являются частью Веб 2.0 — фотохостинг Flicker[2], электронная почта Google[3], сервис закладок Delicious[4], поисковая система A9[5] компании Amazon, Windows Live[6] и др.

Ресурсы по технологии Ajax

Технология Ajax сегодня очень популярна, и во Всемирной паутине о ней можно найти множество ресурсов и статей. Ниже указаны некоторые полезные сайты:

- как начать работу с Ajax: **www.alistapart.com/articles/gettingstartedwithajax**
- руководство по Ajax: **www.tizag.com/ajaxTutorial**

[1] webdevfoundations.net/css
[2] www.flickr.com
[3] gmail.google.com
[4] www.delicious.com
[5] www.a9.com
[6] www.live.com

11.11. Элемент canvas

***Элемент* canvas** в спецификации HTML5 конфигурирует динамическую графику. Он дает возможность динамически рисовать и преобразовывать на веб-страницах линии, фигуры, изображения и текст. Если этого недостаточно, элемент canvas также предоставляет способы реагирования на действия, совершаемые пользователями, к примеру, перемещение мыши.

Элемент canvas начинается с тега <canvas> и заканчивается тегом </canvas>. Однако он задается с помощью интерфейса API. Это означает, что для применения элемента canvas необходимы языки программирования или скриптов, такие как JavaScript. API для элемента canvas предоставляет способы создания двумерной (2D) растровой графики, в том числе линий, штрихов, дуг, заливок, градиентов, изображений и текста.

Однако вместо того чтобы рисовать с помощью графического приложения, вы рисуете программным путем, записывая выражения JavaScript. Очень простой пример использования JavaScript для рисования с помощью элемента canvas показан на рис. 11.26 (см. файл *Примеры\Глава_11\canvas.html* на прилагающемся к книге диске).

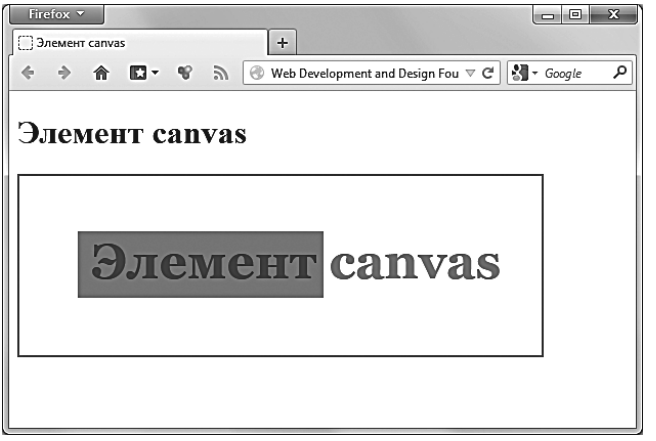


Рис. 11.26. Элемент canvas

Код следующий:

```
<!DOCTYPE HTML>
<HTML lang="en">
```

```
<head>
<title>Элемент canvas</title>
<meta charset="utf-8">
<script type="text/javascript">
function drawMe() {
var canvas = document.getElementById("myCanvas");
if (canvas.getContext) {
var ctx = canvas.getContext("2d");
ctx.fillStyle = "rgb(255, 0, 0)";
ctx.font = "bold 3em Georgia";
ctx.fillText("Элемент canvas", 70, 100);
ctx.fillStyle = "rgba(0, 0, 200, 0.50)";
ctx.fillRect(57, 54, 240, 65);
}
}
</script>
<style>
canvas { border: 2px solid red; }
</style>
</head>
<body onload="drawMe()">
<h1>Элемент canvas</h1>
<canvas id="myCanvas" width="510" height="175"></
canvas>
</body>
</HTML>
```

Не волнуйтесь, если часть кода кажется вам совершенно незнакомой, будто написана на неизвестном языке. Язык JavaScript отличается от CSS и HTML, у него собственный синтаксис и правила. Ниже приведен краткий обзор кода:

- Красный контур был создан путем применения CSS к селектору `canvas`.
- Функция `drawMe()` языка JavaScript вызывается, когда браузер загружает страницу. JavaScript ищет элемент `canvas`, которому

присвоен идентификатор "myCanvas". JavaScript проверяет, поддерживают ли браузеры элемент canvas и, если это так, выполняет следующие действия:

- задает значение 2d методу getContext;
- рисует надпись «Элемент canvas»;
- использует атрибут fillStyle, чтобы задать красный цвет нарисованной надписи;
- использует атрибут font, чтобы задать толщину шрифта, его размер и семейство шрифтов;
- применяет метод fillText, чтобы указать, что текст будет отображаться, а затем добавляет значения x (количество пикселов слева направо) и y (количество пикселов сверху вниз);
- рисует прямоугольник;
- использует атрибут fillStyle, чтобы задать синий цвет рисунка с прозрачностью 50%;
- применяет метод fillRect, чтобы задать значения x (количество пикселов слева направо), y (количество пикселов сверху вниз), ширину и высоту прямоугольника.

Потенциал элемента canvas заключается в том, что его можно использовать для обеспечения интерактивности, не менее сложной, чем разрабатываемая с помощью Adobe Flash.

На момент написания книги все современные браузеры (за исключением Internet Explorer) поддерживали элемент canvas.

С помощью любого браузера, кроме Internet Explorer, просмотрите виртуозные примеры действия элемента canvas, представленные на сайте **www.canvasdemos.com**.

В этом разделе мы очень кратко рассмотрели элемент canvas. Если вам нужны дополнительные примеры и упражнения по работе с этим занимательным элементом, посетите следующие ресурсы:

- увеличение изображения при наведении указателя мыши благодаря элементу canvas HTML5: **www.html5canvastutorials.com**
- шпаргалка по работе с элементом canvas HTML5: **blog.nihilogic.dk/2009/02/HTML5-canvas-cheat-sheet.html**

11.12. Доступность и мультимедиа/интерактивность

Мультимедийность и интерактивность могут помочь в создании неповторимого, запоминающегося впечатления у посетителей вашего веб-сайта. Но помните, что не каждый посетитель вашего сайта сможет использовать эти возможности, поэтому добавьте на сайт следующие элементы:

- Предоставьте ссылки на бесплатную загрузку плагинов, которые используют ваши мультимедийные файлы.
- Текстовые описания и альтернативный контент (такой как субтитры) для аудио- и видеофайлов сделают информацию доступной для людей с проблемами со слухом, а также помогут пользователям с двигательными расстройствами или медленным интернет-соединением.
- Когда вы работаете совместно с разработчиками мультимедийных файлов и программистами для создания Flash-анимации или Java-апплетов для вашего сайта, обязательно попросите учесть возможности для обеспечения доступности контента — доступ с клавиатуры, текстовые описания и т. д. Если вы используете Flash, Java-апплет для навигации по сайту — убедитесь, что к ним можно получить доступ с клавиатуры, и/или создайте простую навигацию из текстовых ссылок в нижней части страницы. Корпорация Adobe предоставляет отличный ресурс для веб-разработчиков, Accessibility Resource Center, по адресу **www.adobe.com/resources/accessibility**.
- Инструкция 2.3.1 стандарта WCAG 2.0 рекомендует не размещать на веб-странице элементов, мигающих чаще трех раз в секунду. Это необходимо, поскольку частые вспышки в некоторых случаях могут привести к возникновению эпилептических припадков. Возможно, вам придется сотрудничать с разработчиком мультимедийного контента, чтобы убедиться, что динамические эффекты находятся в пределах безопасного диапазона.
- Если вы используете JavaScript, помните, что у некоторых пользователей JavaScript может быть отключен, или, возможно, они не могут управлять мышью. Желательно, чтобы ваш сайт функционировал на необходимом уровне, даже если браузер вашего посетителя не поддерживает Java-Script. У сайта, использующего Ajax для отображения части окна браузера, могут возникнуть проблемы при получе-

нии к нему доступа посредством программ экранного доступа или текстовых браузеров. Важность тестирования невозможно переоценить. Консорциум W3C разработал стандарт ARIA (Accessible Rich Internet Applications — доступные активные веб-приложения), который является протоколом, поддерживающим доступность динамического содержимого и контента, связанного со скриптами, такого как приложения, созданные с помощью технологии Ajax. На момент написания этой книги стандарт ARIA поддерживался не всеми популярными браузерами. Вы можете узнать больше о стандарте ARIA по адресу **w3.org/WAI/intro/aria.php**.

Когда вы проектируете сайт, не забывая об обеспечении доступности мультимедийных и интерактивных элементов, вы помогаете людям с физическими недостатками, низкоскоростным доступом в Интернет и тем, у кого не установлены необходимые плагины в браузере. Тем не менее, если мультимедийные и/или интерактивные возможности, использованные на вашей странице, не могут удовлетворять требованиям доступности, подумайте о создании отдельной текстовой версии страницы. Стандарт ГОСТ-52872-2007 требует предоставления такой альтернативы.

Глава 12
ЭЛЕКТРОННАЯ КОММЕРЦИЯ

Цели главы

В этой главе вы узнаете следующее:

- определение электронной коммерции;
- выгоды и риски электронной коммерции;
- описание бизнес-моделей электронной коммерции;
- безопасность и шифрование данных в электронной коммерции;
- электронный обмен данными (EDI, Electronic Data Interchange);
- тенденции и прогнозы электронной коммерции;
- вопросы, касающиеся электронной коммерции;
- возможные варианты обработки заказа и оплаты.

Электронная коммерция — это продажа и покупка товаров и услуг во Всемирной паутине. Будь это деятельность Бизнес-Бизнесу, Бизнес-Потребителю или Потребитель-Потребителю, веб-сайты, поддерживающие электронную коммерцию, есть везде. Эта глава представляет собой обзор данной темы.

12.1. Что такое электронная коммерция?

Формальное определение ***электронной коммерции*** — это объединение коммуникаций, управления данными и технологий безопасности, которое позволяет отдельным людям и организациям обмениваться информацией, относящейся к продаже товаров и услуг. Главные функции электронной коммерции включают в себя покупку товаров, продажу товаров и оформление финансовых транзакций через Всемирную паутину.

Преимущества электронной коммерции

Существует ряд преимуществ как для бизнеса, так и для покупателя, когда речь идет об электронной коммерции. Некоторыми из многих преимуществ для бизнеса являются:

- **Снижение затрат.** Онлайн-бизнес может работать 24 часа в сутки без накладных расходов, которые вынужден нести традиционный магазин. Многие бизнесы, прежде чем перейти к электронной коммерции, запускают веб-сайт. После этого они добавляют функции электронной коммерции на свой сайт, сайт становится источником дохода и, чаще всего, оправдывает расходы на него в самые короткие сроки.
- **Увеличение уровня удовлетворенности покупателя.** Компания может использовать веб-сайт для улучшения своего уровня коммуникации с клиентами и увеличения уровня удовлетворенности покупателя. Сайты электронной коммерции нередко содержат страницу Часто Задаваемых Вопросов (ЧаВо). Доступность службы работы с клиентами по электронной почте, на форумах или даже в онлайн-чате (см. **liveperson.com**) может улучшить отношения с клиентами.
- **Более эффективное управление данными.** В зависимости от уровня автоматизации, сайты электронной коммерции могут выполнять верификацию и авторизацию банковских карт, обновлять информацию о количестве товара на складе, а также обрабатывать формы выполнения заказа, таким образом управляя данными организации более эффективно.
- **Потенциально повышенные продажи.** Магазин системы электронной коммерции открыт 24 часа в сутки, семь дней в неделю и доступен каждому человеку на планете (при условии доступа в Интернет), что делает уровень его продаж потенциально выше, чем у традиционных магазинов.

Электронная коммерция выгодна не только компаниям, покупатели тоже получают свои преимущества, среди которых можно привести следующие:

- **Удобство.** Потребитель может совершать покупки в любое время суток. Не нужно тратить время на поездки в магазин. Некоторые покупатели предпочитают совершать покупки во Всемирной паутине вместо выбора товаров по каталогам, потому что это позволяет им увидеть дополнительные изображения и присоединится к форумам, обсуждающим те или иные продукты.
- **Легкость в сравнении магазинов.** Не нужно ездить из магазина в магазин, чтобы сравнивать цены на товары. Покупатели могут легко просмотреть интернет-магазины и сравнить цены и достоинства товаров.

- **Более широкий выбор продуктов.** Так как во Всемирной паутине удобнее посещать магазины и сравнивать продукты, покупатели получают более широкий выбор товаров для приобретения.

Как видите, электронная коммерция несет в себе множество преимуществ как для бизнеса, так и покупателя.

Риски электронной коммерции

Любая бизнес-операция заключает в себе определенный уровень рисков, и электронная коммерция не исключение. Возможные факторы риска перечислены ниже:

- **Падение продаж при технических проблемах сайта.** Если ваш сайт недоступен или ваши формы обработки заказов не работают, покупатель может и не вернуться на сайт. Всегда важно иметь удобный для пользователя надежный веб-сайт, но когда дело касается электронной коммерции, надежность и легкость использования становятся критически важными факторами успеха вашего бизнеса.
- **Мошеннические транзакции.** Мошеннические покупки с помощью банковских карт или фиктивные заказы, помещенные на сайт конкурентами (или тринадцатилетними подростками с избытком свободного времени), — это риски, с которыми придется столкнуться любому бизнесу.
- **Нерасположенность покупателей.** Хотя все больше и больше покупателей стремятся совершать покупки во Всемирной паутине, целевая аудитория вашего бизнеса, возможно, не относится к ним. Тем не менее, предлагая такие средства поощрения, как бесплатная доставка или политика возврата «без указания причины», ваш бизнес все же может привлечь этих покупателей.
- **Повышенная конкуренция.** Так как уровень расходов интернет-магазина гораздо ниже, чем у традиционных магазинов, только открывшаяся компания может быть настолько же впечатляющей, как и давно утвердившаяся организация, если ее веб-сайт будет выглядеть профессионально. Так как гораздо легче войти на рынок через электронную коммерцию, ваш магазин столкнется с высоким уровнем конкуренции.

Не только бизнесу приходится сталкиваться с рисками, связанными с электронной коммерцией. Покупатели могут столкнуться со следующими рисками:

- **Вопросы безопасности.** Позже в этой главе вы узнаете, как определять, осуществляется ли соединение с веб-сайтом через протокол SSL (Secure Sockets Layer, уровень защищенных сокетов), служащий для шифрования и обеспечения безопасности информации. Обычные пользователи могут не уметь определять, использует ли сайт этот метод шифрования, и опасаться помещать заказ для оплаты банковской картой. Другой, возможно, более важный вопрос безопасности: что сайт сделает с информацией после того, как она поступит к нему через Интернет. Защищена ли его база данных? Защищены ли резервные копии базы данных? На эти вопросы сложно ответить. Лучшим выходом будет совершать покупки только на сайтах, имеющих хорошую репутацию.
- **Вопросы конфиденциальности.** Многие сайты публикуют положения своей политики конфиденциальности. Они описывают, что сайт будет (или не будет) делать с полученной от клиента информацией. Некоторые сайты используют эту информацию только в целях внутреннего маркетинга. Другие сайты продают информацию внешним компаниям. Веб-сайты со временем могут поменять (и меняют) политику конфиденциальности. Покупатель может не решиться на покупку в интернет-магазине из-за опасности недостаточной конфиденциальности.
- **Покупка на основе фотографий и описаний.** Ничто не заменит возможности дотронуться и подержать вещь перед покупкой. Покупатель рискует приобрести продукт, которым он будет недоволен, так как он принимает решение о покупке на основе фотографий и текстовых описаний. Если интернет-магазин поддерживает политику возврата, покупатель будет более уверен в покупке.
- **Возвраты.** Чаще всего вернуть вещь в интернет-магазин сложнее, чем в традиционный. Покупатель может опасаться подобных трудностей.

12.2. Бизнес-модели электронной коммерции

Как бизнес, так и покупатели вовлечены в электронную коммерцию. Существуют четыре основные модели бизнеса электронной коммерции: Бизнес–Потребителю, Бизнес–Бизнесу, Потребитель–Потребителю и Бизнес–Правительству[1].

[1] Существуют также модели: G2B (Правительство–Бизнесу) и G2C (Правительство–Гражданину). — *Прим. пер.*

- **Бизнес–Потребителю (B2C, business-to-consumer).** Большинство продаж модели Бизнес–Потребителю происходит в онлайн-магазинах. Некоторые из них, такие как Ozon.ru (**ozon.ru**), существуют только во Всемирной паутине. Другие — это электронно-традиционные магазины от известных традиционных брендов, таких как М.Видео (**www.mvideo.ru**).

- **Бизнес–Бизнесу (B2B, business-to-business).** Электронная коммерция между двумя бизнесами часто принимает форму обмена информацией о сети поставщиков среди продавцов, партнеров и клиентов бизнеса. Электронный обмен данными (Electronic Data Exchange, EDI) тоже находится в этой категории.

- **Потребитель–Потребителю (C2C, consumer-to-consumer).** Люди могут торговать друг с другом во Всемирной паутине. Наиболее частый формат такой торговли — это аукцион. Самый известный сайт-аукцион — это eBay (**ebay.com**), который основали в 1995 году.

- **Бизнес–Правительству (B2G, business-to-government).** Бизнес торгует с правительством через Интернет. Примером модели Бизнес–Правительству могут служить системы электронных госзакупок.

Бизнес начал обмениваться информацией в электронном виде задолго до появления Всемирной паутины, используя Электронный обмен данными.

12.3. Электронный обмен данными

Электронный обмен данными (EDI, Electronic Data Interchange) — это передача данных между компаниями по сети. Это облегчает обмен стандартными бизнес-документами, включая заказы на поставку и инвойсы. Электронный обмен данными давно не новинка; он существует с 1960 года. Организации, которые используют стандарты EDI для передачи информации, называются торговыми партнерами.

Комитет по аккредитованным стандартам X12 (Accredited Standards Committee, ASC X12) создан Американским Национальным Институтом Стандартов (ANSI, American National Standards Institute) для разработки и поддержания стандартов EDI. Эти стандарты включают в себя транзакционные наборы для стандартных бизнес-форм, таких как заявки и инвойсы. Это позволяет бизнесу уменьшить бумажную работу и передавать данные с помощью цифровых каналов связи.

Сообщения стандарта EDI помещаются в транзакционные наборы. Транзакционный набор состоит из заголовка, одного или нескольких информационных сегментов, которые являются строками информационных элементов, отделенных разделителями, и трейлера. Более новые технологии, такие как XML и веб-службы, позволяют торговым партнерам использовать виртуальные неограниченные возможности для настройки их параметров обмена информацией через Интернет.

Теперь, когда вы знакомы с возможностями электронной коммерции и основными бизнес-моделями, вы, наверное, хотите узнать, где же лучше всего можно заработать. Следующий раздел рассматривает некоторую статистику, которая касается электронной коммерции.

12.4. Статистика электронной коммерции

Возможно, вы удивитесь, узнав, что наибольшая прибыль извлекается в секторе B2B. По расчетам исследовательского агентства Datainsight[1], в 2011 году объем рынка интернет-торговли в России достиг 310 млрд рублей. К 2015 году, по прогнозам[2] Datainsight, объем рынка онлайн-торговли в России вырастет более чем в 2 раза, покупки в сети будут совершать на сумму, примерно равную 710 млрд рублей в год.

Вам также может быть интересно, что люди покупают во Всемирной паутине. Отчет, составленный компанией RUметрика[3] установил, что четырьмя наиболее популярными категориями розничных продаж во Всемирной паутине по состоянию на декабрь 2010 года были следующие:

- Книги (в том числе электронные) (13,7%);
- Одежда, обувь (13,2%);
- Бытовая техника (12,3%);
- Аудио-, видео-, цифровая техника (10,8%);
- Компьютеры и комплектующие (10,3%);
- Туристические путевки, бронирование (10,0%).

Теперь, когда вы знаете, что лучше всего продается во Всемирной паутине, кто ваши потенциальные онлайн-покупатели? Как правило, самая активная часть аудитории интернет-торговли — это молодые люди в возрасте 25–35 лет. Исследование, проведенное компанией

[1] www.datainsight.ru
[2] www.slideshare.net/Data_Insight/data-insight-bonline2012
[3] rumetrika.rambler.ru

RUметрика[1] выявило, что большинство потребителей готовы оплачивать покупки наличными и наложенными платежами (свыше 50% по данным на 2010 год.). Наиболее популярны покупки стоимостью от 500 до 3000 рублей и от 30 000 до 100 000 рублей — 35 и 26% соответственно по данным на 2010 год.

12.5. Проблемы, касающиеся электронной коммерции

Ведение бизнеса во Всемирной паутине тоже несет в себе определенные проблемы. Вот некоторые наиболее частые из них:

- **Интеллектуальная собственность.** В последнее время было много споров касательно прав интеллектуальной собственности и доменных имен. *Киберсквоттингом* называют регистрацию доменных имен, являющихся торговой маркой какой-либо организации, с целью получить прибыль от продажи доменного имени этой организации. Ассоциация по присвоению имен и номеров [портов] Интернета (ICANN, Internet Corporation for Assigned Names and Numbers) спонсирует Единую политику рассмотрения споров о доменных именах (Uniform Domain Name Dispute Policy)[2], которую можно использовать для борьбы с киберсквоттерами.
- **Безопасность.** Безопасность — это постоянная проблема во Всемирной паутине. Распределенные атаки типа DDoS не раз отключали популярные сайты. Некоторые из этих атак совершались малолетними горе-программистами, которым, как можно выразиться, больше нечего делать, кроме как сеять хаос во Всемирной паутине.
- **Мошенничество.** Мошеннические веб-сайты, требующие номера банковских карт без доставки самого продукта или как-то иначе обманывающие покупателя, являются вполне понятным источником опасений покупателей.
- **Международная коммерция.** У веб-сайтов, нацеленных на всемирную аудиторию, есть дополнительные поводы переживать. Если вебсайт должен работать на нескольких языках, существует ряд программ автоматического перевода[3,4] и компаний, которые предлагают

[1] rumetrika.rambler.ru/review/26/4622
[2] www.icann.org/udrp/udrp.htm
[3] www.ru.conveythis.com
[4] www.systranlinks.com

услуги перевода веб-сайтов под заказ[1,2]. Помните, что графический интерфейс пользователя (GUI, Graphical User Interface), корректно работающий на русском языке, может работать неправильно на других языках. Например, аналогичные слова и фразы на немецком языке зачастую длиннее. Если вашему графическому интерфейсу не хватает пространства в русской версии сайта, как он будет выглядеть в немецкой?

Как ваши международные покупатели будут платить вам? Если вы принимаете банковские карты, конвертацию валюты проведет компания, работающая с банковскими картами. Как насчет культуры вашей целевой международной аудитории? Вы изучили культуру целевых стран и убедились в том, что ваш сайт привлекателен и не выглядит для представителей этих стран оскорбительным или неприятным? Другим вопросом, относящимся к международной коммерции, является стоимость пересылки и возможность доставки в отдаленные пункты назначения.

Теперь, когда вы знакомы с концепцией электронной коммерции, давайте рассмотрим методы шифрования и безопасность. Следующий раздел ознакомит вас с методами шифрования, протоколом SSL и электронными сертификатами.

12.6. Безопасность в электронной коммерции

Шифрование

Шифрование обеспечивает конфиденциальность данных как внутри организации, так и при передаче в Интернете. ***Шифрование*** — это преобразование данных в нечитаемую форму, которую называют ***зашифрованным текстом***. Люди, не обладающие доступом к этой информации, понять зашифрованный текст не смогут. ***Расшифровка*** — это процесс преобразования зашифрованного текста в обычный или ***открытый текст*** таким образом, чтобы он приобрел понятную человеку форму. Процесс шифрования и расшифровки требует алгоритм и ключ.

Шифрование в Интернете важно потому, что информация в пакете данных может быть перехвачена в то время, пока она путешествует по

[1] otkrmir.ru/local/
[2] worldlingo.com

среде передачи данных. Если хакер или конкурент по бизнесу перехватит зашифрованный пакет, он не сможет использовать информацию (такую как номер банковской карты или бизнес-стратегию), потому что ее невозможно будет прочитать.

В Интернете широко используется несколько типов шифрования, включая ***симметричное шифрование с секретным ключом*** и ***асимметричное шифрование с открытым ключом***.

Симметричное шифрование

Симметричное шифрование, показанное на рис. 12.1, также называют шифрованием с единым ключом, потому что и шифрование, и расшифровка используют один и тот же ключ. Так как ключ нужно держать в тайне от других, оба — и отправитель, и получатель должны знать ключ, прежде чем связываться, используя шифрование. Преимущество симметричного шифрования — скорость.

Рис. 12.1. Симметричное шифрование использует единый ключ

Асимметричное шифрование. Асимметричное шифрование также называют шифрованием с открытым ключом, потому что он не является секретным. Вместо этого одновременно создаются *два* ключа. Эта пара ключей содержит открытый и секретный ключ. Секретный и открытый ключи так математически соотносятся, что вряд ли кто-то сможет разгадать второй ключ, даже зная один из них. Только открытый ключ может расшифровать сообщение, зашифрованное секретным ключом, и только секретный ключ может расшифровать сообщение, зашифрованное открытым ключом (рис. 12.2). Открытый ключ можно найти в электронном сертификате (больше об этом позднее). Секретный ключ нужно держать в секрете и безопасности. Он хранится на веб-сервере (или другом компьютере) владельца ключа. Асимметричное шифрование гораздо медленнее, чем симметричное.

Рис. 12.2. Асимметричное шифрование использует пару ключей

Целостность

Методы шифрования, описанные выше, помогают содержать сообщения в секрете. Тем не менее понятие безопасности в электронной коммерции включает в себя убежденность в том, что сообщения не изменятся и не повредятся во время передачи. Если сообщение не повредилось в процессе передачи, его называют ***целостным***. ***Хеш-функция*** дает вам возможность убедиться в целостности сообщения. Хеш-функция, или хеш-алгоритм, преобразует строку символов в обычно более короткое значение или ключ фиксированной длины, который называется ***дайджест сообщения***, и представляет собой первоначальную строку.

Эти методы безопасности — особенно техники симметричного и асимметричного шифрования — используются как часть протокола SSL, технологии, которая помогает обезопасить торговлю во Всемирной паутине. Следующий раздел познакомит вас с этой технологией.

Протокол безопасных соединений SSL

Протокол безопасных соединений SSL (Secure Sockets Layer, уровень защищенных сокетов) — это криптографический протокол, позволяющий обмениваться конфиденциальной информацией через сети

общего доступа. Он разработан компанией Netscape и используется для шифрования информации, передаваемой между клиентом (обычно веб-браузером) и веб-сервером. Протокол SSL использует и симметричное, и асимметричное шифрование.

Протокол SSL обеспечивает безопасность сообщения между клиентом и сервером, используя следующие возможности:

- цифровые сертификаты для аутентификации сервера и (по необходимости) клиента;
- симметричное шифрование «сеансовым ключом» для группового шифрования;
- шифрование открытым ключом для передачи сеансового ключа;
- дайджест сообщения (хеш-функция для подтверждения целостности передачи).

