ВЛАД МЕРЖЕВИЧ

HTML и CSS

НА ПРИМЕРАХ

ОФОРМЛЕНИЕ ЭЛЕМЕНТОВ ФОРМ, ТАБЛИЦ И СПИСКОВ С ПОМОЩЬЮ СТИЛЕЙ

ИСПОЛЬЗОВАНИЕ СЛОЕВ В ДИЗАЙНЕ ВЕБ-СТРАНИЦ

СОЗДАНИЕ ВКЛАДОК, ВЕРТИКАЛЬНЫХ, НИСПАДАЮЩИХ, ПЛАВАЮЩИХ И ДРУГИХ МЕНЮ

СОВЕТЫ, РЕЦЕПТЫ И ХИТРОСТИ ПРИ РАБОТЕ С HTML И CSS

bhv

Влад Мержевич

HTML и CSS
НА ПРИМЕРАХ

Санкт-Петербург
«БХВ-Петербург»
2005

УДК 681.3.068+800.92HTML
ББК 32.973.26-018.1
М52

Мержевич В. В.

М52 HTML и CSS на примерах. — СПб.: БХВ-Петербург, 2005. — 448 с.: ил.

ISBN 5-94157-360-X

На практических примерах раскрываются технологии HTML и CSS в плане решения различных аспектов создания веб-страниц. Приведены возможности и средства по оформлению текста и изменению его вида. Описаны основные графические форматы, которые используются на сайтах. Освещены вопросы работы с рисунками, ссылками, списками, линиями и рамками. Раскрыты возможности управления видом таблицы, ускорения загрузки табличных данных, использования шаблонов. Рассмотрены элементы форм, их параметры и примеры изменения оформления с помощью цвета, изображений и рамок. Приведены способы выравнивания рисунков, слоев и текста, использования отступов и полей и многие другие приемы оформления веб-страниц. Даются подробные сведения об особенностях популярных браузеров и о том, как они работают с тегами и стилями. Объясняется, как учитывать различия между браузерами и создавать универсальные документы, которые будут корректно в них отображаться.

Детальные примеры и пошаговое описание действий позволяют реализовать приведенные рекомендации на практике.

Для веб-разработчиков

УДК 681.3.068+800.92HTML
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Евгений Рыбаков*
Зав. редакцией	*Григорий Добин*
Редактор	*Наталья Довгулевич*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Дизайн обложки	*Игоря Цырульникова*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 22.04.05.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 36,12.
Тираж 5000 экз. Заказ №
"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 5-94157-360-X

Оглавление

Введение

HTML (HyperText Markup Language, язык разметки гипертекста) — это прежде всего система верстки, которая определяет, как и какие элементы должны располагаться на веб-странице. Информация на сайте, способ ее представления и оформления зависят исключительно от разработчика и тех целей, которые он перед собой ставит. Вместе с тем, HTML имеет ряд ограничений, которые породили самые неожиданные способы верстки, в частности применение изображений вместо текста, активное использование таблиц с невидимой границей, прозрачных рисунков для контроля пустого пространства и т. д. Подобные методики увеличивают сложность разработки *сайта* — совокупности связанных между собой веб-страниц — ведь вместо того, чтобы заниматься творчеством, приходится решать, как обойти то или иное ограничение. Стили частично решают эти проблемы, в то же время, не заменяя собой HTML, но дополняя его механизмы.

Что же такое стили или *CSS* (Cascading Style Sheets, каскадные таблицы стилей)? *Стилем* называется набор параметров форматирования, который применяется к элементам документа, чтобы изменить их внешний вид. Возможность работы со стилями издавна включают в развитые издательские системы и текстовые редакторы, тем самым позволяя одним нажатием кнопки придать тексту заданный, заранее установленный вид. Теперь это доступно и создателям сайта, когда цвет, размеры текста и другие параметры хранятся в определенном месте и легко "прикручиваются" к любому тегу. Еще одним преимуществом стилей является то, что они предлагают намного больше возможностей для форматирования, чем обычный HTML. CSS представляет собой мощную систему, расширяющую возможности дизайна и верстки веб-страниц.

Любой сайт отображается в специальной программе просмотра, называемой *браузером*. Таким образом, получается, что разработчики сайта зависят от производителей браузеров, от того, насколько корректно и правильно они

воплощают стандарты, разработанные Консорциумом W3. Дело осложняется тем, что популярных браузеров существует несколько, и они по-разному интерпретируют указанные стандарты.

У разработчика сайта в подобной ситуации существует два подхода. Первый — руководствоваться существующими стандартами и верстать сайт с их учетом, а второй — понимать специфику поведения браузеров в разных случаях и следовать ей. При этом браузеры могут использовать противоречивые подходы, и чтобы создать действительно универсальный код, требуется приложить немало усилий.

Несмотря на то, что второй подход сложнее, лучше руководствоваться именно им. Здесь следует принять во внимание следующий момент. Большинство пользователей, которые заходят на сайты, интересуются размещенной на них информацией и вообще не знают ни о каком HTML и CSS. Они пользуются преимущественно своим любимым браузером или тем, что установлен в операционной системе по умолчанию. Если сайт отображается с ошибками, то пользователю проще перейти на другой сайт, ведь существует их достаточное количество по любой теме. Разработчику сайта необходимо удержать пользователя, не просто предоставив ему нужную информацию, но и правильно ее подав.

Таким образом, при создании сайта необходимо решить следующие задачи — придать веб-странице желаемый внешний вид и обеспечить его корректное отображение в популярных браузерах.

Верстка веб-страниц — это не просто знание приемов и хитростей создания различных эффектов. Это умение предугадывать результат действий с элементами веб-страниц и понимание особенностей различных браузеров, которые могут по-разному отображать сайт. Деятельность разработчика сайта сродни работе шеф-повара, который точно знает, какие ингредиенты и в каком количестве нужно положить, чтобы улучшить вкус готового блюда. В большинстве своем простые, но действенные рецепты, содержащиеся в данной книге, помогут вам создавать впечатляющие и работоспособные сайты.

Возможно, многие описанные техники вам и не пригодятся, но их главная задача — показать те возможности и перспективы, которые доступны при создании веб-страниц.

Для кого предназначена книга

Эта книга будет полезна всем тем, кто занимается версткой веб-страниц, создает сайты и так или иначе задает себе вопрос: "Как сделать..?". И дело совсем не в опыте разработчика, поскольку технологии не стоят на месте и к любой задаче можно подойти творчески и с разных сторон.

Для новичков мы расскажем, как создавать и добавлять на веб-страницу основные элементы — ссылки, рисунки, таблицы, формы и списки. Подробные примеры, описание всех действий и комментарии в листингах позволяют легко воспроизвести приведенные технологии на практике и модернизировать код под свои нужды. Множество советов, посвященных разным аспектам создания веб-страниц, помогут закрепить первоначальные навыки и знания или сделать первые шаги в этом направлении. Кроме того, в книге приводятся основы веб-дизайна при работе с текстом, изображениями, таблицами и т. д.

Опытные пользователи получат более подробные сведения об особенностях популярных браузеров и о том, как они работают с тегами и стилями. Это позволит учитывать различия между браузерами и создавать универсальные документы, которые будут корректно в них отображаться. Не всегда элементы веб-страниц следует применять в той роли, которая для них отведена. На протяжении всей книги наряду с традиционным использованием объектов веб-страницы показаны расширенные функции, которые можно на них возлагать, открывая таким образом неожиданную сферу применения разных элементов и их параметров.

Наконец, искушенные специалисты по созданию сайтов смогут использовать книгу как справочное руководство, заглядывая в нее для получения информации о тегах, стилях и их параметрах. Подробное содержание и предметный указатель позволяют быстро получить ответ на множество вопросов, возникающих при работе с HTML и CSS.

Как организована книга

Материал в книге представлен так, что ее можно читать последовательно с самого начала или пользоваться ею в качестве справочника, заглядывая в ту главу, которая в данный момент больше всего интересует. Все главы следует рассматривать как независимые — это позволяет вам сразу перейти к теме, которую вы считаете для себя более важной. Однако между ними существует и взаимосвязь, поэтому, чтобы не повторять дважды одни и те же приемы, на них дается ссылка. Кроме того, следует понимать, что множество техник очень тесно переплетены между собой, и их невозможно без повторений отнести к одной теме. Так что разные приемы в некотором смысле разбросаны по главам, но стройно и логично следуют один из другого, вдобавок, используемые в книге определения и термины вводятся постепенно, начиная с первых глав. В первый раз рекомендуется прочитать книгу с самого начала, а затем обращаться к нужному разделу по необходимости.

Книга состоит из одиннадцати глав и приложения.

- *Глава 1 "Текст"*. В ней приведены основы *типографики* — науки о шрифтах и текстах и особенности ее применения к веб-страницам. Также по-

казаны возможности и средства по оформлению текста и изменению его вида.

- *Глава 2 "Изображения"*. Описываются графические форматы, которые используются на сайтах, добавление рисунков в документ и манипуляции с ними.
- *Глава 3 "Ссылки"*. В этой главе представлено создание ссылок, изменение их вида и способы акцентирования на них внимания.
- *Глава 4 "Списки"*. Приводятся виды списков и способы добавления их на страницу, а также изменение оформления нумерованных, маркированных и многоуровневых списков.
- *Глава 5 "Линии и рамки"*. Описываются различные способы создания горизонтальных и вертикальных линий, добавления рамок разного стиля вокруг текста, фреймов и таблиц. Указаны способы создания рамок с заголовками, панелей.
- *Глава 6 "Таблицы"*. Рассматриваются возможности и особенности таблиц, а также их создание. В главе рассказывается, как управлять видом таблицы, ускорять загрузку табличных данных, выравнивать их, а, кроме того, приведены шаблоны оформления таблиц, которые можно брать за основу.
- *Глава 7 "Формы"*. В главе рассматривается, что такое формы и как их использовать; описаны элементы форм, приведены их возможные параметры и примеры изменения оформления с помощью цвета, изображений и рамок.
- *Глава 8 "Выравнивание элементов"*. Приводятся способы выравнивания на веб-странице рисунков, слоев и текста, кроме того, поясняется, как выравнивать элементы относительно друг друга и элементы, которые находятся внутри других.
- *Глава 9 "Отступы и поля"*. В главе объясняется, чем различаются между собой поля и отступы, и даны способы их применения для верстки веб-страниц.
- *Глава 10 "Элементы оформления"*. В этой главе представлены различные приемы, не вошедшие в предыдущие разделы, — изменение цвета полосы прокрутки, вида курсора мыши, ссылка на значок сайта и добавление тени к элементам.
- *Глава 11 "Создание меню"*. Описаны виды меню, которые встречаются на веб-странице, и способы их создания с использованием скриптов и стилей.
- *Приложение "Свойства CSS"*. Описаны основные стилевые параметры, браузеры, которые их поддерживают, приводятся примеры их использования.

Соглашения, используемые в книге

- В тексте неоднократно используется термин *браузер*. Под ним подразумевается не конкретный, а некоторый обобщенный обозреватель вебстраниц. Предполагается, что все листинги, если это не оговорено особо, правильно работают в основных браузерах: Internet Explorer 5.5, Netscape 6, Opera 6, Firefox 1.0 и старше. В предшествующих версиях не поддерживаются современные технологии и спецификации, поэтому использование таких браузеров нецелесообразно. Тем не менее для информации в *приложении* приведены свойства CSS с указанием версий браузеров, в том числе и относительно старых версий, в которых они работают.
- Один и тот же документ может отображаться в разных браузерах не совсем идентично из-за некоторых различий между ними, поэтому результат выполнения примера в одном браузере может незначительно отличаться при использовании другого браузера. Такие различия в книге не акцентируются, кроме случаев явных несоответствий.
- Хотя любые теги нечувствительны к регистру, и их можно писать как строчными, так и прописными символами, в тексте книги теги всегда обозначены заглавными буквами. В листингах теги также пишутся прописными символами, когда они упоминаются в таблице стиля. Такое выделение предназначено для акцентирования внимания на тегах, встречающихся по ходу текста, и отделения их от названий классов, параметров и значений.

Поддержка

Ваши замечания, предложения и вопросы направляйте автору на адрес электронной почты **vlad@htmlbook.ru**.

Множество материалов и примеров, связанных с темой книги, можно посмотреть в Интернете на сайте автора по адресу **http://www.htmlbook.ru**.

ГЛАВА 1

Текст

Ради чего люди заходят на сайты, а потом вновь и вновь возвращаются на них? Вопрос риторический, и каждый может ответить на него по-своему. Тем не менее большая часть сайтов, несмотря на их разнородную направленность, имеет нечто общее. Это интересная, привлекающая посетителей информация, а также интерактивная возможность пообщаться с другими людьми. И в том, и другом случае дело не обходится без текста. Именно текст служит основным компонентом практически любого сайта. Хотя в CSS существует множество атрибутов, ориентированных на речевые браузеры, стоит признать, что в данный момент они должным образом не работают. Поэтому текстовые описания продолжают оставаться главной формой передачи информации посетителям сайтов. Конечно, в проектах, явно ориентированных на содержание, вроде сайта с анекдотами или статьями, текст выполняет наиболее важную роль, но к изображениям с музыкой также стоит добавлять их описания.

Красиво и элегантно оформленный текст может лучше передать задумку автора и привлечь к себе внимание. К тому же с таким текстом приятнее работать, он лучше воспринимается, и пользователи это ценят.

HTML и CSS предоставляют множество средств для работы с текстом — от управления шрифтом до форматирования параграфов. Сюда относится изменение размера шрифта, его гарнитуры и начертания, а также выравнивание текста, регулировка межбуквенного и межстрочного расстояния и многое другое.

Аз и буки шрифтовой науки

Типографика — наука о работе со шрифтами и текстами — это скорее искусство, соединенное с психологией восприятия. Ее задача состоит в том, чтобы точно и с нужным настроем донести печатное слово до читателя.

К сожалению, типографика может применяться на сайтах не в полной мере, поскольку HTML и CSS предлагают ограниченный набор средств, предназначенных для работы со шрифтами. Это обстоятельство подталкивает к техническому финту, когда заголовки или другие текстовые элементы, требующие особого подхода и шрифтов, готовятся в графической программе, а на веб-страницу помещаются уже в виде изображений. Несмотря на столь грустную особенность, управлять видом и типом шрифта на сайте, хоть и с ограничениями, все же возможно.

Шрифты

Существуют тысячи шрифтов, которые предназначены для оформления текстов. При этом следует отметить, что число шрифтов, применяемых для набора текста на сайтах, существенно ниже. Конечно, ничто не мешает выбрать и задать, например, для заголовка, вычурный шрифт, установленный на локальном компьютере. Но если такого шрифта на компьютере пользователя нет, то текст будет отображаться шрифтом, установленным в браузере по умолчанию. Получается, что усилия разработчика пропадут даром. Есть три способа избежать этого.

- Создать надпись в графическом редакторе и вставить ее на веб-страницу как изображение. Этот метод подключает всю мощь графических систем по управлению текстом, при этом допустимо включать любые шрифты. Из недостатков самый главный состоит в том, что рисунок так просто не изменишь, и придется вновь обращаться к редактору. К тому же существует вероятность того, что пользователь отключил показ рисунков в браузере. Так что рисунки в качестве текста применяются только для создания небольших и постоянных надписей вроде заголовков.
- Загрузить пользовательский шрифт. Действительно, если можно загружать по сети изображения, почему бы то же самое не проделать и со шрифтами? CSS поддерживает эту возможность, но в реальности она используется весьма редко. Связано это с тем, что формат шрифта должен отличаться от привычного и применяемого в системе. К тому же не все браузеры поддерживают данную технологию, что снижает универсальность подхода, а пользователи не любят загружать лишнюю информацию.
- Воспользоваться стандартными шрифтами, встроенными в браузер и операционную систему. Этот способ является пока самым распространенным для указания шрифта на веб-странице.

Все многообразие шрифтов (несмотря на то, что они совершенно разные по виду и форме) можно поделить на определенные группы: шрифты с засечками, шрифты без засечек, моноширинные и декоративные шрифты.

Шрифты с засечками

Такие шрифты характеризуются засечками — поперечными элементами на концах букв, которые называются еще *серифами* (serif) (рис. 1.1).

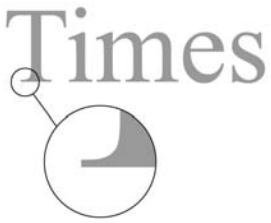


Рис. 1.1. Текст, в котором используется шрифт с засечками

Засечки придают шрифту черты рукописных текстов, словно написанных пером, хотя современные виды шрифтов с засечками в большинстве своем уже не несут в себе следов каллиграфии.

Шрифты с засечками активно применяются для набора основного текста, поскольку это облегчает восприятие больших объемов текста. Засечки заставляют взгляд читателя скользить вдоль них и одновременно разделяют отдельные буквы, чтобы они не сливались между собой. Также такие шрифты могут использоваться и для написания заголовков. На экране монитора при уменьшении размера шрифта шрифт с засечками начинает хуже передавать начертание, поэтому для мелких надписей рекомендуется воспользоваться шрифтом без засечек.

Шрифты без засечек

Шрифты без засечек, называемые также *гротесками* или *рублеными шрифтами*, не имеют серифов на концах букв, поэтому для их обозначения используется термин sans-serif (в переводе с французского — "без серифа"). На сайтах подобные шрифты нашли применение в самых разнообразных элементах: заголовках, надписях на кнопках форм, основном тексте и т. д. Преимущество шрифта без засечек состоит в том, что он одинаково хорошо передает текст как в крупном, так и в мелком начертании (рис. 1.2).

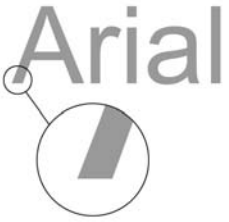


Рис. 1.2. Текст со шрифтом без засечек

Следует понимать, что удобочитаемость текста зависит от множества факторов, которые определяются шрифтом, разрешением и размером монитора, настройками системы, длиной строки и т. д. Поэтому решение о выборе шрифтов с засечками или без можно сделать только исходя из готового макета веб-страницы.

Моноширинный шрифт

Все символы *моноширинного шрифта* имеют одинаковую ширину, независимо от начертания буквы отводимое под нее пространство не меняется (рис. 1.3).

Рис. 1.3. Текст с моноширинными символами

Такой подход имеет как преимущества, так и недостатки. Из достоинств следует отметить то, что текстом удобно манипулировать, если требуется точно разместить одну надпись под другой. Благодаря тому, что символы выступают в роли кирпичиков одинаковых размеров, можно строить из них целые текстовые картины. Такого рода искусство до сих пор популярно, и подобные "изображения" встречаются порой в Интернете.

Опять же в силу того, что все символы одинаковы по ширине, это накладывает отпечаток на комфортность чтения текста. Понятно, что хотя буквы "Г" и "Ш" могут быть равными по размеру, это приводит к излишнему растягиванию символов, что идет в ущерб их красоте и элегантности.

Что касается применения моноширинного шрифта на сайте, то обычно он используется, когда требуется привести код программы. Добавляя пробелы перед строками, их можно выравнивать между собой, а это позволяет лучше ориентироваться в коде.

К моноширинным шрифтам, применяемым на сайтах, относится шрифт Courier и его разновидности.

Декоративные шрифты

Эту категорию составляют шрифты, не вошедшие в предыдущие описания. Подобные шрифты используют для создания определенного настроения на сайте, поскольку в большинстве своем они трудны для восприятия и для чтения. Наиболее часто декоративные шрифты применяются для заголовков, текстовых выделений и, как правило, никогда для основного текста (рис. 1.4).