Вы можете узнать, использует ли веб-сайт протокол SSL, по адресной строке браузера — вместо протокола http там появится протокол https. Кроме того, браузеры отображают значок замка, когда используется протокол SSL, как показано на рис. 12.3.

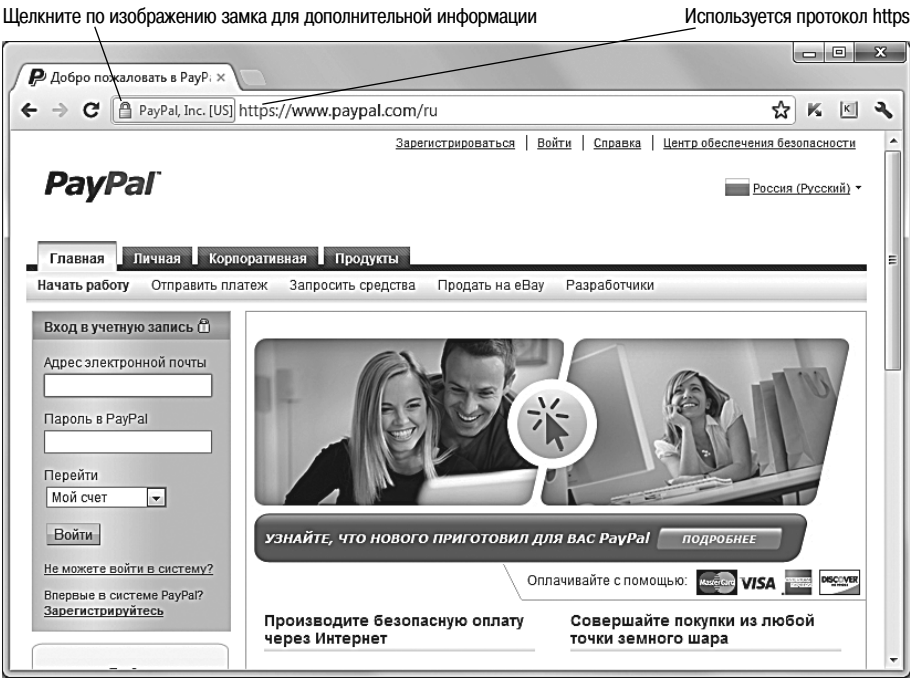


Рис. 12.3. Браузер Google Chrome указывает на использование протокола SSL

ЧаВо

КОГДА НЕКОТОРЫЕ ВЕБ-САЙТЫ ОТОБРАЖАЮТСЯ В БРАУЗЕРЕ, В АДРЕСНОЙ СТРОКЕ ПОЯВЛЯЕТСЯ ЦВЕТНАЯ СТРОКА. В ЧЕМ ДЕЛО?

Если веб-сайт отображает цветную строку в адресной строке браузера в дополнение к значку замка в строке состояния, это значит, что он использует технологию расширенной проверки — **Extended Validation SSL (EV SSL)**.

Использование EV SSL означает, что бизнес прошел более строгую проверку, чтобы получить свой цифровой сертификат, включая подтверждение того, что:

- заявитель является владельцем домена;
- заявитель работает на организацию;
- заявитель уполномочен обновлять веб-сайт;
- организация законна, а место расположения предприятия подтверждено.

Цифровой сертификат

Протокол SSL позволяет двум компьютерам безопасно сообщаться с помощью отправки цифрового сертификата для аутентификации.

Цифровой сертификат — это форма асимметричного ключа, которая также содержит информацию о сертификате, владельце и удостоверяющем центре сертификата. Содержание цифрового сертификата включает в себя следующие элементы:

- открытый ключ;
- дата вступления сертификата в силу;
- дата истечения срока действия сертификата;
- детали о Центре Сертификации, удостоверяющем центре сертификата;
- детали о владельце сертификата;
- дайджест содержания сертификата.

VeriSign[1] и Entrust[2] — широко известные Центры Сертификации.

[1] verisign.com
[2] www.entrust.net

Чтобы получить сертификат, вы запрашиваете его у Центра Сертификации и платите регистрационный взнос. Центр Сертификации подтверждает вашу личность, выдает вам сертификат и предоставляет вам пару из открытого и секретного ключа.

Вы храните ваш сертификат в своем программном обеспечении — таком как веб-сервер, веб-браузер или приложение по работе с электронной почтой. Центр Сертификации делает ваш сертификат публично известным.

ЧаВо

НУЖНО ЛИ МНЕ ПОДАВАТЬ ЗАЯВКУ НА СЕРТИФИКАЦИЮ?

Если вы принимаете любую конфиденциальную информацию на вашем веб-сайте, такую как номера банковских карт, вам нужно использовать протокол SSL. Одним из вариантов станет посещение Центра Сертификации (такой как VeriSign или Thawte по адресу **www.thawte.com**) и подача заявки на свой собственный сертификат. Возможно, придется немного подождать, кроме того, нужно будет оплатить ежегодный взнос.

Как альтернативный вариант, ваш хостинг-провайдер может позволить вам использовать его собственный сертификат. Обычно веб-хост назначает вам папку на своем защищенном сервере. В этой папке вы помещаете веб-страницы (и связанные с ней файлы, например изображения), которые требуют безопасной обработки. При перенаправлении на эти веб-страницы используется протокол https вместо http в ваших абсолютных ссылках. Свяжитесь с вашим хостинг-провайдером, чтобы узнать больше.

Протокол SSL и цифровые сертификаты

Процесс аутентификации SSL включает несколько шагов. Веб-браузер и веб-сервер сначала проходят стадии подтверждения установления связи, обмениваясь информацией о сертификатах и ключах сервера.

Как только рукопожатие выполнено, веб-браузер зашифровывает единый секретный ключ (симметричный ключ), который будет использован при дальнейшей коммуникации. Начиная с этого шага, вся информация зашифровывается через секретный ключ. Таблица 12.1 демонстрирует этот процесс.

Таблица 12.1. Обзор процесса шифрования протокола SSL

Браузер	→	«Приветствие»	→	Сервер
Браузер	←	«Приветствие» + сертификат сервера	←	Сервер
Браузер	←	Секретный ключ сервера используется для шифровки сообщения. Только открытый ключ может расшифровать сообщение	←	Сервер
Теперь браузер подтвердил подлинность веб-сервера. Он получил сертификат Центра Сертификации (ЦС), который выдал сертификат серверу. Потом браузер расшифрует дайджест сертификата сервера. Браузер сравнивает два дайджеста и проверяет дату истечения срока действия сертификата. Если он действенен, совершается следующий шаг				
Браузер	→	Браузер создает сеансовый ключ и шифрует с помощью открытого ключа сервера	→	Сервер
Браузер	←	Сервер посылает сообщение, зашифрованное сеансовым ключом	←	Сервер
Все будущие передачи между браузером и сервером будут зашифрованы сеансовым ключом				

Теперь вы понимаете, как работает протокол SSL для защиты целостности информации в Интернете, включая информацию, которой обмениваются при совершении транзакций, связанных с электронной коммерцией.

12.7. Процессы обработки заказа и его оплаты

В электронной коммерции типа B2C (Бизнес–Потребителю) продаваемые товары отображаются в онлайн-каталоге. На больших сайтах страницы этих каталогов создаются динамически с использованием скриптов, выполняющихся на стороне сервера, для доступа к базам данных. У каждого товара обычно есть кнопка «Купить» или «Добавить в корзину». Выбранные товары помещаются в виртуальную корзину покупок. Когда посетитель закончил выбирать покупки, он нажимает кнопку или изображение-ссылку с надписью вида «Подтвердить заказ» или «Оформить заказ». На этой стадии товары в корзине покупок обычно отображаются на веб-странице с формой заказа.

Обеспечение безопасности достигается посредством использования протокола SSL. После размещения заказа вы можете оплатить товар или услугу одним из нескольких способов; методы оплаты, которые называют моделями оплаты, — это наличные, чек, а также банковские карты.

Модель оплаты наличными

Модель оплаты наличными наиболее трудна в реализации — как вы пошлете наличные через компьютер? Вы не посылаете их. Вы используете электронные деньги. Вы покупаете электронные деньги у банка и переводите их на свой электронный кошелек. Перевод средств мгновенен. Среди сервисов, предоставляющих такие услуги, можно отметить Яндекс.Деньги[1] и WebMoney[2]. Стоит отметить, что в России наиболее популярны способы оплаты наличными (курьеру), наложенным платежом или с помощью сервисов электронных денег (источник: компания RUметрика).

Модель оплаты чеком

При использовании ***модели оплаты чеком*** покупатель выписывает электронный чек для совершения покупки. Как и в случае с обычными чеками, доступность денежных средств должна быть подтверждена и они не передаются немедленно. Один из продавцов такого рода услуг — сервис Paymer[3].

Модель оплаты банковской картой

Обработка оплаты банковской картой — очень важный компонент веб-сайта, работающего по системе электронной коммерции. Денежные средства покупателя нужно перевести в банк продавца. Чтобы принимать расчет банковскими картами, владелец сайта должен подать заявку на создание специального ***аккаунта продавца***, который должны одобрить. Аккаунт продавца — это соглашение между бизнесом и банком, которое позволяет вам принимать оплату заказов посредством банковской карты. Вам также может понадобиться проверка подлинности банковской карты в реальном времени с помощью торгового шлюза, или третьей стороны, такой как Authorize.Net[4]. Существуют сервисы, предоставляющие недорогие аккаунты продавца, такие как PayPal[5]. Изначально предназначавшийся для расчетов по банковским картам между потребителями, сегодня PayPal предлагает услуги покупок с помощью

[1] money.yandex.ru
[2] www.webmoney.ru
[3] www.paymer.ru
[4] www.authorizenet.com
[5] www.paypal.com

банковских карт и услугу корзины покупок для владельцев коммерческих веб-сайтов. Стоит отметить, что в России способ оплаты товаров и услуг остается мало популярным из-за сомнения пользователей в обеспечении конфиденциальности передаваемых данных (источник: компания RUметрика).

Модель оплаты смарт-картой

Модель оплаты смарт-картой (также называемой чиповой картой) популярна в Европе, Австралии и Японии. Смарт-карта похожа на обычную карту с магнитной полосой, но вместо магнитной ленты в нее встроен микропроцессор. Смарт-карту вставляют в небольшое устройство для чтения данных с микропроцессора, в остальном процедура применения не отличается от карт с магнитной полосой.

Модель оплаты с помощью мобильного телефона

Компания Juniper Research прогнозирует[1] значительный рост числа мобильных платежей и утверждает, что к 2015 году 2,5 млрд человек будут использовать мобильные устройства для совершения платежа того или иного типа. ***Мобильный платеж*** — это электронная оплата товаров или услуг с помощью мобильного устройства, например, смартфона. Новая тенденция заключается в использовании технологии «коммуникации ближнего поля» (Near Field Communication (NFC)), которую характеризуют[2] как вариант бесконтактной передачи данных.

По данным журнала *The Wall Street Journal*, Google заключил договора с несколькими компаниями, выпускающими кредитные карты, о том, чтобы использовать технологию NFC в мобильных устройствах на платформе Android, что позволило бы клиентам при совершении покупки просто провести мобильным телефоном по устройству считывания на кассе. Приложение Google Wallet[3], работающее в мобильных устройствах на платформе Android с технологией NFC, может эффективно «Превратить ваш телефон в кошелек».

Конкуренцию Google в области мобильных платежей составляет мобильный кошелек ISIS, PayPal и цифровой бумажник Visa. Стоит ожидать дальнейших разработок этой новой технологии.

[1] www.ecommercetimes.com/story/72807.html
[2] www.3dnews.ru/editorial/625196/
[3] www.google.com/wallet

12.8. Варианты решений интернет-магазинов

Вы, наверное, совершали покупки в интернет-магазинах, и в одних все было просто, а в других сложнее. Основная проблема сайтов электронной коммерции — брошенные корзины покупателей, которые начинают выбирать товары, но так и не оформляют заказ. В этом разделе рассматриваются варианты решений интернет-магазинов и типы корзин покупателя. Для владельцев бизнеса и веб-разработчиков доступно множество различных вариантов оформления интернет-магазинов. Они разнятся от простого мгновенного онлайн-каталога на базе другого веб-сайта до создания собственной системы электронной коммерции. Этот раздел рассматривает некоторые из них.

Мгновенный онлайн-каталог

Вы просто поставляете товары — ***мгновенный онлайн-каталог*** делает все остальное. Нет нужды устанавливать программное обеспечение. Все, что вам нужно, — это использовать веб-браузер для указания адреса вашего виртуального магазина. Вы используете шаблон, предоставленный онлайн-каталогом, выбираете его свойства, настраиваете его и добавляете в него свои товары — загружаете изображения, названия, цены и описания товаров. У этого варианта есть свои недостатки. Вы ограничены тем количеством шаблонов, которые предоставляет сервис создания онлайн-каталога. Количество продуктов, которые вы продаете, тоже может быть ограничено. Ваш магазин может «по впечатлению и ощущениям» быть идентичным другим мгновенным онлайн-каталогам этого сервиса. Тем не менее у этого варианта низкие затраты, низкий уровень риска, и он может быть весьма удобным для малого бизнеса с ограниченными техническими знаниями или возможностями. Сервисы онлайн-каталогов часто предоставляют коммерческий счет и систему автоматической оплаты заказа.

Некоторые онлайн-каталоги бесплатны, но ограничивают уровень услуг или количество товаров. Другие платны и могут взимать плату за хостинг, за обработку заказов и ежемесячный платеж. Во Всемирной паутине можно найти много сайтов, предоставляющих сервис онлайн-каталогов, среди которых InSales[1], AllTrades.ru[2] и др.

[1] www.insales.ru
[2] www.alltrades.ru

Готовое программное обеспечение системы электронной коммерции

При выборе этого варианта приобретается программное обеспечение со стандартным набором возможностей системы электронной коммерции, после чего оно устанавливается на ваш веб-сервер и настраивается под ваши нужды. Многие хостинг-провайдеры предлагают такое программное обеспечение, которое обычно включает в себя корзину покупателя, систему обработки заказов и, по желанию, систему обработки оплаты заказа по банковской карте. ***Программное обеспечение системы электронной коммерции*** обеспечивает вас онлайн-каталогом, который ваши посетители могут просматривать, добавлять из него товары в свою корзину и оформлять заказ через форму заказа, когда они готовы совершить покупку. Среди популярных систем электронной коммерции, которые предлагают хостинг-провайдеры, можно назвать AgoraCart[1], osCommerce[2], ZenCart[3], 1С-Битрикс[4], CS-cart[5] и Mercantec SoftCart[6].

Магазин «под заказ»

Создание крупномасштабного веб-сайта для ведения электронной коммерции полностью с нуля обычно требует опыта, времени и немалого бюджета! Преимущество состоит в том, что вы получаете именно то, что вам нужно. Программные инструменты разработки для изготовляемого под заказ магазина могут включать в себя такие приложения, как Adobe Dreamweaver, Microsoft Visual Studio.NET, Adobe ColdFusion, WebSphere Commerce Studio компании IBM, систему управления базами данных (СУБД) и CGI или другой стандарт скриптов, выполняющихся на стороне сервера. Вариант магазина «под заказ» также может потребовать ***коммерческий сервер***, который является улучшенной версией веб-сервера с поддержкой некоторых коммерческих операций. Возможные варианты: WebSphere Commerce Studio компании IBM и Commerce Server компании Microsoft.

Бюджетный магазин «частично под заказ»

Если размах вашего электронного предприятия небольшой, но вы хотите избежать клонирования сайта, которым грешат мгновенные

[1] agoracart.com
[2] oscommerce.com
[3] zencart.com
[4] www.1c-bitrix.ru
[5] www.cs-cart.com
[6] www.mercantec.com

онлайн-каталоги, возможно, вам стоит рассмотреть другие варианты. Они включают в себя приобретение готовой системы электронной коммерции и системы обработки заказов, наем сторонней компании, такой как PayPal, а также покупку дополнительных модулей электронной коммерции для популярных инструментов верстки веб-страниц.

Во Всемирной паутине существует немало бесплатных систем электронной коммерции. Альтернативные решения можно найти на сайтах ASPCode.net[1], PHP Resource[2] или Mal[3].

Также вы можете найти примеры на сайтах **www.oscommerce.ru**, **virtuemart.ru**, или **www.magentocommerce.com/ru** или воспользоваться поисковыми системами, чтобы узнать об альтернативных вариантах. Уровень сложности и функциональность таких систем могут различаться. У каждого веб-сайта есть инструкции и документация, сопровождающие их продукт. Некоторые сервисы могут требовать от вас регистрации и предоставлять вам специальный HTML-код. Другие могут потребовать загрузить и установить скрипт на ваш собственный веб-сервер.

Сервис PayPal предлагает любому бизнесу свою систему корзины покупателя и верификации платежей по очень низким ценам. Сервис генерирует код, который вы должны поместить на ваши веб-страницы, чтобы разместить на них интерфейс системы PayPal. Вам нужно просто скопировать и вставить его.

Сервис Google Checkout[4] предлагает различные варианты добавления компонента электронной коммерции на веб-сайт, в том числе специальных кнопок, подсказок для корзины покупателя, а также гаджета оформления заказов.

Некоторые дополнения, или расширения, приложения Adobe Dreamweaver предоставляют функционал корзины покупателя. Одним из простых вариантов будет расширение JustAddCommerce[5], которое позволит вам настраивать и добавлять кнопки «Добавить в корзину» и «Заказать» на ваши страницы так же легко, как вы добавляете изображения и таблицы. Бюджетные варианты, такие как PayPal и JustAddCommerce, лучше всего подходят бизнесу, который вписывается в стандартную модель бизнеса и не требует специальных функций обработки.

[1] aspcode.net
[2] php.resourceindex.com
[3] www.mals-e.com
[4] checkout.google.com/seller
[5] www.richmediatech.com

Глава 13
ПРОДВИЖЕНИЕ САЙТА

Цели главы

В этой главе вы узнаете:

- о популярных поисковых системах и поисковых каталогах;
- об устройстве поисковых систем;
- научитесь создавать веб-страницы, дружественные для поисковых систем;
- научитесь подавать заявку на включение вашего веб-сайта в поисковые системы или поисковые каталоги;
- как следить за листингами поисковых систем;
- научитесь осуществлять другие действия по продвижению веб-сайта;
- научитесь создавать внутренние фреймы с помощью элемента `iframe`.

Итак, мы создали свой веб-сайт, а что теперь нужно сделать, чтобы привлечь на него посетителей? Как только у вас появились посетители, как заставить их снова вернуться к вам? Индекс поисковых систем, партнерские программы и баннерная реклама — вот некоторые из тем, обсуждаемых в этой главе.

13.1. Обзор поисковых систем

Что вы делаете, когда нужно найти веб-сайт? Большинство людей запускают свои любимые поисковые системы. Согласно исследованию Нильсена (Nielsen/Net Ratings), девять из десяти пользователей во Всемирной паутине один раз в месяц заходят в поисковую систему, портал или сайт сообщества.

Использование ***поисковых систем*** — это популярный способ навигации во Всемирной паутине и нахождения веб-сайтов. Веб-проект

PEW[1] сообщает, что 87% взрослого населения США ежедневно используют поисковые системы. Отчет DM News на основе исследований компании Harris Interactive сообщает, что 80% интернет-трафика начинается с поисковой системы.

Листинги поисковых систем помогают клиентам обнаружить ваш сайт и увеличивают шансы на то, что они сделают покупку. Листинги поисковых систем могут быть превосходным маркетинговым инструментом для вашего бизнеса. Чтобы воспользоваться всеми возможностями поисковых систем и поисковых индексов (которые иногда называют поисковыми каталогами), полезно знать, как они работают.

13.2. Популярные поисковые системы

Согласно отчету сервиса StatCounter[2], ресурсы Google и Яндекс были двумя самыми популярными сервисами, используемыми в Российской Федерации для поиска во Всемирной паутине по данным на август 2012. Из опрошенных, 56,43% использовали поисковую систему Google, а 41,55% — Яндекс. Далее следуют поисковые системы Bing, Yahoo и Lycos. Посетите сайт StatCounter[3], на нем вы найдете последние результаты исследований. Еще один статистический ресурс, полезный для веб-дизайнеров, — **www.liveinternet.ru/stat/ru/**.

Популярность поисковой системы Google продолжает расти с момента ее основания в конце 1990-х годов. Простой и запоминающийся интерфейс совместно с быстрой загрузкой и полезными результатами сделали ее популярной у российских пользователей Всемирной паутины. Второй наиболее популярной поисковой системой является Яндекс. Во всем мире среди поисковых систем на втором месте (после Google с 83,06%[4]) находится Yahoo! (6,71%). Хотя Yahoo! сегодня является поисковой системой, она изначально была ***поисковым индексом*** (который также называется ***поисковым каталогом***). Каждый сайт, на который подана заявка на включение в поисковый каталог, просматривается человеком. Пример существующего поискового каталога — это DMOZ[5]. Он содержит иерархию тем и сайты, распределенные по этим темам. Поучаствовать в работе каталога может любой пользователь (бесплат-

[1] www.pewinternet.org/Reports/2010/Generations-2010/Activities/All-age-groups.aspx
[2] gs.statcounter.com/#search_engine-RU-monthly-201108-201208-bar
[3] gs.statcounter.com
[4] marketshare.hitslink.com/search-engine-market-share.aspx?qprid=4
[5] www.dmoz.org/World/Russian

но) в качестве редактора. Дополнительным преимуществом попадания в листинг DMOZ является то, что базу, содержащую одобренные сайты, использует определенное количество поисковых систем, включая Google, Ask.com и AOL.

13.3. Устройство поисковых систем

Поисковые системы состоят из следующих компонентов:

- робот;
- юаза данных (ее также называют поисковым каталогом);
- форма поиска (ее также используют поисковые каталоги).

Робот

Поисковый робот (его иногда называют паук или бот) — это программа, которая автоматически прочесывает гипертекстовую структуру Всемирной паутины, разыскивая документы веб-страниц и двигаясь по гиперссылкам на странице. Он двигается, как паук по паутине, получая доступ и документируя страницы. Робот делит страницы по категориям и хранит информацию о веб-сайте и веб-страницах в базе данных. Различные роботы работают по-разному, но в общем, они получают доступ и могут хранить следующие разделы веб-страниц: заголовок, метатеги ключевых слов, метатеги описаний и небольшую часть текста на странице (обычно это первые несколько предложений текста, содержащегося внутри тегов заголовка). Зайдите на сайт Википедия[1], если хотите больше узнать о поисковых роботах.

База данных

База данных — это собранная воедино информация, организованная так, чтобы ее контент был легко досягаем, управляем и обновляем. Системы управления базами данных (СУБД), такие как Oracle, Microsoft SQL Server или IBM DB2, используются для конфигурации и управления базой данных.

Веб-страница, отображающая результат вашего поиска, предоставляет информацию из базы данных, к которой имеет доступ сайт поис-

[1] ru.wikipedia.org/wiki/Поисковый_робот

ковой системы. Некоторые поисковые системы получают часть своего контента от других поисковых систем[1]. Например, поисковая система AOL получает свой основной контент от поисковой системы Google.

Поисковые формы

Поисковая форма — это компонент поисковой системы, с которым большинство из вас знакомы. Вы, вероятно, много раз пользовались поисковыми системами, но не думали о том, что происходит «за кадром». Поисковая форма — это графический пользовательский интерфейс, позволяющий пользователю вводить слово или фразу, которую он ищет. Обычно это просто текстовое поле и кнопка отправки данных. Посетитель поисковых систем вводит слова (которые называются ***ключевыми словами***), связанные с его или ее поисковым запросом, в текстовое поле. Когда форма отправлена, данные, введенные в текстовое поле, отправляются серверному скрипту, который осуществляет поиск по базе данных, используя введенные ключевые слова. ***Результат поиска*** — это информация, такая как URL-адрес веб-страниц, отвечающих вашим критериям. Результаты поиска указаны со ссылкой отдельно на каждую страницу и дополнительной информацией, которая может включать в себя название страницы, краткое описание, первые несколько строк текста или графическую миниатюру страницы. Вид дополнительной информации отличается в зависимости от поисковой системы. Потом веб-сервер на сайте поисковой системы отправляет ***страницу результатов поиска*** для отображения в вашем браузере.

Порядок страниц в результатах поиска может зависеть от платной рекламы, алфавитного порядка и популярности ссылки (более подробно мы об этом поговорим далее). У каждой поисковой системы своя политика упорядочивания результатов поиска.

Компоненты поисковой системы (роботы, база данных и поисковая форма) работают вместе для получения информации о веб-странице, хранения информации о веб-странице и для предоставления графического пользовательского интерфейса для удобства поиска и отображения списка веб-страниц, релевантных заданным ключевым словам.

Теперь, когда вы знаете о компонентах поисковых систем, давайте перейдем к наиболее важной части — как создавать ваши веб-страницы так, чтобы они продвигали ваш веб-сайт.

[1] www.bruceclay.com/searchenginerelationshipchart.htm

13.4. Создание веб-страниц для продвижения

Если вы следовали рекомендациям, приведенным в практических заданиях по веб-дизайну, то уже создали веб-сайт с такими страницами, которые привлекательны и убедительны для вашей целевой аудитории. Как сделать так, чтобы ваш сайт работал с поисковыми системами? В данном разделе предоставлено несколько предложений и идей, как сделать ваши веб-страниц привлекательными для поисковых систем — этот процесс называется ***поисковая оптимизация*** (SEO, search engine optimization).

Ключевые слова

Подумайте о терминах и фразах, которые люди могут использовать при поиске вашего сайта. Эти термины или фразы, описывающие ваш веб-сайт или бизнес, являются ***ключевыми словами***. Составьте из них список и не забудьте добавить ключевые слова с распространенными ошибками (опечатками), которые могут совершить пользователи.

Названия страниц

Название, описывающее страницу (текст между тегами `<title>`, который включает название вашей компании и/или вашего веб-сайта), уже само по себе поможет маркетингу вашего сайта. Общепринятой практикой для поисковых систем является отображение названия сайта на странице результатов поиска. Название страницы также сохраняется по умолчанию, когда посетитель добавляет ваш сайт в Избранное и часто присутствует, когда посетитель распечатывает страницу вашего сайта. Избегайте использования одинаковых названий для каждой страницы, включайте в описание страницы ключевые слова, подходящие именно для нее. Например, измените текст названия: вместо «Агентство интерактивного дизайна» укажите название компании и цель страницы: «Агентство интерактивного дизайна: индивидуальные решения для электронной коммерции».

Теги заголовков

Используйте структурные теги, такие как `<h1>`, `<h2>` и т. д. для организации контента на вашей странице. Если это подходит под контент

веб-страницы, также добавьте несколько ключевых слов в текст, содержащийся внутри тегов заголовков. Некоторые поисковые системы посчитают страницу более релевантной, если ключевые слова включены в название или заголовок страницы. Также включите подходящие ключевые слова в контент страницы. Но избегайте спама ключевых слов — не повторяйте их снова и снова. Программы поисковых систем с каждым разом становятся все более искушенными и могут отказать вам в зачислении в список, если заподозрят, что вы были нечестны или попытались обмануть систему.

Описания

Какой особенностью обладает ваш веб-сайт, что должно привлечь посетителей? Думая над этим вопросом, напишите несколько предложений о вашем веб-сайте или бизнесе. Описание должно быть привлекательным и интересным, чтобы человек, пользующийся поиском, выбрал ваш сайт из списка, предоставленного поисковой системой или поисковым каталогом. Некоторые поисковые системы будут отображать описание вашего сайта в результатах поиска.

Вам, должно быть, интересно, как это описание можно применить в веб-странице. Описание размещается на веб-странице с помощью метатегов HTML в области заголовка.

Метатег `description`

Метатег — это контейнерный тег, которые размещают в области заголовка веб-страницы. Вы уже использовали метатег для указания кодировки символов на странице. Есть еще ряд других применений метатегов. Здесь мы сконцентрируемся на их использовании для обеспечения описания сайта и перечисления ключевых слов для поисковых систем. Содержимое метатега `description` отображается на странице результатов поиска некоторыми поисковыми системами, например Google. Атрибут **`name`** указывает на использование метатега. Атрибут **`content`** указывает значения, необходимые именно для этого конкретного использования. Значение **`description`** атрибута **`name`** указывает на то, что использование метатега нужно для обеспечения описания. Например, метатег `description` для веб-сайта консультационной фирмы по веб-разработкам под названием Acme Design может быть задан следующим образом:

```
<meta name="description" content="Acme Design -
ведущая консультационная фирма во Всемирной
паутине, которая специализируется на электронной
коммерции, дизайне веб-сайтов, разработке веб-сайтов
и изменении дизайна веб-сайтов.">
```

ЧаВо

ЧТО ДЕЛАТЬ, ЕСЛИ Я НЕ ХОЧУ, ЧТОБЫ ПОИСКОВАЯ СИСТЕМА ИНДЕКСИРОВАЛА СТРАНИЦУ?

Некоторые страницы вы можете не захотеть индексировать, например, текстовые страницы или страницы, предназначенные только для узкого круга лиц (членам семьи или коллегам). Метатеги могут быть использованы также и для этой цели. Для указания поисковым роботам, что страницу не надо индексировать и не нужно следовать по ссылкам, не размещайте метатеги с ключевыми словами на странице. Вместо этого добавьте метатег `"robots"` в страницу, как указано ниже:

```
<meta name="robots" content="noindex, nofollow">
```

Ссылки

Проверьте, все ли гиперссылки работают. Каждая страница вашего веб-сайта должна быть доступна по текстовой гиперссылке. Текст должен быть описательным (избегайте фраз типа «больше информации» и «нажмите сюда») и включать соответствующие ключевые слова. Входящие ссылки также играют в SEO-оптимизации большую роль (см. раздел о популярности ссылок далее в этой главе).

Изображения и мультимедийные файлы

Помните, что поисковые роботы «не видят» текст, встроенный в ваши изображения и мультимедийные файлы. Разместите значимый замещающий текст для изображений. Включите подходящие по смыслу ключевые слова в замещающий текст. Хотя некоторые поисковые роботы, такие как Googlebot, недавно освоили функцию индексации текста и гиперссылок, содержащихся внутри мультимедийных файлов Flash, вы должны знать, что веб-сайты, которые существенно зависят от использования таких технологий, как Flash и Silverlight, будут видны не всем поисковым системам, и в результатах поиска займут более низкие позиции.

Валидный код

Поисковые системы не требуют, чтобы код HTML и CSS ваших страниц проходил тест на валидность. Тем не менее валидный и хорошо структурированный код намного проще обрабатывается роботами поисковых систем. А это, в свою очередь, может помочь занять вашему сайту более высокое положение в результатах поиска.

Значимый контент

Вероятно, самый важный компонент SEO-оптимизации после соблюдения всех стандартов веб-дизайна (см. главу 5), о котором часто забывают, — это размещение значимого контента на вашем веб-сайте. Ваш веб-сайт должен содержать высококачественный и хорошо организованный контент, который будет иметь смысл для ваших посетителей.

13.5. Регистрация сайта в поисковых машинах и каталогах

В соответствии с исследованиями Ассоциации прямого маркетинга[1] 66% маркетологов относят поисковые системы к лучшему методу привлечения трафика на сайты. Прежде чем вы задумаетесь о том, как зарегистрировать свой веб-сайт в поисковой системе, убедитесь, что сайт завершен и применены все основные техники SEO-оптимизации (описанные выше). Проверив готовность веб-сайта, выполните перечисленные ниже шаги, чтобы зарегистрировать сайт для индексации поисковой системой.

Шаг 1. Зайдите на сайт поисковой системы (к примеру, **webmaster.yandex.ru**) и нажмите кнопку **Добавить сайт**. Этот элемент управления обычно присутствует на главной странице (или в разделе для веб-дизайнера сайта) поисковой системы. Будьте терпеливыми, эти ссылки не всегда бывают перед глазами.

Шаг 2. Следуйте указаниям, приведенным на странице, и заполните форму запроса о добавлении вашего сайта в поисковую систему. В других поисковых системах может быть установлена плата за автоматическое включение в список, которая называется платное включение/регистрация. Подробнее об этом мы поговорим позднее. В данный

[1] www.the-dma.org

момент добавление URL-адреса в поисковую систему Яндекс является бесплатным.

Шаг 3. Паук поисковой системы проиндексирует ваш сайт. На это может уйти несколько недель.

Шаг 4. Через несколько недель после того, как вы подали заявку на включение вашего сайта, проверьте в поисковой системе или поисковом каталоге, появился ли ваш сайт в списке или нет. Если он не появился, проверьте ваши страницы и убедитесь, что они «дружественны» для поисковых систем и отображаются в распространенных браузерах. Если веб-сайт создан для бизнеса, вы можете захотеть рассмотреть вариант его платного включения в листинг поисковой системы или поискового каталога (также называют «быстрое включение»), оплатить первоочередное появление при показе результатов поиска (это называется спонсорством или рекламой) и платить каждый раз, когда посетитель переходит на сайте поисковой системы по ссылке на ваш сайт. Многие компании рассматривают оплату за эти виды услуг как еще одну статью расхода, такую как плату за размещение рекламы в газете или в телефонном справочнике.

ЧаВо

ОКУПАЕТСЯ ЛИ РЕКЛАМА В ПОИСКОВЫХ СИСТЕМАХ?

Здесь возможны варианты. Сколько стоит вашему клиенту появляться на первой странице результатов поисковой выдачи? Вы выбираете ключевые слова, которые вызовут показ вашей рекламы. Вы также устанавливаете ежемесячный бюджет и максимальную цену за каждый клик. В то время как стоимость и тарифы отличаются у разных поисковых систем, в данный момент тарифы поисковой системы Google базируются на стоимости за клик. С вас снимают плату каждый раз, когда посетитель Google щелкает мышью по вашей рекламе. Узнать об этой программе подробнее можно на сайте **google.com/adwords.**

Если вы подробнее ознакомитесь с платными программами, которые предлагают поисковые системы, вы наткнетесь на ряд сокращений, связанных с интернет-маркетингом. Наиболее популярные перечислены ниже:

- **Цена за клик** (CPC, Cost Per Click).

  Это понятие часто употребляется в связке с ***оплатой за клик*** (PPC, Pay Per Click) — это цена, которую вы заплатите, если вы участвуете в программе платного спонсорства или рекламы, и посетитель переходит по ссылке на ваш веб-сайт.

- **Цена за тысячу показов** (CPI, Cost Per Thousand Impressions).

  При модели Цена за тысячу показов вы платите за каждые 1000 раз демонстрации вашего баннера во Всемирной паутине (вне зависимости, щелкал ли посетитель по нему или нет).

- **Показатель кликабельности** (CTR, Click Through Rate).

  Показатель кликабельности (CTR) — определяется как отношение числа кликов на баннер или рекламное объявление к числу показов, измеряется в процентах. Например, если ваша реклама была показана 100 раз и 20 человек перешли по ней, ваш показатель кликабельности равен 20/100 или 20%.

Карта сайта

Инструкции, приведенные в Центре веб-мастеров поисковой системы Google[1], описывают два типа карт сайта (файлов Sitemap), которые являются полезными для поисковой оптимизации:

- Карта сайта — это веб-страница, которая содержит иерархический список гиперссылок на основные страницы на вашем веб-сайте (см. рис. 5.19). Информация на такой странице не только полезна посетителям вашего веб-сайта, но также может помочь роботам поисковых систем, поскольку они следуют по гиперссылкам в вашем сайте.
- ***Sitemap*** — это файл XML, который используют поисковые системы, но к нему нет доступа у посетителей веб-сайта. Файл Sitemap предоставляет информацию поисковым системам, например Google, о вашем веб-сайте и обязательно список ссылок вместе с этой информацией: дата последнего изменения каждой страницы, индикатор того, насколько часто на странице происходят изменения, и уровень приоритета для каждой страницы. Часть файла *sitemap.xml* приведен ниже:

```
<url>
<loc>http://webdevfoundations.net/</loc>
<lastmod>2011-07-03T08:10:09+00:00</lastmod>
<changefreq>monthly</changefreq>
<priority>1.00</priority>
</url>
<url>
<loc>http://webdevfoundations.net/index.html</loc>
<lastmod>2011-07-03T08:10:09+00:00</lastmod>
```

[1] www.google.ru/webmasters

```
<changefreq>monthly</changefreq>
<priority>1.00</priority>
</url>
<url>
<loc>http://webdevfoundations.net/6e/chapter1.
html</loc>
<lastmod>2011-08-22T15:09:07+00:00</lastmod>
<changefreq>monthly</changefreq>
<priority>0.800</priority>
</url>
```

Онлайн-генераторы файлов Sitemap, такие как **xml-sitemaps.com**, автоматически создадут для вас файлы с картой сайта с именем *sitemap.xml*. Вам нужно будет загрузить файл Sitemap на свой веб-сайт и сообщить поисковой системе Google его URL-адрес. Для получения более подробной информации о файлах Sitemap, см. Центр веб-мастеров Google.

Альянсы

Есть несколько альянсов между определенными поисковыми системами и поисковыми каталогами. Проект DMOZ[1] предоставляет услуги каталога для ряда поисковых систем, включая Google. Обратите внимание, что эти альянсы могут меняться время от времени. Осведомленность об альянсах поисковых систем поможет вам максимизировать шансы на то, что ваш веб-сайт появится в результатах поиска.

13.6. Мониторинг сайта в каталогах

Несмотря на ваше желание постоянно видеть свой сайт в поисковых системах и поисковых каталогах, вам может потребоваться терпение, пока ваш сайт появится на страницах результатов поиска. Согласно информации с ресурса **www.searchengineposition.com**, пройдет от двух дней до двух недель, пока ваш веб-сайт появится в листинге поисковой системы Google. Также помните о том, что при подаче заявки нет гарантии, что сайт будет включен в листинг. Однако редко бывает так, что качественный веб-сайт с полезным контентом не индексируется поисковыми системами и не попадает в списки поисковых каталогов.

[1] www.dmoz.org

Как только ваши сайты попали в листинг, важно определить, какие ключевые слова работают. Обычно вам нужно будет со временем корректировать и менять ключевые слова:

- **Ручная проверка.** Зайдите на сайт поисковой системы и введите ключевые слова. Посмотрите результаты. Рекомендуется записывать название поисковой системы, ключевые слова и рейтинг страницы.
- **Веб-аналитика.** Ассоциация Web Analytics Association[1] определяет ***веб-аналитику*** как «измерение, сбор, анализ, представление и интерпретацию информации о посетителях веб-сайтов с целью их улучшения и оптимизации». Каждый посетитель вашего веб-сайта, включая тех, которые пришли с поисковых систем, записывается в файл журнала вашего веб-сайта. **Журнал веб-сайта** состоит из одного или более текстовых файлов, которые записывают каждое посещение вашего веб-сайта, собирая информацию о посетителях и сайтах, с которых они пришли. Анализируя журнал, вы можете определить, успешны ли ваши ключевые слова и какая поисковая система была использована. Вы также можете определить дни и время, в которое пользователи посещают ваш сайт, операционные системы и браузеры, которые они использовали, маршруты передвижения посетителей по вашему сайту и многое другое. Журнал — это достаточно загадочный текстовый файл (рис. 13.1, в котором приведена часть журнала).

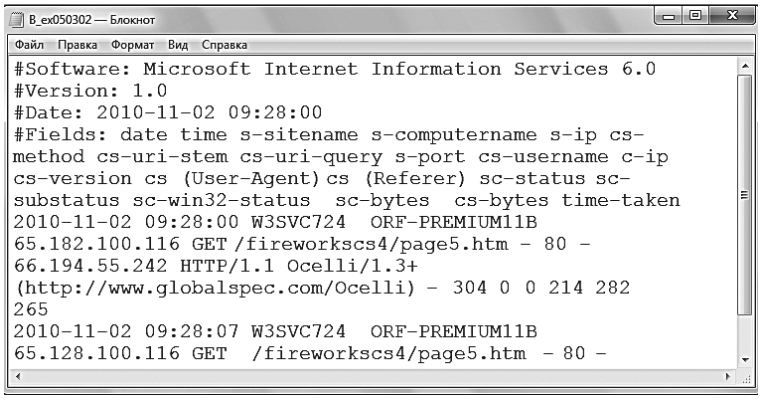


Рис. 13.1. Журнал веб-сайта содержит полезную информацию, но его бывает нелегко прочитать

Программное обеспечение для веб-аналитики может анализировать ваш журнал и создать простые графики и отчеты. Если у вас

[1] www.webanalyticsassociation.org

есть собственный веб-сайт и доменное имя, многие хостинги предоставят бесплатный доступ к журналам и могут даже делать отчеты по веб-аналитике, показывающие, какие ключевые слова фактически используются в поисковой системе при поиске определенного веб-сайта.

При проверке информации в журнале, можно определить не только, какие ключевые слова работают, но и какими поисковыми системами пользуются посетители. Для анализа журнала веб-сайта часто используется программа Webtrends[1]. На рис. 13.2 представлена информация из отчета об анализе журнала сайта, содержащая список 10 ключевых слов, используемых фактическими посетителями при поиске определенного сайта в поисковой системе Yahoo!.

Главные поисковые системы и ключевые слова			
Поисковая система	Ключевые слова	Найденные ключевые слова	% итого
Yahoo!	Цитата	280	37.68%
	Образование	182	24.49%
	Веб	144	19.38%
	Фон	83	11.17%
	JavaScript	73	9.82%
	Картинка	59	7.94%
	Java	54	7.26%
	Образовательный	36	4.84%
	Изображение	31	4.17%
	Web	28	3.76%

Рис. 13.2. Фрагмент отчета об анализе журнала сайта

Анализ журналов — это мощный маркетинговый инструмент, поскольку вы можете точно определить, как посетители находят ваш сайт. Это позволяет узнать, какие ключевые слова работают, а какие нет. Вероятно, подумав еще, вы сможете добавить в ваш список новые вариации эффективных ключевых слов. Если вы изучите рис. 13.2, то заметите, что образовательные материалы, выложенные на этом конкретном веб-сайте, довольно популярны. Разработчики веб-сайта могут добавить еще материалы и потенциально увеличить количество посетителей сайта.

Поисковая система Google предлагает бесплатную услугу по веб-аналитике по адресу **www.google.com/intl/ru_ALL/analytics/index.html**. Категории отчетов предоставляют вам следующую информацию:

[1] webtrends.com

- посетители (включая географическую карту и информацию о браузере);
- источники трафика (сайты-источники переходов, ключевые слова и AdWords);
- контент (включая самые популярные страницы входа, пути переходов по сайту и страницы выхода с сайта);
- цели (отслеживает бизнес-цели).

Другой вариант — покупка программы, которая поможет вам отслеживать положение вашего сайта в листингах поисковых систем. Такие приложения, как WebPosition[1], могут создавать отчеты о вашем рейтинге в поисковых системах, анализировать их, отслеживать ключевые слова и даже подавать заявку о внесении вашего сайта в каталог поисковых систем.

13.7. Популярность ссылок

Популярность ссылок — это рейтинг, определяемый поисковой системой на основании количества сайтов, с которых ведут ссылки на конкретный веб-сайт, и качества этих сайтов. Например, ссылка с хорошо известного сайта, такого как «Вести»[2], считается ссылкой более высокого качества, по сравнению со ссылкой с личной страницы вашего друга на бесплатном веб-сервере. Популярность ссылок веб-сайта может определять его положение на странице с результатами поиска. Один из способов проверить, ссылки с каких сайтов ведут на ваш, — это проанализировать журнал вашего сайта. Другой способ — зайти на сервис по проверке популярности ссылок, к примеру LinkPopularity.com, и запустить отчет, который проверяет популярность ссылок в определенном количестве поисковых систем. Третий способ — перейти на сайт поисковой системы и проверить самостоятельно. В поисковой системе Google введите запрос: **link:*mysite.ru*** в поле поиска и вы увидите список сайтов, ссылки с которых ведут к сайту ***mysite.ru***. Поисковые системы и поисковые каталоги — не единственные инструменты, которые вы можете использовать, чтобы привести посетителей на ваш веб-сайт. Следующий раздел расскажет о некоторых других возможностях.

[1] webposition.com
[2] vesti.ru

13.8. Продвижение сайта в социальных сетях

Получить существующих и потенциальных посетителей вашего веб-сайта можно с помощью ***оптимизации под социальные сети*** (SMO, Social Media Optimization). Рохит Боргава (Rohit Bhargava) описывает, как лучше оптимизировать веб-сайт, чтобы было «проще предоставить на него ссылку, чтобы он был ярче виден при поиске в социальных сетях (таких как Technorati) и чаще включался в релевантные публикации в блогах, подкастах и видеоблогах». Преимущества оптимизации сайта под социальные сети включают в себя повышение осведомленности о вашем бренде и/или сайте вместе с повышением количества входящих ссылок (что может помочь в поисковой оптимизации). Для начала посетите следующие ресурсы, чтобы получить дополнительные советы и подсказки по оптимизации сайта под социальные сети:

- правила оптимизации сайтов для социальных сетей: **whiteseo.ru/content/view/227/6/**
- 5 правил оптимизации сайта под социальные сети (SMO): **rohitbhargava.typepad.com/weblog/2006/08/5_rules_of_soci.html**
- SMO, или Раскрутка в социальных сетях: **vlada-rykova.com/smo-ili-raskrutka-v-socialnyx-setyax/**

Ключевой принцип в оптимизации сайта под социальные сети заключается в простоте для пользователя сделать закладку или добавить в Избранное. Сайты, использующие социальные закладки, такие как БобрДобр[1] и Delicious[2], обеспечивают возможность людям сохранять, рекомендовать и классифицировать веб-сайты. Сделайте так, чтобы вашим посетителям было просто добавить ваш сайт в социальные сервисы закладок, добавив кнопки или ссылки, такие как «Забобрить!», которые предлагает сайт БобрДобр по ссылке **bobrdobr.ru/tools/**.

Блоги и RSS-ленты

В главе 1 была представлена информация о ***блогах***, которые легко обновлять и которые являются общедоступными журналами во Всемирной паутине. Преимущество блогов заключается в том, что в них

[1] bobrdobr.ru
[2] www.delicious.com

можно делиться информацией, а публикации в них используют различные компании (начиная от Nike до Adobe) с целью расширения отношений с клиентами. Большинство сайтов-хостингов для блогов, такие как **blogspot.com** и **wordpress.com**, предлагают бесплатную услугу ленты RSS (Really Simple Syndication, очень простой сбор сводной информации) с контентом вашего блога. ***RSS-лента*** блога — это XML-файл (с расширением *.rss*), который содержит краткую информацию о посте со ссылкой на блог или другой веб-сайт. Ваши клиенты или партнеры по бизнесу могут подписаться на ленту RSS, которая будет автоматически обновляться, когда вы будете добавлять новый контент. Кнопка подписки на RSS-ленту чаще всего окрашена в оранжевый цвет и содержит текст XML или RSS. Многие браузеры способны отображать содержимое RSS-лент. Существует бесчисленное множество бесплатных и недорогих программ для чтения RSS-лент, включая Headline Viewer[1] и NetNewsWire[2]. Чтобы просмотреть, как выглядит блог, посетите сайт **sadalskij.livejournal.com**.

Социальные сети

Присоединяйтесь к группам на сайтах социальных сетей, таких как В Контакте[3], Мой Круг[4], Мой Мир[5], Одноклассники.ru[6] или В кругу друзей[7], чтобы найти и связаться с существующими и потенциальными посетителями. Создайте контент, который будет продвигать ваш веб-сайт, и опубликуйте его на сервисе YouTube[8], Slideshare[9] и других подобных сайтах. Будьте активны на сервисах микроблогинга, таких как Twitter[10]. Издание Bloomberg сообщило, что использование компанией Dell сайта Twitter в результате привело к дополнительным заказам на сумму в 6,5 млн долларов за два года. Расскажите о вашем контенте в блоге и микроблоге. Позвольте вирусному маркетингу работать на вас, когда потенциальные посетители найдут и порекомендуют ваш контент, который должен увеличить осведомленность и привлечь новых и вернувшихся посетителей на ваш сайт.

[1] www.headlineviewer.com
[2] ranchero.com/netnewswire
[3] vkontakte.ru
[4] moikrug.ru
[5] my.mail.ru
[6] www.odnoklassniki.ru
[7] vkrugudruzei.ru
[8] youtube.com
[9] slideshare.net
[10] twitter.com

13.9. Другие способы продвижения сайта

Существует ряд других способов продвижения вашего веб-сайта, включая партнерские программы, баннерную рекламу, обмен баннерами, взаимные ссылки, новостные письма, личные рекомендации, традиционную медийную рекламу и указание URL-адресов во всех рекламных материалах.

QR-коды

QR-код — это двухмерный штрихкод в виде квадрата, который распознается приложением смартфона или устройством для чтения QR-кода. Закодированные данные могут быть текстом, телефонным номером или даже URL-адресом веб-сайта. Онлайн-издание ClickZ сообщает, что использование QR-кодов увеличивается, поскольку все больше людей пользуется смартфонами[1]. Существует множество бесплатных онлайн-генераторов, в том числе **qrcode.kaywa.com, http://www.labeljoy.com/en/generate-qr-code.html http://www.qrstuff.com**. Доступны бесплатные приложения, такие как ScanLife и QR Code Scanner для смартфонов под управлением операционных систем iOS, Android и BlackBerry, которые используют функции камеры для сканирования QR-кода. QR-код полезен для продвижения сайта — добавьте его на визитную карточку или даже футболку! QR-код на рис. 13.3 скрывает ссылку на страницу **http://www.eksmo.ru/**.

Рис. 13.3. QR-код сайта **http://www.eksmo.ru/**

Партнерские программы

Суть ***партнерских программ*** заключается в том, что один веб-сайт (партнер) продвигает товары или услуги другого сайта (продавца) в обмен на комиссию. Оба веб-сайта выигрывают от такого сотрудничества. По имеющимся данным, сайт Amazon.com первым начал партнерскую маркетинговую программу, которая до сих пор успешно функционирует.

[1] www.clickz.com/clickz/column/2039242/qr-codes-matter

Присоединившись к этой программе, ваш веб-сайт может представлять книги со ссылкой на сайт компании Amazon. Если один из ваших посетителей совершает покупку, вы получаете комиссию. Компания Amazon выигрывает, потому что вы привели заинтересованного посетителя, который может купить товар сейчас или в будущем. Ваш сайт выигрывает от того, что является партнером такого известного сайта, как Amazon, и дохода от программы.

Зайдите на веб-сайт **www.cj.com**, чтобы увидеть программу, связывающую веб-сайты с потенциальными партнерскими программами. Их служба позволяет издателям (владельцам сайтов и разработчикам) выбрать из широкого круга рекламодателей и партнерских программ. К преимуществам для веб-разработчиков относится возможность сотрудничать с ведущими рекламодателями, получать дополнительный доход от посетителей сайта или рекламного пространства, а также отслеживать работу сайтов и получать отчеты в режиме реального времени.

На сайте Википедия[1] вы найдете больше информации о партнерских программах. На сайтах **www.affiliate.ru**, **www.affiliateportal.ru**, **allpp.ru/dir/** и **www.ppmoney.ru** вы найдете каталоги партнерских программ.

Баннерная реклама

Баннерная реклама обычно представляет собой графические изображения, которые используют для объявления и рекламы имени или предназначения сайта. Баннер — это изображение с гиперссылкой, после щелчка мышью по которой открывается рекламируемый сайт. Вы, скорее всего, видели их много раз во время серфинга во Всемирной паутине. Баннерная реклама существует уже достаточно давно, сайт **www.hotwired.com** представил первую баннерную рекламу в 1994 году для продвижения компании AT&T.

Не существует официальных размеров для баннерной рекламы. Тем не менее исследования, проведенные Interactive Advertising Bureau (**www.iab.net**), предоставляют рекомендации для типичных размеров баннеров, в том числе «перетяжки» вверху страницы (728×90 пикселов) и средний квадрат (300×250 пикселов). На этом сайте вы найдете[2] полный список типов рекламы и распространенные размеры. Стоимость показа вашей баннерной рекламы может варьироваться. Некоторые веб-сайты берут плату за показ (обычно цена назначается за тысячу по-

[1] ru.wikipedia.org/wiki/Партнерская_программа
[2] www.iab.net/iab_products_and_industry_services/508676/508767/Ad_Unit

казов). Другие берут плату только за переход по баннеру. Некоторые поисковые системы отображают вашу баннерную рекламу на странице с результатами поиска по ключевым словам, относящимся к вашему сайту (за плату, разумеется).

Эффективность баннерной рекламы является отдельной темой для изучения. Если вы один из большинства посетителей веб-сайта, то не будете обращать внимания на баннерную рекламу. Компания Interactive Advertising Bureau провела исследования отношений между баннерной рекламой и узнаваемостью бренда. Отчет на сайте ClickZ.com[1] демонстрирует, что в то время как баннерная реклама стандартного размера положительно сказывается на узнаваемости бренда, другие ее форматы, такие как «небоскребы» (длинные, узкие рекламные объявления, располагаемые вдоль одной из сторон страницы), и большие прямоугольные рекламные баннеры в шесть раз более эффективны в вопросе повышения узнаваемости бренда и ассоциирования потребителем рекламы компании с самой компанией. Мультимедийные технологии, такие как аудио, видео и Flash, также оказывают огромное воздействие и повышают эффективность брендинга. Безусловно, повышение узнаваемости бренда увеличит вероятность того, что этот веб-сайт будут посещать и в будущем.

Если стоимость баннерной рекламы кажется вам слишком высокой по сравнению с ее эффективностью, рассмотрите «бесплатный» вариант — обмен баннерами.