В Windows для размещения на сайте по умолчанию обычно доступен только один декоративный шрифт — Comic Sans MS.

Рис. 1.4. Декоративный шрифт

Размер текста

Размер шрифта или, как он еще называется в типографике — *кегль*, определяется высотой символов, которая, в свою очередь, на веб-странице может задаваться как в относительных (пикселы (px), проценты (%)), так и в абсолютных единицах (дюймы (in), миллиметры (mm), пункты (pt) и др). Пожалуй, самой распространенной единицей для указания размера шрифта является *пункт*. Многие люди привыкли задавать размер шрифта в текстовых редакторах, например 12. А что это число означает, не понимают. Так это и есть пункты, они родные. Конечно, они нам не родные, мы привыкли измерять все в миллиметрах и подобных единицах, но пункт, пожалуй, единственная величина не из метрической системы измерения, которая используется у нас повсеместно. И все благодаря текстовым редакторам и издательским системам.

На вид шрифта влияет не только заданный размер, но и выбор гарнитуры. Шрифт Arial выглядит крупнее, чем шрифт Times того же размера, а шрифт Courier New чуть меньше шрифта Arial (рис. 1.5). Учитывайте эту особенность при выборе шрифта и его размеров.

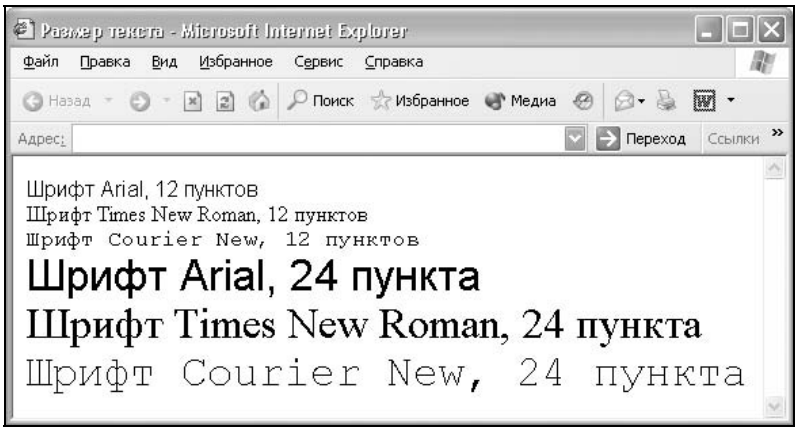


Рис. 1.5. Размеры текста на веб-странице

На рис. 1.5 приведены три типа шрифтов с размером 12 и 24 пункта. Легко заметить, что при одинаковых заданных размерах текст различается как по высоте, так и по насыщенности.

Насыщенность

Насыщенностью называют увеличение толщины линий шрифта и соответственно контраста. Обычно различают четыре вида насыщенности: светлое начертание, нормальное, полужирное и жирное (рис. 1.6).

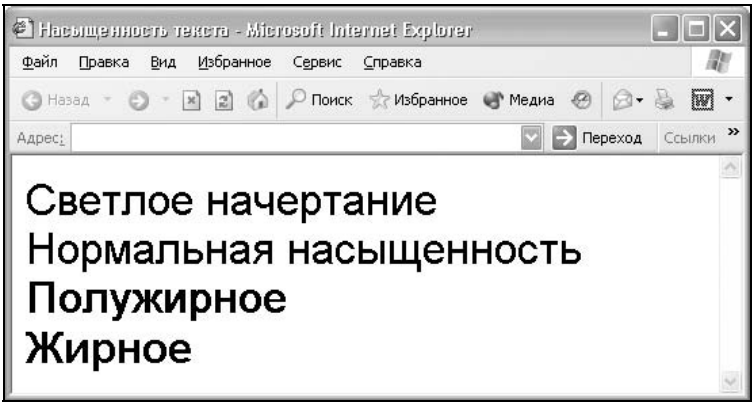


Рис. 1.6. Разная насыщенность шрифта

Браузер не всегда может адекватно показать требуемую насыщенность шрифта, это зависит от размеров текста и вида шрифта.

Наклон текста

Наклон определяется сдвигом шрифта на определенный угол. Различают два типа наклона: просто наклонный шрифт (oblique) и курсив (italic). Курсивный шрифт представляет собой не просто наклон отдельных символов, для шрифтов с засечками это полная переделка под новый стиль, имитирующий рукописный.

Следует отметить, что хотя браузеры и различают параметры italic и oblique, при этом они отображают текст как курсив.

Начертание

Существует несколько видов начертаний шрифта, которые встречаются на сайтах: нормальное, жирное, курсивное, подчеркнутое, перечеркнутое и выворотное (рис. 1.7). Все указанные вариации шрифта составляют его *гарнитуру*.

Большая часть начертаний в той или иной мере встречается в различных текстовых редакторах и допускает смешивание с другими. Например, можно сразу установить жирный курсивный текст. Что касается выворотки, то

здесь светлый текст выводится на темном фоне. Для больших объемов подобный текст (темные буквы на светлом фоне) читать труднее, чем в традиционном исполнении, поэтому выворотка обычно применяется для привлечения внимания читателя к тексту, например заголовку.

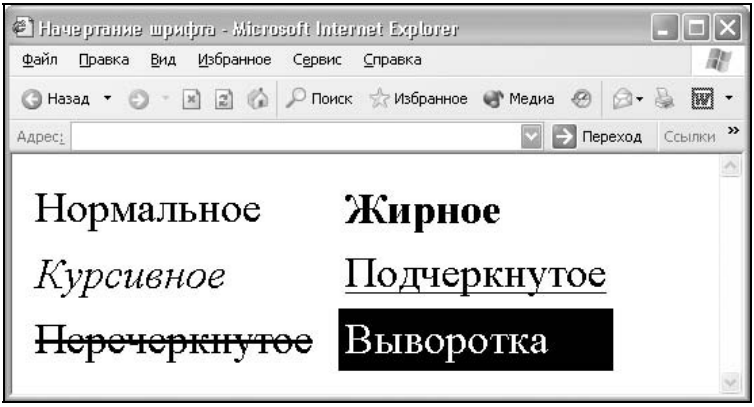


Рис. 1.7. Разное начертание шрифта текста

Работа со шрифтами

Чтобы управлять форматированием текста, в частности изменять начертание шрифта, можно воспользоваться специальными тегами или обратиться к стилям. В табл. 1.1 перечислены основные теги, которые применяются для изменения оформления текста.

Таблица 1.1. *Теги для управления видом текста*

Код HTML	Описание
`<b>Текст</b>`	Определяет жирное начертание шрифта
`<i>Текст</i>`	Устанавливает курсивное начертание шрифта
`<u>Текст</u>`	Текст, выделенный с помощью тега `U`, отображается подчеркиванием
`<sup>Текст</sup>`	Воспроизводит текст как верхний индекс. Шрифт при этом отображается выше базовой линии текста и уменьшенного размера
`<sub>Текст</sub>`	Отображает шрифт в виде нижнего индекса. Текст при этом располагается ниже базовой линии остальных символов строки и уменьшенного размера

Таблица 1.1 *(окончание)*

Код HTML	Описание
<s>Текст</s>	Тег S отображает текст как перечеркнутый. Этот тег аналогичен тегу STRIKE, но имеет сокращенную форму записи подобно тегам B, I и U. В любом случае, теги S и STRIKE осуждаются спецификацией HTML, взамен их рекомендуется использовать стили
<pre>Текст</pre>	Элемент PRE определяет блок предварительно форматированного текста. Такой текст отображается обычно моноширинным шрифтом и со всеми пробелами между словами. По умолчанию любое количество пробелов, идущих в коде подряд, на веб-странице отображается как один. Тег PRE позволяет обойти эту особенность и отображать текст, как требуется разработчику. Внутри контейнера PRE допустимо применять некоторые теги, кроме APPLET, BASEFONT, BIG, FONT, IMG, OBJECT, SMALL, SUB и SUP

Любые перечисленные в таблице теги можно использовать совместно друг с другом. Например, чтобы сделать шрифт одновременно жирным и курсивным, используется сочетание тегов B и I (листинг 1.1). Их порядок в данном случае не важен.

Листинг 1.1. Создание жирного курсивного текста

```
<html>
<body>
 <b><i>"А где же печенье и самогоноваренье?!"</i></b>, - воскликнул
Мальчиш-плохиш.
</body>
</html>
```

Жирный шрифт и курсив активно применяются для выделения текста в документах и привлечения к нему внимания читателя. Что касается подчеркивания, то его на веб-странице лучше вообще не использовать нигде, кроме ссылок. Подчеркивание давно уже ассоциируется именно со ссылками, поэтому любое его появление без должных на то причин просто вводит посетителя сайта в заблуждение. Да и читабельность такого текста хуже по сравнению с обычным написанием.

Теги STRONG и EM выполняют те же функции, что и теги B и I, но написание последних короче, привычнее и удобнее. Следует отметить, что теги B и STRONG, так же как I и EM, являются не совсем эквивалентными и заменяемыми. Тег B является тегом физической разметки и устанавливает жирный текст, а тег STRONG — тегом логической разметки и определяет важность по-

меченного текста. Такое разделение тегов на логическое и физическое форматирование изначально предназначалось для того, чтобы сделать HTML универсальным, в том числе не зависящим от устройства вывода информации. Теоретически, если воспользоваться, к примеру, речевым браузером, то текст, оформленный с помощью тегов `B` и `STRONG`, будет отмечен поразному. Однако получилось так, что в популярных браузерах результат использования этих тегов равнозначен.

Установка шрифта

Выбор шрифта для оформления текста осуществляется двумя способами: с помощью тега `FONT` или через стилевое свойство `font-family`. Добавление тега `FONT` осуждается спецификацией HTML как противоречащее концепции разделения оформления и содержания. Кроме того, при активном использовании этого тега повышается объем кода, и разработчику становится сложнее править текст. Так что в настоящее время про тег `FONT` можно забыть, и для указания нужного шрифта обратиться к стилям.

Атрибут `font-family` устанавливает семейство шрифтов, которое будет использоваться для оформления текста. Список шрифтов может включать одно или несколько названий, разделенных запятой. Если в имени шрифта содержатся пробелы, например, Trebuchet MS, оно должно заключаться в двойные или одинарные кавычки.

Когда браузер встречает первый шрифт в списке, он проверяет его наличие на компьютере пользователя. Если такого шрифта нет, берется следующее имя из списка и также анализируется на присутствие. Поэтому несколько шрифтов увеличивает вероятность того, что хотя бы один из них будет обнаружен на клиентском компьютере. Заканчивают список обычно ключевым словом, которое описывает тип шрифта — serif, sans-serif, cursive, fantasy или monospace. Таким образом, последовательность шрифтов лучше начинать с экзотических типов и заканчивать обобщенным именем, которое задает вид начертания (листинг 1.2).

Универсальные семейства шрифтов:

- `serif` — шрифты с засечками (антиквенные), типа Times;
- `sans-serif` — рубленые шрифты (шрифты без засечек или гротески), типичный представитель — Arial;
- `cursive` — курсивные шрифты, обычно подставляется Comic Sans MS;
- `fantasy` — декоративные шрифты; как правило, в браузере используется шрифт Zapf, если его нет, то текст отображается шрифтом, установленным по умолчанию;
- `monospace` — моноширинные шрифты, ширина каждого символа в таком семействе одинакова, например, шрифт Courier.

Листинг 1.2. Установка шрифта с помощью стилей

```
<html>
<head>
<style type="text/css">
 BODY {
  font-family: 'Times New Roman', Times, serif;
  font-size: 100%
}
 TD {
  font-family: Arial, sans-serif;
  font-size: 90%;
  font-weight: bold
}
 H1, H2, H3 {
  font-family: Verdana, Tahoma, Arial, sans-serif
}
</style>
</head>
<body>
...
</body>
</html>
```

В примере показано создание *глобального стиля*, который действует в пределах всего веб-документа, с помощью тега STYLE. Он обычно располагается в заголовке документа HEAD, но может также быть и в теле документа BODY. Чтобы сообщить браузеру, что он имеет дело с таблицей стилей, к тегу STYLE добавляется параметр type="text/css". Следует заметить, что современные браузеры корректно работают со стилями и при отсутствии этого параметра, он необходим для некоторых старых браузеров, которые могут не распознать содержимое контейнера STYLE.

Внутри STYLE указывается имя селектора, а в фигурных скобках после него имена параметров, которые отделяются от своего значения двоеточием. Все параметры разделяются точкой с запятой, в конце, перед закрывающей скобкой, этот символ можно опустить.

Так, в примере приведено добавление шрифтов к селектору BODY, он управляет видом всех шрифтов на веб-странице; к селектору TD, задающему шрифт для ячеек таблицы, и заголовкам разделов H1, H2, H3.

Селектором называется имя стилей, в котором указаны параметры форматирования. Селекторы делятся на несколько типов: селекторы тегов, классы и идентификаторы.

Селекторы тегов используются для определения стилей встроенных тегов HTML, как показано в листинге 1.2. Для тега определяются правила форматирования, такие как цвет, фон, размер и др. После того, как стиль тега переопределен, можно использовать данный тег как обычно, но при этом его вид будет уже другим.

Классы используются для создания стилей, которые можно применять к любому тегу HTML, для выделений или изменения стиля блока текста; они обозначаются точкой перед именем. Обращение к селектору класса происходит через параметр `class`, значением которого выступает имя класса без точки (см. листинг 1.4).

Идентификаторы обычно используются совместно с клиентскими программами, чтобы через них можно было динамически управлять параметрами стиля; для определения того, что это идентификатор, к имени добавляется символ решетки. Стиль любого тега можно связать с идентификатором, если к тегу добавить параметр `id` с указанием имени идентификатора без символа решетки (см. листинг 1.7).

Идентификаторы и классы обычно выполняют одну роль, поэтому они одинаково часто встречаются в текстах примеров.

Учитывая, что список шрифтов на компьютерах пользователей может сильно различаться в зависимости от операционной системы и собственных предпочтений, приведем список шрифтов, которые широко распространены и которые можно использовать без опаски.

Для Windows:

Arial, Comic Sans MS, Courier, Courier New, Lucida Console, Tahoma, Times, Times New Roman, Trebuchet MS, Verdana.

Для Linux:

Courier, Helvetica, Lucida, Times.

Для Mac:

Helvetica, Times, Times New Roman, Verdana.

Шрифты с одинаковыми именами в разных браузерах и системах могут незначительно отличаться друг от друга по форме или по размеру.

Цвет текста

Цвета текста и фона под ним можно задавать тремя способами.

1. *По его названию.* Браузеры поддерживают некоторые цвета по их названию. Самые распространенные ключевые слова — black (черный), white (белый), red (красный), green (зеленый), blue (синий) и др.
2. *По шестнадцатеричному значению.* Для задания цветов используются числа в шестнадцатеричном коде. Шестнадцатеричная система, в отличие от

десятичной системы, базируется, как следует из ее названия, на числе 16. Цифры будут следующие: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Цифры от 10 до 15 заменены латинскими буквами. Числа больше 15 в шестнадцатеричной системе образуются объединением двух чисел в одно. Например, числу 255 в десятичной системе соответствует число FF в шестнадцатеричной системе. Чтобы не возникало путаницы в определении системы счисления, перед шестнадцатеричным числом ставят символ решетки #, например #666999. Каждый из трех цветов — красный, зеленый и синий — может принимать значения от 00 до FF. Таким образом, обозначение цвета разбивается на три составляющие #rrggbb, где первые два символа отмечают красную компоненту цвета, два средних — зеленую, а два последних — синюю. Допускается использовать сокращенную форму вида #rgb, где каждый символ следует удваивать. Так, запись #fe0 следует расценивать как #ffee00. Регистр в данном случае значения не имеет, поэтому текст можно набирать как прописными, так и строчными символами.

3. *С помощью RGB.* Можно определить цвет, используя значения красной, зеленой и синей составляющей в десятичном исчислении. Количество каждого из трех цветов может принимать значения от 0 до 255. Также можно задавать цвет в процентном отношении, например `rgb(90%, 30%, 60%)`.

В листинге 1.3 приведены разные способы задания цвета текста.

Листинг 1.3. Изменение цвета символов

```
<html>
<body>
<p><span style="color: red">П</span>ервая буква в строке - красная.</p>
<p style="color: rgb(49, 151, 116)"><span style="color:
#fc0">Желтое</span> слово в строке зеленого цвета.</p>
</body>
</html>
```

В примере показано добавление стиля к тегу путем использования параметра `style`. В качестве его значения принимаются атрибуты CSS, перечисленные через точку с запятой, при этом их обязательно заключить в кавычки. Стиль, описанный прямо в теге, называется *внутренним* или *встроенным*.

Таким образом, цвет фона под текстовой строкой в примере устанавливается через внутренний стиль с помощью свойства `background-color` или `background`, а цвет текста — через свойство `color`. Так, с помощью сочетания атрибутов `color` и `background` можно создать выворотку, как показано в листинге 1.4.

Листинг 1.4. Создание выворотки

```
<html>
<head>
<style type="text/css">
 .inverse {
  font-family: Arial;         /* Шрифт Arial */
  font-weight: bold;          /* Жирное начертание */
  background: black;          /* Черный фон */
  color: white;               /* Символы белого цвета */
  padding: 3px                /* Поля между текстом и границей */
}
</style>
</head>
<body>
  <span class=inverse>Выворотка</span>
</body>
</html>
```

Выворотка лучше смотрится, когда текст набран рубленым и жирным шрифтом, поэтому в примере установлен шрифт Arial.

Размер символов

Размер текста может быть установлен несколькими способами. Набор констант (`xx-small`, `x-small`, `small`, `medium`, `large`, `x-large`, `xx-large`) задает размер, который называется абсолютным. По правде говоря, он не совсем абсолютный, поскольку зависит от настроек браузера и операционной системы.

Другой набор констант (`larger`, `smaller`) устанавливает относительные размеры шрифта. Поскольку размер унаследован от родительского элемента, эти относительные размеры применяются к родительскому элементу, чтобы определить размер шрифта текущего элемента.

Также разрешается использовать любые допустимые единицы CSS: `em` (высота шрифта элемента), `ex` (высота символа x), пункты (`pt`), пикселы (`px`), проценты (`%`) и др. За 100 % принимается размер шрифта родительского элемента.

В конечном итоге, размер шрифта сильно зависит от значения этого параметра у родительского элемента. На рис. 1.8 показано изменение размера текста в зависимости от использованной константы.

Размер текста удобнее всего задавать через свойство `font-size`, применяя его в стилях к различным элементам веб-страницы, как показано в листинге 1.5.

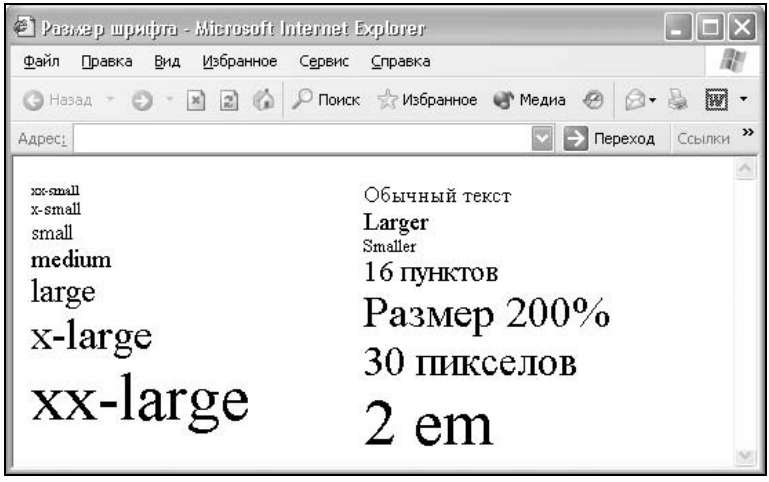


Рис. 1.8. Использование разных единиц для задания размера текста

Когда размер шрифта задается в пунктах или пикселах, то изменить эту величину с помощью опции браузера **Вид > Размер шрифта** нельзя. Если шрифт установлен слишком мелким, то исправить этот недостаток читателю простыми средствами не представляется возможным. Поэтому лучше использовать другие единицы размеров шрифта, например проценты.