Обмен баннерами

Условия программ ***обмена баннерами*** различаются, но основная идея заключается в том, что вы соглашаетесь демонстрировать баннеры других сайтов, а другие сайты будут показывать ваш баннер. Информацию об обмене баннерами вы можете найти на сайтах **www.rle.ru**, **www.clickhere.ru** и **qle.ru**. Обмен баннерами может быть выигрышным для всех сторон, потому что это бесплатная реклама.

Обмен ссылками

Обмен ссылками обычно производится между двумя сайтами со схожим или дополняющим контентом. Вы соглашаетесь размещать ссылки друг на друга. В результате на каждом сайте должно быть больше посетителей. Если вы найдете сайт, с которым хотели бы обменяться ссылками,

[1] www.clickz.com/stats/sectors/advertising/article.php/804761

свяжитесь с его веб-мастером (к примеру, по электронной почте). Несмотря на то что некоторые поисковые системы частично формируют рейтинг, исходя из количества качественных ссылок на ваш веб-сайт, правильно расположенные взаимные ссылки могут помочь обоим сайтам.

Новостные рассылки

Рассылка новостных писем может вернуть посетителей на ваш сайт. Первый шаг — собрать адреса электронной почты. Предоставьте возможность посетителям веб-сайта подписаться на получение вашего новостного письма, заполнив форму на сайте. На рис. 13.4 приведен пример формы подписки на рассылку новостных писем сайта Softodrom.ru.

Рис. 13.4. Пример формы подписки на рассылку новостных писем

Предоставьте вашим посетителям действительно ценную и своевременную информацию по актуальным темам, скидкам и т. д. Регулярно рассылайте новостные письма со свежим и убедительным контентом. Это помогает напоминать вашим прошлым посетителям о вашем веб-сайте. Они даже могут переслать эти письма коллегам и привлечь новых посетителей на ваш сайт.

Элементы сайта, удерживающие посетителей

Частое обновление вашего веб-сайта и постоянное добавление свежего контента воодушевит посетителей вернуться на ваш веб-сайт. Как же их удержать? Сделайте свой веб-сайт привлекательным. Отображайте интересный и значимый контент вместе с такими элементами, способствующими удерживанию посетителей, как обновление новостей, голосования и опросы, а также форумы и чаты.

Персональные рекомендации

Отправить другу — это форма личной рекомендации, некоторые сайты помогают вам рассказать о них вашим друзьям. Они содержат такие ссылки, как «Отправить эту статью по e-mail» или «Отправить эту страницу другу» или «Рассказать коллеге об этом веб-сайте». Эта личная рекомендация приведет на сайт нового посетителя, которому, вероятно, будет интересен контент сайта. На рис. 13.5 приведен снимок одной из страниц сайта Brita[1], которая содержит форму для отправки интересующей вас статьи другу.

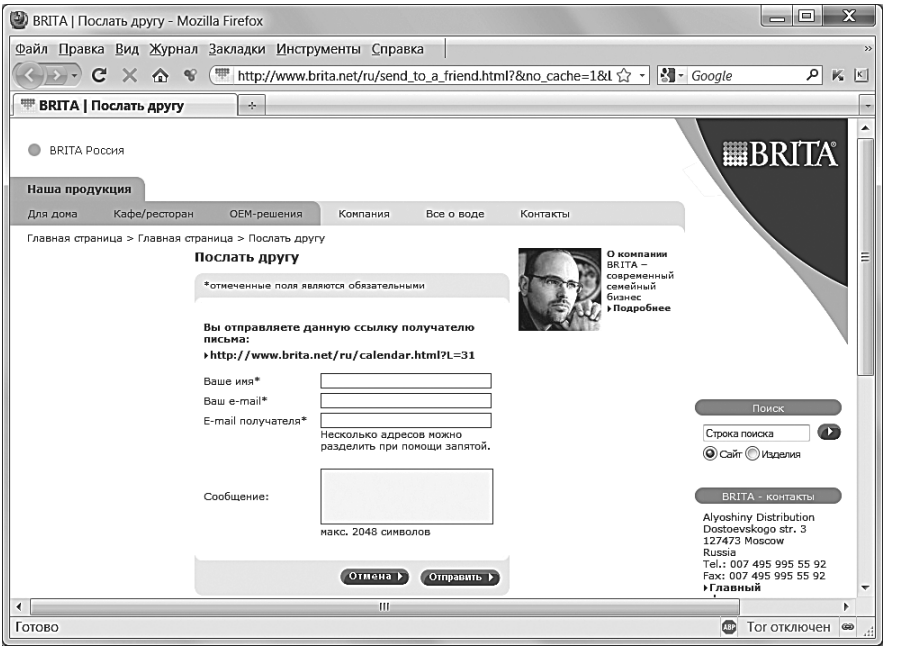


Рис. 13.5. Этот сайт содержит форму для отправки интересующей вас статьи другу

[1] www.brita.net

Группы новостей и серверы подписки

Подпишитесь на релевантные ***новостные группы***, ***рассылки*** или форумы, связанные с контентом вашего веб-сайта в Usenet (компьютерной сети, используемой для общения и публикации файлов). Не отвечайте на посты рекламой своего сайта. Вместо рекламы предлагайте помощь или совет. Включите строку с URL-адресом вашего веб-сайта. Будьте тактичны, вас могут забанить в некоторых рассылках, если модератор решит, что вашей единственной целью является реклама. Однако, предложив дружелюбный и полезный совет в новостной группе или рассылке, вы можете продвинуть ваш сайт в тактичной и позитивной манере.

Ваш провайдер интернет-услуг может предоставить доступ к новостным группам Usenet. Поисковая система Google также предоставляет доступ по ссылке **groups.google.ru**. Рассылки могут проводить как индивидуальные лица, так и организации.

Традиционная мультимедийная реклама и существующие маркетинговые материалы

Не забудьте упомянуть о вашем сайте в рекламе в любом печатном СМИ, на телевидении или радио. Напечатайте URL-адрес вашего веб-сайта во всех брошюрах, корпоративных бланках и визитках. Существующим и потенциальным клиентам вы таким образом упростите поиск вашего веб-сайта. В зависимости от целевой аудитории подумайте также о размещении QR-кода своего веб-сайта в печатных средствах массовой информации.

13.10. Сопровождение динамичного контента с помощью внутренних фреймов

Каким образом **auto.ru**, сайт с объявлениями о продаже автомобилей, отображает баннерную рекламу на своей главной странице, которая размещена на хостинге и находится под управлением другой организации? Как сайты команды «Зенит»[1] и телеканала BBC[2] с легкостью отображают целый ряд мультимедийных клипов? Как создаются и отслеживаются рефералы потенциальных посетителей в партнерской

[1] www.fc-zenit.ru/main/
[2] www.bbc.co.uk/russian/

программе Ozon.ru? Как поисковая система Google обеспечивает показ рекламных объявлений AdSense и переходы на сторонние веб-сайты? На момент написания этой книги ответом на все эти вопросы являлось использование внутренних фреймов. Использование внутренних фреймов широко распространено во Всемирной паутине в различных целях маркетинга и продвижения сайтов, включая демонстрацию рекламных баннеров, проигрывание мультимедийных файлов, которые могут быть размещены на отдельном веб-сервере, и показ контента партнерских сайтов. Преимущество заключается в разделении контроля. Динамический контент — такой как баннерная реклама или мультимедийные клипы — может менять команда разработчиков проекта, не имеющая разрешения на внесение изменений во всем веб-сайте. Например, в случае рекламного баннера на веб-сайте **auto.ru**, организация, являющаяся третьим лицом (такая как компания AdRiver[1]), имеет контроль над контентом рекламного баннера, однако у нее нет доступа к обновлению других частей страницы **auto.ru**. Это достигается конфигурированием динамического контента (в форме рекламных баннеров) во внутренних фреймах. Давайте рассмотрим, как конфигурируются внутренние фреймы.

Элемент `iframe`

Внутренние фреймы (они также называются встроенными или плавающими фреймами) можно размещать в теле любой веб-страницы, идентично тому, как вы бы вставили изображение в веб-страницу. Элемент **`iframe`** задает внутренний фрейм, который отображает содержимое другой веб-страницы внутри документа веб-страницы, что называется *встроенный фрейм*. Элемент `iframe` начинается с тега `<iframe>` и заканчивается тегом `</iframe>`. Резервный контент, который должен отображаться, если браузер не поддерживает встроенные фреймы (например, текстовое описание или гиперссылка на фактическую веб-страницу), помещается между тегами.

Рисунок 13.6 демонстрирует использование внутреннего фрейма (вы также можете найти эту страницу на диске, прилагающемся к книге, в папке *Примеры\Глава_13\внутренние_фреймы*). Белый прямоугольник справа является внутренним фреймом, он отображает другую веб-страницу, которая содержит изображение цветка и текстовое описание.

На рис. 13.7 показаны два снимка одной и той же веб-страницы, на которой разные страницы отображаются в области внутреннего фрейма.

[1] www.adriver.ru

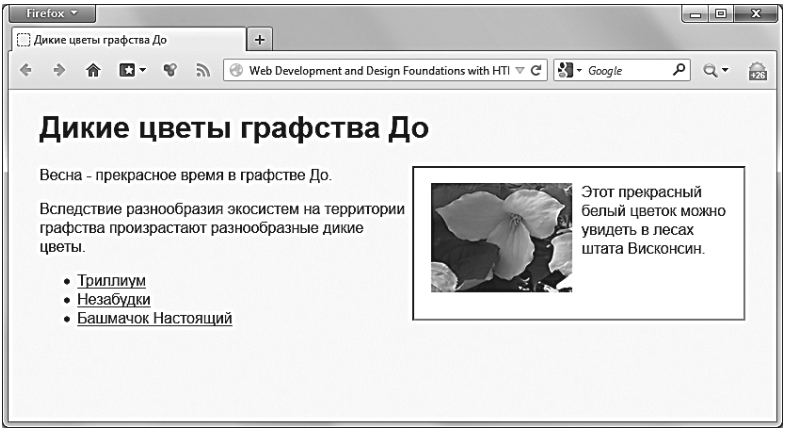


Рис. 13.6. Белый прямоугольник справа на странице — это внутренний фрейм, отображающий другую веб-страницу

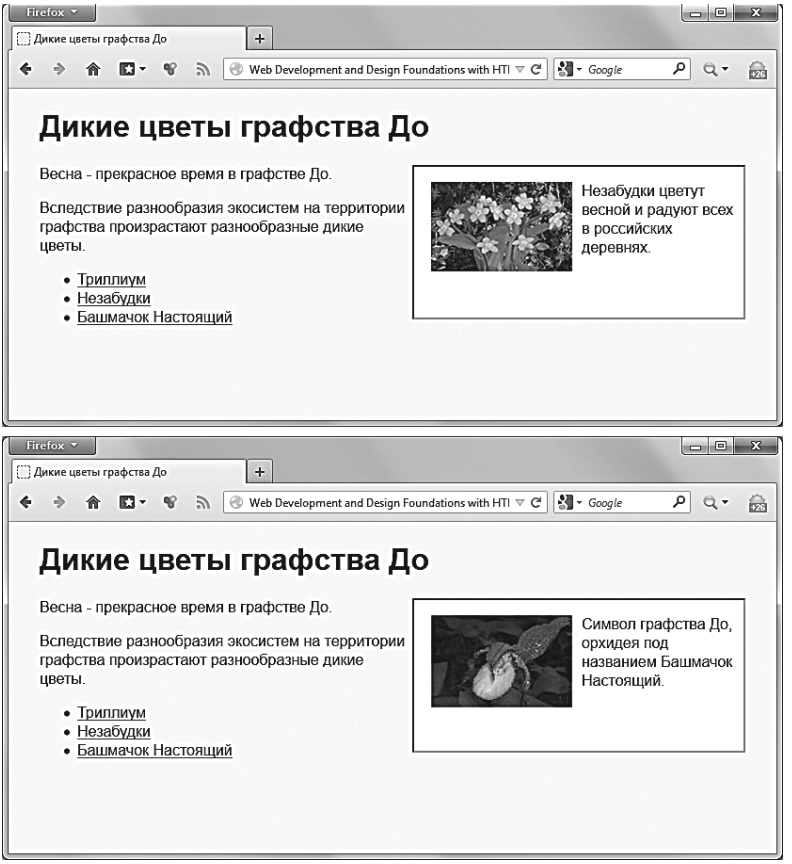


Рис. 13.7. Одна и та же страница с разным контентом в области внутреннего фрейма

Для создания этого эффекта был использован приведенный ниже код внутреннего фрейма:

```
<iframe src="trillium.html" title="Wild Flowers"
height="160" width="350" name="flowers">Описание
весеннего цветка <a href="trillium.html" target="_
blank">Триллиум</a> </iframe>
```

Таблица 13.1 перечисляет атрибуты элемента `iframe`.

Таблица 13.1. Атрибуты элемента `iframe`

Атрибут	Описание
`height`	Высота внутреннего фрейма в пикселах
`id`	Необязателен. Латинские буквы или цифры, без пробелов. Значение должно быть уникальным и не использоваться в других значениях идентификатора в одном и том же HTML-документе
`name`	Необязателен. Латинские буквы и цифры, без пробелов, начинается с буквы. Конфигурирует имя внутреннего фрейма
`sandbox`	Необязателен. Запрещает/отключает функции, такие как плагины, шрифты, формы (нововведение в HTML5)
`seamless`	Необязателен. Установите значение `seamless="seamless"`, чтобы задать более плавное отображение браузером содержимого внутреннего фрейма (нововведение в HTML5)
`src`	URL-адрес веб-страницы, отображаемой во внутреннем фрейме
`title`	Необязателен. Определяет краткое текстовое описание, которое может отображаться браузерами или вспомогательными технологиями
`width`	Ширина внутреннего фрейма в пикселах

Размещение видео с сайта YouTube во встроенном фрейме

YouTube[1] — популярный веб-сайт для обмена видеороликами для личного и делового использования. Когда видео загружается на YouTube, создатель может поделиться своим видео с другими. Отобразить видео с сайта YouTube на вашей веб-странице несложно, просто нажмите кнопку **Поделиться**, а затем **Сгенерировать HTML-код**. Скопируйте и вставьте HTML-код в исходный код вашей страницы. Для отображения файла на вашей веб-странице в коде используется элемент `iframe`. YouTube определяет браузер и операционную систему посетителя вашей веб-страницы и предоставляет контент в соответствующем формате с помощью Flash или видео HTML5.

[1] www.youtube.com

Практическое задание 13.1

В этом практическом задании запустите программу Блокнот (Notepad) и создайте веб-страницу, показанную на рис. 13.8, на которой отображается видео с сайта YouTube в элементе `iframe`.

В этом примере в веб-страницу вставляется видеоролик, находящийся по адресу **http://www.youtube.com/watch?v=A3JC24p0YsA**. Вы можете выбрать любой другой ролик. Зайдите на сайт YouTube и скопируйте идентификатор видеоролика (текст после знака равенства (=) в URL-адресе). В нашем примере идентификатор видео A3JC24p0YsA.

В качестве отправной точки используйте файл из папки *Примеры\Глава_02\template.html.*

Конфигурируйте веб-страницу с заголовком «Видеоролик с сайта YouTube» и элементом `iframe`, отображающим видео. Добавьте в код атрибут `src` со значением `http://www.youtube.com/embed/A3JC24p0YsA`. Как запасной вариант, создайте гиперссылку на страницу YouTube. HTML-код для отображения видео, показанного на рис. 13.8, приведен ниже:

```
<iframe width="560" height="315" src="http://
www.youtube.com/embed/A3JC24p0YsA"
frameborder="0">Смотрите <a href="http://www.youtube.
com/watch?v=A3JC24p0YsA">веселый мульт про кота Саймо-
на</a> </iframe>
```

Сохраните веб-страницу под именем *iframe.html* и отобразите ее в браузере. Протестируйте веб-страницу в различных версиях браузера. Сравните свою работу с рис. 13.8 и образцом *Примеры\Глава_13\iframe.html*, находящемся на прилагающемся к книге диске.

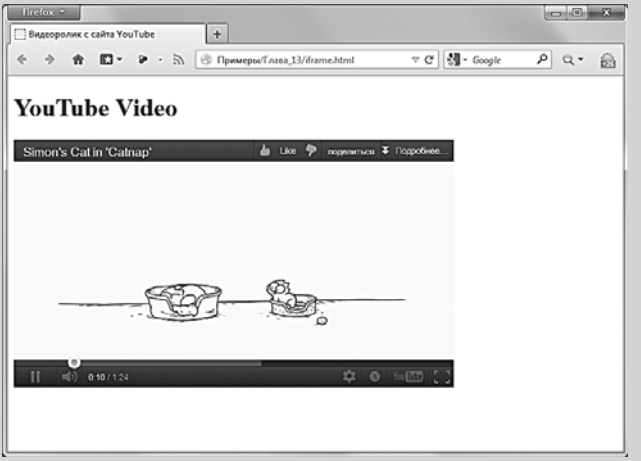


Рис. 13.8. Размещение видео с YouTube

Глава 14
КРАТКИЙ ОБЗОР JAVASCRIPT

Цели главы

В этой главе вы узнаете следующее:

- основные способы применения языка JavaScript в коде веб-страниц;
- предназначение объектной модели документа (DOM, Document Object Model) и несколько основных событий;
- научитесь создавать простые скрипты JavaScript, используя элемент `script` и метод `alert()`;
- научитесь использовать переменные, операторы и изучите структуру оператора `if`;
- научитесь создавать простой скрипт валидации формы.

Если во время вашего серфинга во Всемирной паутине волшебным образом появляется всплывающее окно, это значит, что вы столкнулись с JavaScript. JavaScript — это скриптовый язык программирования, а команды JavaScript могут быть включены в HTML-файл. Используя язык JavaScript, вы можете объединить техники и эффекты, которые оживят ваши веб-страницы. Вы можете отобразить окно с предупреждением, содержащее важное сообщение для пользователей. Вы можете сделать так, чтобы отображалось изображение, когда пользователь наводит указатель мыши на ссылку, и еще многое другое. В этой главе мы познакомим вас с языком программирования JavaScript и некоторыми его возможностями, а также приведем примеры, которые вы сможете использовать для создания ваших собственных веб-страниц.

14.1. Обзор языка программирования JavaScript

Существует целый ряд способов добавить на веб-страницу интерактивность. Как вы узнали из главы 7, можно применить каскадные таблицы стилей для достижения эффекта, возникающего при наведении указателя мыши на гипертекстовую ссылку. В главе 11 вы видели примеры, как мо-

жет быть использована мультимедийная платформа Adobe Flash для добавления интерактивности и анимации на веб-страницу. Кроме того, CSS используются для создания интерактивных эффектов, в том числе галереи изображений, преобразований и переходов (CSS3). Из главы 11 вы также узнали, что JavaScript также может быть использован для тех же целей.

Итак, что такое JavaScript? Это ***объектно-ориентированный скриптовый язык***, работающий на стороне клиента, интерпретируемый веб-браузером. Язык JavaScript рассматривается как объектно-ориентированный, поскольку его используют для работы с ***объектами*** документа веб-страницы: окном браузера, самим документом и такими элементами, как формы, изображения и ссылки. Поскольку язык JavaScript интерпретируется браузером, то он является скриптовым языком, работающим на стороне клиента. Скриптовый язык — это разновидность языка программирования, но не стоит паниковать: вам необязательно быть программистом, чтобы в нем разобраться.

Рассмотрим клиентов и серверы. В главе 10 мы обсудили хостинг веб-сайта на веб-сервере. Как вы уже знаете, провайдер веб-хостинга хранит у себя ваш веб-сайт и позволяет вам загружать ваши файлы на веб-сервер. Посетители вашего сайта (их также называют пользователями) могут просмотреть ваш веб-сайт с помощью браузера, введя URL, предоставляемый вашим провайдером хостинга. Как вы помните, веб-браузер пользователя называется клиентом.

Язык программирования JavaScript интерпретируется клиентом. Это означает, что код JavaScript, встроенный в HTML-документ, будет преобразован для просмотра веб-браузером. Работа сервера заключается в том, чтобы отправить HTML-документ. Работа веб-браузера — интерпретировать код в HTML-файл и, соответственно, отобразить веб-страницу. Поскольку все операции осуществляются клиентом (в данном случае веб-браузером), то это является ***обработкой на стороне клиента***. Существуют языки программирования, которые используются на сервере, они называются языки программирования на стороне сервера. ***Обработка на стороне сервера*** может включать в себя отправку электронной почты, хранение объектов в базе данных или направление объектов в корзину для покупок в интернет-магазине. Из главы 9 вы узнали, как задать форме набор действий, чтобы направить ее скрипту на стороне сервера.

Итак, JavaScript — это объектно-ориентированный скриптовый язык, работающий на стороне клиента и интерпретируемый веб-браузером.

Код JavaScript встраивают в HTML-файл, а веб-браузер интерпретирует и отображает необходимые результаты.

14.2. Развитие языка программирования JavaScript

Есть распространенное заблуждение, что Java и JavaScript — это одно и то же. Java и JavaScript — это совершенно разные языки, у которых очень мало общего. Как было сказано в главе 11, Java — это объектно-ориентированный язык программирования. Java — это технически очень сложный язык программирования, который может быть использован для создания больших приложений для бизнеса, например, для разработки системы инвентаризации и системы расчета заработной платы. Компания Sun Microsystems (в конце января 2010 года вошла в состав компании Oracle) создала язык программирования Java в 90-х, этот язык был предназначен для работы в операционных системах Windows или Unix.

Разработчики языка программирования Java также стремились к гибкости и популярности, которая могла быть достигнута, если бы их язык смог работать в веб-браузерах. Независимо от них команда компании Netscape разрабатывала скриптовый язык под названием LiveScript и со временем стала партнером компании Sun Microsystems. Это партнерство было взаимовыгодным, поскольку привело к созданию плагина Java, который наделял веб-браузеры способностью работать с Java-апплетами в браузерах. Развитие языка программирования LiveScript продолжалось, и в дальнейшем он был переименован в JavaScript. Однако JavaScript — это не то же самое, что и язык программирования Java. JavaScript намного проще, чем Java. У этих двух языков больше различий, чем общего.

14.3. Популярные приемы использования JavaScript

Способы использования языка программирования JavaScript варьируются от создания различных украшений, таких как простая анимация или дизайнерские меню, до функционального применения, такого как всплывающие окна, содержащие информацию о продукте, нахождение ошибок в форме или определение версии браузера. Давайте рассмотрим некоторые примеры его использования.

Сообщение с предупреждением

Сообщение с предупреждением — это популярная техника, используемая, чтобы привлечь внимание пользователя к происходящему событию. Например, интернет-магазин может использовать сообщения с предупреждениями, чтобы информировать об ошибках в форме заказа или чтобы напомнить пользователям о предстоящей распродаже.

Рисунок 14.1 демонстрирует сообщение, напоминающее пользователю ввести название фотоальбома до того, как начнется загрузка фотографий

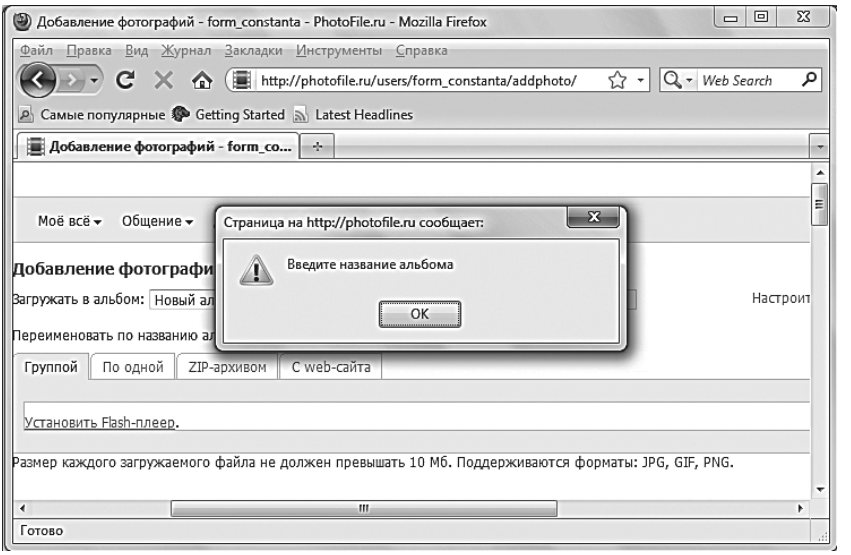


Рис. 14.1. Сообщение о том, чтобы посетитель ввел название альбома до загрузки фотографий на сайт

Обратите внимание, что пользователь должен нажать кнопку **ОК** перед следующим шагом. Всплывающие окна эффективно привлекают внимание пользователей, но быстро начинают раздражать, если их используют слишком часто.

Всплывающие окна

Всплывающие окна — это окна веб-браузера, которые могут появиться, когда вы совершаете какие-либо действия на веб-странице (нажимаете на изображения, наводите указатель мыши на определенную

область веб-страницы) или просто возникающие загадочным образом. У этой техники есть несколько обоснованных вариантов использования, например, всплывающее информационное окно, содержащее изображение большего размера и описание товара, когда пользователь щелкает мышью по товару в основном окне. К сожалению, использование всплывающих окон было настолько агрессивным, что большинство браузеров позволяет пользователям их блокировать. Это означает, что полезные всплывающие окна также не будут показаны.

Рисунок 14.2 демонстрирует всплывающее окно, которое появляется, когда пользователь щелкает мышью по ссылке на главной странице.

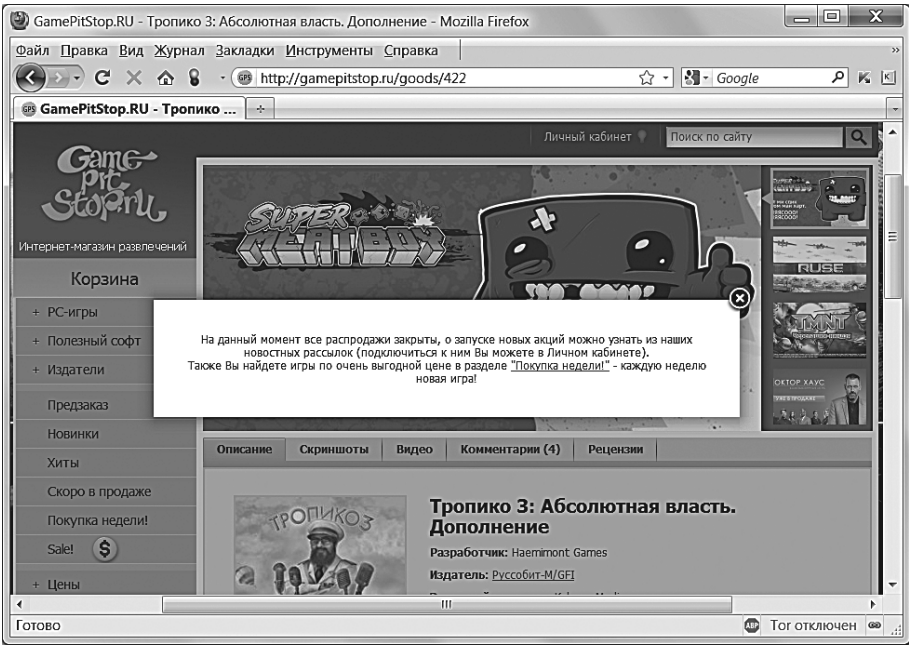


Рис. 14.2. Всплывающее окно меньшего размера появляется, когда пользователь щелкает мышью по ссылке в окне обычного размера

Jump-меню

JavaScript также может быть использован для создания ***Jump-меню***, основанных на раскрывающихся списках. Пользователь может выбрать веб-страницу из раскрывающегося списка и перейти на нее.