Листинг 1.5. Изменение размера символов

```
<html>
<head>
<style type="text/css">
 BODY { font-size: 90% }      /* Размер шрифта для основного текста */
 TH { font-size: 80% }        /* Размер шрифта заголовка таблицы */
 H1 { font-size: 180% }       /* Размер шрифта заголовка первого уровня */
</style>
</head>
<body>
  ...
</body>
</html>
```

Некоторые замечания по поводу размера текста. При задании абсолютного размера шрифта параметром `small`, Internet Explorer 5 будет показывать шрифт таким же размером, что и шрифт без стиля, который имеет размер `medium`.

Задание размера шрифта font-size: medium приведет к различным размерам шрифта в браузерах Internet Explorer и Netscape, что противоречит спецификации CSS и вводит в заблуждение многих разработчиков.

Изменение насыщенности

Насыщенность шрифта управляется тегами B, STRONG или (более гибкий вариант) через свойство font-weight. Этот параметр определяет насыщенность шрифта с помощью ключевых слов: bold — полужирное, bolder — жирное, lighter — светлое, normal — нормальное начертание. Также допустимо использовать условные единицы от 100 до 900 с шагом 100. Сверхсветлое начертание шрифта, которое может отобразить браузер, имеет значение 100, а сверхжирное — 900. Нормальное начертание шрифта (которое установлено по умолчанию) эквивалентно значению 400, стандартный полужирный текст — 700.

Поскольку свойство font-weight предоставляет больше возможностей по управлению жирным начертанием текста, то можно переназначить свойства тега B через стили (листинг 1.6).

Листинг 1.6. Насыщенность шрифта

```
<html>
<head>
<style type="text/css">
 B {
  color: #000080;     /* Темно-синий цвет текста */
  font-weight: 900    /* Сверхжирная насыщенность */
}
</style>
</head>
<body>
  <b>Жирный текст</b><br>
  Обычный текст
</body>
</html>
```

В примере показано изменение параметров тега B, в частности текст при использовании этого тега будет темно-синего цвета и максимальной насыщенности.

Курсивный текст

Для создания курсивного текста применяются теги I и EM, а также атрибут font-style, который относится к свойствам CSS. Поскольку с тегами все понятно, рассмотрим изменение начертания через стили.

Атрибут font-style определяет начертание шрифта — normal (обычное), italic (курсивное) или oblique (наклонное), как показано в листинге 1.7. Когда для текста установлено курсивное или наклонное начертание, браузер обращается к системе для поиска подходящего шрифта. Если заданный шрифт не найден, браузер использует специальный алгоритм для имитации нужного вида текста. Результат и качество при этом могут получиться неудовлетворительными, особенно при печати документа.

Листинг 1.7. Курсивное и наклонное начертание

```
<html>
<head>
<style type="text/css">
 #cursive { font-style: italic }    /* Курсивный текст */
 #oblique { font-style: oblique }   /* Наклонный текст */
}
</style>
</head>
<body>
  <p id=cursive><span style="font-style: normal">Курсивный текст</span>
хорошо использовать для цитат и прямой речи</p>
  <p id=oblique>Наклонный шрифт пригодится для создания ненавязчивого
выделения в тексте</p>
</body>
</html>
```

Курсив и наклонный шрифт при всей их похожести — не одно и то же. Курсив — это специальный шрифт, имитирующий рукописный, наклонный же образуется путем наклона обычных знаков. Несмотря на то, что между указанными начертаниями существует различие, браузеры, как правило, отображают их одинаково.

Верхний и нижний индекс

Индексом по отношению к тексту называется смещение символов относительно базовой линии вверх или вниз. В зависимости от положительного или отрицательного значения смещения, индекс называется, соответственно, верхним или нижним. Они активно применяются в математике, физике, химии и для обозначения единиц измерения. HTML предлагает два тега для создания индекса: SUP — верхний индекс и SUB — нижний индекс. Текст, помещенный в один из этих контейнеров, обозначается меньшим размером, чем базовый текст, и смещается относительно горизонтали. В листинге 1.8 приведено совместное использование указанных тегов и стилей для изменения вида текста.

Листинг 1.8. Создание верхнего и нижнего индекса

```
<html>
<head>
<style type="text/css">
 SUP, SUB {
  font-style: italic;        /* Курсивное начертание */
  color: red                 /* Красный цвет символов */
}
</style>
</head>
<body>
<p>Характеристическое уравнение поверхности второй степени</p>
&lambda;<sup>3</sup> - I<sub>1</sub>&lambda;<sup>2</sup> +
I<sub>2</sub>&lambda; - I<sub>3</sub> = 0
</body>
</html>
```

В примере одновременно встречается как нижний, так и верхний индекс. Для изменения начертания шрифта индекса применяются стили, которые задают единое оформление, греческая буква лямбда воспроизводится через зарезервированное слово `&lambda;`.

Можно вообще отказаться от использования тегов `SUB` и `SUP` в пользу стилей. Аналогом этих тегов служит свойство `vertical-align`, заставляющее текст смещаться по вертикали на заданное расстояние. В частности, в листинге 1.9 в качестве аргумента применяется значение `0.8em`. `Em` — это относительная единица, равная высоте шрифта. Поскольку размер шрифта на веб-странице может меняться в самых широких пределах, то этой величиной удобно пользоваться, когда мы имеем дело именно с текстом. Например, `0.5em` говорит о том, что текст надо сдвинуть на половину размера шрифта. Подобный результат можно получить, если в качестве значений применять процентную запись вроде `vertical-align: 60%`.

Листинг 1.9. Использование стилей для управления индексами

```
<html>
<head>
<style type="text/css">
 .math { font-style: italic; font-size: 150% }
 .sup {
   vertical-align: 0.8em;          /* Сдвигаем текст вверх */
   color: red;                     /* Красный цвет верхнего индекса */
```

```
   font-size: 70%                    /* Уменьшаем размер шрифта */
}
 .sub {
   vertical-align: -0.8em;           /* Сдвигаем текст вниз */
   color: blue;                      /* Синий цвет нижнего индекса */
   font-size: 70%
}
</style>
</head>
<body>
Многочлен степени <i>n</i>
<p class=math>f(x) = a<span class=sub>0</span> + a<span
class=sub>1</span> x + ... + a<span class=sub>n-1</span> x<span
class=sup>n-1</span> + a<span class=sub>n</span> x<span
class=sup>n</span></p>
</body>
</html>
```

В примере сама формула выводится увеличенным размером, символы верхнего индекса устанавливаются красным цветом, а нижние — синим (рис. 1.9).

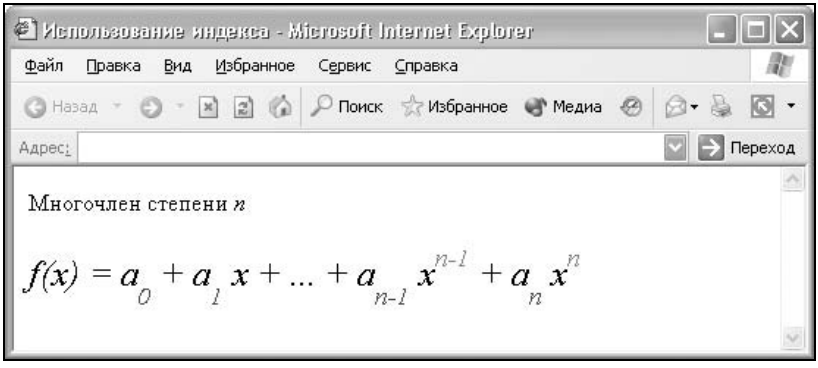


Рис. 1.9. Управление положением и видом нижнего и верхнего индекса

Браузер Internet Explorer до шестой версии не отображает результат правильно, если в качестве значений параметра `vertical-align` используется процентная запись или конкретные значения. Другими словами, приведенный пример в браузере Internet Explorer 5 работать не будет. Воспользовавшись ключевыми словами `sub` (нижний индекс) и `super` (верхний индекс), можно поправить дело, все браузеры их понимают корректно, но точно задавать положение индекса с помощью них нельзя. В этом смысле результат не отличается от использования тегов `SUB` и `SUP` (листинг 1.10).

Листинг 1.10. Создание индексов через стили

```
<html>
<head>
<style type="text/css">
 .sup { vertical-align: super; color: red; font-size: 70% }
 .sub { vertical-align: sub; color: blue; font-size: 70% }
</style>
<head>
<body>
...
</body>
</html>
```

Капитель

Капителью в типографике называется текст, набранный прописными буквами уменьшенного размера. Не стоит путать ее с обычными заглавными символами, капитель отображается и выглядит несколько по-иному. Чтобы не мучаться с изменением регистра символов и размером шрифта, стоит воспользоваться свойством `font-variant` со значением `small-caps`, как показано в листинге 1.11.

Листинг 1.11. Создание капители

```
<html>
<head>
<style type="text/css">
 H1.capitel { font-variant: small-caps; font-size: 200% }
</style>
</head>
<body>
<h1>Lorem ipsum dolor sit amet</h1>
<h1 class=capitel>Lorem ipsum dolor sit amet</h1>Lorem ipsum dolor sit
amet...
<body>
</html>
```

Как выглядит текст, набранный капителью, можно посмотреть на рис. 1.10.

Для сравнения на рисунке приведен обычный заголовок, заголовок, оформленный с помощью капители, и обычный текст. Обратите внимание, что первая буква предложения больше, чем остальной текст. Особенность капители в том, что заглавные и строчные буквы при ее использовании сохра-

няются. Браузер Internet Explorer до шестой версии отображает текст неправильно, заменяя все символы прописными. Остальные браузеры корректно преобразуют символы.

Рис. 1.10. Вид капительного текста

Прописные и строчные буквы

Обычно автор сам указывает, какие буквы будут прописными, а какие строчными, исходя из правил русского языка и собственных предпочтений. Тем не менее этот процесс можно автоматизировать, переложив работу по изменению регистра символов на стили. Для этого применяется свойство `text-transform`, которое может принимать следующие значения:

- `none` — текст пишется как есть, без каких-либо изменений;
- `capitalize` — каждое слово будет начинаться с заглавного символа;
- `lowercase` — все символы становятся строчными (нижний регистр);
- `uppercase` — все символы становятся прописными (верхний регистр).

Пример изменения регистра показан в листинге 1.12. Заголовок, определяемый тегом `H1`, полностью отображается заглавными символами, а каждое слово текста параграфа начинается с прописной буквы.

Листинг 1.12. Использование верхнего регистра символов

```
<html>
<head>
<style type="text/css">
 H1 { text-transform: uppercase }
 P { text-transform: capitalize }
```

```
</style>
</head>
<body>
<h1>Lorem ipsum dolor sit amet</h1>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat. Ut wisis enim ad minim veniam, quis nostrud exerci tution
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo
consequat.</p>
</body>
</html>
```

Результат примера показан на рис. 1.11. Обратите внимание, что независимо от того, какой регистр символов был задан в тексте первоначально, преобразование происходит в любом случае.

Рис. 1.11. Изменение регистра символов

Автоматическое изменение регистра символов удобно задавать для аббревиатур, названий тегов, заголовков и других элементов текста, где требуется писать прописными или строчными символами.

Работа с текстом

Поскольку текст является одним из главных способов предоставления информации посетителю сайта, следует уделить внимание его оформлению. Для ознакомления можно просматривать различные журналы и газеты, в которых дизайнеры вволю экспериментируют со шрифтами и текстами.

Все-таки веб-дизайн уходит своими корнями в полиграфический дизайн, и многое позаимствовано у "старшего брата". Стоит ли обращать внимание на дизайн журналов, пытаясь верстать сайт, ведь в этом случае мы имеем дело с разной средой? Безусловно стоит, определенные принципы работы с текстом сохраняются независимо от того, с чем мы имеем дело. Кроме того, использование CSS расширяет возможности при работе с текстом, и мы можем управлять его характеристиками практически как в издательских системах. Тем не менее доступен не весь арсенал средств, поэтому до сих пор практикуется метод написания текста в графическом редакторе с последующим добавлением его в виде изображения. Кроме того, существует и множество других приемов при работе со шрифтами, которые характерны только для сайтов.

Подчеркивание текста

Хотя подчеркивание текста лучше использовать только в ссылках, правильное его применение может придать эстетичный вид элементам веб-страницы.

Вначале разберемся, почему к подчеркиванию создалось такое отношение, чтобы не повторять чужих ошибок.

- Подчеркнутый текст читается хуже, чем обычный. С учетом того, что чтение с экрана монитора по сравнению с чтением с листа бумаги менее комфортно, данный фактор становится важным для повышения удобства восприятия информации.
- На веб-страницах подчеркивание ссылок стало определенным стандартом. Когда мы видим на сайте подчеркнутый текст, то воспринимаем его в силу привычки как ссылку. Если это не так, возникает раздражение: почему этот текст выглядит как ссылка, но при этом ссылкой не является?

Получается, что если включить подчеркивание, то оно должно быть либо декоративным, чтобы не возникала ассоциация со ссылками, либо применяться для небольших текстов, например заголовков. Но и в этом случае текст должен оставаться самим собой и не восприниматься как ссылка. Рассмотрим разные варианты создания подзаголовочных линий без применения тега `U`, который устанавливает подчеркивание текста.

Использование тега *HR*

Тег `HR` создает горизонтальную линию определенной толщины и ширины, перед ней всегда автоматически добавляется перенос строки. Чтобы заставить линию плотнее прижиматься к тексту, тег `HR` можно поместить прямо внутрь тега `H1` или `H2`, определяющего заголовок (листинг 1.13).

Листинг 1.13. Создание заголовка с линией

```
<html>
<body>
<H2>Заголовок с линией (первый способ)</H2>
<HR noshade size=2>

<H2>Заголовок с линией (второй способ)<HR noshade size=2></H2>
</body>
</html>
```

Отличие между приведенными способами размещения линии показано на рис. 1.12.

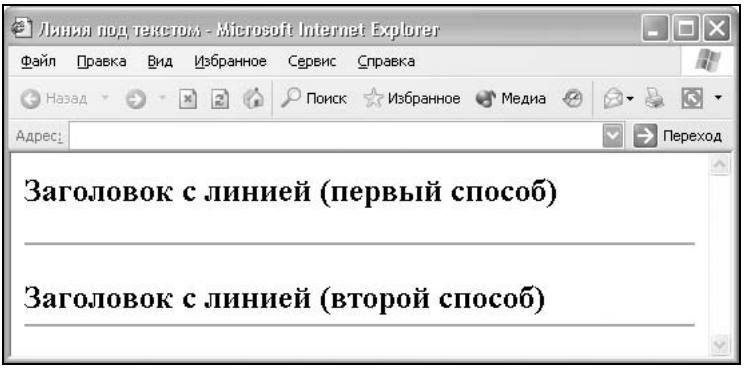


Рис. 1.12. Линия под текстом, созданная с помощью тега `HR`

Такие линии, как показано в примере, имеют фиксированную длину. Если она явно не указана, длина линии становится равной ширине окна браузера или родительского элемента, например таблицы. Когда захочется создать подчеркивание строго под текстом, тег `HR` уже не поможет, и придется переходить к таблицам или стилям.

Применение таблиц

С помощью таблиц легко создавать разные линии, заливая нужным цветом ячейку фиксированной высоты. Тег `H1` и подобные ему лучше не применять, т. к. при их использовании получается перенос строки и ненужный отступ. В листинге 1.14 размер шрифта варьируется стилем, который применяется к тегу `SPAN`.

Листинг 1.14. Создание подчеркивания с помощью таблицы

```
<html>
<body>
```

```
 <table border=0 cellpadding=0 cellspacing=0>
  <tr>
   <td>
    <span style="font-size: 220%">Видео не для всех</span>
   </td>
 </tr>
  <tr>
   <td height=4 bgcolor=red></td>
  </tr>
 </table>
</body>
</html>
```

Расстоянием от линии до текста можно управлять с помощью параметров таблицы `cellpadding` и `cellspacing` или задавая настройки стиля. Цвет линии определяется параметром `bgcolor`. Обратите внимание, что ширина таблицы специально не указывается, в этом случае она автоматически подстраивается под содержащийся в ней текст.

Использование стилей

Пожалуй, один из самых удобных способов создания линий под текстом — это воспользоваться стилями. Для этого применимо свойство CSS `border-bottom`. Данный атрибут сразу задает стиль линии под текстом, ее толщину и цвет (листинг 1.15).

Листинг 1.15. Создание стиля заголовка с подчеркиванием

```
<html>
<head>
<style type="text/css">
 H1 { border-bottom: 2px solid red; font-size: 220% }
</style>
</head>
<body>
  <h1>Часть 4, в которой Пол и Пропилен наносят ответный удар</h1>
</body>
</html>
```

Поскольку теги `H1`, ..., `H6` являются блочными, то линия под ними получается на всю ширину окна браузера. *Блочным* называется элемент, заключенный в условный прямоугольник, после которого всегда автоматически добавляется перенос строки. Обычно блочные элементы занимают всю отве-

денную им свободную ширину, хотя это и не всегда обязательно так. Антагонистами блочных элементов выступают *встроенные* элементы, они помещаются в любое место веб-страницы, например строку текста, занимают заданную ширину, и после них переносов нет.

Таким образом, чтобы создать линию на ширину текста, стиль надо установить для встроенного элемента `SPAN`, который располагается внутри контейнера `H1`, как показано в листинге 1.16.

Листинг 1.16. Стиль для линии на ширину текста

```
<html>
<head>
<style type="text/css">
 H1 SPAN { border-bottom: 4px solid red }
</style>
</head>
<body>
 <h1><span>Как поймать льва</span></h1>
</body>
</html>
```

Результат создания подобной линии показан на рис. 1.13.

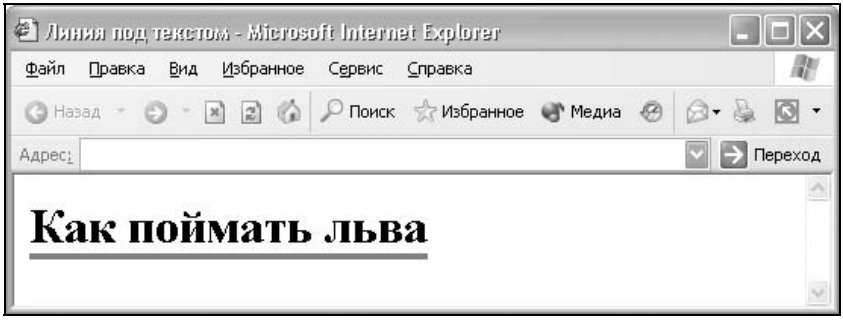


Рис. 1.13. Линия под текстом, созданная с помощью стилей

Замечание

Браузер Internet Explorer 5 содержит ошибку и не показывает результат использования стилей для встроенных элементов вроде `SPAN`. В этом браузере приведенный пример работать должным образом не будет. Начиная с версии 5.5, эта ошибка была устранена.