Рисунок 14.3 демонстрирует эту технику. В этом примере пользователь выбрал пункт **Контакты** в раскрывающемся списке. Страница «Контакты» загрузится окне браузера.

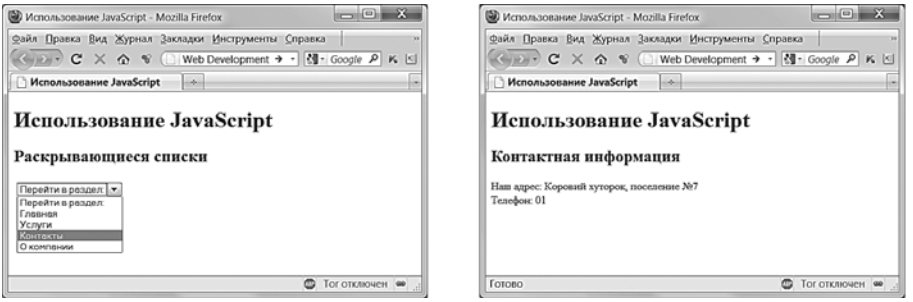


Рис. 14.3. Jump-меню с выбранной страницей «Контакты», которая открылась на новой вкладке

Техники перемещения мыши

Язык программирования JavaScript может быть использован для осуществления задач, выполняющихся при перемещении мыши в окне браузера.

Одна из популярных техник заключается в отображении подменю, когда пользователь наводит указатель мыши на пункт меню. Рисунок 14.4 показывает эту технику.

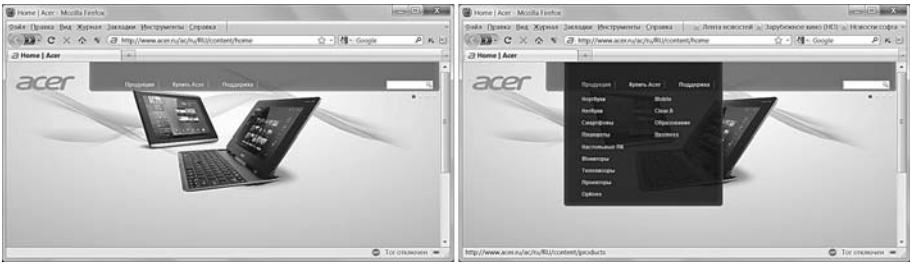


Рис. 14.4. Основное меню и подменю

Окно слева показывает основное меню, а окно справа показывает подменю, отображающееся, когда пользователь устанавливает указатель мыши на пункт меню **Продукция**. Когда пользователь перемещает мышь в сторону от пункта меню **Продукция**, подменю исчезает. Эта техника также используется для смены графических изображений при создании иллюзии нажатия кнопки или смены цвета пункта меню. В случае изменения изображения, также известного как ***ролловер-эффект***, изображение отображается на веб-странице в то время, как веб-страница загружается. Когда пользователь наводит указатель мыши на изображе-

ние, оригинальное изображение меняется на новое. Когда пользователь отодвигает указатель мыши с изображения, появляется оригинальное изображение.

Эта техника широко используется для элементов панели навигации, а также для интересных эффектов. Рисунок 14.5 показывает технику изменения графического изображения.

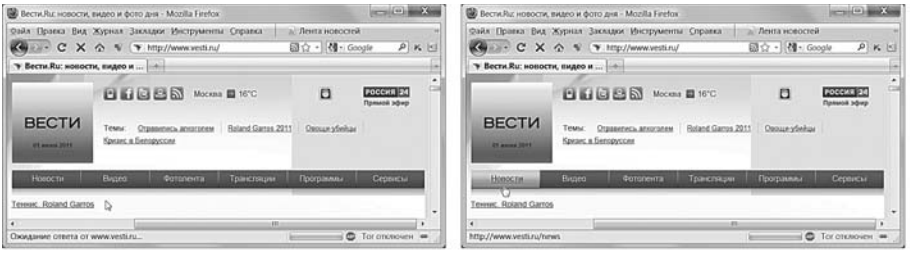


Рис. 14.5. Оригинальное изображение слева и измененное изображение справа после того, как на него был наведен указатель мыши

В этом разделе мы рассказали о некоторых основных моментах и концепциях использования языка программирования JavaScript. Мы создадим несколько скриптов, позволяющих выводить сообщения с предупреждениями, реагировать на движения указателя мыши и используем некоторые техники, предназначенные для валидации форм на веб-страницах. Эта глава лишь поверхностно рассматривает JavaScript, но вы сможете разобраться в некоторых приемах.

14.4. Добавление в веб-страницу кода JavaScript

Код JavaScript встраивается в HTML-страницу и интерпретируется веб-браузером. Это означает, что веб-браузер может понять код и обработать его. В примерах в этой главе используется браузер Mozilla Firefox. Код, который мы создадим, будет работать в большинстве браузеров. Тем не менее мы будем использовать браузер Firefox, поскольку он выведет сообщения об ошибках, которые очень полезны в процессе создания и тестирования страниц. Если вы еще не установили браузер Mozilla Firefox на ваш компьютер, скачайте его бесплатно по ссылке **www.mozilla.com/ru/firefox/**. Для отображения ошибок и анализа кода также понадобится расширение Web developer. Установить его можно на веб-странице **chrispederick.com/work/web-developer**.

Элемент `script`

Когда код JavaScript встраивают в HTML-документ, он должен находиться между контейнерными тегами `<script>` и `</script>`. Вебстраницы визуализируются браузером сверху вниз. Особенностью наших скриптов является то, что они будут выполняться, где бы они ни находились в документе, мы это увидим далее. Для согласованной работы с HTML элемент `script` должен включать в себя атрибут для определения скриптового языка — `"text/javascript"`.

Шаблон блока операторов JavaScript

Между тегов `script` мы добавим комментарии, до и после определения блока JavaScript. ***Комментарии*** находятся между символами разметки `<!--` и `-->`. Символ `<!--` обозначает начало комментария, а `-->` — его конец. Комментарий JavaScript начинается с двойного слеша `//`. Браузеры игнорируют текст этой строки. Мы будем использовать тег комментариев в начале блока операторов JavaScript, чтобы скрыть JavaScript от устаревших браузеров. Некоторые устаревшие версии браузеров будут отображать сам код JavaScript, а не выполнять его. Вставка блока JavaScript в теги комментариев XHTML прячет этот блок от устаревших браузеров, и они игнорируют этот код, если не поддерживают. Каждый блок JavaScript должен иметь следующую структуру:

```
<script type="text/javascript">
<!--
... Здесь располагается код JavaScript ...
// -->
</script>
```

Код JavaScript вы размещаете внутри блока операторов. Этот блок может находиться в любой части HTML-документа, код все равно будет выполнен. Давайте посмотрим, как он работает в практическом задании, где код будет выводить сообщение с предупреждением.

Окно с сообщением

Окно с сообщением появляется благодаря использованию метода `alert()` со следующей структурой:

```
alert("сообщение");
```

Каждая команда JavaScript обычно заканчивается точкой с запятой (`;`). JavaScript также ***чувствителен к регистру***, поэтому существует разница между строчными и заглавными буквами, и очень важно точно указать код JavaScript.

Практическое задание 14.1

В этом практическом задании вы создадите простой скрипт с диалоговым окном `alert()`.

Запустите программу Блокнот (Notepad). Введите приведенный ниже HTML-код и JavaScript. Обратите внимание, что в значении `alert()` нет пробела между словом `alert` и символом открывающей круглой скобки.

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Пример работы с JavaScript</title>
<meta charset="utf-8">
</head>
<body>
<h1>Пример работы с JavaScript</h1>
<script type="text/javascript">
<!--
alert("Добро пожаловать на мою страницу!");
// -->
</script>
<h2>Окно с предупреждением</h2>
</body>
</HTML>
```

Сохраните этот файл с именем *alert.html*. Запустите браузер Firefox и откройте файл *alert.html*, чтобы протестировать вашу страницу. Обратите внимание, что появляется первый заголовок, а потом открывается всплывающее окно с сообщением, как показано на рис. 14.6. После того как вы нажмете кнопку **OK**, появляется вторая строка текста — заголовок `h2`. Этот пример иллюстрирует нисходящую обработку кода веб-страницы. Блок JavaScript находится между строками текста и именно после первой строки появляется сообщение.

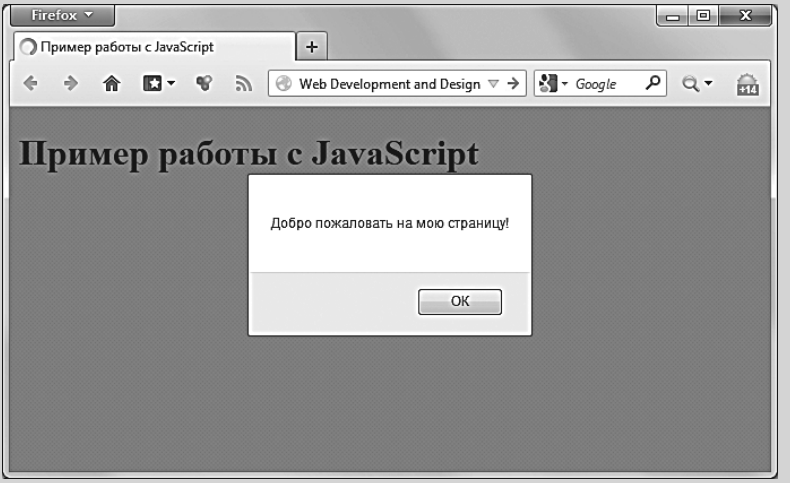


Рис. 14.6. Файл *alert.html* отображает окно с сообщением JavaScript

Практикум отладки

Иногда ваш код JavaScript не работает сразу, когда вы его тестируете. Когда это происходит, вам нужно ***отладить*** код, т.е. найти ошибки и исправить их.

Рассмотрим технику отладки. Измените код JavaScript, чтобы в нем появилась типичная опечатка, как показано ниже:

```
aalert("Приветствую вас на моей страничке!");
```

Сохраните файл и снова откройте страницу в браузере. Обратите внимание, что на этот раз окно с сообщением не отображается. Браузер Firefox укажет на некоторые ошибки в коде JavaScript, для чего нужно открыть окно **Консоль ошибок** (Error Console). Данное окно доступно, если установлено и включено расширение Web developer.

В браузере Firefox на панели Web developer выберите команду меню **Инструменты** ⇒ **Консоль ошибок** (Tools ⇒ Error Console). Откроется окно **Консоль ошибок** (Error Console) с сообщением об ошибке (вторая строка), как показано на рис. 14.7.

Обратите внимание, что ошибка отображается вместе с номером строки (прокрутите содержимое окна вправо), в которой она была обнаружена. Полезно верстать ваши документы в редакторе, который отображает номера строк, но это не обязательно. Если вы используете программу Блокнот (Notepad), примените функцию **Перейти** (Go To) в меню **Правка** (Edit).

Исправьте ошибку в файле *alert.html* и снова протестируйте страницу в браузере. На этот раз скрипт должен быть выполнен успешно.

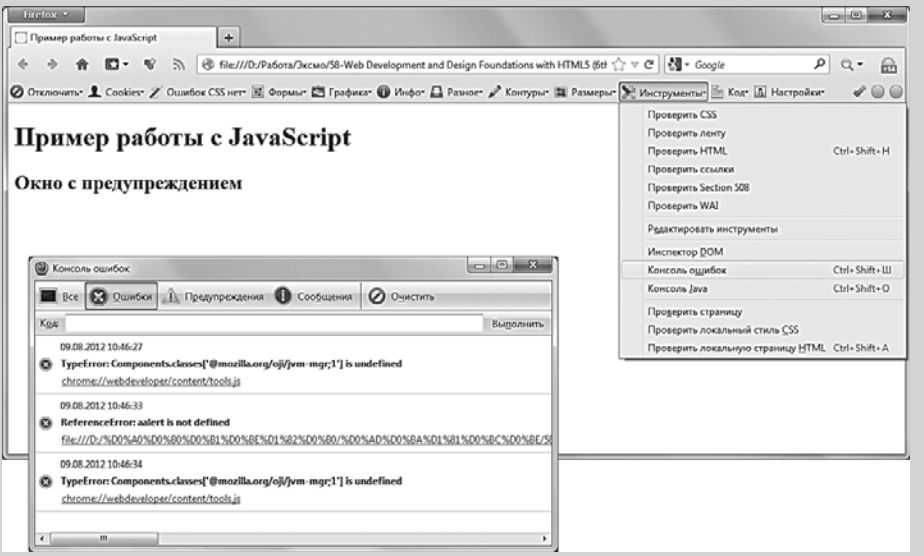


Рис. 14.7. Окно **Консоль ошибок** (Error Console) расширения Web developer отображает ошибку

ЧаВо

БУДЕТ ЛИ КОНСОЛЬ С ОШИБКАМИ ОТОБРАЖАТЬ ВСЕ ОШИБКИ МОЕГО КОДА JAVASCRIPT?

Будут отображаться синтаксические ошибки, включающие в себя такие, как недостающая скобка и элементы, которые консоль не может опознать. Иногда ошибка находится выше указанной строки, в частности, если не хватает скобки или цитаты. Отображаемые ошибки обозначают, что что-то не так, и направляют вас в позицию, в которой может находиться ошибка. Начните с проверки указанной строки и, если на ней нет ошибок, взгляните на строку над ней.

14.5. Краткий обзор объектной модели документа

Язык программирования JavaScript может управлять объектами HTML-документа, которые включают в себя контейнерные элементы, такие как `p`, `span` или `div`. Объекты также включают в себя изображения, формы и отдельные элементы форм, такие как текстовые поля и раскрывающиеся списки. Чтобы получить доступ к этим объектам, вам нужно иметь представление об объектной модели документа (DOM).

В целом, ***объект*** — это сущность или «предмет». При использовании объектной модели документа окно браузера, документ веб-страницы и любой HTML-элемент считаются объектами. Окно браузера — это объект. Когда веб-страница загружается в браузере, она рассматривается как документ. Сам документ — это объект. Документ может содержать такие объекты, как изображения, заголовки, параграфы и отдельные элементы формы, например, текстовые поля. Объекты могут иметь свойства, которые можно указывать и управлять ими. Например, свойство документа — это его название. Другое свойство документа — это цвет фона.

Есть несколько действий, которые можно выполнить по отношению к объектам. Например, ***объект окно*** может отображать сообщение с предупреждением или отображать подсказку. Эти действия называются ***методами.*** Команда отображения сообщения с предупреждением относится к методу объекта окно. Объектная модель документа (DOM) — это коллекция объектов, свойств и методов. Язык JavaScript использует объектную модель документа (DOM) для определения и управления элементами в HTML-документе.

По-другому взглянем на эту систему объектов, свойств и методов. Представьте, что ваша машина — это объект. У нее есть свойства, такие как цвет, производитель и год. У вашей машины есть элементы, такие как капот и багажник. Капот и багажник можно открыть и закрыть. Если бы нам нужно было использовать язык программирования для того, чтобы открыть и закрыть капот и багажник, то команды могли выглядеть следующим образом:

```
машина.капот.открыть()
машина.капот.закрыть()
машина.багажник.открыть()
машина.багажник.закрыть()
```

Если бы мы хотели узнать цвет, год и производителя машины, команды могли выглядеть следующим образом:

```
машина.цвет
машина.год
машина.модель
машина.производитель
```

Когда мы используем значения, `машина.цвет` может быть равно `"синий"` и `машина.производитель` может быть равно `"Форд"`, а `машина.`

модель может быть равной "Фокус". Мы можем поменять значения или только прочитать их, не внося изменений. В этом примере машина — это объект, а его свойства — капот, багажник, цвет, год и производитель. Капот и багажник могут также быть рассмотрены как свойства.

Открыть и закрыть — это методы капота и методы багажника. Для объектной модели документа (DOM) мы можем написать документ, используя метод `write()`. Структура следующая:

```
document.write("текст документа");
```

Мы можем использовать язык JavaScript, чтобы написать текст и HTML-теги в документе, а браузер преобразует их для просмотра.

Метод `alert()` был использован в практическом задании, это метод объекта окна. Его можно указать следующим образом:

```
window.alert("сообщение");
```

Предполагается, что объект окна уже существует и может быть опущен. Если окно не существует, скрипт также не существует.

Одним из свойств документа является `lastModified`. Это свойство содержит данные о том, когда файл в последний раз был сохранен или изменен, и мы можем получить к нему доступ, используя `document.lastModified`. Эта информация доступна только для чтения, и мы можем записать ее в документ или использовать для других целей.

Практическое задание 14.2

В этом практическом задании вы будете практиковаться применять метод `write()` документа и свойство `lastModified`. Вы будете использовать `document.write()` для того, чтобы добавить текст и HTML-теги в документ, а также для записи данных о том, когда файл был в последний раз сохранен.

Откройте документ *alert.html* и отредактируйте блок скрипта следующим образом:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Пример работы с JavaScript</title>
<meta charset="utf-8">
</head>
```

```
<body>
<h1>Пример работы с JavaScript</h1>
<script type="text/javascript">
<!--
document.write("<p>Использование document.write для
добавления текста</p>");
document.write("<h2>Мы также можем добавить
и HTML-теги!</h2>");
// -->
</script>
<h3>Последнее изменение документа:
<script type="text/javascript">
<!--
document.write(document.lastModified);
// -->
</script>
</h3>
</body>
</HTML>
```

Сохраните этот документ с именем *documentwrite.html* и откройте его в браузере. В окне браузера должен появиться текст, как показано на рис. 14.8. Если текст не появляется, откройте окно **Консоль ошибок** (Error Console) и исправьте ошибки.

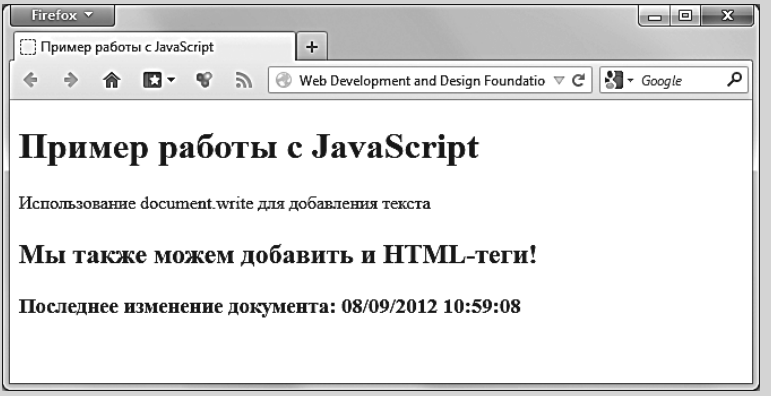


Рис. 14.8. Документ *documentwrite.html* в браузере Firefox

Скрипт JavaScript можно посмотреть в исходном коде, выбрав в браузере команду **Вид** ⇒ **Исходный код страницы** (View ⇒ Page

Source). Закройте окно с исходным кодом, когда закончите проверку кода. Образец файла *documentwrite.html* вы найдете на диске, прилагающемся к книге, в папке *Примеры\Глава_14*.

ЧаВо

ЗАЧЕМ МНЕ ИСПОЛЬЗОВАТЬ МЕТОД `DOCUMENT.WRITE`, ЕСЛИ Я МОГУ ПРОСТО СВЕРСТАТЬ HTML-КОД?

На практике вы обычно не будете использовать метод `document.write` при верстке вашей веб-страницы, если требуется лишь HTML-код. Метод `document.write` применяется вместе с другими техниками. Например, вы можете использовать JavaScript для определения времени и, если посетитель появился на сайте до полудня, применить метод `document.write`, чтобы на странице отображалось приветствие «Доброе утро». Аналогичным образом указываются приветствия «Добрый день» и «Добрый вечер» (после 18:00).

14.6. События и обработчики событий

Когда пользователь просматривает веб-страницу, браузер обнаруживает движение мыши и события. ***События*** — это действия, предпринимаемые посетителями веб-страницы, такие как щелчок мышью, загрузка страницы или отправка данных формы. Например, когда вы перемещаете указатель мыши по гиперссылке, браузер определяет событие, связанное с движением мыши. Таблица 14.1 перечисляет несколько событий и их описание.

Таблица 14.1. События и их описание

Событие	Описание
`click`	Пользователь щелкает мышью по объекту. Это может быть изображение, гиперссылки или кнопка
`load`	Браузер отображает веб-страницу
`mouseover`	Пользователь наводит указатель мыши на объект. Указатель мыши не обязательно должен оставаться на объекте. Это может быть гиперссылка, изображение, параграф или другой объект
`mouseout`	Пользователь перемещает указатель мыши с объекта, на который до этого он был установлен
`submit`	Пользователь нажимает кнопку отправки формы
`unload`	Веб-страница загружается в браузере. Это событие выполняется непосредственно перед загрузкой новой веб-страницы

Когда событие происходит, оно может инициировать выполнение некоторого кода JavaScript. Одна широко распространенная техника заключается в том, что перемещение указателя мыши на/с объект изменяет изображение или раскрывает меню.

Нам нужно определить, какое событие должно инициировать действия и что произойдет, когда это событие наступит. Мы можем использовать ***обработчик событий*** для указания целевого события. Обработчик событий встраивается в HTML-тег как атрибут и определяет какой-либо код JavaScript для дальнейшего выполнения, когда событие наступает. Имена обработчиков событий состоят из имени события с префиксом `on`. В таблице 14.2 перечислены обработчики, соответствующие событиям, описанным в табл. 14.1. Например, событие **`onload`** наступает, когда браузер визуализирует страницу (загружает ее). Когда вы наводите указатель мыши на текстовую гиперссылку, происходит событие **`mouseover`**, и оно определяется браузером. Если эта гиперссылка содержится в коде обработчика событий `onmouseover`, то будет выполнен код JavaScript, заданный обработчиком. Этот код может вывести сообщение с предупреждением, отобразить изображение или раскрыть меню. Другие обработчики событий, такие как **`onclick`** и **`onmouseout`** могут инициировать выполнение кода JavaScript при наступлении соответствующего события.

Таблица 14.2. События и обработчики событий

Событие	Обработчик событий
`click`	`onclick`
`load`	`onload`
`mouseover`	`onmouseover`
`mouseout`	`onmouseout`
`submit`	`onsubmit`
`unload`	`onunload`

Практическое задание 14.3

Попрактикуемся использовать обработчики событий `onmouseover` и `onmouseout`, а также выводить сообщение с предупреждением для определения момента, когда обработчик событий инициирован. Мы будем использовать простые гипертекстовые ссылки и вставлять обработчики событий в теги `a`. Нам не понадобится блок `script`, поскольку обработчики событий размещены в виде атрибутов в HTML-тегах. Мы поместим гипертекстовые ссылки в неупоря-

доченный список, чтобы в окне браузера было больше пространства для перемещения указателя мыши и тестирования скрипта.

Откройте текстовый редактор и введите приведенный ниже код. Обратите внимание на использование двойных и одиночных кавычек в обработчике событий `onmouseover` и `onmouseout`. Требуются кавычки вокруг сообщения в методе `alert()` и кавычки, инкапсулирующие обработчик событий в коде JavaScript. Языки HTML и JavaScript позволяют использовать либо двойные кавычки, либо одинарные. По правилам они должны совпадать. Поэтому в ситуации, когда вам нужно два набора кавычек, вы можете использовать и двойные, и одинарные. В этом случае мы использовали двойные кавычки для внешнего набора и одинарные кавычки для внутреннего набора. В теге привязки символ # используется в качестве значения атрибута `href`, потому что нам не требуется загрузка какой-либо другой веб-страницы. Необходима лишь гиперссылка, чтобы связать события `mouseover` и `mouseout`.

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Пример работы с JavaScript</title>
<meta charset="utf-8">
</head>
<body>
<h1>Пример работы с JavaScript</h1>
<ul>
<li><a href="#" onmouseover="alert('Указатель мыши находится на ссылке');">Тест события mouseover</a>
</li>
<li><a href="#" onmouseout="alert('Указатель мыши находится вне ссылки');">Тест события mouseout</a>
</li>
</ul>
</body>
</HTML>
```

Сохраните файл с именем *mouseovertest.html* и откройте его в браузере. Установите указатель мыши на ссылку с текстом «Тест события mouseover». Как только указатель вашей мыши будет помещен поверх ссылки, произойдет событие `mouseover` и запустится обработчик событий `onmouseover`. Он отобразит окно с предупреждением, показанное на рис. 14.9.

Нажмите кнопку **OK** и установите указатель мыши на текстовую ссылку «Тест события mouseout». Обратите внимание на то, что ничего не происходит. Это потому, что событие `mouseout` еще не произошло.

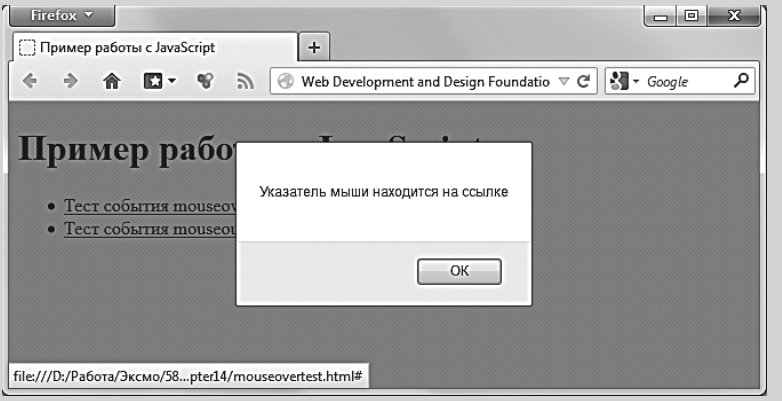


Рис. 14.9. Демонстрация события `onmouseover`

Сместите указатель мыши от ссылки. Как только указатель мыши больше не находится на ссылке, активируется событие `mouseout` и запустится обработчик событий `onmouseout`. Появится окно с предупреждением, как показано на рис. 14.10.

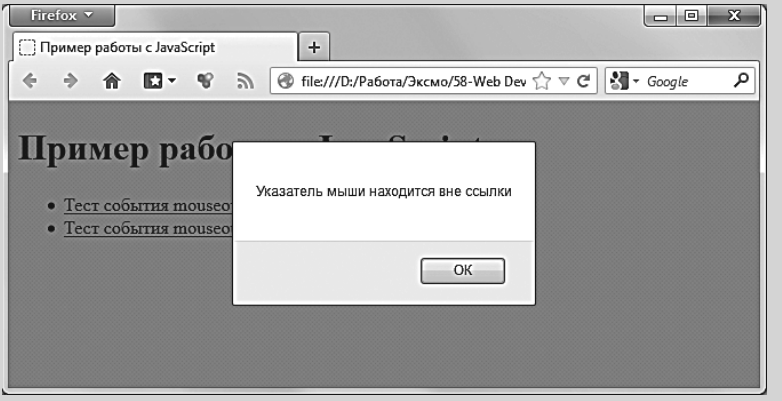


Рис. 14.10. Демонстрация события `onmouseover` в документе *mouseovertest.html*

Образец файла *mouseovertest.html* вы найдете на диске, прилагающемся к книге, в папке *Примеры\Глава_14*.

Вы можете объединять обработчики событий в одной гипертекстовой ссылке. Так работает техника смены изображений. Обработчик событий `onmouseover` меняет изображение на новое, а обработчик

событий `onmouseout` возвращает исходное изображение. Эта техника выходит за рамки главы, но, возможно, данное практическое задание прольет немного света на то, как производится эффект смены изображений.

14.7. Переменные

Иногда нам нужно собрать данные от пользователей и произвести с ними какие-либо действия. Простой пример — запрос у пользователя имени и запись имени в документ. Мы будем хранить имя в ***переменной***. Когда вы изучали математику, то использовали переменные X и Y в уравнении взамен каких-либо значений. Этот же принцип применяется при использовании переменных в языке программирования JavaScript. Переменные JavaScript также являются заменой для данных, и их значение может меняться. В сложных языках программирования, таких как C++ и Java, существует много правил переменных и их типов данных. JavaScript в этом вопросе предоставляет свободу. Вам не придется беспокоиться о том, какой тип данных содержится в переменной.

ЧаВо

СУЩЕСТВУЮТ ЛИ РЕКОМЕНДАЦИИ ПО СОЗДАНИЮ ИМЕН ПЕРЕМЕННЫХ?

Это своего рода форма искусства, но прежде всего вам нужно будет придумать имя переменной, которое описывает содержащиеся в нем данные. Символ нижнего подчеркивания или заглавные буквы могут быть использованы для удобства чтения, если вы используете более одного слова. Не используйте другие специальные символы. Ограничьтесь латинскими буквами и цифрами. Будьте осторожны, не используйте ***зарезервированные*** или ***ключевые слова*** языка программирования JavaScript, такие как `var`, `return`, `function` и т. д.

Список ключевых слов JavaScript можно найти на диске, прилагающемся к книге, в файле *Примеры\Глава_14\Зарезервированные слова JavaScript.pdf*. Далее представлены некоторые имена переменных, которые могут быть использованы для кода продукта:

- `productCode`;
- `prodCode`;
- `product_code`.

Создание переменной для веб-страницы

Перед тем как мы начнем использовать переменную, мы можем представить ее с помощью ключевого слова **var**. Этот шаг не обязателен, но рекомендован. Мы можем назначить данные переменной, используя оператор присваивания — символ равно (=).

Переменная может содержать число или строку. Строка должна быть заключена в кавычки и может содержать буквы алфавита, пробелы, цифры и специальные символы. Например, строка может быть фамилией, адресом электронной почты, почтовым адресом, кодом продукта или параграфом информации. Далее вы выполните практическое задание по назначению данных для переменной и запишем это в документ.

Практическое задание 14.4

В этом практическом задании вы объявите переменную, присвоите ей строку с данными, и запишите это в документ.

Откройте текстовый редактор и введите код:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Пример работы с JavaScript</title>
<meta charset="utf-8">
</head>
<body>
<h1>Пример работы с JavaScript</h1>
<h2>Привет,
<script type="text/javascript">
<!--
var userName;
userName = "Олеся";
document.write(userName);
// -->
</script>
</h2>
</body>
</HTML>
```

Обратите внимание, что тег `<h2>` расположен перед блоком скрипта, а тег `</h2>` — после. Таким образом, значение `userName` при просмотре будет отформатировано как заголовок `<h2>`. Также указаны запятая и один пробел после «т» в слове «Привет». Если вы не укажите этот пробел, то значение `userName` будет отображено сразу после запятой.

Обратите внимание, что переменная написана в смешанном регистре. Эта конвенция используется во многих языках программирования для удобства чтения кода. Некоторые разработчики могут использовать символ нижнего подчеркивания, например `user_name`. Выбор имени переменной — это своего рода искусство, но постарайтесь выбирать имена, которые обозначают содержание переменной.

Также обратите внимание, что метод `document.write()` не содержит кавычек. Контент переменной будет записан в документ. Если бы мы использовали кавычки вокруг имени переменной, то оно само было бы записано в документ, а не контент переменной.

Сохраните документ с именем *variablewrite.html* и откройте его в браузере.

Рисунок 14.11 демонстрирует файл *variablewrite.html* в браузере.

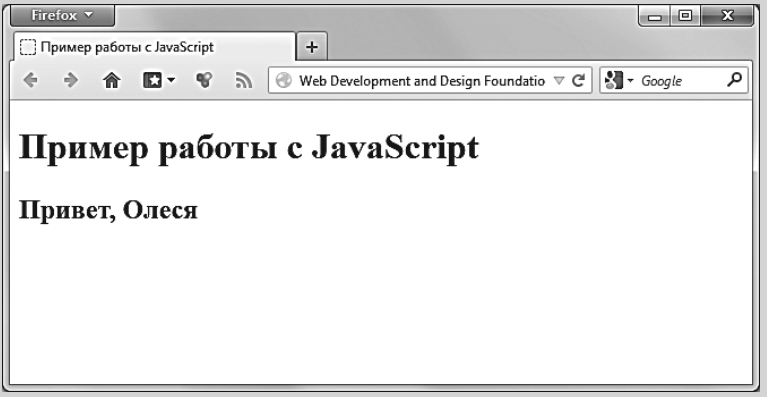


Рис. 14.11. Документ *variablewrite.html,* отображенный в браузере

Разделение заголовка `h2` так, чтобы он был размещен до и после скрипта, несколько обременительно. Мы можем объединить строчки, используя символ `+`. Позже в этой главе вы увидите, что символ `+` также может использоваться для добавления чисел.

Объединение строк с использованием символа `+` называется ***конкатенацией***. Мы соединим строку данных `<h2>` с значением `username` и тегом `</h2>`.

Отредактируйте документ *variablewrite.html* следующим образом:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Пример работы с JavaScript</title>
<meta charset="utf-8">
</head>
<body>
<h1>Пример работы с JavaScript</h1>
<script type="text/javascript">
<!--
var userName;
userName = "Олеся";
document.write("<h2>Привет, " + userName + "</h2>");
// -->
</script>
</body>
</HTML>
```

Убедитесь, что убрали элемент `h2` с данными над и под блоком скрипта. Сохраните файл с именем *variablewrite2.html* и откройте его в браузере. Никаких изменений на странице в браузере не должно быть.

Сбор значений переменной с помощью метода `prompt()`

Для демонстрации интерактивных возможностей языка программирования JavaScript и переменных мы будем использовать метод `prompt()` для запроса данных у пользователя и записи этих данных в веб-страницу.

В качестве примера мы продолжим практическое задание 14.4: запросим имя у пользователя, а не будем вводить эти данные в переменную `userName`.

`prompt()` — это метод объекта окна. Мы могли бы использовать метод `window.prompt()`, но объект окна обозначен, поэтому мы укажем его так: `prompt()`. Метод `prompt()` может вывести сообщение пользо-

вателю. Этот метод обычно используется в связке с переменной, так что входящие данные хранятся в переменной. Структура выглядит следующим образом:

```
someVariable = prompt("сообщение запроса");
```

Когда эта команда выполняется, появляется окно с запросом, которое отображает сообщение и поле ввода данных. Пользователь вводит данные в поле ввода, нажимает кнопку **ОК** и данные отправляются в переменную.

Теперь мы добавим эту функцию в файл *variablewrite2.html*.

Практическое задание 14.5

В этом практическом задании вы будете использовать метод `prompt()` для сбора информации от пользователя и ее записи в документ.

Отредактируйте код скрипта в файле *variablewrite2.html* следующим образом:

```
<script type="text/javascript">
<!--
var userName;
userName = prompt("Пожалуйста, введите свое имя");
document.write("<h2>Привет, " + userName + "</h2>");
// -->
</script>
```

Изменилась только команда запроса данных переменной — `userName`. Данные, введенные пользователем, будут связаны с переменной `userName`.

Сохраните файл с именем *variablewrite3.html* и откройте его в браузере. Появится диалоговое окно с запросом, в поле которого можно ввести имя и нажать кнопку **ОК**, как показано на рис. 14.12. Имя должно появиться в окне браузера.

Теперь создадим вариацию этого скрипта и позволим пользователям ввести название цвета. Предпочтение пользователя будет использовано в качестве фона документа. Мы используем свойство объекта документа `bgColor` для установки цвета, который предпочтет пользователь. Обратите внимание на код этого примера и убедитесь, что буква «C» написана в верхнем регистре, когда будете имя переменной `bgColor`. Кроме того, следует учитывать, что названия цветов надо вводить на английском языке.

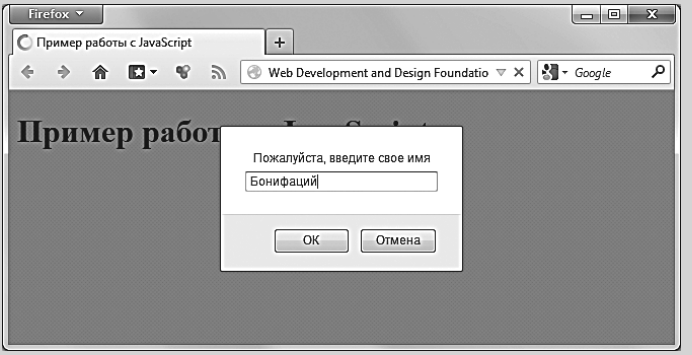


Рис. 14.12. Окно с запросом данных отображается в браузере после загрузки страницы

Отредактируйте документ *variablewrite3.html* следующим образом и сохраните его с именем *changebackground.html*. Код скрипта следующий:

```
<script type="text/javascript">
<!--
var userColor;
userColor = prompt("Напечатайте название цвета - blue или red");
document.bgColor = userColor;
// -->
</script>
```

Мы запрашиваем у пользователя название цвета, например blue (синий) или red (красный). Вы знаете из своего опыта работы с языком HTML, что существует множество названий цветов (см. файл *Примеры\Таблицы_цветов.html* на диске, прилагающемся к книге). Экспериментируйте!

Сохраните документ и отобразите его в браузере. Появится окно с запросом, в котором следует ввести название цвета и нажать кнопку **OK**. Цвет фона изменится незамедлительно. Образцы файлов *changebackground.html* и *variablewrite3.html* вы найдете на диске, прилагающемся к книге, в папке *Примеры\Глава_14*.

14.8. Введение в концепции программирования

До сих пор мы использовали объектную модель документа (DOM) для доступа к свойствам и методам окна и документа. Мы также создали

несколько простых обработчиков событий. Есть еще один аспект языка программирования JavaScript, который больше похож на программирование. В этой главе мы затронем только небольшую его часть, чтобы почувствовать силу использования концепций программирования, и поработаем с ним позднее, чтобы протестировать ввод данных в форму.

Арифметические операции

Работая с переменными, часто полезно уметь выполнять арифметические подсчеты. Например, вы можете создать веб-страницу, которая считает налог на товар. Как только пользователь выбрал продукт, вы можете использовать язык JavaScript, чтобы подсчитать налог и записать результат в документ. Таблица 14.3 показывает список ***арифметических операторов***, дает их описание и несколько примеров.

Таблица 14.3. Распространенные арифметические операторы

Оператор	Описание	Пример	Значение количества
=	присвоить	`quantity = 10`	10
+	прибавить	`quantity = 10 + 6`	16
-	вычесть	`quantity = 10 - 6`	4
*	умножить	`quantity = 10 * 2`	20
/	разделить	`quantity = 10 / 2`	5

Языки программирования очень сильно отличаются по своим возможностям, но у них всех есть несколько общих черт. Они все позволяют использовать переменные, и у них есть команды для принятия решений, повторения команд, а также повторно используемые блоки кода. Принятие решений будет использовано, когда требуются разные результаты в зависимости от ввода или действий пользователя. В примере из нашего практического задания мы запросим у пользователя ввести число и проиллюстрируем различные сообщения на основе значения возраста, введенного пользователем. Повторение команд становится полезным при осуществлении одинаковой задачи много раз. Например, утомительно создавать раскрывающийся список, содержащий числа от 1 до 31 по количеству дней в месяце. Взамен мы можем использовать язык программирования JavaScript — и получим несколько строк кода. Повторно используемые блоки кода полезны, когда вы хотите привязать блок кода к обработчику событий, а не вводить много команд в HTML-теги обработчика событий. Поскольку целью этой главы является очень краткий

обзор некоторых концепций, то мы не можем раскрывать тему дальше. Мы затронем темы принятия решений и повторного использования кода в примерах в практическом задании.

Принятие решений

Как мы уже знаем, в языке программирования JavaScript мы можем использовать переменные. Мы можем протестировать значение переменной и осуществить различные задачи, основанные на переменных. Например, форма заказа, вероятнее всего, требует, чтобы пользователь ввел количество большее, чем 0. Мы можем протестировать поле ввода количества, чтобы убедиться, что введенное количество больше, чем 0. Если количество не больше 0, мы можем сделать так, чтобы появилось всплывающее окно с сообщением, инструктирующим пользователя ввести количество больше, чем 0. Структура оператора `if` выглядит следующим образом:

```
if (условие)
{
. . . команды для выполнения, если условие верное
} else {
. . . команды для выполнения, если условие неверное
}
```

Обратите внимание, что есть два типа используемых группирующих символов: круглые скобки и фигурные. Круглые скобки располагаются вокруг условия, а фигурные используют для инкапсулирования блока команд. Оператор `if` включает в себя два блока команд для выполнения: один, если условие является `true` (верным), и другой — если условие `false` (неверное). Квадратные скобки выравниваются так, чтобы вы могли с легкостью видеть открывающие и закрывающие фигурные скобки. Очень просто пропустить фигурные скобки, когда вы программируете, а затем искать недостающую. Выравнивание позволяет намного проще отследить скобки визуально. Когда вы верстаете код JavaScript, помните, что круглые скобки, фигурные скобки и кавычки обычно используют в парах. Если скрипт не работает, как должен, проверьте, есть ли у каждого из этих символов пара.

Если событие определяется как `true` (верное), будет выполнен первый блок команд, а блок `else` будет пропущен. Если условие `false` (неверное), то первый блок команд будет пропущен, а блок `else` выполнен.

Данный обзор должен дать вам понимание, как условия и структура оператора `if` могут быть вам полезны. Условие должно представлять собой то, что может быть определено, как `true` или `false`. Мы можем рассматривать это как математическое условие. Условие обычно задействует оператор. В таблице 14.4 перечислены наиболее распространенные ***операторы сравнения***. Примеры в табл. 14.4 можно использовать как условия в структуре `if`.

Таблица 14.4. Распространенные операторы сравнения

Оператор	Описание	Пример (количество)	Примерные значения количества, которые дадут результат `true`
`==`	Двойной знак равенства (эквивалент) «точно равный»	`quantity == 10`	10
`>`	Больше, чем	`quantity > 10`	11, 12 ... (но не 10)
`>=`	Больше или равно	`quantity >= 10`	10, 11, 12...
`<`	Меньше, чем	`quantity < 10`	... 8, 9 (но не 10)
`<=`	Меньше или равно	`quantity <= 10`	... 8, 9, 10

ЧаВо

ЧТО ДЕЛАТЬ, ЕСЛИ МОЙ КОД JAVASCRIPT НЕ РАБОТАЕТ?

Вы можете попробовать перечисленные ниже техники отладки.

- Откройте окно **Консоль ошибок** (Error Console) в программе Firefox, чтобы проверить, есть ли в коде какие-либо ошибки. Распространенные ошибки включают в себя пропущенную точку с запятой в конце строки и опечатки в командах.
- Используйте метод `alert()`, чтобы напечатать переменные и проверить содержимое. Например, если у вас переменная с именем `quantity`, выполните `alert(quantity);`, чтобы увидеть, что содержится в переменной.
- Попросите знающего человека проверить ваш код. Трудно редактировать свой собственный код, потому что у вас есть тенденция видеть то, что вы предполагали написать, а не то, что вы написали на самом деле. Легче редактировать чужой код.
- Попробуйте объяснить ваш код знающему человеку. Это часто помогает найти ошибки по мере проверки кода.
- Убедитесь, что вы не используете зарезервированные слова JavaScript в именах переменных или именах функций. Вы найдете список зарезервированных слов на диске, прилагающемся

к книге, в файле *Примеры\Глава_14\Зарезервированные слова JavaScript.pdf*.

Практическое задание 14.6

В этом практическом задании вы напишите код примера с количеством, описанным ранее. У пользователя мы запросим количество, и он должен будет ввести значение больше 0. Мы допустим, что пользователь введет число. Если пользователь введет значение 0 или отрицательное число, то отобразится сообщение об ошибке. Если пользователь вводит значение больше 0, будет отображено сообщение, благодарящее пользователя за заказ. Далее мы используем запрос и запишем сообщения в документ.

Откройте программу Блокнот (Notepad) и введите приведенный ниже код:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Пример работы с JavaScript</title>
<meta charset="utf-8">
</head>
<body>
<h1>Пример работы с JavaScript</h1>
<script type="text/javascript">
<!--
var quantity;
quantity = prompt("Напечатайте количество больше 0");
if (quantity <= 0) {
document.write("<p>Количество меньше 0.</p>");
document.write("<p>Пожалуйста, обновите веб-страницу.</
p>");
} else {
document.write("<p>Количество больше 0.</p>");
}
// -->
</script>
</body>
</HTML>
```

Сохраните документ с именем *quantityif.html* и откройте его в браузере. Если окно с запросом данных не появилось, не забудьте проверить ошибки в окне **Консоль ошибок** (Error Console). Когда появится окно с запросом, введите число 0, а затем нажмите кнопку **OK**. Вы должны увидеть сообщение об ошибке в окне браузера, как показано на рис. 14.13.

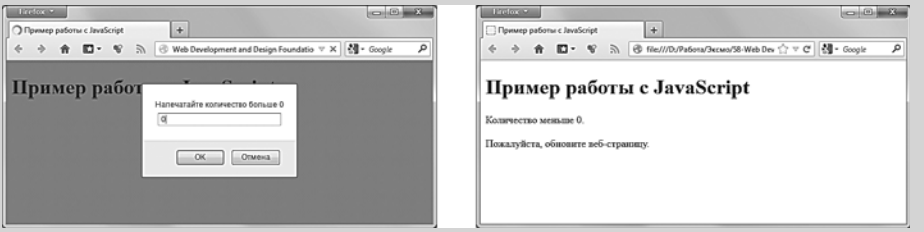


Рис. 14.13. Браузер слева показывает окно с запросом и введенной цифрой 0, а окно браузера справа показывает результат

Теперь обновите страницу и на этот раз введите значение больше 0, как показано на рис. 14.14.

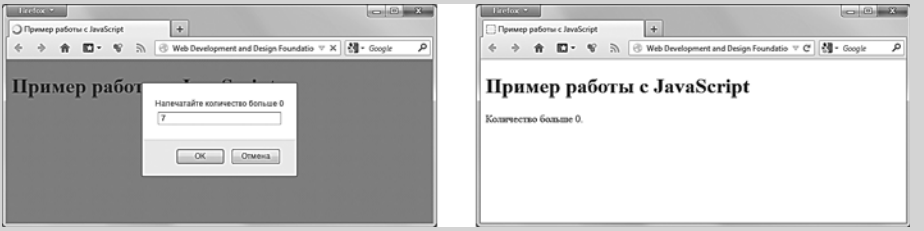


Рис. 14.14. Браузер слева показывает окно с запросом и введенным значением больше, чем 0, а браузер справа показывает результат

Функции

В практическом задании 14.6 окно с запросом появляется, как только страница загрузилась. А что если позволить пользователю решить, когда определенный скрипт должен быть интерпретирован или выполнен браузером? Мы бы могли использовать обработчик событий `onmouseover` и запустить скрипт, когда пользователь наводит указатель мыши на ссылку или изображение. Другой метод, вероятно, более интуитивный для пользователей — сделать кнопку и направить пользователей нажать ее, чтобы запустить скрипт. Посетителям веб-страницы не нужно знать, что скрипт запущен, но они могут нажать кнопку, чтобы инициировать своего рода функциональность.

Три типа кнопок были представлены в главе 9:

- кнопка отправки данных `<input type="submit">` используется для передачи информации из элементов формы;
- кнопка сброса `<input type="reset">` используется для очистки введенных значений в форму;
- третий тип кнопки, `<input type="button">`, не несет в себе каких-либо действий по умолчанию, связанных с формой.

Сейчас мы будем использовать кнопку `<input type="button">` и обработчик событий `onclick` для выполнения скрипта. Обработчик событий `onclick` может выполнять единичную или множественные команды. К примеру:

```
<input type="button" value="Нажмите, чтобы увидеть
сообщение"
onclick="alert('Добро пожаловать!');">
```

В этом примере на кнопке будет написан текст «Нажмите, чтобы увидеть сообщение». Когда пользователь нажимает кнопку, происходит событие нажатия кнопкой мыши и обработчик событий `onclick` выполняет команду `alert('Добро пожаловать!');`. Появляется окно с сообщением. Этот метод очень эффективный, когда выполняются только операторы JavaScript. И быстро выходит из-под контроля, когда указано больше операторов для выполнения. Когда это происходит, имеет смысл разместить все операторы JavaScript в блок и каким-либо образом дать задание блоку выполняться. Если у блока операторов есть название, мы можем выполнить блок, указав на его имя. В дополнение к предоставлению сокращенного имени этот код также легко может быть использован повторно. Мы можем предоставить имя блока операторов, создав функцию.

Функция — это блок операторов JavaScript с конкретной целью, которая может быть выполнена, когда это необходимо. Функция может содержать одиночный оператор или группу операторов и определяется следующим образом:

```
function function_name()
{
... операторы JavaScript
}
```

Определение функции начинается с ключевого слова, за которым следует имя функции. Круглые скобки обязательны и более сложные функции их используют. Вы можете выбрать имя функции так же, как вы выбираете имя для переменной. Имя функции должно каким-либо образом определять ее цель. Операторы содержатся внутри фигурных скобок. Блок операторов будет выполнен, когда функция будет ***вызвана***.

Вот пример определения функции:

```
function showAlerts()
{
alert("Пожалуйста, нажмите ОК, чтобы продолжить.");
alert("Пожалуйста, нажмите ОК снова.");
alert("Нажмите ОК в последний раз, чтобы
продолжить.");
}
```

Эта функция может быть вызвана при использовании следующего оператора:

```
showAlerts();
```

Теперь мы можем включить функцию `showAlerts()` в кнопку, как указано далее:

```
<input type="button" value="Нажмите, чтобы увидеть
сообщение">
onclick="showAlerts();">
```

Когда пользователь нажимает кнопку, вызывается функция `showAlerts()` и три сообщения с предупреждением появятся одно за другим. Обычно определения функций располагают внутри области `head` HTML-документа. Это загружает код определения функции, но он не выполняется, пока не вызван. Обеспечивается то, что определение функции загружено и готово к использованию перед тем, как функция вызвана.

Практическое задание 14.7

В этом практическом задании вы отредактируете документ *quantityif.html* так, чтобы переместить скрипт с запросом в функцию, чтобы его можно было вызвать обработчиком событий `onclick`.