Создание буквицы

Буквица является художественным приемом оформления текста и представляет собой увеличенную первую букву, базовая линия которой находится на одну или несколько строк ниже базовой линии основного текста. Буквица привлекает внимание читателя к началу текста, особенно если страница лишена других ярких деталей. Обычно кроме самого символа используются изображения растений, животных и других объектов. Это, конечно, не обязательно, но может придать определенный настрой содержанию. Если хочется применить на сайте этот эффект, лучше всего для этого подойдет рисунок, выровненный по левому краю (листинг 1.17).

Листинг 1.17. Создание буквицы с помощью рисунка

```
<html>
<body>
<img src=n.gif width=50 height=50 align=left vspace=2 hspace=2>еобходимо,
манипулируя полученными предметами материального мира и фрагментами
информационного поля, эмпирическим путем достигнуть логического
завершения конкурса.
</body>
</html>
```

Отступ от буквицы до текста задается параметрами `vspace` — по вертикали и `hspace` — по горизонтали. Результат примера показан на рис. 1.14.

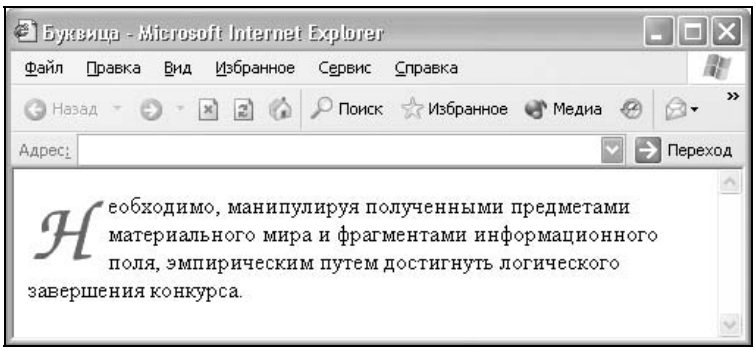


Рис. 1.14. Рисунок в качестве буквицы

Достоинства рисунков в качестве буквицы заключаются в следующем: применение любого шрифта и разных эффектов, простота метода, а также возможность использования в качестве формата анимированный GIF-файл, что дает дополнительные возможности оформления. Но есть и недостатки: если буквица на сайте применяется довольно часто, то нужно подготовить

множество рисунков разных букв; также усложняется возможность редактирования текста, т. к. придется вместо простого изменения буквы вставлять новый рисунок.

Буквица в тексте

Можно создать буквицу не в виде рисунка, а в качестве простого текста, увеличенного по сравнению с базовым шрифтом. Для того чтобы текст огибал первую букву, ее необходимо поместить в ячейку таблицы, выровненную по левому краю, как показано в листинге 1.18.

Листинг 1.18. Применение таблицы для создания буквицы

```
<html>
<body>
<head>
<style type="text/css">
 .letter { font-family: Arial; font-size: 24pt; color: navy }
</style>
</head>
<body>
<table align=left border=0 cellspacing=0 celpadding=0>
  <tr><td><span class=letter>П</span></td></tr>
</table>
оутру Паша проснулся после порядочного похмелья.
</body>
</html>
```

От таблиц можно легко отказаться в пользу использования стилей. В данном случае потребуется только поместить начальную букву текста в контейнер `SPAN` и для него задать свойство `float:left`, как показано в листинге 1.19.

Листинг 1.19. Использование стилей для создания буквицы

```
<html>
<head>
<style type="text/css">
 .letter {
  font-size: 200%;           /* Размер шрифта буквицы */
  float: left;               /* Выравнивание по левому краю */
  color: red;                /* Цвет буквицы */
  padding: 3px               /* Отступ между буквицей и текстом */
}
```

```
</style>
</head>
<body>
<span class=letter>Н</span>еобходимо, манипулируя полученными предметами материального мира и фрагментами информационного поля, эмпирическим путем достигнуть логического завершения конкурса.
</body>
</html>
```

С помощью стилей также удобно сразу задать цвет буквицы и необходимые отступы между символами.

Буквица на поле

Ничто не мешает расположить буквицу слева, четко отделив ее от основного текста, как показано на рис. 1.15. Таким образом, можно добиться простого выразительного эффекта.

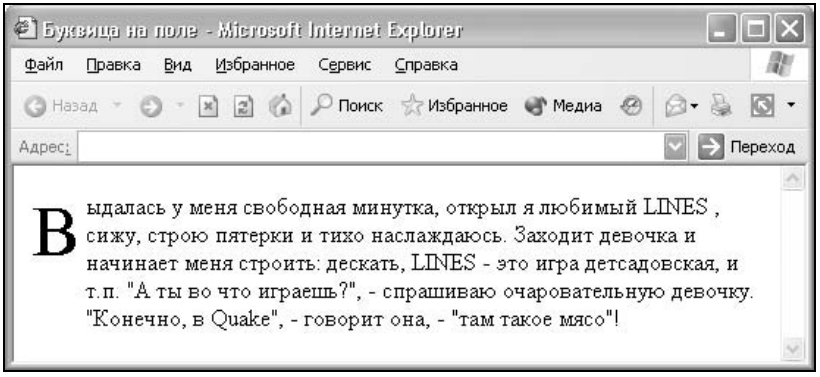


Рис. 1.15. Расположение буквицы на поле

Для размещения буквицы таким образом необходимо создать таблицу с двумя ячейками, в левой расположить первый символ, а в правой — остальной текст (листинг 1.20).

Листинг 1.20. Таблица для создания буквицы на поле

```
<html>
<head>
<style type="text/css">
 .letter { font-family: Arial; font-size: 24pt; color: navy }
</style>
</head>
```

```
<body>
<table width=100% border=0>
 <tr>
  <td valign=top><span class=letter>B</span></td>
  <td valign=top>
ыдалась у меня свободная минутка, открыл я любимый LINES, сижу, строю
пятерки и тихо наслаждаюсь. Заходит девочка и начинает меня строить:
дескать, LINES - это игра детсадовская, и т. п. "А ты во что играешь?", -
спрашиваю очаровательную девочку. "Конечно, в Quake", - говорит она, -
"там такое мясо"!
  </td>
 </tr>
</table>
</body>
</html>
```

Обратите внимание, содержимое обеих ячеек выравнивается по верхнему краю параметром `valign=top`. Расстояние между буквицей и текстом можно изменять с помощью атрибутов `cellpadding` и `cellspacing`.

Выступающий инициал

Аналогом буквицы в некотором роде можно считать *выступающий инициал* — увеличенную прописную букву, базовая линия которой совпадает с базовой линией основного текста. Допустимо воспользоваться способами, которые предназначены для создания буквицы, но в данном случае ситуация упрощается, поскольку не требуется, чтобы основной текст обтекал первую букву. Поэтому лучше сразу применить стили, тем более что существует псевдокласс `first-letter`, который определяет вид первого символа текста.

Псевдоклассы используются в CSS для того, чтобы применять к элементам стилей разные процедуры форматирования. В частности псевдокласс `first-letter`, добавленный, как показано в листинге 1.21, к параграфу (тег `P`), позволяет получить универсальный и компактный код.

Листинг 1.21. Использования псевдокласса `first-letter`

```
<html>
<head>
<style type="text/css">
 P:first-letter {
  color: red;
  font-size: 200%
}
```

```
</style>
</head>
<body>
<p>При выходе длины ногтей за нормы регламентируемые ГОСТом во избежание риска поцарапать поверхность дорогостоящей техники, оператор допускается к работе на компьютере только в верхонках.</p>
</body>
</html>
```

Результат выполнения примера приведен на рис. 1.16.

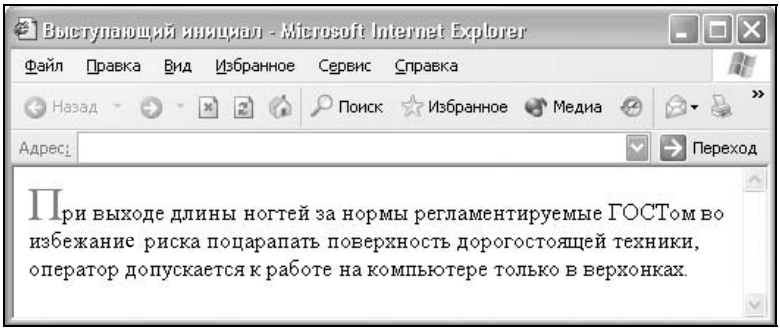


Рис. 1.16. Текст с использованием выступающего инициала

Отступ первой строки

Использование красной строки с отступом в 2—5 пробелов повышает читаемость текста, позволяя легко отыскивать взглядом начало следующего абзаца. В HTML любое количество пробелов заменяется одним, в этом случае стоит использовать специальный символ — ` `, который называется *неразделяемым пробелом* (листинг 1.22).

Листинг 1.22. Создание отступа с помощью неразделяемого пробела

```
<html>
<body>
      Сварить на медленном огне воду. Положить в нее небольшого червяка, лучше живого, мелко покрошенное яблоко, крылышко мотылька и жабу. Соль и сахар по вкусу. По готовности перелить в бутылки. Хранить в прохладном месте.
</body>
</html>
```

Следующий способ также имеет право на существование, хотя применяется весьма редко. Вместо символов пробела надо поставить невидимый рисунок

нужной ширины (листинг 1.23). Учтите, что при использовании в качестве отступа невидимого рисунка есть опасность, что пользователь отключил загрузку изображений, тогда вместо отступа будет показываться некрасивая полоса.

Листинг 1.23. Создание отступа невидимым рисунком

```
<html>
<body>
<img src=empty.gif width=20 height=1>Сварить на медленном огне воду.
Добавить в нее хлива и хрольва. Довести до кипения.
</body>
</html>
```

В качестве рисунка можно использовать прозрачный GIF-файл размером 1 × 1 пиксел.

Наконец, всегда можно воспользоваться стилями. Параметр `text-indent` задает отступ первой строки текста (листинг 1.24). Отступ можно указывать в пунктах (`pt`), пикселах (`px`), дюймах (`in`), миллиметрах (`mm`) и др.

Листинг 1.24. Создание отступа с помощью параметра `text-indent`

```
<html>
<head>
<style type="text/css">
 P { text-indent: 20px }
</style>
</head>
<body>
<p>Смешать 2 части соляной кислоты и 1 часть азотной со льдом. Слить
охлажденную смесь в фужер. Пить залпом.</p>
</body>
</html>
```

Конечно, использование стилей более универсально, вдобавок оно менее обременительно. Кроме того, стили позволяют создать не только отступ, но и выступ, при котором первая строка выдвигается влево, а остальной текст остается на месте. Для этого требуется только установить отрицательное значение у параметра `text-indent`, как показано в листинге 1.25.

Листинг 1.25. Создание выступа в тексте

```
<html>
<head>
```

```
<style type="text/css">
 P {
  text-indent: -20px;        /* Выступ первой строки */
  padding-left: 20px         /* Отступ всего текста влево */
}
</style>
</head>
<body>
<p>Грог по рыбацки<br>
1 рыбу залить кипятком, через 5 минут процедить и добавить грога.
Подавать в чашках.</p>
</body>
</html>
```

Отступ текста с помощью параметра `padding-left` необходим для того, чтобы выступ первой строки не уходил далеко за пределы текста. Иначе может получиться, что текст будет начинаться левее окна браузера.

Пробелы между словами

Особенность языка HTML такова, что любое количество пробелов между словами будет отображаться как один. Но иногда у разработчика сайта возникает необходимость изменить расстояние между словами текста, в частности увеличить его. Чтобы задать расстояние между словами в тексте, используется параметр `word-spacing` (листинг 1.26).

Листинг 1.26. Изменение пробелов с помощью атрибута `word-spacing`

```
<html>
<html>
<head>
<style type="text/css">
 P { word-spacing: 10px }
</style>
</head>
<body>
 <p>Слон + хорошая пища = два слона</p>
</body>
</html>
```

Преимущество указанного подхода состоит в его удобстве и централизованности. Стиль можно описать один раз в глобальной таблице стилей и применять его к нужным элементам веб-страницы.

Замечание

Если в тексте установлен параметр выравнивания `justify`, то атрибут `word-spacing` не действует, поскольку интервал между словами будет установлен принудительно, чтобы строка текста была выровнена по правому и левому краю.

Выравнивание текста

Выравниванием, или на типографский манер, *выключкой*, называется размещение левого или правого края блока текста вдоль невидимой вертикальной линии. Различают четыре типа выравнивания.

- *По левому краю.* В этом случае строки текста выравниваются по левому краю, а правый край располагается "лесенкой". Такой способ выравнивания является наиболее популярным на сайтах, поскольку он позволяет пользователю легко отыскивать взглядом новую строку и комфортно читать большой текст. Также выравнивание по левому краю является включенным по умолчанию.
- *По правому краю.* Этот способ выравнивания выступает в роли антагониста предыдущему типу: строки текста равняются по правому краю, а левый остается "рваным". Из-за того, что левый край не выровнен, а именно с него начинается чтение новых строк, такой текст читать труднее, чем если бы он был выровнен по левому краю. Поэтому выравнивание по правому краю применяется обычно для коротких заголовков объемом не более трех строк. Мы не рассматриваем специфичные сайты, где текст приходится читать справа налево, там, возможно, подобный способ выравнивания и пригодится. Но где вы у нас в стране видели такие сайты?
- *По центру.* Текст помещается по центру горизонтали окна браузера или контейнера, где расположен текстовый блок. Строки текста словно нанизываются на невидимую ось, которая проходит по центру веб-страницы. Подобный способ выравнивания активно используется в заголовках и различных подписях, вроде подрисуночных, он придает официальный и солидный вид оформлению текста. Во всех других случаях выравнивание по центру применяется редко, опять же по той причине, что читать большой объем такого текста неудобно.
- *По ширине.* Здесь выравнивание происходит одновременно по левому и правому краю. При этом данное выравнивание таит в себе некоторые особенности. Известно, что HTML не умеет работать с переносами текста, как это реализовано в текстовых редакторах. Чтобы обеспечить выравнивание блока текста, браузер добавляет пустое пространство между словами, удлиняя таким образом текстовые строки. Из-за этого в некоторых местах появляются некрасивые "дыры", которые портят впечатление и снижают читабельность текста.

Таким образом, для заголовков лучше воспользоваться выравниванием по центру, по левому или правому краю. Различные подрисуночные подписи и заголовки таблиц обычно выравнивают по центру, а основной текст — по левому краю или, что реже, по ширине.

Выравнивание текста на веб-странице задается двумя основными способами. Первый заключается в использовании параметра `align`, он применяется обычно к тегу параграфа `P`. У этого параметра может быть четыре значения:

- `left` — выравнивание текста по левому краю; это значение задано по умолчанию, поэтому его требуется указывать, когда в родительском элементе используется выравнивание другого типа;
- `center` — выравнивание по центру;
- `right` — выравнивание по правому краю;
- `justify` — выравнивание по ширине.

В листинге 1.27 приведен код для создания выравнивания таким методом.

Листинг 1.27. Выключка текста с помощью параметра `align`

```
<html>
<body>
<h2>Выравнивание по левому краю</h2>
<p>Строки прижимаются к левому краю окна и образуют неровный правый
край.</p>
<h2>Выравнивание по правому краю</h2>
<p align=right>Строки текста выровнены по правому краю, а левый край
получается "рваный". Поскольку мы привыкли читать текст слева направо, то
чтение затрудняется, когда левая сторона текста плохо определена.</p>
<h2>Выравнивание по центру</h2>
<p align=center>Каждая строка текста выравнивается по центру, и оба края
оказываются неровными. </p>
<h2>Выравнивание по ширине</h2>
<p align=justify>Строки текста выровнены по левой и правой стороне окна
браузера одновременно. Чем у́же область вывода текста, тем больше
вероятность появления пустого пространства внутри строк. Эта же
особенность проявляется и при использовании длинных слов.</p>
</body>
</html>
```

Посмотреть, как будет выглядеть каждый тип выравнивания, можно на рис. 1.17.

Когда требуется сделать однотипным выравнивание большого числа блоков текста, нет нужды каждый раз писать параметр `align`. Второй способ подразумевает использование стилей и атрибута `text-align`, в частности. Допустимые значения у него те же самые, что и у `align` (листинг 1.28).

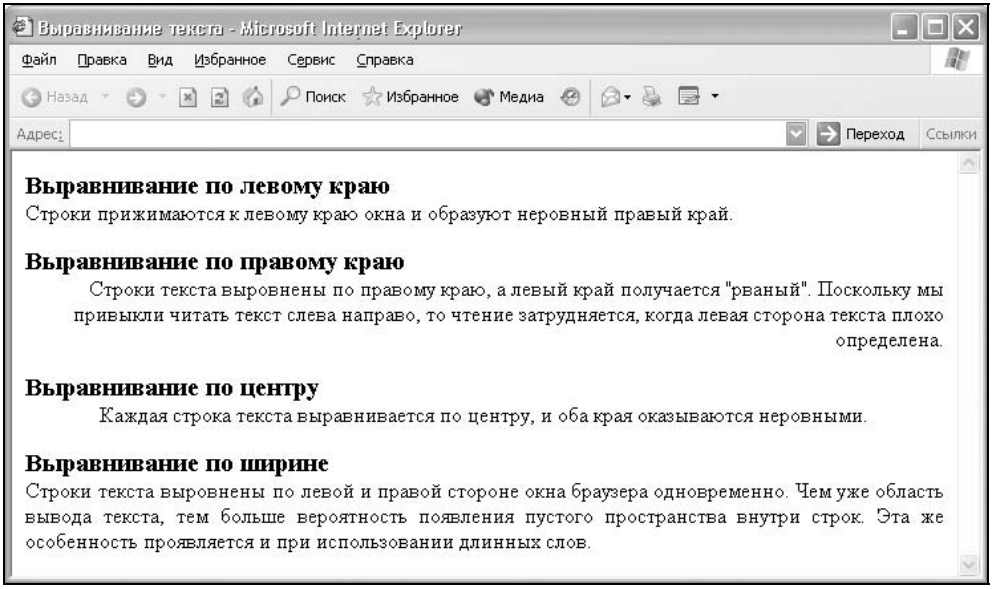


Рис. 1.17. Различное выравнивание параграфов текста

Листинг 1.28. Выключка текста с помощью свойства `text-align`

```
<html>
<head>
<style type="text/css">
 P { text-align: justify }
</style>
</head>
<body>
 <p> ... </p>
</body>
</html>
```

В данном примере для всех параграфов текущей страницы задано выравнивание текста по ширине.

Отбивка строк

Чтобы визуально отделить один параграф от другого, после каждого блочного тега `P` браузер добавляет пустой интервал, который называется *отбивкой*. Такого рода промежутки разделяют однородные прямоугольники текста, позволяя читателю легче отыскивать взглядом новые абзацы. С позиции

дизайна в этом тоже есть некоторые плюсы — подобное пустое пространство улучшает вид текста и документа в целом.

Но нельзя отдавать слишком многое на волю браузера, иногда хочется и поменять величину отбивки, например, вообще ее убрать или установить разный интервал до и после параграфа. Здесь нам на помощь опять придут стили, в частности воспользуемся свойствами, регулирующими отступ сверху — `margin-top` и снизу — `margin-bottom`. Так, если требуется вообще убрать отступы, следует установить значения этих параметров, равное нулю, как показано в листинге 1.29.

Листинг 1.29. Убираем интервалы у параграфа

```
<html>
<head>
<style type="text/css">
 P {
  margin-bottom: 0px;        /* Отступ снизу */
  margin-top: 0px            /* Отступ сверху */
}
</style>
</head>
<body>
 <p> ... </p>
</body>
</html>
```

Использование отбивки пригодится и для управления видом заголовков, создаваемых с помощью тегов `H1`, `H2`, `H3` и т. д. Чтобы связать заголовок с текстом под ним, интервал до заголовка должен быть больше, а после заголовка — меньше. Сравните два варианта одного и того же текста, приведенные на рис. 1.18.

В левой колонке на рис. 1.18 показано положение заголовков и абзацев, как оно отображается по умолчанию. Названия разделов находятся на одинаковом расстоянии от текста сверху и снизу, из-за этого не прослеживается четкая связь между заголовком и текстом под ним. Тот же вариант в правой колонке за счет изменения отбивки выглядит более выигрышно и наглядно. В данном случае перед заголовком раздела добавлен интервал, который отделяет его от предыдущего абзаца, а сам заголовок сближен с текстом под ним. В таком виде организация текста более четкая и понятная.

Код для изменения интервалов в абзаце приведен в листинге 1.30.

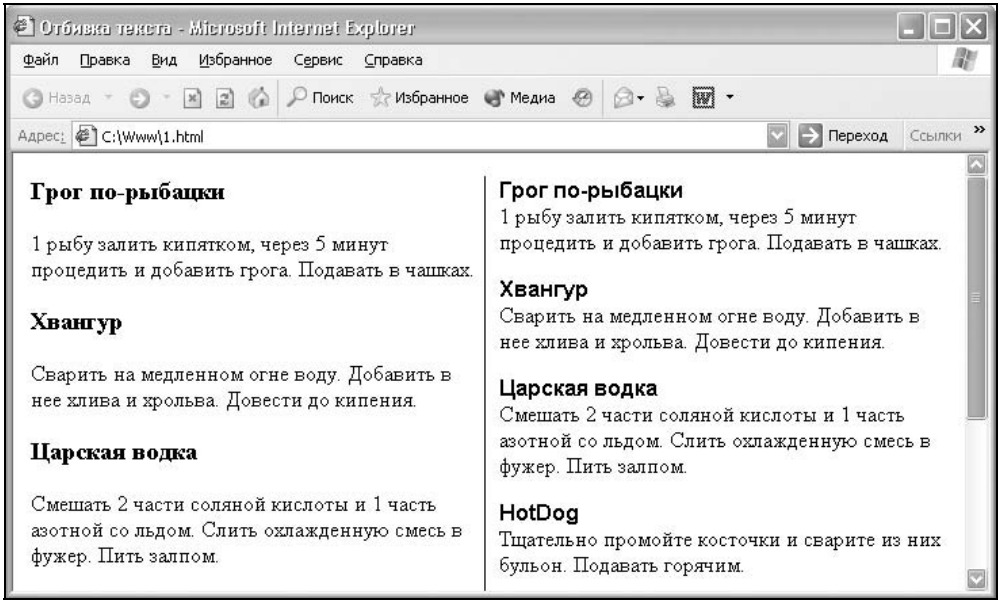


Рис. 1.18. Применение отбивки в заголовках текста

Листинг 1.30. Использование отбивки в абзацах

```
<html>
<head>
<style type="text/css">
 H3 {
  margin-bottom: 0.1em;            /* Расстояние между заголовком и
                                   текстом под ним */
  margin-top: 0px;                  /* Отступа сверху нет */
  font-family: Arial, sans-serif;   /* Рубленый шрифт для заголовков
                            смотрится лучше, чем шрифт с засечками */
  font-size: 100%                   /* Размер шрифта */
}
 P {
  margin-bottom: 1em;               /* Отступ после параграфа */
  margin-top: 0px                   /* Отступа сверху нет */
}
</style>
</head>
<body>
  <h3>HotDog</h3>
```

```
  <p>Тщательно промойте косточки и сварите из них бульон. Подавать
горячим.</p>
</body>
</html>
```

Для тега `H3` значения отступов обнуляем, чтобы не получилось лишнего сдвига текста перед заголовком. По желанию можно задать небольшой интервал между абзацем и названием к нему, так, в примере он установлен в `0.1em`. Нужный отступ получается добавлением параметра `margin-bottom` к селектору `P`. В примере используется значение `1em`, равное высоте строки текста.

Интерлиньяж (межстрочное расстояние)

Интерлиньяжем называется расстояние между базовыми линиями близких друг к другу строк. При обычных обстоятельствах расстояние между строками зависит от вида и размера шрифта и определяется браузером автоматически. Но это значение может быть изменено с помощью свойства CSS `line-height`. Заданное по умолчанию значение этого параметра `normal` заставляет браузер вычислять расстояние между строк автоматически. Любое число больше нуля воспринимается как множитель от размера шрифта текущего текста. Например, значение 1,5 устанавливает полуторный межстрочный интервал. В качестве значений допустимо использовать также любые единицы длины, принятые в CSS, — пикселы (`px`), дюймы (`in`), пункты (`pt`) и др. Разрешается использовать процентную запись, в этом случае за 100 % принимается высота шрифта. Отрицательное значение межстрочного расстояния не допускается.

На рис. 1.19 приведен один и тот же текст с разным интервалом.

Как изменить интерлиньяж текста, показано в листинге 1.31.

Листинг 1.31. Установка полуторного межстрочного интервала

```
<html>
<body>
<div style="line-height: 1.5">
<b>Полуторный интервал</b><br>
И вот однажды пришел добрый волшебник по имени Нортон, посмотрел он на
страдания людей, покачал головой и сделал специальную колотушку, с
помощью которой любой человек мог получить плоды с самого высокого
дерева.
</div>
</body>
</html>
```

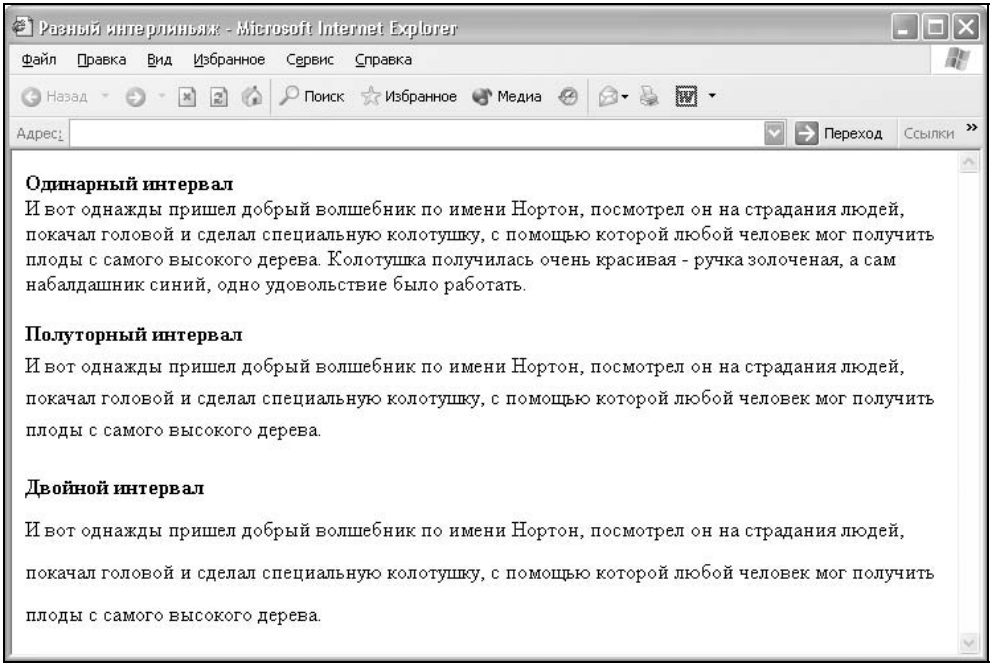


Рис. 1.19. Вид текста при разном интерлиньяже

Интерлиньяж можно применять к параграфам, слоям и другим блочным элементам, как показано в примере.

Межбуквенный интервал

Браузер автоматически подбирает интервалы между символами исходя из размера и типа шрифта. В некоторых случаях установка по умолчанию не является оптимальной, и требуется подкорректировать расстояние между буквами для облегчения чтения текста. Точно так же можно улучшить расположение слов, переместив их ближе или дальше друг от друга с помощью *межсловных интервалов*.

Расстояние между словами текста изменяется с помощью стилевого атрибута `word-spacing`, который был рассмотрен в листинге 1.26. Что касается межбуквенного интервала, то для управления им существует параметр `letter-spacing`. В качестве значений применяются любые единицы CSS, но лучший результат дает использование относительных единиц, основанных на размере шрифта (`em` и `ex`). На рис. 1.20 показан вид текста при разных значениях параметра `letter-spacing`.

Термины *жидкий* и *плотный интервал* пришли из типографики, где коррекция межбуквенных интервалов называется *трекингом*. Как правило, изда-

тельские системы предлагают несколько ступеней для изменения трекинга, но при использовании текста на веб-странице мы лишены этой возможности. Приходится подбирать межбуквенный интервал "вручную" исходя из параметров текста. В листинге 1.32 приведено несколько вариантов параметра `letter-spacing` для текста параграфа с разными значениями и единицами измерения.

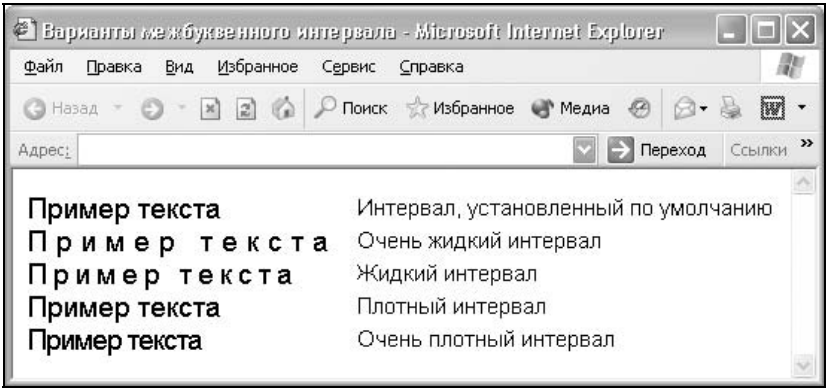


Рис. 1.20. Влияние межбуквенного интервала на отображение текста

Листинг 1.32. Изменение межбуквенного интервала в тексте

```
<html>
<body>
 <p>Интервал, установленный по умолчанию</p>
 <p style="letter-spacing: 0.3em">Очень жидкий интервал</p></td>
 <p style="letter-spacing: 0.2em">Жидкий интервал</p></td>
 <p style="letter-spacing: 0px">Плотный интервал</p>
 <p style="letter-spacing: -1px">Очень плотный интервал</p>
</body>
</html>
```

Обратите внимание, что параметр `letter-spacing` допускает использование отрицательного значения, но в этом случае надо убедиться, что сохраняется читабельность текста.

ГЛАВА 2

Изображения

Изображения придают сайту уникальный вид, делают его более выразительным и привлекательным, а разработчику сайта позволяют расширить арсенал средств для модификации и управления дизайном. Определенную информацию вроде графиков и схем проще, нагляднее и понятнее передать через рисунки, чем долго описывать, что к чему. Так что правильное использование картинок на сайте только улучшит его.

Добавление изображений

Для встраивания изображения в документ используется тег `IMG`, имеющий единственный обязательный параметр `src`, который определяет адрес файла с картинкой. Если необходимо, то рисунок можно сделать ссылкой на другой файл, поместив тег `IMG` в контейнер `A`, при этом вокруг изображения отображается рамка.

Рисунки также могут применяться в качестве карт-изображений, когда картинка содержит активные области, выполняющие функцию ссылки. Такая карта по внешнему виду ничем не отличается от обычного изображения, но при этом оно может быть разбито на невидимые зоны разной формы, где каждая из областей служит ссылкой.

Далее описаны параметры тега `IMG`, отвечающие за отображение изображений и их характеристики.

- `ALIGN`. Управляет положением изображений относительно текста или других объектов на веб-странице.
- `ALT`. Устанавливает альтернативный текст для изображений. Такой текст позволяет получить текстовую информацию о рисунке при отключенной в браузере загрузке изображений. Поскольку загрузка изображения происходит после получения браузером информации о нем, то замещающий

рисунок текст появляется раньше. А уже по мере загрузки текст будет сменяться изображением. Браузеры также отображают альтернативный текст в виде подсказки, появляющейся при наведении курсора мыши на изображение. В качестве аргумента параметра `alt` служит любая подходящая текстовая строка. Ее обязательно надо брать в двойные или одинарные кавычки.

- `BORDER`. Изображение, используемое на веб-странице, можно поместить в рамку различной ширины. Для этого служит параметр `border`. По умолчанию рамка вокруг изображения не отображается за исключением случая, когда рисунок является ссылкой. При этом рамка добавляется автоматически, толщина ее составляет один пиксел, а цвет рамки совпадает с цветом ссылки. Чтобы убрать рамку, следует задать параметр `border=0`.
- `HEIGHT` и `WIDTH`. Для изменения размеров изображения средствами HTML предусмотрены параметры `height` и `width`. Допускается использовать значения в пикселах или процентах. Если установлена процентная запись, то размеры изображения вычисляются относительно родительского элемента — контейнера, где находится тег `IMG`. В случае отсутствия родительского контейнера, его роль играет окно браузера. Иными словами, `width=100%` означает, что рисунок будет растянут на всю ширину веб-страницы. Добавление только одного параметра `width` или `height` сохраняет пропорции и отношение сторон изображения. Браузер при этом ожидает полной загрузки рисунка, чтобы определить его первоначальную высоту и ширину.

Обязательно задавайте размеры всех изображений на веб-странице. Это несколько ускоряет загрузку страницы, поскольку браузеру нет нужды вычислять размер каждого рисунка после его получения. Это утверждение особенно важно для изображений, которые располагаются внутри таблиц.

Ширину и высоту изображения можно менять как в меньшую, так и в большую сторону. Однако на скорость загрузки рисунка это никак не влияет, поскольку размер файла остается неизменным. Поэтому с осторожностью уменьшайте изображение, т. к. это может вызвать недоумение у читателей, отчего такой маленький рисунок так долго загружается. А вот увеличение размеров приводит к обратному эффекту — размер изображения велик, но файл относительно изображения аналогичного размера загружается быстрее, хотя качество рисунка при этом ухудшается.

- `HSPACE` и `VSPACE`. Для любого изображения можно задать невидимые отступы по горизонтали и вертикали с помощью параметров `hspace` и `vspace`. Особенно это актуально при обтекании рисунка текстом. В этом случае, чтобы текст не "наезжал" плотно на изображение, необходимо вокруг него добавить пустое пространство.
- `ISMAP` Параметр `ismap` сигнализирует браузеру о том, что рисунок является серверной картой-изображением. Карты-изображения позволяют при-

вязывать ссылки к разным областям одного изображения. Параметр реализуется в двух различных вариантах — серверном и клиентском. В случае применения серверного варианта браузер посылает запрос на сервер для получения адреса выбранной ссылки и ждет ответа с нужной информацией. Такой подход требует дополнительного времени на ожидание результата и отдельные файлы для каждой карты-изображения. Отправка данных на сервер происходит следующим образом. Необходимо поместить тег `IMG` в контейнер `A`, где в качестве значения параметра `href` указать адрес CGI-программы. Программа анализирует полученные координаты нажатия мыши, которые считаются от левого верхнего угла изображения, и возвращает требуемую веб-страницу. Например, если пользователь установил координаты мыши на изображении 100, 50, то после нажатия на ссылку будет открыт файл по адресу **http://www.htmlbook.ru/cgi-bin/map.cgi?100,50**. Последние цифры передаются в CGI-программу по методу GET и интерпретируются на сервере

- `LOWSRC`. Параметр `lowsrc` используется для отображения рисунка низкого качества перед полной загрузкой конечного изображения. Такой подход предназначен для снижения времени ожидания пользователя. Предварительно ему демонстрируется изображение в низком разрешении или черно-белый рисунок, который быстро загружается из-за небольшого исходного объема файла. Пока пользователь рассматривает его, происходит загрузка изображения хорошего качества, которое постепенно сменяет первоначальное. Размеры этих изображений обязательно должны совпадать.
- `SRC`. Адрес графического файла, который будет отображаться на веб-странице. В качестве значения принимается полный или относительный путь к файлу.

Форматы графических файлов

Широкое распространение для веб-графики получили два формата — GIF и JPEG. Их многофункциональность, универсальность, небольшой объем исходных файлов при достаточном для сайта качестве сослужили им хорошую службу, фактически определив их как стандарт веб-изображений. Есть еще формат PNG, который также поддерживается браузерами при добавлении изображений и существует в двух ипостасях — PNG-8 и PNG-24. Однако популярность PNG сильно уступает форматам GIF и JPEG.

GIF

GIF (Graphics Interchange Format, формат обмена графическими данными) — формат графических файлов, широко применяемый при создании

сайтов. GIF использует 8-битовый цвет и эффективно сжимает сплошные цветные области, при этом сохраняя детали изображения.

Особенности

- Количество цветов в изображении может быть от 2 до 256, но это могут быть любые цвета из 24-битной палитры.
- Файл в формате GIF может содержать прозрачные участки. Если используется отличный от белого цвета фон, он будет проглядывать сквозь "дыры" в изображении.
- Формат поддерживает покадровую смену изображений, что делает его популярным для создания баннеров и простой анимации.
- Использует свободный от потерь метод сжатия.

Область применения

Текст, логотипы, иллюстрации с четкими краями, анимированные рисунки, изображения с прозрачными участками, баннеры.

JPEG

JPEG (Joint Photographic Experts Group, объединенная группа экспертов в области фотографии) — популярный формат графических файлов, широко применяемый при создании сайтов и для хранения изображений. JPEG поддерживает 24-битный цвет и сохраняет неизменными яркость и оттенки цветов в фотографиях. Данный формат называют сжатием с потерями, поскольку алгоритм JPEG выборочно отвергает данные. Метод сжатия может внести искажения в рисунок, особенно содержащий текст, мелкие детали или четкие края. Формат JPEG не поддерживает прозрачность. Когда вы сохраняете фотографию в этом формате, прозрачные пикселы заполняются определенным цветом.

Особенности

- Количество цветов в изображении — около 16 миллионов, что вполне достаточно для сохранения фотографического качества изображения.
- Основная характеристика формата — качество, позволяющее управлять конечным размером файла.
- Поддерживает технологию, известную под названием прогрессивный JPEG, при которой версия рисунка с низким разрешением появляется в окне просмотра до полной загрузки самого изображения.

Область применения

Используется преимущественно для фотографий. Не очень подходит для рисунков, содержащих прозрачные участки, мелкие детали или текст.

PNG-8

PNG-8 (Portable Network Graphics, портативная сетевая графика) — формат по своему действию аналогичен GIF. По заверению разработчиков, он использует улучшенный формат сжатия данных, но как показывает практика, это не всегда так.

Особенности

- Использует 8-битную палитру (256 цветов) в изображении, за что и получил в своем названии цифру восемь. При этом можно выбирать, сколько цветов будет сохраняться в файле, — от 2 до 256.
- В отличие от GIF, не отображает анимацию ни в каком виде.

Область применения

Текст, логотипы, иллюстрации с четкими краями, изображения с градиентной прозрачностью.

PNG-24

PNG-24 — формат, аналогичный PNG-8, но использующий 24-битную палитру цвета. Подобно формату JPEG, сохраняет яркость и оттенки цветов в фотографиях. Подобно GIF и формату PNG-8, сохраняет детали изображения как, например, в линейных рисунках, логотипах или иллюстрациях.