Обратите внимание на несколько моментов. Скрипт был перемещен в раздел `head` и включает в себя определение функции. Метод `document.write()` был заменен на метод `alert()` и сообщения были немного изменены. Метод `document.write()` не будет нормально функционировать после того, как страница уже была написана, как в примере в этом упражнении. Также было добавлено несколько комментариев к закрывающим фигурным скобкам для оператора `if` и определения функции. Эти комментарии помогут вам отслеживать блоки с кодом внутри скрипта. Идентификация блоков с кодом также помогает определить, какие прямые скобки открывают и закрывают различные операторы. Запустите программу Блокнот (Notepad) и отредактируйте документ *quantityif.html* следующим образом:

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Пример работы с JavaScript</title>
<meta charset="utf-8">
<script type="text/javascript">
<!--
function promptQuantity() {
var quantity;
quantity = prompt("Напечатайте количество больше 0");
if (quantity <= 0) {
alert("Количество меньше 0.");
} else {
alert("Спасибо вам, что ввели количество больше 0.");
} // end if
} // конец функции promptQuantity
// -->
</script>
</head>
<body>
<h1>Пример работы с JavaScript</h1>
<input type="button" value="Нажмите для ввода количества" onclick="promptQuantity();">
</body>
</HTML>
```

Сохраните документ с именем *quantityif2.html* и откройте его в браузере. Откройте окно **Консоль ошибок** (Error Console) в случае, если вы допустили опечатки, и скрипт не работает. Нажмите кнопку, чтобы

протестировать скрипт. Если окно с запросом данных не появляется, проверьте окно **Консоль ошибок** (Error Console) и исправьте ошибки.

Рисунок 14.15 демонстрирует снимок окна браузера с окном запроса информации, появляющегося после нажатия кнопки, и появившееся в результате сообщение. Протестируйте значение больше 0, а также значение равное или меньше 0.

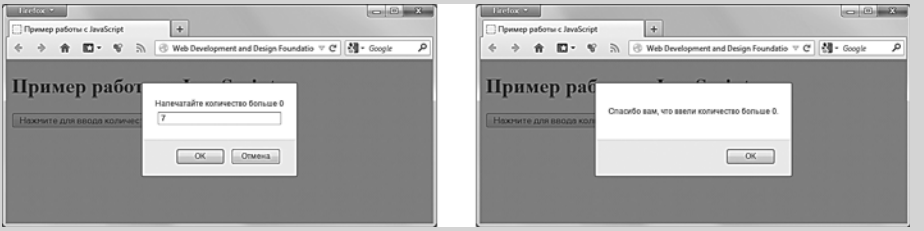


Рис. 14.15. Слева показано окно с запросом информации и полем для ее ввода; браузер справа показывает окно с предупреждением, появляющееся после ввода информации

Образец файла *quantityif2.html* вы найдете на диске, прилагающемся к книге, в папке *Примеры\Глава_14*.

14.9. Обработка форм

Как вы узнали из главы 9, данные из веб-формы могут быть предоставлены скрипту CGI или скрипту на стороне сервера. Эта информация может быть добавлена в базу данных или использована для других целей. Поэтому очень важно, чтобы данные, предоставленные пользователем, были как можно более точными. Когда пользователь вводит информацию в форму, всегда есть шанс, что информация будет неверной или неточной. Это особенно важно, когда используются текстовые поля, поскольку пользователь может с легкостью ввести ошибочные данные. Часто данные формы проверяют на наличие неверной информации перед ее отправкой. Проверка данных формы может быть осуществлена скриптом на стороне сервера, но это также может быть сделано на стороне клиента, используя язык программирования JavaScript. Опять-таки, эта тема представлена здесь в очень упрощенном виде, но вы сможете понять, как это реализуется.

Когда пользователь нажимает кнопку отправки данных формы, происходит соответствующее событие (submit). Мы можем использовать обработчик событий **`onsubmit`**, чтобы вызвать функцию, которая тестирует данные формы на валидность. Эта техника относится к ***обработке***

формы. Веб-разработчик может проверять все указанные данные формы, некоторые данные или только один вид данных формы. Ниже приведен список некоторых типов данных, которые могут быть проверены:

- обязательное поле ввода, такое как имя и адрес электронной почты;
- обязательный флажок согласия с лицензионным соглашением;
- переключатель, позволяющий выбрать метод оплаты или условия доставки;
- количество, которое представляет собой число в определенном диапазоне.

Когда пользователь нажимает кнопку отправки данных, обработчик событий `onsubmit` вызывает функцию, которая проверяет все соответствующие элементы формы на валидность данных. Затем функция проверки подтверждает, что данные верны (`true`) или неверны (`false`). Форма отправляется на URL, указанный в действии `form`, если данные определены как верные (`true`). Форма не будет принята, если данные неверные (`false`), и пользователю будут отображены сообщения о допущенных ошибках. Общая структура кода веб-страницы, относящаяся к декларированию функции и обработки события `onsubmit`, является следующей:

```
...HTML-код начинает веб-страницу
function validateForm()
{
...команды JavaScript для тестирования данных формы
if form data is valid
return true
else
return false
}
...HTML-код продолжается
<form method="post" action="URL" onsubmit="return
validateForm();">
...элементы формы
<input type="submit" value="отправить форму">
</form>
...HTML-код продолжается
```

Существует новая концепция по отношению к приведенным здесь функциям. Функция может инкапсулировать группу операторов, но она также может отправлять значение обратно туда, откуда оно было вызвано. Это так называемое ***возвращаемое значение***, и для него используется ключевое слово `return`, оно используется в коде JavaScript для определения значения, которое будет отправлено обратно. В нашем примере обратно вернется значение `true`, если данные верные, и значение `false`, если данные не проходят тест на валидность. Обратите внимание, что обработчик событий `onsubmit` также содержит ключевое слово `return`. Это работает следующим образом: функция `validateForm()` возвращает значение `true` и значение `false`, обработчик событий `onsubmit` становится `return true`, и данные формы отправляются. Если функция `validateForm()` возвращает значение `false`, обработчик событий `onsubmit` становится `return false`, и данные формы не отправляются. Когда функция возвращает значение, ее выполнение заканчивается независимо от того, есть ли в ней еще операторы.

Практическое задание 14.8

В этом практическом задании вы создадите форму с текстовыми полями для ввода имени и возраста, и будете использовать язык программирования JavaScript для проверки данных, чтобы информация была введена в поле для имени, а возраст был указан 18 лет или больше. Если поле для ввода имени пустое, то будет отображено окно с сообщением, предлагающее пользователю ввести имя. Если введенный возраст меньше 18 лет, то появится сообщение, инструктирующее пользователя ввести возраст 18 лет или больше. Если вся информация верна, будет отображено сообщение, обозначающее, что данные верны и форма будет отправлена.

Давайте начнем с создания формы. Откройте программу Блокнот (Notepad) и введите код, указанный ниже. Обратите внимание, что обработчик событий `onsubmit` встраивается в тег `<form>`, а код JavaScript мы добавим позднее. Каскадная таблица стилей используется для выравнивания и добавления места вокруг элементов формы.

```
<!DOCTYPE HTML>
<HTML lang="en">
<head>
<title>Пример работы с JavaScript</title>
<meta charset="utf-8">
<style>
input { display: block;
```

```
margin-bottom: 10px; }
label { float: left;
width: 120px;
padding-right: 10px;
text-align: right; }
#submit {margin-left: 130px; }
</style>
</head>
<body>
<h1>Обработка данных формы с помощью JavaScript</h1>
<form method="post"
action="http://webdevfoundations.net/scripts/formdemo.
asp"
onsubmit="return validateForm();">
<label for="userName">Имя:</label>
<input type="text" name="userName" id="userName">
<label for="userAge">Возраст:</label>
<input type="text" name="userAge" id="userAge">
<input type="submit" value="Отправить данные"
id="submit">
</form>
</body>
</HTML>
```

Сохраните файл с именем *formvalidation.html* и откройте его в браузере. Рисунок 14.16 демонстрирует форму в браузере.

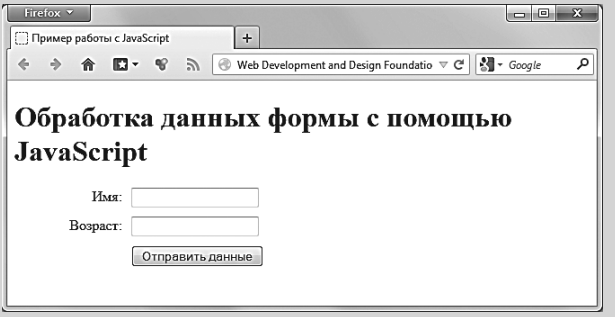


Рис. 14.16. Файл *formvalidation.html*, открытый в браузере

Нажмите кнопку **Отправить данные**. Вы заметите, что данные формы будут отправлены. Мы еще не написали код для функции `vali-`

dateForm(), поэтому данные формы были отправлены без валидации.

Доступ к введенным в форму данным немного сложный. Форма — это свойство объекта документа. Каждый элемент формы — это свойство объекта form. Свойство элемента формы может быть значением. Таким образом, доступ к контенту из поля ввода может выглядеть следующим образом:

```
document.forms[0].inputbox_name.value
```

Форма определяется значением forms[0]. HTML-документ может содержать несколько форм. Обратите внимание, что слово forms[0] указано во множественном числе. Первая форма — это forms[0]. Для доступа к значению, введенному пользователем в поле ввода userAge, потребуется использовать document.forms[0].userAge.value. Это достаточно трудоемкий процесс.

Также обратите внимание, что значения true и false не заключаются в кавычки. Это важно, поскольку true и false являются не строками, а зарезервированными, или ключевыми, словами JavaScript и представляют особые значения. Если вы добавите к ним кавычки, они станут строками, и эта функция не будет работать надлежащим образом.

Начнем с добавления кода для проверки возраста. Отредактируйте файл *formvalidation.html*, добавив следующий блок скрипта между тегами <head> и </head>:

```
<script type="text/javascript">
<!--
function validateForm() {
if (document.forms[0].userAge.value < 18) {
alert("Указанный возраст меньше 18 лет. Вы еще слишком малы.");
return false;
} // end if
alert("Указан допустимый возраст.");
return true;
} // конец функции validateForm
// -->
</script>
```

Функция validateForm() проверяет возраст в поле ввода userAge. Если он меньше 18, то будет отображено сообщение с предупреждением, возвращено значение false и выполнение функции прекратится. Обработчик событий onsubmit изменит значение на "return

false" и данные формы не будут отправлены. Если возраст 18 лет или больше, операторы в структуре if будут пропущены и появится сообщение alert("Указан допустимый возраст."); . После того как пользователь нажмет кнопку **OK** в сообщении с предупреждением, оператор return true; будет выполнен, обработчик событий onsubmit примет значение "return true", а данные формы будут отправлены. Давайте это протестируем!

Введите значение меньше 18 в поле ввода **Возраст:** и нажмите кнопку **Отправить данные**. Если данные формы будут отправлены, то, вероятно, закралась ошибка в код JavaScript. Если так и произошло, откройте окно **Консоль ошибок** (Error Console) и исправьте найденные ошибки.

Рисунок 14.17 демонстрирует ввод данных в поле **Возраст:** и сообщение, появляющееся после того, как вы нажали кнопку **OK**.

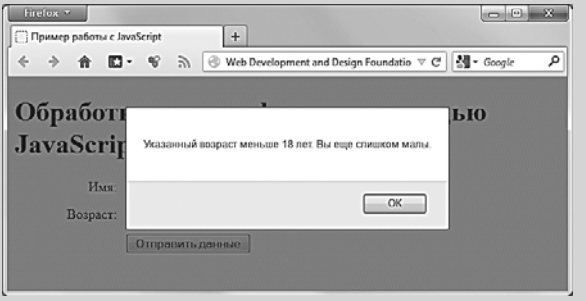


Рис. 14.17. Файл *validateform.html* отображается в браузере. Окно с предупреждением появляется при вводе значения возраста меньше 18

Нажмите кнопку **OK** и введите значение возраста, равное или больше 18 в поле ввода **Возраст:**. Нажмите кнопку **Отправить данные**.

Рисунок 14.18 демонстрирует ввод информации в поле ввода **Возраст:**, сообщение с предупреждением, появившееся после того, как данные о возрасте были отправлены, и веб-страницу, появившуюся после того, как форма была принята.

Теперь добавим другой оператор if для проверки имени. Чтобы убедиться, что какие-либо данные были введены в поле ввода username, мы протестируем форму, чтобы проверить, пустое ли поле ввода. Строка null представлена двойными кавычками "" или двумя одинарными апострофами '' без пробела или любого другого символа между ними. Вы можете сравнить значение текстового поля userName со строкой null. Если значение поля userName равно строке null, тогда мы знаем, что пользователь не ввел никаких данных в поле. В нашем примере мы будем отправлять только одно сообщение с ошибкой за один раз.

Рис. 14.18. Файл *validateform.html* отображается в браузере с введенным значением возраста больше или равным 18. Обратите внимание на сообщение с предупреждением. Браузер справа отображает страницу, появляющуюся после того, как данные формы были отправлены

Если пользователь не указал имя в поле `userName` и также у него неподходящий возраст в поле `userAge`, пользователь увидит только ошибку о неверном значении `userName`. После исправления имени и повторной отправки информации пользователь увидит сообщение о неверно введенном значении `userAge`. Это поверхностная информация об обработке форм, но она позволит вам разобраться, как можно работать с формами. Более сложная обработка форм заключается в проверке каждого поля формы и указания на все ошибки каждый раз при ее отправке.

Добавим код для проверки данных `userName`. Отредактируйте файл *formvalidation.html* и блок скрипта следующим образом. Обратите внимание, что два знака равно представляют равенство в операторе `if`. Некоторые пользователи читают символы `==` как «точно равен чему-то».

```
<script type="text/javascript">
<!--
function validateForm() {
if (document.forms[0].userName.value == "" ) {
alert("Поле Имя не может быть пустым.");
return false;
} // end if
if (document.forms[0].userAge.value < 18) {
alert("Указанный возраст меньше 18 лет. Вы еще слишком
малы.");
return false;
} // end if
alert("Имя и возраст допустимы.");
return true;
} // конец функции validateForm
```

```
// -->
</script>
```

Сохраните документ и откройте его в окне браузера. Нажмите кнопку **Отправить данные**, не указав предварительно данные в поле **Имя:** или **Возраст:**.

Рисунок 14.19 демонстрирует сообщение, которое отображается, если не введена какая-либо информация, а кнопка **Отправить данные** была нажата.

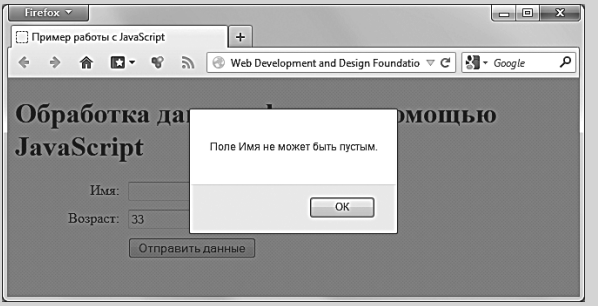


Рис. 14.19. Файл *validateform.html* открыт в браузере, в поля **Имя:** и **Возраст:** не введена информация, вследствие этого при отправке данных формы отображается сообщение с предупреждением

Нажмите кнопку **OK** и введите какой-нибудь текст в поле ввода **Имя:** и снова отправьте форму.

Рисунок 14.20 показывает данные в поле вводе **Имя:** и сообщение с предупреждением, которое появляется после проверки возраста. Поле ввода возраста не содержит возраст, и значение в нем интерпретируется как `0`.

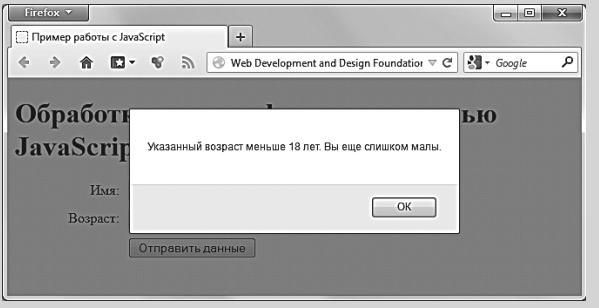


Рис. 14.20. Файл *validateform.html*, открытый в браузере, с введенной информацией в поле **Имя:** и с пустым полем **Возраст:**; сообщение с предупреждением появляется после нажатия кнопки **Отправить данные**

Нажмите кнопку **OK** и введите возраст — 18 лет или больше. Нажмите кнопку **Отправить данные**.

Рисунок 14.21 демонстрирует данные в полях **Имя:** и **Возраст:** и сообщение, которое отображается после того, как вы нажали кнопку **Отправить данные**. Он также показывает появляющуюся в результате веб-страницу после успешно отправленных данных формы, когда все данные валидны.

Рис. 14.21. Файл *validateform.html* отображается в браузере с верными введенными данными в полях **Имя:** и **Возраст:**, а также показано сообщение об этом. Браузер справа показывает веб-страницу, отображающуюся после отправки валидных данных

Образец файла *validateform.html* вы найдете на диске, прилагающемся к книге, в папке *Примеры\Глава_14*.

14.10. Доступность и JavaScript

Интерактивность и функциональность, которые язык программирования JavaScript позволяет добавить на веб-страницу, — это прекрасные качества. Тем не менее вы должны знать, что некоторые посетители могут отключить JavaScript и не увидеть ваших визуальных эффектов или не смогут управлять мышью. ГОСТ-52872-2007 требует, чтобы ваш сайт был функциональным на базовом уровне, даже если браузер ваших посетителей не поддерживает интерактивные включения.

Если вы используете JavaScript для обработки событий, связанных с движением мыши во время навигации по вашему сайту, вы должны также предоставить обычную текстовую навигацию, которая не требует участия мыши и к которой легко могут получить доступ программы экранного доступа.

Если вы используете JavaScript для валидации данных формы, укажите также и адрес электронной почты, чтобы предоставить людям с ограниченными возможностями способ связаться с вашей организацией и получить помощь.

14.11. Ресурсы по JavaScript

В этой главе предоставлена краткая информацию об использовании языка программирования JavaScript в веб-проектах. У вас может возникнуть желание узнать о нем более подробно. Для получения дополнительной информации посетите следующие ресурсы:

- JSToolbox — все о JavaScript **(www.jstoolbox.com)**
- JavaPortal.ru — все о Java и JavaScript **(www.javaportal.ru)**
- Справочник на Mozilla.org **(developer.mozilla.org/ru/JavaScript)**
- Википедия **(ru.wikipedia.org/wiki/JavaScript)**
- Центральный JavaScript-ресурс **(javascript.ru)**
- JavaScript and HTML DOM Reference **(www.w3schools.com/jsref/default.asp)**

СПРАВОЧНИК ВЕБ-РАЗРАБОТЧИКА

В следующих приложениях вы найдете множество ресурсов, которые могут помочь вам стать профессиональным веб-разработчиком.

Приложение А. «Справочник HTML5» содержит список распространенных элементов и атрибутов HTML5.

Приложение Б. «Специальные символы» содержит список кодов, необходимых для отображения знаков и других специальных символов в веб-страницах.

Приложение В. «Справочник свойств CSS» содержит список обычно используемых свойств и значений.

Приложение Г. «Использование протокола FTP для публикации сайтов» содержит краткое введение по использованию протокола передачи файлов в Интернете (FTP).

Приложение А
СПРАВОЧНИК HTML5

Распространенные элементы HTML5.

Элемент	Назначение	Часто используемые атрибуты
`<!-- -->`	Комментарий	*Не применимы*
`<a>`	Тег привязки: конфигурирует гиперссылки	`accesskey`, `class`, `href`, `id`, `name`, `rel`, `style`, `tabindex`, `target`, `title`
`<abbr>`	Конфигурирует аббревиатуру	`class`, `id`, `style`
`<address>`	Конфигурирует контактную информацию	`class`, `id`, `style`
`<area>`	Задает область карты-изображения	`accesskey`, `alt`, `class`, `href`, `hreflang`, `id`, `media`, `shape`, `style`, `tabindex`, `target`
`<article>`	Конфигурирует независимый раздел документа, к примеру статью	`class`, `id`, `style`
`<aside>`	Конфигурирует второстепенный контент	`class`, `id`, `style`
`<audio>`	Конфигурирует панель управления аудиофайлом браузера	`autoplay`, `class`, `controls`, `id`, `loop`, `preload`, `src`, `style`, `title`
`<b>`	Задает выделение полужирным текста	`class`, `id`, `style`
`<blockquote>`	Конфигурирует длинную цитату	`class`, `id`, `style`
`<body>`	Конфигурирует раздел тела документа	`class`, `id`, `style`
` `	Задает разрыв строки	`class`, `id`, `style`
`<button>`	Конфигурирует кнопку	`accesskey`, `autofocus`, `class`, `disabled`, `form`, `formaction`, `formenctype`, `formmethod`, `formtarget`, `formnovalidate`, `id`, `name`, `type`, `style`, `value`
`<canvas>`	Конфигурирует динамические изображения	`class`, `height`, `id`, `style`, `title`, `width`
`<caption>`	Конфигурирует подпись для выравнивания таблицы	`align` (устарел), `class`, `id`, `style`

Элемент	Назначение	Часто используемые атрибуты
<cite>	Конфигурирует заголовок цитаты	class, height, id, style, title, width
<code>	Конфигурирует фрагмент компьютерного кода	class, id, style
<col>	Задает колонку таблицы	class, id, span, style
<colgroup>	Конфигурирует группу из нескольких столбцов в таблице	class, id, span, style
<command>	Задает область для отображения команд	class, id, style, type
<data>	Предоставляет текстовые комментарии в формате, который легко прочитать машинам	class, id, style, value
<datalist>	Конфигурирует элемент управления, содержащий один или несколько вариантов выбора	class, id, style
<dd>	Задает область описания в списке описаний	class, id, style
<del>	Конфигурирует удаленный текст (посредством зачеркивания)	cite, class, datetime, id, style
<details>	Конфигурирует средство управления для предоставления дополнительной информации по требованию пользователя	class, id, open, style
<dfn>	Задает определение термина	class, id, style
	Конфигурирует раздел или секцию в документе	align (устарел), class, id, style
<dl>	Конфигурирует список описаний (раньше назывался список определений)	class, id, style
<dt>	Конфигурирует понятие в списке описаний	class, id, style
	Задает выделение текста (обычно курсивом)	class, id, style
<fieldset>	Задает группировку элементов формы с границей	class, id, style
<figcaption>	Конфигурирует подпись к изображению	class, id, style
<figure>	Задает изображение	class, id, style
<footer>	Конфигурирует область нижнего колонтитула	class, id, style

Элемент	Назначение	Часто используемые атрибуты
`<form>`	Конфигурирует фигуру	`accept-charset`, `action`, `autocomplete`, `class`, `enctype`, `id`, `method`, `name`, `novalidate`, `style`, `target`
`<h1> … <h6>`	Задает заголовки	`align` (устарел), `class`, `id`, `style`
`<head>`	Конфигурирует раздел заголовка	*Не применимы*
`<header>`	Конфигурирует область заголовка	`class`, `id`, `style`
`<hgroup>`	Конфигурирует группу заголовков	`class`, `id`, `style`
`<hr>`	Конфигурирует горизонтальную линию; обозначает новую тему в HTML5	`class`, `id`, `style`
`<HTML>`	Конфигурирует элемент `root` документа веб-страницы	`lang`, `manifest`
`<i>`	Задает выделение текста курсивом	`class`, `id`, `style`
`<iframe>`	Конфигурирует встроенный фрейм	`class`, `height`, `id`, `name`, `sandbox`, `seamless`, `src`, `style`, `width`
`<img>`	Конфигурирует изображение	`alt`, `class`, `height`, `id`, `ismap`, `name`, `src`, `style`, `usemap`, `width`
`<input>`	Конфигурирует средства ввода: текстовое поле, поле для ввода адреса электронной почты, URL-адреса, номера телефона, поле поиска, текстовое поле с прокруткой, кнопку отправки данных, кнопку сброса, поле ввода пароля, календарь, счетчик, палитру цветов или скрытое поле	`accesskey`, `autocomplete`, `class`, `checked`, `disabled`, `form`, `id`, `list`, `max`, `maxlength`, `min`, `name`, `pattern`, `placeholder`, `readonly`, `required`, `size`, `step`, `style`, `tabindex`, `type`, `value`
`<ins>`	Конфигурирует текст, вставленный в документ	`cite`, `class`, `datetime`, `id`, `style`
`<kbd>`	Задает обозначения текста, вводимого с клавиатуры	`class`, `id`, `style`
`<keygen>`	Конфигурирует генератор пары ключей «открытый-секретный ключ» или отправляет открытый ключ	`autofocus`, `challenge`, `class`, `disabled`, `form`, `id`, `keytype`, `style`
`<label>`	Определяет подпись для элемента управления формы	`class`, `for`, `form`, `id`, `style`
`<legend>`	Конфигурирует подпись для элемента `fieldset`	`class`, `id`, `style`

Элемент	Назначение	Часто используемые атрибуты
	Задает пункт неупорядоченного или упорядоченного списка	class, id, style, value
<link>	Связывает веб-страницу с внешним ресурсом	class, href, hreflang, id, rel, media, sizes, style, type
<map>	Конфигурирует карту-изображение	class, id, name, style
<mark>	Задает помеченный (выделенный) текст, который можно легко найти	class, id, style
<menu>	Конфигурирует список команд	class, id, label, style, type
<meta>	Задает метаданные	charset, content, http-equiv, name
<meter>	Конфигурирует видимую шкалу измерения для определенного значения	class, id, high, low, max, min, optimum, style, value
<nav>	Конфигурирует область с гиперссылками навигации	class, id, style
<noscript>	Конфигурирует контент для браузеров, не поддерживающих скрипты на стороне клиента	
<object>	Задает встроенный объект	classid, codebase, data, form, height, name, id, style, title, tabindex, type, width
	Конфигурирует упорядоченный список	class, id, reversed, start, style, type
<optgroup>	Конфигурирует группу связанных вариантов в списке выбора	class, disabled, id, label, style
<option>	Конфигурирует вариант в списке выбора	class, disabled, id, selected, style, value
<output>	Определяет результат вычисления	class, for, form, id, name, style
<p>	Задает абзац	class, id, style
<param>	Задает параметр для плагина	name, value
<pre>	Конфигурирует переформатированный текст	class, id, style
<progress>	Конфигурирует видимый индикатор прогресса	class, id, max, style, value
<q>	Задает текст цитаты	class, id, style
<rp>	Конфигурирует комментарий в теге <ruby>	class, id, style

Элемент	Назначение	Часто используемые атрибуты
<rt>	Задает текстовый компонент комментария в теге <ruby>	class, id, style
<ruby>	Конфигурирует аннотацию	class, id, style
<s>	Определяет текст, не являющийся точным или значимым	class, id, style
<samp>	Задает образец ввода компьютерной программы или системы	class, id, style
<script>	Задает скрипт на стороне клиента (обычно JavaScript)	async, charset, defer, src, type
<section>	Конфигурирует раздел документа	class, id, style
<select>	Конфигурирует список выбора	class, disabled, form, id, multiple, name, size, style, tabindex
<small>	Отображает текст мелким шрифтом	class, id, style
<source>	Определяет файл мультимедиа и тип MIME	class, id, media, src, style, type
	Конфигурирует общий раздел документа со строковым отображением	class, id, style
	Задает выделение важного текста (обычно он отображается полужирным начертанием и зачастую — крупнее)	class, id, style
<style>	Задает глобальные стили документа	media, scoped, type
<sub>	Конфигурирует нижний индекс	class, id, style
<summary>	Конфигурирует текст в виде резюме или подписи, содержащий детальное описание	class, id, style
<sup>	Задает верхний индекс	class, id, style
<table>	Конфигурирует таблицу	class, id, style, summary
<tbody>	Конфигурирует раздел тела таблицы	class, id, style
<td>	Конфигурирует ячейку данных в таблице	class, colspan, id, headers, rowspan
<textarea>	Создает текстовую область с прокруткой	accesskey, autofocus, class, cols, disabled, id, maxlength, name, placeholder, readonly, required, rows, style, tabindex, wrap

Элемент	Назначение	Часто используемые атрибуты
<tfoot>	Конфигурирует раздел нижнего колонтитула таблицы	class, id, style
<th>	Задает ячейку заголовка таблицы	class, colspan, id, headers, rowspan, scope, style
<thead>	Конфигурирует раздел заголовка таблицы	class, id, style
<time>	Определяет дату и/или время	class, datetime, id, style
<title>	Конфигурирует заголовок документа веб-страницы	
<tr>	Определяет строку в таблице	class, id, style
<track>	Задает подзаголовок или подпись к мультимедийным файлам	class, default, id, kind, label, src, srclang, style
<u>	Задает подчеркнутый текст	class, id, style
	Конфигурирует неупорядоченный список	class, id, style
<var>	Конфигурирует текст как переменную или метку-заполнитель	class, id, style
<video>	Конфигурирует панель управления видеофайлом браузера	autoplay, class, controls, height, id, loop, poster, preload, src, style, width
<wbr>	Задает область разрыва строки	class, id, style

Приложение Б
СПЕЦИАЛЬНЫЕ СИМВОЛЫ

В таблице перечислены некоторые специальные символы, отсортированные в порядке числовых кодов. Часто употребляемые специальные символы выделены полужирным шрифтом. Список специальных символов Консорциума W3C находится на странице **www.w3.org/MarkUp/HTML-spec/HTML-spec_13.html**.

Имя сущности	Числовой код	Код описания	Символ
Двойная кавычка	**"**	**"**	**"**
Амперсанд	**&**	**&**	**&**
Апостроф	**'**		**'**
Знак «меньше»	**<**	**<**	**<**
Знак «больше»	**>**	**>**	**>**
Вертикальная черта	&#124;		|
Одиночная левая кавычка	**‘**	**‘**	**‘**
Одиночная правая кавычка	**’**	**’**	**’**
Неразрывный пробел	** **	** **	**пробел**
Перевернутый восклицательный знак	&#161;	&iexcl;	¡
Знак цента	&#162;	&cent;	¢
Знак фунта стерлингов	&#163;	&pound;	£
Общий знак валюты	&#164;	&curren;	¤
Знак иены	&#165;	&yen;	¥
Разорванная вертикальная черта	&#166;	&brvbar;	¦
Знак параграфа	&#167;	&sect;	§
Умлаут	&#168;	&uml;	¨
Знак авторского права	**©**	**©**	**©**
Порядковый индикатор (женский род)	&#170;	&ordf;	ª
Левая двойная угловая кавычка	&#171;	&laquo	«
Знак отрицания	&#172;	&not;	¬
Мягкий перенос	&#173;	&shy;	
Знак зарегистрированной торговой марки	**®**	**®**	**®**

Имя сущности	**Числовой код**	**Код описания**	**Символ**
Макрон	&#175;	&macr;	¯
Знак градуса	&#176;	&deg;	°
Плюс-минус	&#177;	&plusm;	±
Верхний индекс, 2	&#178;	&sup2;	²
Верхний индекс, 3	&#179;	&sup3;	³
Акут (оксия, острое ударение)	&#180;	&acute;	´
Микро	&#181;	&micro;	µ
Знак абзаца	&#182;	&para;	¶
Интерпункт	&#183;	&middot;	·
Седиль	&#184;	&cedil;	¸
Верхний индекс, 1	&#185;	&supl;	¹
Порядковый индикатор (мужской род)	&#186;	&ordm;	º
Правая угловая двойная кавычка	&#187;	&raquo	»
Дробь одна четверть	&#188;	&frac14;	¼
Дробь одна вторая	&#189;	&frac12;	½
Дробь три четверти	&#190;	&frac34;	¾
Перевернутый знак вопроса	&#191;	&iquest;	¿
Латинская «е» с грависом	&#232;	&egrave;	è
Латинская «е» с акутом	&#233;	&eacute;	é
Короткое тире	&#8211;	&ndash;	–
Длинное тире	**—**	**—**	**—**

Приложение В
СПРАВОЧНИК СВОЙСТВ CSS

Свойства каскадных таблиц стилей.

Свойство	Описание
background	Общее свойство для обозначения всех свойств конфигурирования фона. Значения: background-color, background-image, background-repeat, background-position
background-attachment	Задает фоновое изображение как фиксированное на месте или прокручиваемое. Значения: scroll (по умолчанию) или fixed
background-clip	Свойство CSS3; конфигурирует область для отображения фона. Значения: border-box, padding-box или content-box
background-color	Определяет фоновый цвет элемента. Значения: допустимое значение цвета
background-image	Конфигурирует файл изображения как фон элемента. Значения: url (имя файла или путь к изображению), none (по умолчанию). Необязательные функции в CSS3: linear-gradient() и radial-gradient()
background-origin	Свойство CSS3; конфигурирует область позиционирования фона. Значения: padding-box, border-box или content-box
background-position	Определяет позицию фонового изображения элемента. Значения: два процентных отношения, числовые значения в пикселах или значения позиционирования (left, top, center, bottom, right)
background-repeat	Определяет, как будет повторяться фоновое изображение элемента. Значения: repeat (по умолчанию), repeat-y, repeat-x no-repeat
background-size	Свойство CSS3; определяет размер фоновых изображений. Значения: числовое значение (в пикселах или единицах em), величина в процентах, contain, cover
border	Определяет границу, окружающую элемент. Значения: border-width, border-style и border-color
border-bottom	Определяет нижнюю границу элемента. Значения: border-width, border-style и border-color
border-collapse	Объединяет границы таблицы и ячеек в ней или отображает их с раздельными границами. Значения: separate (по умолчанию), collapse

Свойство	Описание
`border-color`	Определяет цвет границ элемента. Значения: допустимое значение цвета
`border-image`	Свойство CSS3; определяет изображение для построения границы элемента. См.: **www.w3.org/TR/css3-background#the-border-image**
`border-left`	Определяет левую границу элемента. Значения: `border-width`, `border-style` и `border-color`
`border-radius`	Свойство CSS3; конфигурирует скругленные углы Значения: от одного до четырех числовых значений в пикселах, единицах em или процентах, задающих радиусы углов. Если указывается одно значение, оно конфигурирует все четыре угла. Углы скругляются в таком порядке: левый верхний, правый верхний, правый нижний и левый нижний. Связанные свойства: `border-top-left-radius`, `border-top-right-radius`, `border-bottom-left-radius` и `border-bottom-right-radius`
`border-right`	Определяет правую границу элемента. Значения: `border-width`, `border-style` и `border-color`
`border-spacing`	Задает расстояние между ячейками в таблице. Значения: числовое значение (в пикселах или единицах em)
`border-style`	Определяет стиль границы вокруг элемента. Значения: `none` (по умолчанию), `double`, `groove`, `inset`, `outset`, `ridge`, `solid`, `dashed`, `dotted`
`border-top`	Определяет верхнюю границу элемента. Значения: `border-width`, `border-style` и `border-color`
`border-width`	Определяет ширину границы вокруг элемента. Значения: числовое значение (например, `1px`) или значения `thin`, `medium`, `thick`
`bottom`	Определяет позицию смещения относительно нижнего края элемента-контейнера. Значения: числовое значение (в пикселах или единицах em), процентное отношение или `auto` (по умолчанию)
`box-shadow`	Свойство CSS3; конфигурирует тень элемента. Значения: указывается от трех до четырех числовых значений в пикселах или единицах em для обозначения смещения по горизонтали и по вертикали, радиуса размытия и размера (необязательно), а также допустимое значение цвета. Используйте ключевое слово `inset` для создания внутренней тени
`caption-side`	Задает расположение подписи к таблице Значения: `top` (по умолчанию) или `bottom`
`clear`:	Назначает отображение элемента относительно обтекаемых элементов. Значения: `left`, `right`, `both`, `none` (по умолчанию)
`color`	Определяет фоновый (текст) цвет элемента. Значения: допустимое значение цвета
`display`	Управляет, будет ли отображаться элемент, и если да, то каким образом. Значения: `inline`, `none`, `block`, `list-item`, `table`, `table-row` или `table-cell`

Свойство	Описание
float	Определяет горизонтальное положение (левое или правое) элемента в родительском элементе. Другими словами, задает обтекание элемента. Значения: right", left, none (по умолчанию)
font-family	Определяет шрифт, используемый для отображения элемента. Значения: допустимое имя шрифта или семейство шрифтов
font-size	Определяет размер шрифта, используемого для отображения элемента. Значения: числовое значение (в точках, пикселах или единицах em), процентное отношение, xx-small, x-small, small, medium (по умолчанию), large, x-large, xx-large, smaller или larger
font-stretch	Свойство CSS3; задает нормальный, разреженный или уплотненный интервал шрифта. Значения: normal (по умолчанию), wider, narrower, condensed, semi-condensed, expanded или ultra-expanded
font-style	Определяет стиль текста. Значения: normal (по умолчанию), italic или oblique
font-variant	Определяет формат текста — обычный текст или малые прописные буквы. Значения: normal (по умолчанию), small-caps
font-weight	Определяет жирность шрифта. Значения: числовое значение, normal (по умолчанию), bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800 или 900
height	Определяет высоту элемента. Значения: числовое значение (в пикселах или единицах em), процентное отношение или значение auto (по умолчанию)
left	Определяет позицию смещения относительно левого края элемента-контейнера. Значения: числовое значение (в пикселах или единицах em), процентное отношение или значение auto (по умолчанию)
letter-spacing	Определяет расстояние между символами в тексте. Значения: числовое значение (в пикселах или единицах em) или normal (по умолчанию)
line-height	Определяет интервалы, разрешенные в строке текста. Значения: числовое значение (в пикселах или единицах em), процентное отношение, множественное числовое значение или normal (по умолчанию)
list-style	Общее для всех свойств списка. Значения: list-style-type, list-style-position, list-style-image
line-style-image	Определяет изображение для замещения «маркеров» в неупорядоченном списке. Значения: url (имя файла или путь к изображению) или none (по умолчанию)
list-style-position	Задает положение маркеров списка. Значения: inside или outside (по умолчанию)
line-style-type	Определяет стиль маркера (пункта списка). Значения: none, circle, disc (по умолчанию), square, decimal, decimal-leading-zero, georgian, lower-alpha, lower-roman, upper-alpha или upper-roman

Свойство	Описание
margin	Определяет размеры поля, окружающего элемент. Значения: от одного до четырех числовых значения (в пикселах или единицах em), процентное отношение, 0 или auto
margin-bottom	Определяет размер нижнего поля элемента. Значения: числовое значение (в пикселах или единицах em), процентное отношение, 0 или auto
margin-left	Определяет размер левого поля элемента. Значения: числовое значение (в пикселах или единицах em), процентное отношение, 0 или auto
margin-right	Определяет размер правого поля элемента. Значения: числовое значение (в пикселах или единицах em), процентное отношение, 0 или auto
margin-top	Определяет размер верхнего поля элемента. Значения: числовое значение (в пикселах или единицах em), процентное отношение, 0 или auto
max-height	Определяет максимальную высоту элемента. Значения: числовое значение (в пикселах или единицах em), процентное отношение или none (по умолчанию)
max-width	Определяет максимальную ширину элемента. Значения: числовое значение (в пикселах или единицах em), процентное отношение или none (по умолчанию)
min-height	Определяет минимальную высоту элемента. Значения: числовое значение (в пикселах или единицах em), процентное отношение или none (по умолчанию)
min-width	Определяет минимальную ширину элемента. Значения: числовое значение (в пикселах или единицах em), процентное отношение или none (по умолчанию)
opacity	Свойство CSS3; задает степень непрозрачности элемента и его дочерних элементов. Значения: числовые значения от 1 (совершенно непрозрачный) до 0 (полностью прозрачный).
overflow	Определяет отображение контента, если он превышает свою высоту или ширину. Значения: visible (по умолчанию), hidden, auto или scroll
padding	Определяет величину отступа, связанную с элементом. Значения: от одного до четырех числовых значения (в пикселах или единицах em), процентное отношение или 0
padding-bottom	Определяет пустое пространство между элементом и его нижней границей. Значения: числовое значение (в пикселах или единицах em), процентное отношение или 0
padding-left	Определяет пустое пространство между элементом и его левой границей. Значения: числовое значение (в пикселах или единицах em), процентное отношение или 0
padding-right	Определяет пустое пространство между элементом и его правой границей. Значения: числовое значение (в пикселах или единицах em), процентное отношение или 0

Свойство	Описание
padding-top	Определяет пустое пространство между элементом и его верхней границей. Значения: числовое значение (в пикселах или единицах em), процентное отношение или 0
page-break-after	Определяет, будет ли выполняться разрыв страницы после элемента. Значения: auto (по умолчанию), always, avoid, left или right
page-break-before	Определяет, будет ли выполняться разрыв страницы до элемента. Значения: auto (по умолчанию), always, avoid, left или right
page-break-inside	Определяет, будет ли выполняться разрыв страницы внутри элемента. Значения: auto (по умолчанию) или avoid
position	Конфигурирует позиционирование элемента. Значения: static (по умолчанию), absolute, fixed или relative
right	Определяет позицию смещения относительно правого края элемента-контейнера. Значения: числовое значение (в пикселах или единицах em), процентное отношение или значение auto (по умолчанию)
text-align	Определяет выравнивание текста по горизонтали в элементе. Значения: left (по умолчанию), right, center или justify
text-decoration	Определяет, будет ли текст декорирован. Значения: none (по умолчанию), underline, overline, line-through или blink
text-indent	Определяет отступ первой строки блочного элемента. Значения: числовое значение (в пикселах или единицах em) или процентное отношение
text-outline	Свойство CSS3; конфигурирует контур вокруг текста, отображаемого в элементе. Значения: от одного до двух числовых значений (в пикселах или единицах em) для указания толщины и (необязательно) радиуса размытия, а также допустимое значение цвета
text-shadow	Свойство CSS3; задает эффект тени тексту, отображаемому внутри элемента. Значения: от трех до четырех числовых значений (в пикселах или единицах em) для обозначения смещения по горизонтали и по вертикали, радиуса размытия и размера тени (необязательно), а также допустимое значение цвета
text-transform	Изменяет регистр шрифта текста в элементе. Значения: none (по умолчанию), capitalize, uppercase или lowercase
top	Определяет позицию смещения относительно верхнего края элемента-контейнера. Значения: числовое значение (в пикселах или единицах em), процентное отношение или значение auto (по умолчанию)
transform	Свойство CSS3; определяет преобразование или трансформацию при отображении элемента. Значения: функция преобразования, к примеру, scale(), translate(), matrix(), rotate(), skew() или perspective()
transition	Свойство CSS3; общее для конфигурирования переходов свойства CSS.

Свойство	Описание
	Значения: перечислите свойства `transition-property`, `transition-duration`, `transition-timing-function` и `transition-delay`, разделив их пробелами. Установленные по умолчанию значения можно опустить, однако первая единица времени будет относиться к свойству `transition-duration`
`transition-delay`	Свойство CSS3; определяет начало перехода; по умолчанию установлено значение `0` (без задержки) или вы можете ввести числовое значение, чтобы указать время (обычно в секундах)
`transition-duration`	Свойство CSS3; указывает период времени, на который применяется переход. Значение по умолчанию 0 задает немедленный переход, в других случаях для определения времени используйте числовые значения (обычно указывается в секундах)
`transition-property`	Свойство CSS3; указывает свойство CSS, к которому применяется переход. Значения: список подходящих свойств можно найти на сайте **www.w3.org/TR/css3-transitions**
`transition-timing-function`	Свойство CSS3; задает изменения скорости перехода, описывая, как рассчитываются непосредственные значения свойства. Значения: `ease` (по умолчанию), `linear`, `ease-in`, `ease-out`, `ease-in-out`
`vertical-align`	Определяет выравнивание по вертикали элемента. Значения: числовое значение (в пикселах или единицах em), процентное отношение, значения `baseline` (по умолчанию), `sub`, `super`, `top`, `text-top`, `middle`, `bottom` или `text-bottom`
`visibility`	Определяет, отображается ли элемент. Значения: `visible` (по умолчанию), `hidden` или `collapse`
`white-space`	Задает пробельный символ внутри элемента. Значения: `normal` (по умолчанию), `nowrap`, `pre`, `pre-line` или `pre-wrap`
`width`	Определяет ширину элемента. Значения: числовое значение (в пикселах или единицах em), процентное отношение или значение `auto` (по умолчанию)
`word-spacing`	Конфигурирует пробелы между словами в тексте. Значения: числовое значение (в пикселах или единицах em) или `auto` (по умолчанию)
`z-index`	Определяет порядок расположения элементов. Значения: числовое значение или `auto` (по умолчанию)

Приложение Г
ИСПОЛЬЗОВАНИЕ ПРОТОКОЛА FTP ДЛЯ ПУБЛИКАЦИИ САЙТОВ

После того как вы получите пространство на жестком диске сервера хостинговой компании, необходимо будет загрузить файлы. Обычно файлы передаются с помощью протокола File Transfer Protocol (протокола передачи файлов). Протокол — это соглашение или стандарт, который позволяет компьютерам общаться друг с другом. Протокол FTP используется для копирования файлов и папок в Интернете.

FTP использует два порта для соединения по сети: один для данных (обычно это порт 20) и один для команд управления (порт 21). Номера портов, используемые в Интернете, можно найти по адресу **www.iana.org/assignments/port-numbers.**

Приложения FTP

Существует много приложений для FTP, доступных во Всемирной паутине, в том числе:

- **Filezilla**

  Для операционных систем Windows, OS X, Linux

  filezilla-project.org

  Бесплатная программа

- **SmartFTP**

  Для операционной системы Windows

  www.smartftp.com

  Бесплатная программа

- **CuteFTP**

  Для операционных систем Windows, Mac

  www.cuteftp.com

  Доступна для скачивания бесплатная пробная версия

- **WS_FTP**

  Для операционной системы Windows

  www.ipswitch.com

  Доступна для скачивания бесплатная пробная версия.

Соединение с FTP

Ваш хостинг-провайдер предоставит вам следующую информацию наряду с особыми указаниями касательно того, должен ли FTP-сервер работать в активном или пассивном режиме:

FTP-хостинг: *адрес сервера*.

Имя пользователя: *ваше имя пользователя*.

Пароль: *ваш пароль*.

Обзор приложения FileZilla

В этом разделе описаны приемы работы с программой FileZilla, бесплатным FTP-клиентом для операционных систем Windows, OS X и Linux. Загрузить дистрибутив программы FileZilla вы можете по ссылке **filezilla-project.org/download.php?type=client** или же воспользоваться дистрибутивом, записанном на диске, прилагающемся к книге. После того как вы загрузите FTP-клиент, следуя инструкциям установщика, инсталлируйте программу на ваш компьютер.

Запуск и авторизация

Запустите программу FileZilla или другой FTP-клиент. Введите информацию, которую требует ваш веб-хостинг (такую как адрес FTP-хоста, логин (имя пользователя) и пароль), и начните соединение. Снимок экрана FTP-клиента FileZilla после успешного соединения показан на рис. 1.

Изучая рис. 1, обратите внимание на текстовые поля в верхней части экрана для ввода адреса хоста (сервера), имени пользователя и пароля. Ниже расположена область журнала сообщений FTP-сервера. Просмотрите информацию в этой области и убедитесь, что соединение прошло

успешно. Далее обратите внимание, что окно приложения делится две части — панели: правую и левую. Панель слева, **Локальный сайт** (Local site), отображает информацию о вашем компьютере и позволяет переходить к его дискам, папкам и файлам. Панель справа, **Удаленный сайт** (Remote site), отображает список каталогов удаленного компьютера и позволяет взаимодействовать с ними.

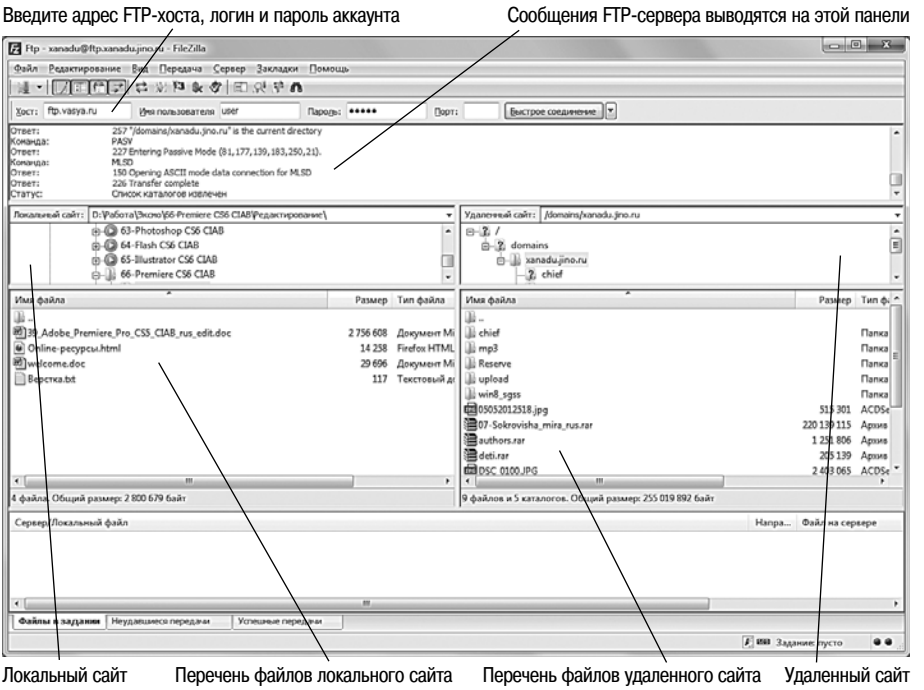


Рис. 1. Главное окно FTP-клиента FileZilla

Загрузка файлов

Перенести файл с локального компьютера на удаленный веб-сайт очень легко: просто выделите файл, щелкнув по нему правой кнопкой мыши в левой панели (список файлов локального сайта) и перетащите его в правую панель (список файлов удаленного сайта).

Скачивание файлов

Если вам необходимо скачать файл с вашего сайта на локальный компьютер, просто перетащите файл с помощью мыши из правой пане-

ли (список файлов удаленного сайта) в левую (список файлов локального сайта).

Удаление файлов

Для удаления файла с сервера щелкните по нему правой кнопкой мыши (в правой панели) и выберите команду **Удалить** (Delete) из контекстного меню.

Смело изучайте другие функции, предлагаемые программой FileZilla (и большинством других FTP-клиентов). Щелкните правой кнопкой мыши по перечню файлов удаленного сайта. Откроется контекстное меню, содержащее множество команд, в том числе команды для переименования файла, создания новой директории (папки) и просмотра файла.

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ

В

Г

Д

Ж

З

И

Р

С

Т

У

Ф

Х

Ц

Ш

Э

Я

Производственно-практическое издание

МИРОВОЙ КОМПЬЮТЕРНЫЙ БЕСТСЕЛЛЕР

Терри Фельке-Моррис

БОЛЬШАЯ КНИГА ВЕБ-ДИЗАЙНА

Директор редакции *Е. Кальёв*
Ответственный редактор *В. Обручев*
Художественный редактор *Г. Федотов*

ООО «Издательство «Эксмо»
127299, Москва, ул. Клары Цеткин, д. 18/5. Тел. 411-68-86, 956-39-21.
Home page: **www.eksmo.ru** E-mail: **info@eksmo.ru**

Подписано в печать 25.09.2012. Формат 70x100$^{1}/_{16}$.
Печать офсетная. Усл. печ. л. 49,26.
Тираж экз. Заказ

ISBN 978-5-699-55404-1

Оптовая торговля книгами «Эксмо»:
ООО «ТД «Эксмо». 142700, Московская обл., Ленинский р-н, г. Видное,
Белокаменное ш., д. 1, многоканальный тел. 411-50-74.
E-mail: **reception@eksmo-sale.ru**

По вопросам приобретения книг «Эксмо» зарубежными оптовыми покупателями *обращаться в отдел зарубежных продаж ТД «Эксмо»*
E-mail: **international@eksmo-sale.ru**

International Sales: *International wholesale customers should contact Foreign Sales Department of Trading House «Eksmo» for their orders.*
international@eksmo-sale.ru

По вопросам заказа книг корпоративным клиентам, в том числе в специальном оформлении,
обращаться по тел. 411-68-59, доб. 2299, 2205, 2239, 1251.
E-mail: **vipzakaz@eksmo.ru**

Оптовая торговля бумажно-беловыми и канцелярскими товарами для школы и офиса «Канц-Эксмо»:
Компания «Канц-Эксмо»: 142702, Московская обл., Ленинский р-н, г. Видное-2,
Белокаменное ш., д. 1, а/я 5. Тел./факс +7 (495) 745-28-87 (многоканальный).
e-mail: **kanc@eksmo-sale.ru**, сайт: **www.kanc-eksmo.ru**

Полный ассортимент книг издательства «Эксмо» для оптовых покупателей:
В Санкт-Петербурге: ООО СЗКО, пр-т Обуховской Обороны, д. 84Е.
Тел. (812) 365-46-03/04.
В Нижнем Новгороде: Филиал ООО «Торговый Дом «Эксмо» в Нижнем Новгороде,
ул. Маршала Воронова, д. 3. Тел. (8312) 72-36-70.
В Казани: Филиал ООО «РДЦ-Самара», ул. Фрезерная, д. 5. Тел. (843) 570-40-45/46.
В Ростове-на-Дону: Филиал ООО «Издательство «Эксмо» в г. Ростове-на-Дону,
пр-т Стачки, 243 «А». Тел. +7 (863) 305-09-12/13/14.
В Самаре: ООО «РДЦ-Самара», пр-т Кирова, д. 75/1, литера «Е».
Тел. (846) 269-66-70.
В Екатеринбурге: ООО «РДЦ-Екатеринбург», ул. Прибалтийская, д. 24а.
Тел. +7 (343) 272-72-01/02/03/04/05/06/07/08.
В Новосибирске: ООО «РДЦ-Новосибирск», Комбинатский пер., д. 3.
Тел. +7 (383) 289-91-42. E-mail: **eksmo-nsk@yandex.ru**
В Киеве: ООО «РДЦ Эксмо-Украина», Московский пр-т, д. 6.
Тел./факс: (044) 498-15-70/71.
В Донецке: ул. Артема, д. 160. Тел. +38 (062) 381-81-05.
В Харькове: ул. Гвардейцев Железнодорожников, д. 8. Тел. +38 (057) 724-11-56.
Во Львове: ул. Бузкова, д. 2. Тел. +38 (032) 245-01-71.
Интернет-магазин: www.knigka.ua. Тел. +38 (044) 228-78-24.
В Казахстане: ТОО «РДЦ-Алматы», ул. Домбровского, д. 3а.
Тел./факс (727) 251-59-90/91. RDC-Almaty@eksmo.kz

Полный ассортимент продукции издательства «Эксмо»
можно приобрести в магазинах «Новый книжный» и «Читай-город».
Телефон единой справочной: 8 (800) 444-8-444.
Звонок по России бесплатный.

В Санкт-Петербурге в сети магазинов «Буквоед»:
«Парк культуры и чтения», Невский пр-т, д. 46. Тел. (812) 601-0-601
www.bookvoed.ru

По вопросам размещения рекламы в книгах издательства «Эксмо» обращаться в рекламный отдел. Тел. 411-68-74.

Интернет-магазин ООО «Издательство «Эксмо»
www.fiction.eksmo.ru
Розничная продажа книг с доставкой по всему миру.
Тел.: +7 (495) 745-89-14 . E-mail: **imarket@eksmo-sale.ru**

За **HTML5** и **CSS3** будущее Всемирной паутины, и «Большая книга веб-дизайна» расскажет вам обо всем, что необходимо знать для овладения этими инновационными стандартами.

Читая книгу, вы изучите современный мир Всемирной паутины, рассмотрите приемы подготовки успешных и доступных веб-сайтов, научитесь верстке на языке HTML5 с использованием всех новых элементов стандарта. Вы также освоите каскадные таблицы стилей (CSS), с помощью которых сможете создать самые невероятные веб-сайты. Познакомившись с языком JavaScript, вы научитесь создавать простые сценарии, превращающие ваши страницы в интерактивные многофункциональные ресурсы. А изучив главы, посвященные электронной коммерции и поисковой оптимизации, научитесь зарабатывать на своем сайте деньги.

Терри Фельке-Моррис — адъюнкт-профессор, обладатель степени магистра наук по информационным системам. Она более 20 лет работает в области информационных технологий в бизнесе и производстве и является членом оперативной группы по проектному изучению веб-стандартов. Свой первый веб-сайт она создала в 1986 году и с тех пор работает над проектами для Всемирной паутины.

Самый серьезный и комплексный охват всех вопросов веб-дизайна. Практические задания и упражнения отлично иллюстрируют материал. Лучшая книга для начинающих.

Журнал Web designer, http://www.webdesignermag.co.uk/

Книга рассказывает о:

- концепции Интернета
- верстке веб-страниц на языке HTML5
- использовании каскадных таблиц стилей CSS3
- сценариях JavaScript
- добавлении видео - и аудиообъектов средствами HTML5 без плагинов
- элементах веб-форм
- эффективном веб-дизайне и доступности веб-страниц
- разработке веб-страниц для мобильных устройств
- поисковой оптимизации
- электронной коммерции

и многом другом.

ISBN 978-5-699-55404-1