Особенности

- Использует примерно 16,7 млн цветов в файле, из-за чего этот формат применяется для полноцветных изображений.
- Поддерживает многоуровневую прозрачность, это позволяет создавать плавный переход от прозрачной области изображения к цветной, так называемый *градиент*.
- Из-за того, что используемый алгоритм сжатия сохраняет все цвета и пикселы в изображении неизменными, по сравнению с другими форматами у PNG-24 конечный объем графического файла получается наибольшим.

Область применения

Фотографии, рисунки, содержащие прозрачные участки, рисунки с большим количеством цветов и четкими краями изображений.

Рамка вокруг изображения

Когда изображение помещается внутрь контейнера `A`, то вокруг рисунка браузер автоматически устанавливает рамку, цвет которой совпадает с цветом ссылок. Также рамку можно добавить самостоятельно, воспользовавшись параметром `border` тега `IMG` (листинг 2.1).

Листинг 2.1. Добавление рамки вокруг изображения

```
<html>
<body>
  <img src=sample.gif width=50 height=50 border=2>
</body>
</html>
```

Толщина рамки всегда устанавливается в пикселах, поэтому в примере параметру `border` присваивается число 2 без указания единиц измерения. Цвет рамки по умолчанию черный и совпадает с цветом текста.

Чтобы убрать рамку, следует задать нулевое значение у параметра `border` (листинг 2.2).

Листинг 2.2. Удаление рамки вокруг изображения

```
<html>
<body text=#00ff00>
<a href=sample.html><img src=sample.gif width=50 height=50 border=0></a>
</body>
</html>
```

Можно также использовать стили, чтобы убрать рамку для всех изображений, которые являются ссылками. Для этого применяется параметр `border` со значением `none` (листинг 2.3).

Листинг 2.3. Использование стилей для отмены рамки изображений

```
<html>
<head>
<style type="text/css">
  A IMG { border: none }
</style>
</head>
<body>
  <a href=/index.html><img src=/images/home.gif></a>
</body>
</html>
```

Конструкция `A IMG` определяет контекст применения стиля — только для тега `IMG`, который находится внутри контейнера `A` и, следовательно, является ссылкой. Такая конструкция называется *контекстным селектором*.

Можно применить следующую хитрость — цвет рамки установить такой же, как и у фона веб-страницы. Рамка в этом случае, сливаясь с фоном, видна не будет (листинг 2.4).

Листинг 2.4. Цвет рамки, заданный с помощью стилей

```
<html>
<head>
<style type="text/css">
  A IMG { border: none; border-color: #fff }
</style>
</head>
<body bgcolor=#ffffff>
  <a href=/index.html><img src=/images/home.gif></a>
</body>
</html>
```

В данном примере фон веб-страницы и цвет рамки установлены белыми.

Приведенным способом можно устанавливать рамку изображения любого цвета, получая тем самым интересные эффекты. Только предварительно надо убрать параметр `border: none`.

Выравнивание изображений

Для изображений можно задать их положение относительно текста или других элементов на веб-странице. Способ выравнивания изображений задается параметром `align` тега `IMG`. В табл. 2.1 перечислены возможные значения этого параметра и результат его использования.

Таблица 2.1. *Способы выравнивания изображений относительно текста*

Значение `align`	Описание	Пример
`absbottom`	Нижняя граница изображения выравнивается по самому нижнему краю текущей строки	Lorem ipsum dolor sit amet,  consectetuer adipiscing elit...
`absmiddle`	Середина изображения выравнивается по средней линии текста	Lorem ipsum dolor sit amet,  consectetuer adipiscing elit...
`bottom` или `baseline`	Нижняя граница изображения выравнивается по базовой линии текстовой строки. Это значение установлено по умолчанию	Lorem ipsum dolor sit amet, consectetuer  adipiscing elit...

Таблица 2.1 *(окончание)*

Значение `align`	Описание	Пример
`left`	Изображение располагается по левому краю родительского элемента	 Lorem ipsum dolor sit amet, consectetuer adipiscing elit...
`middle`	Середина изображения выравнивается по базовой линии текущей строки текста	Lorem ipsum dolor sit amet,  consectetuer adipiscing elit...
`right`	Изображение выравнивается по правому краю родительского элемента	Lorem ipsum dolor sit amet, consectetuer  adipiscing elit...
`texttop`	Верхняя граница изображения выравнивается по самому высокому текстовому элементу текущей строки	Lorem ipsum dolor sit amet,  consectetuer adipiscing elit...
`top`	Верхняя граница изображения выравнивается по самому высокому элементу текущей строки	 Lorem ipsum dolor sit amet,  consectetuer adipiscing elit...

Замечание

Хотя все значения параметра `align` поддерживаются браузерами, аргументы `absbottom`, `absmiddle`, `baseline` и `texttop` не описаны в спецификации HTML 4.

Разберем более подробно некоторые способы выравнивания изображений. Наиболее очевидны два способа — это значения `left` и `right`, они выравнивают рисунок по левому или правому краю родительского элемента. Под этим термином подразумевается контейнер, куда помещено изображение, например ячейка таблицы или окно браузера. При таком выравнивании текст обтекает картинку со свободных сторон. Более подробно обтекание изображений текстом рассматривается далее.

Остальные способы задают выравнивание изображения и близлежащих элементов по вертикали. Их условно можно разбить на три группы: выравнивание по верхнему краю, по середине и по нижнему краю. Так, при использовании аргумента `texttop` браузер определяет верхнюю границу текстовой строки и выравнивает изображение по ней. Заметьте, что под этой границей не всегда подразумевается край символа, как правило, к высоте буквы добавляются небольшие вертикальные отступы сверху и снизу (рис. 2.1). Чтобы их заметить, установите под текстом фоновый цвет или выделите текст с помощью курсора мыши.

Если рядом с изображением находится только текст, то результат использования значений `texttop` и `top` идентичен. Различие между ними проявляется, когда в строке кроме текста располагается еще один элемент, например другое изображение. Тогда рисунок, у которого установлен параметр `align=top`, будет выравниваться по его верхнему краю.

Аналогично дело обстоит и с выравниванием по нижнему краю, аргумент `absbottom` устанавливает изображение по нижней границе близлежащего элемента, а `bottom` — по базовой линии. Здесь надо отметить, что *базовой линией* называется условная черта под нижним краем текста без учета выступающих частей (рис. 2.1).

Рис. 2.1. Базовая линия и границы элемента

При выравнивании по центру имеет значение размер изображения и других элементов. Например, если высота рисунка меньше высоты текста, то результаты применения аргументов `middle` и `absmiddle` параметра `align` будут выглядеть одинаково. Когда рисунок больше текста, тогда элементы располагаются, как указано в табл. 2.1. Заметим, что эта особенность проявляется только в браузере Internet Explorer, остальные браузеры вообще не видят разницы между значениями `middle` и `absmiddle`. Чтобы сравнить результат поведения браузеров, создадим изображение и поместим его рядом с текстом, в параметрах тега `IMG` при этом укажем `align=absmiddle` (листинг 2.5).

Листинг 2.5. Выравнивание изображения по центру текста

```
<html>
<body>
<img src=help.gif width=66 height=65 align=absmiddle> <span style="font-
size: 200%; font-family: Arial, sans-serif">Помощь по сайту</span>
</body>
</html>
```

Для наглядности размер шрифта в примере увеличен с помощью стилевого свойства `font-size`. Результат выполнения примера показан на рис. 2.2.

На рисунке видно, что все браузеры по-разному подходят к выравниванию изображений. Так, Internet Explorer 6 (рис. 2.2, *а*) при значении `absmiddle` параметра `align` и текст, и рисунок выравнивает строго по центру; Netscape 7 (Firefox) (рис. 2.2, *в*) выравнивает картинку по базовой линии

текста, а в браузере Opera 7 (рис. 2.2, *б*) изображение оказывается чуть выше базовой линии.

а *б* *в*

Рис. 2.2. Выравнивание изображения и текста по центру в разных браузерах: *а* — Internet Explorer 6; *б* — Opera 7; *в* — Netcape 7

Создание паспарту

Разновидностью обычной рамки можно считать *паспарту* — так называется картонная рамка для фотографии или рисунка. Использование паспарту

Рис. 2.3. Пример паспарту

зрительно увеличивает изображение, привлекает к нему внимание и делает картину более эффектной. Конечно, на веб-странице нет нужды имитировать подобную рамку, поэтому паспарту в данном случае мы будем называть прямоугольную цветную область вокруг изображения, как показано на рис. 2.3.

Создать паспарту на веб-странице можно двумя основными способами — с помощью таблиц и через стили.

Использование таблиц

Чтобы получить результат, приведенный на рис 2.3, самое простое — использовать таблицу (листинг 2.6). Цвет паспарту определяется параметром `bgcolor`, в данном случае он серый, а цвет внешней рамки — параметром `bordercolor`. Вокруг самого изображения тоже можно установить рамку, указав ее толщину с помощью параметра `border` тега `IMG`. Ширина и высота паспарту определяется размерами таблицы.

Листинг 2.6. Создание паспарту с помощью таблицы

```
<html>
<body>
<table width=150 height=150 border=1 align=center cellspacing=0
bordercolor=#000000 bgcolor=#f0f0f0>
 <tr>
  <td align=center><img src=images/sample.gif width=71 height=71
border=1></td>
 </tr>
</table>
</body>
</html>
```

Чтобы картинка размещалась строго по центру рамки, необходимо у ячейки таблицы (тег `TD`) установить параметр `align=center`.

Если нужно получить двойную рамку, следует изменить параметр `cellspacing` и задать ему значение, отличное от нуля.

Применение стилей

Использование стилей предпочтительно, когда требуется сделать паспарту для большого количества изображений. Создав один класс с нужными параметрами, его затем можно применять где угодно. Однако не все так просто — атрибуты привязаны к размерам изображения, и сделать универсальный код можно, только пожертвовав одним из браузеров. Все дело в том,

что нужны одинаковые отступы по горизонтали и вертикали от края рамки до изображения. И если с горизонтальными отступами не возникает никаких проблем, то вертикальные отступы категорически не хотят быть одинаковыми в браузерах Internet Explorer и Opera. Придется заняться вычислениями. Берем общую высоту паспарту, которую сами и определяем, отнимаем от нее высоту картинки и делим все пополам. Полученное значение будет отступом сверху и снизу, определяемым параметрами `padding-top` и `padding-bottom` (листинг 2.7).

Листинг 2.7. Создание паспарту с помощью стилей

```
<html>
<head>
<style type="text/css">
 .frame {
  padding-top: 40;           /* Отступ сверху */
  padding-bottom: 40;        /* Отступ снизу */
  background: #f0f0f0;       /* Цвет паспарту */
  border: solid 2px black;   /* Параметры рамки */
  width: 150px;              /* Ширина паспарту */
  text-align: center         /* Выравнивание изображения по центру */
}
</style>
</head>
<body>
  <div class=frame><img src=sample.gif width=70 height=70></div>
</body>
</html>
```

Чтобы не создавать множество классов для разных изображений, часть параметров, которые зависят от размеров картинок, можно включить прямо в описание тега `DIV` (листинг 2.8).

Листинг 2.8. Использование стилей

```
<html>
<head>
<style type="text/css">
 .frame {
  background-color: #f0f0f0;
  border: solid 2px black;
  text-align: center
}
```

```
</style>
</head>
<body>
  <div class=frame style="padding-top: 40; padding-bottom: 40; ↵
  width: 150"><img src=sample1.gif width=70 height=70></div><br>
  <div class=frame style="padding-top: 50; padding-bottom: 50; ↵
  width: 200"><img src=sample2.gif width=100 height=100></div>
</body>
</html>
```

Какой из описанных способов создания паспарту предпочесть, зависит большей частью от изображений. Если их размер одинаков, то стили подойдут как нельзя лучше. При большом разбросе размеров картинок и таблицы, и стили работают практически одинаково.

Подрисуночная подпись

Подрисуночная подпись — это текст, который является комментарием к рисунку и его описывает. Такая подпись важна, поскольку она привлекает внимание читателя к рисунку и дает больше информации об изображении. У тега `IMG` существует, конечно, параметр `alt`, который задает текст подсказки, но чтобы ее получить, приходится наводить курсор мыши на каждый рисунок, что довольно неудобно. Более наглядный способ и, соответственно, более предпочтительный заключается в размещении подрисуночной подписи возле самого изображения. Подпись хоть и называется подрисуночной, но может располагаться и сбоку от рисунка, если это продиктовано соображениями верстки и дизайна (рис. 2.4). Результат напоминает создание паспарту, только к нему добавлен текст.

Рис. 2.4. Варианты размещения подрисуночной подписи

Для размещения на веб-странице и рисунка, и подписи к нему удобно воспользоваться таблицей. В этом случае можно задать цвет и ширину каймы вокруг рисунка, а также расположение текста (листинг 2.9).

Листинг 2.9. Использование таблицы для создания подрисуночной подписи

```
<html>
<body>
<table width=100 border=0 cellspacing=0 cellpadding=4 bgcolor=#cc9900>
  <tr>
   <td><img src=figure.jpg width=100 height=111></td>
  </tr>
  <tr>
   <td>Пример подрисуночной подписи</td>
  </tr>
</table>
</body>
</html>
```

Ширина таблицы определяется исходя из размеров рисунка и желания автора. В данном примере ширина таблицы совпадает с шириной рисунка, ширина цветной рамки вокруг изображения задается с помощью параметра `cellpadding` или `cellspacing`. В данном случае все равно, какой из них предпочесть, результат будет один. Цвет таблицы, указанный параметром `bgcolor`, определяет цвет подложки вокруг изображения и подписи к нему.

В листинге 2.9 представлен простой вариант, когда рисунок с подписью размещен по центру страницы. Если же необходимо поместить нашу конструкцию в текст, чтобы он ее обтекал, можно воспользоваться свойством таблицы `align=left | right`. Это свойство выравнивает таблицу по левому или правому краю страницы. Все бы ничего, да вот отступы от края таблицы до текста, как в теге `IMG`, не предусмотрены. Получается очень некрасиво, когда текст словно наезжает на наш рисунок. Страдают принципы дизайна, и простой здравый смысл подсказывает, что читать такой текст неудобно. Нам надо получить отступы вокруг таблицы, и для этого следует воспользоваться вложенными таблицами. Создаем таблицу, выровненную по левому или правому краю, той же ширины, что и рисунок, а параметр `cellpadding` будет определять расстояние от текста до таблицы (листинг 2.10).

Листинг 2.10. Создание отступов с помощью вложенных таблиц

```
<html>
<body>
```

```
<table width=100 border=0 cellspacing=0 cellpadding=6 align=left>
<tr>
 <td>
  <table width=100 border=0 cellspacing=0 cellpadding=4 bgcolor=#cc9900>
   <tr>
    <td><img src=figure.jpg width=100 height=111></td>
   </tr>
   <tr>
    <td>Пример подрисуночной подписи</td>
   </tr>
  </table>
 </td>
</tr>
</table>
</body>
</html>
```

Применение вложенных таблиц сопряжено с определенными сложностями — это и большой объем конечного кода, и неудобство редактирования данных. Поэтому упростим себе задачу и воспользуемся стилями. Создание полей вокруг картинки происходит с помощью параметра `padding`, а отступы от текста задаются атрибутом `margin`. Выравнивание картинки по левому или правому краю задается свойством `float` (листинг 2.11).

Листинг 2.11. Использование стилей для создания подрисуночной подписи

```
<html>
<head>
<style type="text/css">
 #sign {
  padding: 10px;                    /* Поля вокруг изображения */
  margin: 0px 10px 10px 0px;        /* Отступы от рамки до текста */
  background: #e0e0e0;              /* Цвет фона вокруг изображения */
  border: 1px solid black;          /* Параметры рамки */
  width: 100px;                     /* Ширина */
  float: left;                      /* Выравнивание по левому краю */
  font-size: 90%                    /* Размер текста подписи */
}
</style>
</head>
<body>
<div id=sign>
  <img src=puh.jpg width=100 height=111><br>
```

```
Винни-Пух в гостях у Кролика
</div>
- Ну, - замялся Пух, - я мог бы побыть еще немного, если бы ты...
если бы у тебя... - запинался он и при этом почему-то не сводил глаз
с буфета.<br>
- По правде говоря, - сказал Кролик, - я сам собирался пойти
погулять.<br>
- А-а, ну хорошо, тогда и я пойду. Всего хорошего.<br>
- Ну, всего хорошего, если ты больше ничего не хочешь.<br>
- А разве еще что-нибудь есть? - с надеждой спросил Пух, снова оживляясь.
Кролик заглянул во все кастрюли и банки и со вздохом сказал:<br>
- Увы, совсем ничего не осталось!<br>
- Я так и думал, - сочувственно сказал Пух, покачав головой. - Ну, до
свиданья, мне пора идти.
</body>
</html>
```

В данном примере показано создание паспарту с подрисуночной подписью внизу картинки. Рисунок размещается по левому краю окна браузера, текст обтекает его справа. Размер отступа от текста установлен в 10 пикселов по правому и нижнему краю с помощью универсального параметра `margin`. Выравнивание изображения по краю окна легко меняется с помощью параметра `float`. Установите значение `right`, и рисунок будет размещаться справа.

Результат примера показан на рис. 2.5.

В примере приведен способ, когда текст находится под картинкой. Возможны варианты расположения текста справа или слева от изображения. С этой целью модернизируем листинг 2.11 и добавим новый селектор с именем `caption` к изображению (листинг 2.12).

Листинг 2.12. Подрисуночная подпись справа от рисунка

```
<html>
<head>
<style type="text/css">
 #sign {
  padding: 10px; margin: 0px 10px 10px 0px;
  background: #e0e0e0; border: 1px solid black;
  float: left; font-size: 90%
  width: 190px;                              /* Ширина */
}
 #caption {
  float: left;                       /* Подпись справа от картинки */
  border: 1px solid black;           /* Рамка вокруг изображения */
```

```
  margin: 3px                          /* Отступы вокруг изображения */
}
</style>
</head>
<body>
<div id=sign>
  <img src=puh.jpg width=100 height=111 id=caption><br>
Винни-Пух<br>в гостях<br>у Кролика
</div>
...
</body>
</html>
```

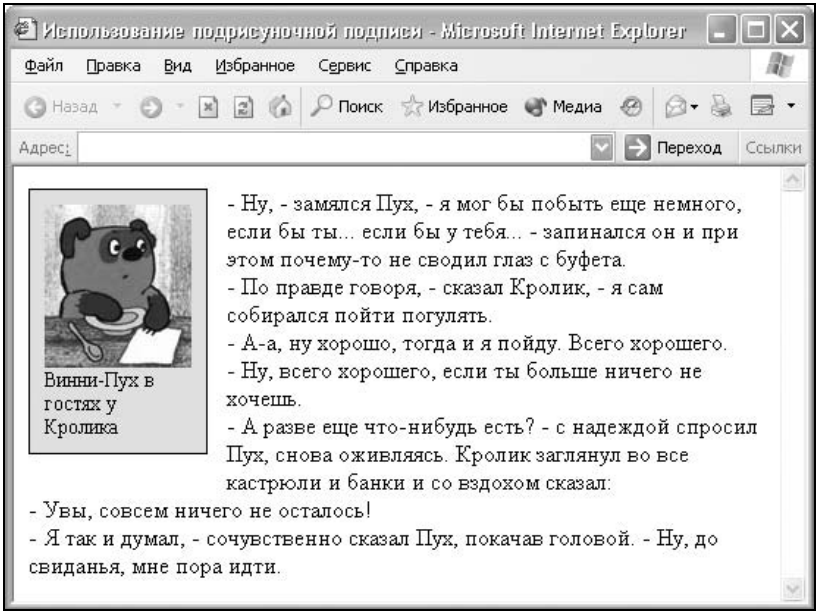


Рис. 2.5. Подрисуночная подпись внизу картинки

В примере к изображению добавляется идентификатор с именем `caption`, его задача состоит в том, чтобы заставить текст располагаться справа от рисунка. Кроме того, изменился ряд параметров, чтобы пример корректно работал в разных браузерах. В частности, необходимо указать новую ширину всей конструкции, поскольку браузер Opera без параметра `width` будет пристраивать текст под рисунок вместо того, чтобы ставить его рядом. Также следует изменить отступы вокруг изображения через параметр `margin`, иначе в некоторых браузерах текст будет слишком плотно прилегать к рисунку.

Обтекание изображения текстом

Обтекание изображения текстом — один из популярных приемов верстки веб-страниц, когда изображение располагается по краю окна браузера, а текст обходит его с разных сторон. Для создания обтекания изображения текстом существует несколько способов, связанных как с возможностями тегов HTML, так и с применением стилей.

Использование параметра *align* тега *IMG*

Выравнивание изображения на веб-странице определяется параметром `align` тега `IMG`. Этот параметр задает, возле какого края окна будет располагаться рисунок, одновременно определяя и способ обтекания текста. Чтобы выровнять изображение по правому краю и задать обтекание слева, используют значение `right`, обратное ему значение — `left`. Параметр `align` часто используют в связке с другими параметрами тега `IMG` – `vspace` и `hspace`. Они устанавливают расстояние от обтекаемого текста до изображения. Без этих атрибутов изображение и текст будут слишком плотно примыкать друг к другу (листинг 2.13).

Листинг 2.13. Использование свойств изображения

```
<html>
<body>
  <img src=sample.gif width=50 height=50 hspace=10 vspace=10 align=left>
  Lorem ipsum dolor sit amet...
</body>
</html>
```

Горизонтальный отступ от картинки до текста управляется параметром `hspace`, он добавляет пустое пространство одновременно слева и справа от изображения. Поэтому рисунок по горизонтали выравнивается не строго по краю окна браузера, а отстоит от него на некотором расстоянии, которое равно значению `hspace`. Чтобы избавиться от данной особенности, можно использовать таблицы.

Применение таблиц

Таблицы с невидимой границей — один из популярных способов верстки, который может пригодиться и в случае обтекания изображения текстом. Для достижения нужного результата воспользуемся свойством `align` тега `TABLE`, имеющим такое же значение и применение, как и у изображения. Но у таблиц больше параметров для управления их отображением, что дает им некоторые преимущества по сравнению с изображениями (листинг 2.14).

Листинг 2.14. Использование таблицы

```
<html>
<body>
<table width=70 height=70 border=0 align=left cellpadding=0
cellspacing=0>
 <tr>
  <td><img src=sample.gif width=50 height=50></td>
 </tr>
</table>
Lorem ipsum dolor sit amet...
</body>
</html>
```

Таблица создает невидимую границу, которая отстоит от самого изображения и не позволяет тексту приблизиться к нему. Размер отступа регулируется шириной и высотой таблицы. Так, в данном примере рисунок выравнивается по левому краю, поскольку этот способ выравнивания установлен у таблицы. Горизонтальный отступ от текста до рисунка будет равен разнице между шириной таблицы и шириной изображения. С вертикальным отступом дело обстоит иначе, по умолчанию выравнивание содержимого ячейки происходит по центру вертикали. Так что если параметр `valign=top` у тега `TD` не задан, отступ по вертикали будет меньше, чем по горизонтали (рис. 2.6).

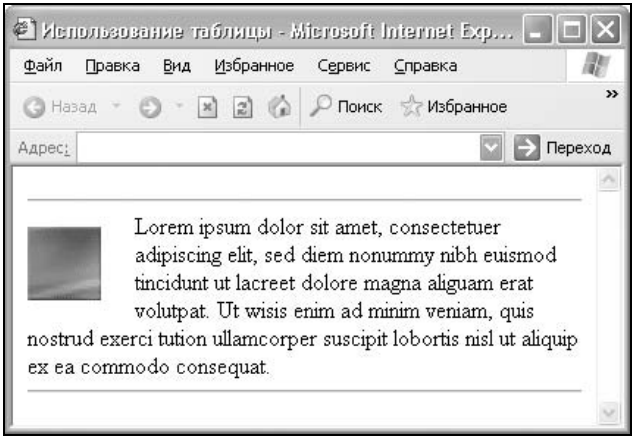


Рис. 2.6. Обтекание рисунка текстом

Использование стилей

Для обтекания картинки текстом можно применить стилевой атрибут `float`. Значение `right` будет выравнивать изображение по правому краю окна брау-

зера, а текст размещать слева от рисунка. Значение `left`, наоборот, выравнивает изображение по левому краю, а текст — справа от рисунка (листинг 2.15). Элемент, для которого установлено значение `float`, обычно называется *плавающим*. Это название, конечно же, условное и говорит лишь о том, что текст или другие объекты будут обходить его с разных сторон, создавая обтекание.

Листинг 2.15. Применение стилей

```
<html>
<head>
<style type="text/css">
 #warp {
  float: left;                /* Выравнивание изображения по левому краю
*/
  margin-right: 10px          /* Отступ справа от изображения */
}
</style>
</head>
<body>
  <img src=sample.gif width=50 height=50 id=warp>
  Lorem ipsum dolor sit amet...
</body>
</html>
```

Применение стилей позволяет отказаться от использования параметров `hspace` и `vspace`, заменив их более гибким атрибутом `margin` или его производными `margin-left`, `margin-right`, `margin-top` и `margin-bottom`. Они позволяют управлять отступами слева, справа, сверху и снизу от рисунка.

Альтернативный текст

Альтернативный текст позволяет получить текстовую информацию о рисунке при отключенной в браузере загрузке изображений. Поскольку загрузка изображения происходит после получения браузером информации о нем, то замещающий рисунок текст появляется раньше. А уже по мере загрузки текст будет сменяться изображением (рис. 2.7).

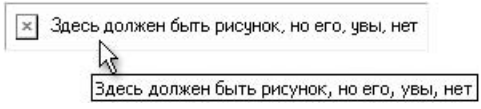


Рис. 2.7. Вид альтернативного текста при отсутствии изображения

Браузеры также отображают альтернативный текст в виде подсказки, появляющейся при наведении курсора мыши на изображение.

Для создания альтернативного текста используется параметр `alt` тега `IMG` (листинг 2.16).

Листинг 2.16. Добавление альтернативного текста

```
<html>
<body>
<img src=home.gif ↳
alt="Здесь должен быть рисунок, но его, увы, нет">
</body>
</html>
```

Текст в параметре `alt` обязательно должен быть взят в кавычки.

Можно создать всплывающую подсказку, которая состоит из нескольких строк текста. Для этого требуется всего лишь установить в требуемом месте кода обычные переносы, как показано в листинге 2.17. Лишние пробелы в этом случае надо убрать.

Листинг 2.17. Создание многострочных подсказок

```
<html>
<body>
<a href=index.html><img src=home.gif alt="Здесь должен быть рисунок, ↳
но его, ↳
увы, нет"></a>
</body>
</html>
```

Стоит отметить, что если в параметрах высота изображения задана небольшой, то альтернативный текст отображаться не будет. Чтобы его увидеть, придется наводить курсор мыши на рисунок и ждать, когда появится всплывающая подсказка.

Замечание

Альтернативный текст не поддерживает браузер Opera до 7 версии.

Добавление фонового рисунка на веб-страницу

В качестве фона можно использовать любое подходящее для этого изображение в формате GIF, JPEG или PNG. Правильно подобранный фон не

отвлекает внимание от текста, хорошо сочетается с цветовой гаммой веб-страницы, при этом файл с фоном желательно должен быть небольшим по объему, чтобы быстро загружаться. Если вы хотите добавить фоновый рисунок на веб-страницу, следует воспользоваться параметром `background` тега `BODY`, как показано в листинге 2.18.

Листинг 2.18. Добавление фонового рисунка

```
<html>
<body background=/image/samplebg.gif>
 ...
</body>
</html>
```

В качестве аргумента параметра `background` принимается путь к графическому файлу. Если изображение меньше размера экрана монитора, оно будет размножено по горизонтали и вертикали.

Поскольку для загрузки файла требуется определенное время, может получиться так, что текст на веб-странице не будет читаться некоторое время, пока не произойдет отображение рисунков. То же самое происходит и при отключенных в браузере рисунках. Поэтому рекомендуется всегда задавать цвет фона наряду с фоновым рисунком (листинг 2.19).

Листинг 2.19. Использование фонового рисунка и цвета фона

```
<html>
<body bgcolor=#c0c0c0 background=/image/samplebg.gif>
 ...
</body>
</html>
```

Цвет фона в примере устанавливается параметром `bgcolor` тега `BODY`.

По умолчанию фоновое изображение заполняет собой всю рабочую область веб-страницы. Изменить эту особенность стандартными средствами HTML не представляется возможным, в этом случае параметрами представления фона лучше управлять с помощью стилей. Универсальный атрибут `background` позволяет задать одновременно цвет фона, фоновое изображение, устанавливает положение рисунка, указывает, фиксировать фон или нет, а также определяет, как будет повторяться изображение. Так, если требуется поместить фоновую картинку в правый верхний угол без дублирования, как показано на рис. 2.8, следует воспользоваться кодом, приведенным в листинге 2.20.

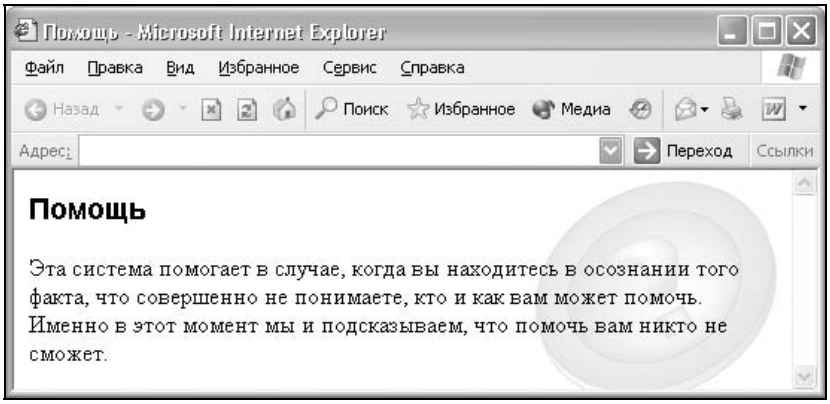


Рис. 2.8. Вид страницы с фоновым рисунком

Листинг 2.20. Добавление фонового рисунка на страницу через стили

```
<html>
<head>
<style type="text/css">
 BODY {
  background:
   white                    /* Цвет фона */
   url(/images/help.gif)    /* Путь к файлу с рисунком фона */
   right top                /* Положение в правом верхнем углу */
   no-repeat                /* Не повторять рисунок */
   fixed                    /* Зафиксировать фон */
}
 H1 {
  font-family: Arial, sans-serif;
  font-size: 120%
}
</style>
</head>
<body>
<h1>Помощь</h1>
Эта система помогает в случае, когда вы находитесь в осознании того
факта, что совершенно не понимаете, кто и как вам может помочь. Именно
в этот момент мы и подсказываем, что помочь вам никто не сможет.
</body>
</html>
```

Значения параметра `background` могут идти в произвольном порядке, браузер сам определит, какое из них соответствует нужному атрибуту. Ни один па-

раметр не является обязательным, поэтому неиспользуемые значения можно опустить. В этом случае будут применяться значения, установленные по умолчанию.

Разберем более подробно параметры в примере. Положением фонового изображения можно управлять с помощью атрибута `background` или `background-position`, который является его производной. У него два значения: положение по горизонтали (`left`, `center`, `right`) и по вертикали (`top`, `center`, `bottom`). Положение также допустимо задавать в процентах, пикселах или других единицах.

Обычно фоновый рисунок при прокрутке содержимого перемещается вместе с ним. Значение `fixed` параметра `background` или `background-attachment` делает фоновое изображение элемента неподвижным, `scroll` — позволяет перемещать фон вместе с содержимым.

Чтобы определить, как будет повторяться фоновое изображение, можно воспользоваться параметром `background-image`, или, как показано в данном примере, `background`. У этого параметра есть следующие аргументы: `no-repeat` — устанавливает одно фоновое изображение в элементе без его повторений (по умолчанию в левом верхнем углу), `repeat` — повторяемость по вертикали и горизонтали, `repeat-x` — только по горизонтали, `repeat-y` — только по вертикали.

Использование фонового рисунка

Используя фоновый рисунок веб-страницы или таблицы, можно создавать различные графические эффекты на сайте, например вертикальные и горизонтальные линии, тени, полоски и многое другое.

Фоновый рисунок

Чтобы на веб-странице с левого края шла вертикальная фоновая полоса, используется следующий прием. Создаем рисунок шириной примерно 1700 пикселов. Такая большая ширина нужна для того, чтобы страница выглядела одинаково на всех мониторах, независимо от их разрешения (рис. 2.9). С левой части рисунок закрашивается нужным цветом желаемой ширины. Можно создавать не только прямоугольные, но и любые декоративные изображения, главное, чтобы они состыковывались друг с другом по вертикали. Полученный рисунок вставляем на веб-страницу как обычный фон параметром `background` тега `BODY`. Поскольку рисунок широкий и не помещается на всю ширину экрана монитора, на странице он будет повторяться только вниз.

В итоге получим страницу, у которой по левому краю идет декоративная полоса (рис. 2.10).

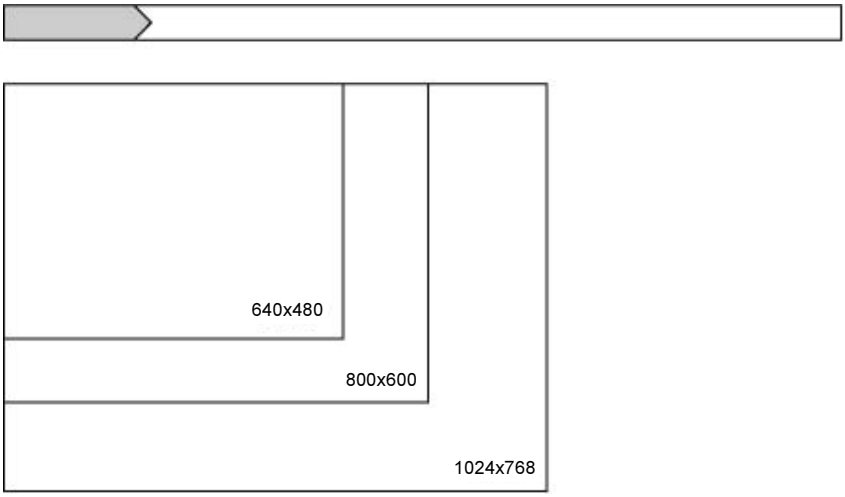


Рис. 2.9. Фоновый рисунок для различных разрешений монитора

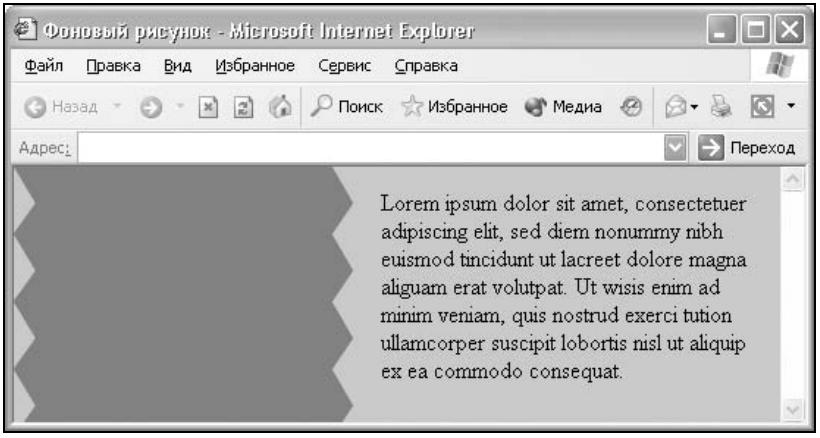


Рис. 2.10. Результат использования широкого фона

Приведенный прием можно использовать и для создания вертикальных полос. В этом случае фоновое изображение должно располагаться вертикально.

Многие авторы делают такой рисунок высотой один-два пиксела, полагая, что объем файла будет минимальным, и загрузка произойдет быстрее. Однако все обстоит наоборот. Компьютер тратит в несколько раз больше времени для отображения одной страницы с узким фоном, что особенно заметно при прокрутке окна браузера. Так что при использовании фонового рисунка делайте изображение высотой 20—30 пикселов — так отображение его на странице будет происходить быстрее.

Создание тени

Очень удобно и просто делать тень от прямоугольной области с помощью таблицы и фонового рисунка. Вначале готовим изображение тени в графическом редакторе, например, как показано на рис. 2.11.

Рис. 2.11. Заготовка для создания тени

Рамка вокруг изображения не нарисована, она показана для того, чтобы были видны границы рисунка. Следующий этап — создание таблицы. Ширина одной из ячеек должна совпадать с шириной тени, в данном случае это расстояние равно 10 пикселов. Внутри ячейки и размещаем фоном изображение, как показано в листинге 2.21.

Листинг 2.21. Создание вертикальной тени

```
<html>
<body>
<table width=300 cellspacing=0 cellpadding=4 bgcolor=#ceffc4>
<tr>
<td width=290><i>Великаны должны быть или большие, или их должно быть
много, если они маленькие.</i></td>
<td width=10 background=shaddow.gif> </td>
</tr>
</table>
</body>
</html>
```

Если ширина таблицы фиксирована, лучше задавать точные размеры всех ячеек. При ширине, заданной в процентах, возможны небольшие изменения размеров ячеек. Вот для такого случая и создается изображение тени (рис. 2.11) с пустым полем справа. Если этого не сделать, тень в некоторых случаях может дублироваться, что нежелательно. А так, если ширина ячейки и будет увеличиваться, несмотря на прописанные размеры, вид тени останется неизменным. Вдобавок, такой "запас" позволяет без негативных последствий изменять параметр `cellpadding`, который управляет отступом от края таблицы до ее содержимого.

Ширины рисунка в 20—30 пикселов вполне достаточно, а цвет фона рисунка должен быть таким же, как у веб-страницы.

Результат представлен на рис. 2.12.

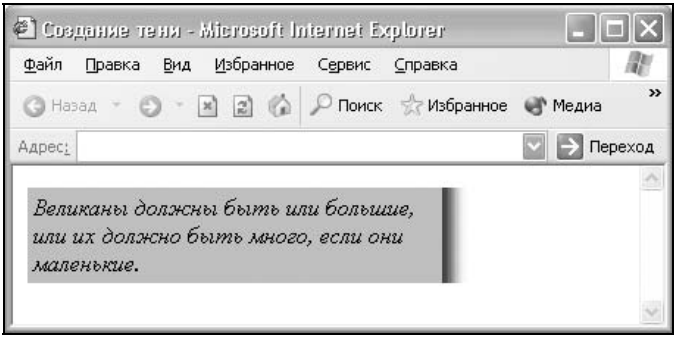


Рис. 2.12. Тень справа от панели с текстом

Результат не впечатляет, явно не хватает тени внизу. Для ее создания добавим ячейку внизу таблицы и разместим в нее фоном еще одну тень, но уже горизонтальную. В правую нижнюю ячейку добавляется рисунок уголка фоном или обычным способом, который соединяет тени по сторонам. Чтобы тень выглядела законченной, следует добавить "заглушки", которые закруглят тень и сместят ее чуть ниже и правее области текста. В итоге понадобится пять изображений, которые представлены в табл. 2.2.

Таблица 2.2. *Изображения, необходимые для создания тени*

Рисунок	Положение	Имя файла
	Вертикальная тень	1.gif
	Горизонтальная тень	2.gif
	Уголок тени	3.gif
	Правая верхняя заглушка	4.gif
	Левая нижняя заглушка	5.gif

Код для создания конечной тени представлен в листинге 2.22.

Листинг 2.22. Создание тени

```
<html>
<body>
<table width=300 cellspacing=0 cellpadding=0 align=center bgcolor=#dddd99>
```

```
<tr>
<td width=288 style="padding: 4px"><i>Великаны должны быть или большие,
или их должно быть много, если они маленькие.</i></td>
<td width=12 background=1.gif valign=top><img src=4.gif width=12
height=20></td>
</tr>
<tr>
<td width=290 height=12 background=2.gif><img src=5.gif width=16
height=12></td>
<td width=10 valign=top background=3.gif><img src=spacer.gif width=10
height=12></td>
</tr>
</table>
</body>
</html>
```

Полученная таблица с тенью приведена на рис. 2.13.

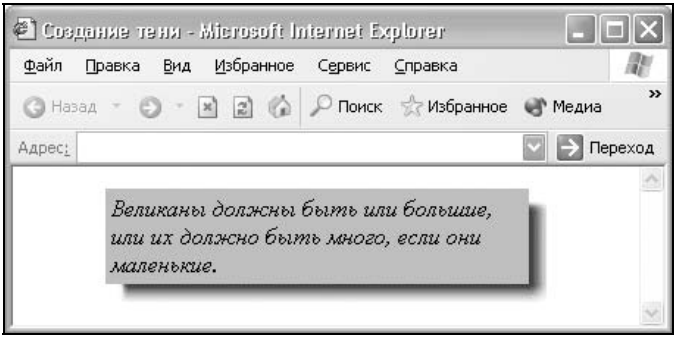


Рис. 2.13. Конечное изображение с тенью

Несколько замечаний относительно приведенного в листинге 2.22 кода.

- Таблицы, в которых используются фоновые рисунки, чувствительны к изменению параметров. Поэтому следует внимательно отнестись ко всем заданным размерам. Если все же появляются пустые промежутки, уберите пробелы после тега `TD`, они иногда могут создавать отступы между ячейками таблицы.
- Значения параметров `cellpadding` и `cellspacing` должны быть нулевыми, иначе получатся ненужные отступы вокруг рисунков. Требуемые отступы задаются стилями, как показано в данном примере (`style="padding: 4px"`).
- По умолчанию содержимое ячеек располагается по центру вертикали, для изменения этой характеристики применяется параметр `valign=top` тега `TD`. Он выравнивает рисунок по верхнему краю ячейки.

- Внутри ячеек, в которых отображается фоновый рисунок, желательно поместить распорку, обычно это прозрачный рисунок в формате GIF. В примере файл называется spacer.gif. Цель использования распорки — не дать ячейке сдвигаться меньше заданных размеров.

Панель на изображении в браузере Internet Explorer

В браузере Internet Explorer, начиная с шестой версии, добавлена новая возможность, предназначенная для работы с изображениями. Если размер рисунка не менее 200 × 200 пикселов, то в его левом верхнем углу отображается специальная панель (рис. 2.14).

Рис. 2.14. Панель инструментов для изображения

Такая панель предназначена для быстрых манипуляций с изображениями: сохранить, распечатать, отправить по почте. Однако в некоторых случаях эта панель только мешает и портит дизайн веб-страницы. Чтобы запретить браузеру выводить панель на экран, добавьте к тегу `IMG` параметр `galleryimg=no`, как показано в листинге 2.23.

Листинг 2.23. Запрет отображения панели инструментов для изображения

```
<html>
<body>
<img src=vedom.jpg width=300 height=400 galleryimg=no>
</body>
</html>
```

Можно также запретить вывод панели для всех изображений на веб-странице одновременно. Требуется только добавить метатег `imagetoolbar` со значением `no`, как показано в листинге 2.24.

Листинг 2.24. Запрет отображения панели инструментов для всех изображений

```
<html>
<head>
  <meta http-equiv="imagetoolbar" content="no">
</head>
<body>
  <img src=vedom.jpg width=300 height=400>
</body>
</html>
```

Карты-изображения

Карты-изображения позволяют создавать ссылки на разные области одного изображения. Использование этого подхода нагляднее, чем обычные текстовые ссылки, и позволяет применять всего один графический файл для организации ссылок. Однако не нужно считать, что карты-изображения следует включать везде, где требуются графические ссылки. Прежде всего, стоит оценить все доводы за и против, а также просмотреть альтернативные варианты.

Преимущества

1. Карты позволяют задать любую форму области ссылки. Учитывая, что изображения по своей природе прямоугольны, сделать графическую ссылку сложной формы, например для указания географического района, без карт-изображений не представляется возможным. Как правило, карты-изображения применяются наиболее часто в географической тематике.
2. С одним файлом удобнее работать — не приходится заботиться о состыковке отдельных фрагментов, и рисунок легко можно поместить в нужное место.

Недостатки

1. Нельзя установить всплывающую подсказку и альтернативный текст для отдельных областей. Альтернативный текст позволяет получить текстовую информацию о рисунке при отключенной в браузере загрузке изображений. Поскольку загрузка изображения происходит после получения браузером информации о нем, то замещающий рисунок текст появляется раньше. А уже по мере загрузки текст будет сменяться изображением. Для карт-изображений эта особенность является актуальной, ведь если отключить просмотр изображений, что делают многие пользователи, то в итоге отображается лишь один пустой прямоугольник.
2. При сложной форме области ссылки увеличивается объем кода HTML. Контур аппроксимируется набором прямых отрезков, для каждой точки такого отрезка следует задать две координаты, а общее количество таких точек может быть достаточно велико. Справедливости ради следует отметить, что сложные формы являются частным случаем и применяются достаточно редко.
3. К картам-изображениям нельзя применять разные эффекты, которые доступны при разрезании одного рисунка на фрагменты: эффект перекатывания, частичная анимация, индивидуальная оптимизация картинок для их быстрой загрузки.

С позиции удобства пользователей карты-изображения имеют только одно преимущество — включение ссылок разнообразной формы. Это добавляет наглядность в представлении информации — мы не ограничены прямоугольной формой ссылки и можем использовать ссылки сложной конфигурации для своих целей. Во всех остальных случаях от них проку нет — обычные текстовые ссылки более информативны, и они не зависят от отключения показа картинок в браузере. Тот факт, что одно изображение загружается быстрее, чем та же картинка, но порезанная на фрагменты и сохраненная в виде набора графических файлов, можно легко обойти. Каждый из таких конечных файлов можно уменьшить, используя индивидуальные настройки оптимизации. В итоге общий объем всех фрагментов будет занимать меньше места, чем одно изображение. Не стоит сбрасывать со счетов и психологический фактор — человеку кажется, что набор маленьких картинок загружается быстрее, чем одна большая.

Основной недочет карт — нет четко выделенных границ ссылок. Поэтому эти границы приходится выделять разными средствами непосредственно на изображении. Если рисунок не загрузился по каким-либо причинам, то разобраться в наборе ссылок становится весьма проблематично.

Карты-изображения реализуются в двух различных вариантах — серверном и клиентском. В случае применения серверного варианта браузер посылает запрос на сервер для получения адреса выбранной ссылки и ждет ответа

с нужной информацией. Такой подход требует дополнительного времени на ожидание результата и отдельные файлы для каждой карты-изображения.

В клиентском варианте карта располагается в том же HTML-документе, что и ссылка на изображение. Для указания того, что изображение является картой, применяется параметр `usemap` тега `IMG`. В качестве значения используется ссылка на описание конфигурации карты, которая устанавливается с помощью тега `MAP`. Значение параметра `name` тега `MAP` должно соответствовать имени в `usemap`. При этом аргумент параметра `usemap` тега `IMG` начинается с символа решетки (листинг 2.25).

Листинг 2.25. Создание карты-изображения

```
<html>
<body>
<img src=map.gif width=411 height=46 border=0 usemap=#navigation
alt="Навигация по сайту">
<map name=navigation>
<area shape=poly coords="210,27, 203,9, 202,6, 197,2, 192,1, 120,1,
115,2, 110,6, 112,9, 119,27, 119,32, 211,32, 210,27" href=link1.html>
<area shape=poly coords="302,27, 295,9, 293,6, 289,2, 283,1, 212,1,
206,2, 202,6, 203,9, 210,27, 211,32, 284,32, 303,32, 302,27"
href=link2.html>
<area shape=poly coords="302,27, 303,32, 394,32, 393,27, 386,9, 382,3,
375,1, 303,1, 298,2, 293,6, 295,9, 302,27" href=link3.html>
</map>
</body>
</html>
```

Внутри контейнера `MAP` располагается один или несколько тегов `AREA`, они задают форму области, ее координаты, устанавливают адрес документа, на который следует сделать ссылку, а также имя окна или фрейма, куда браузер будет загружать документ.

Тег `AREA` имеет следующие параметры.

- `SHAPE`. Определяет форму активной области. Форма может быть в виде окружности (`circle`), прямоугольника (`rect`), полигона (`poly`).
- `ALT`. Добавляет альтернативный текст для каждой области. Служит лишь комментарием для ссылки, поскольку на экран не выводится. Тем не менее иногда рекомендуется добавлять альтернативный текст для повышения рейтинга сайта по ключевым словам в поисковых системах и каталогах.
- `HREF`. Задает адрес документа, на который следует перейти. Поскольку в качестве адреса ссылки может использоваться документ любого типа, то

результат перехода по ссылке зависит от конечного файла. Так, архивы (файлы с расширением zip или rar) будут сохраняться на локальный диск. По умолчанию новый документ загружается в текущее окно браузера, однако это свойство можно изменить с помощью параметра `target`.

- `COORDS`. Задает координаты активной области. Координаты отсчитываются в пикселах от левого верхнего угла изображения, которому соответствует значение 0, 0. Первое число является координатой по горизонтали, второе — по вертикали. Список координат зависит от формы области.

Для окружности задаются три числа — координаты центра круга и радиус.

```
<area shape=circle coords="230, 340, 100" href=circle.html>
```

Для прямоугольника — координаты левого верхнего и правого нижнего угла.

```
<area shape=rect coords="24,18, 210, 56" href=rect.html>
```

Для полигона задаются координаты его вершин, как показано на рис. 2.15.

Рис. 2.15. Координатные точки для полигона

- `TARGET`. По умолчанию, при переходе по ссылке документ открывается в текущем окне или фрейме. При необходимости это условие может быть изменено параметром `target` тега `AREA` (листинг 2.26). В качестве аргумента используется имя окна или фрейма, заданное параметром `name`. Если установлено несуществующее имя, то будет открыто новое окно. В качестве зарезервированных имен используются следующие:
  - `_blank` — загружает страницу в новое окно браузера.
  - `_self` — загружает страницу в текущее окно.
  - `_parent` — загружает страницу во фрейм-родитель; если фреймов нет, то этот параметр работает как `_self`.
  - `_top` — отменяет все фреймы и загружает страницу в полном окне браузера; если фреймов нет, то этот параметр работает как `_self`.

Листинг 2.26. Применение параметра `target`

```
<html>
<body>
<map name=nortland>
```

```
 <area coords="21, 24, 121, 124" shape=rect href=/sch/images/new.html
target=_blank>
</map>
<img src=/images/piter.gif width=200 height=200 usemap=#nortland>
</body>
</html>
```

В данном примере показано создание ссылки прямоугольной области внутри изображения. При активации ссылки открывается новое окно, в которое загружается документ с именем `new.html`.

ГЛАВА 3

Ссылки

Ссылки являются основой гипертекстовых документов и позволяют переходить с одной веб-страницы на другую. Особенность ссылки состоит в том, что она может указывать не только на html-файл, но и на файл любого типа, причем этот файл может размещаться совсем на другом сайте. Главное, чтобы к файлу, на который делается ссылка, был доступ.

Создание ссылок

Для создания ссылки необходимо сообщить браузеру, какой элемент является ссылкой, а также указать адрес документа, на который следует сделать ссылку. Оба действия выполняются с помощью тега-контейнера `A`. Этот тег является одним из важных элементов HTML и предназначен для создания ссылок. В зависимости от наличия параметров `name` или `href` тег `A` устанавливает ссылку или якорь. *Якорем* называется закладка внутри страницы, которую можно указать в качестве цели ссылки. При использовании ссылки, которая указывает на якорь, осуществляется переход к закладке внутри веб-страницы. В качестве значения параметра `href` используется адрес документа (URL, Universal Resource Locator, универсальный указатель ресурсов), на который происходит переход. Адрес ссылки может быть абсолютным и относительным. К *абсолютному адресу* относится полный путь к документу, включая протокол и наименование сайта, например **http://www.htmlbook.ru /about/**. Эта форма обращения работает везде и всюду, независимо от имени сайта или веб-страницы, где прописана ссылка. Как правило, абсолютные адреса применяются для перехода на другой ресурс, а внутри текущего сайта применяются *относительные ссылки*. Подобные ссылки, как следует из их названия, построены относительно текущего документа или корня сайта. Когда путь ведется от корня сайта, в начале пути добавляют слэш (символ /), например **/forum/source/adm.html**. В этом случае сервер понимает,

что ему следует загрузить документ по адресу **http://www.htmlbook.ru/forum/source/adm.html**. Учтите, что ссылки относительно корня сайта не работают на локальном компьютере, а только под управлением веб-сервера. Вот некоторые примеры относительных адресов.

```
/
/demo/
```

Эти две ссылки называются *неполными* и указывают веб-серверу загружать файл index.html (или default.html), который находится в корне сайта или папке demo. Если файл index.html отсутствует, браузер, как правило, показывает список файлов, находящихся в данном каталоге.

```
/images/pic.html
```

Слэш перед адресом говорит о том, что адресация начинается от корня сайта. Ссылка ведет на рисунок pic.html, который находится в папке images. А та, в свою очередь, размещена в корне сайта.

```
../help/me.html
```

Две точки перед именем указывают браузеру перейти на уровень выше в списке каталогов сайта.

```
manual/info.html
```

Если перед именем папки нет никаких дополнительных символов, вроде двух точек, то она размещена внутри текущего каталога.

Далее описаны параметры тега `A`, позволяющие создавать ссылки разных типов, например такие, которые открываются в новом окне.

- `HREF`. Задает адрес документа, на который следует перейти. Поскольку в качестве адреса ссылки может использоваться документ любого типа, то результат перехода по ссылке зависит от конечного файла. Так, архивы (файлы с расширениями zip или rar) будут сохраняться на локальный диск. По умолчанию новый документ загружается в текущее окно браузера, однако это свойство можно изменить с помощью параметра `target`.
- `NAME`. Параметр `name` используется для определения закладки внутри страницы. Вначале следует задать в соответствующем месте закладку и установить ее имя при помощи параметра `name` тега `A`. Имя ссылки на закладку начинается символом `#`, после него идет название закладки. Название выбирается любое, соответствующее тематике. Можно также дать ссылку на закладку, находящуюся в другой веб-странице и даже другом сайте. Для этого в адресе ссылки следует указать ее адрес и в конце добавить символ решетки `#` и имя закладки. Между тегами `<a name=...>` и `</a>` текст писать не обязательно, т. к. требуется лишь указать местоположение перехода по ссылке.

- TARGET. По умолчанию при переходе по ссылке документ открывается в текущем окне или фрейме. При необходимости это условие может быть изменено параметром target тега A. В качестве аргумента используется имя окна или фрейма, заданное параметром name. Если установлено несуществующее имя, то будет открыто новое окно. В качестве зарезервированных имен используются следующие:
  - _blank — загружает страницу в новое окно браузера;
  - _self — загружает страницу в текущее окно;
  - _parent — загружает страницу во фрейм-родитель; если фреймов нет, то этот параметр работает как _self;
  - _top — отменяет все фреймы и загружает страницу в полном окне браузера; если фреймов нет, то этот параметр работает как _self.

Результат применения различных параметров приведен в листинге 3.1.

Листинг 3.1. Создание различных ссылок

```
<html>
<body>
<p><a name=top>Содержание</a></p>
<a href=../wormik/knob.html>Относительная ссылка</a><br>
<a href=http://www.htmlbook.ru/wormik/knob.html>Абсолютная ссылка</a><br>
<a href=new.html target=_blank>Ссылка откроется в новом окне</a><br>
<a href=#top>Наверх к содержанию</a>
</body>
</html>
```

В данном примере продемонстрировано создание закладки с помощью параметра name, использование относительных и абсолютных ссылок, а также параметра target.

Виды ссылок

Обычно в качестве ссылки выступает текст или изображение. Текст ссылки, как правило, помечается подчеркиванием и цветом, чтобы его было легко визуально отличить от обычного текста. С рисунками сложнее: явно определить, ссылка перед нами или нет, можно только после того, как мы наведем курсор мыши на рисунок. Для ссылок курсор превращается в "руку", а в статусной строке браузера отображается путь к документу, на который ссылка указывает. Чтобы показать, что ссылкой служит изображение, вокруг него иногда добавляют рамку параметром border тега IMG. Цвет рамки при

этом совпадает с цветом текстовых ссылок. Однако рамки вокруг картинок часто не вписываются в дизайн веб-страницы, и от их использования обычно отказываются. Поэтому рисунки в качестве ссылок лучше применять там, где они ожидаемы, — баннеры, кнопки, закладки, изображения с надписью "Нажми..." и т. д.

В листинге 3.2 показано создание ссылок с помощью рисунков и текста.

Листинг 3.2. Текстовые и графические ссылки

```
<html>
<body link=#0033cc vlink=#cecece alink=#dd0000>
<p><a href=link1.html>Текстовая ссылка</a>
<p><a href=link2.html><img src=banner.gif width=468 height=60></a>
<p><a href=link3.html><img src=button.gif width=80 height=40 border=0
align=absmiddle> Текстовая ссылка с рисунком</a>
</body>
</html>
```

Первая ссылка в данном примере представляет собой обычный текст, заключенный внутрь контейнера `A`. Следующая строка демонстрирует создание графической ссылки. Изображение, добавляемое на веб-страницу через тег `IMG`, в этом случае надо поместить между тегами `<a href=...>` и `</a>`. Кроме того, внутри контейнера `A` можно одновременно сочетать текст и рисунки, как показано в данном примере.

Любая ссылка на веб-странице может находиться в одном из следующих состояний.

- *Непосещенная ссылка* (`link`). Такое состояние характерно для ссылок, которые еще не открывали. По умолчанию непосещенные текстовые ссылки изображаются синим цветом и с подчеркиванием.
- *Активная ссылка* (`alink`). Ссылка помечается как активная в момент ее открытия. Поскольку время между нажатием на ссылку и началом загрузки нового документа достаточно мало, подобное состояние ссылки весьма кратковременно. Активной ссылка становится также при ее выделении с помощью клавиатуры. Цвет такой ссылки по умолчанию красный.
- *Посещенная ссылка* (`vlink`). Как только пользователь открывает документ, на который указывает ссылка, она помечается как посещенная и меняет свой цвет на фиолетовый, установленный по умолчанию.

Указанные цвета ссылок задаются в качестве параметров тега `BODY`, как показано в листинге 3.2. Параметры не являются обязательными, и если они не указаны, используются значения по умолчанию.

Для управления цветом ссылок с помощью стилей предлагается использовать *псевдоклассы*. Псевдоклассы применяются в CSS, чтобы определять стиль элемента при изменении его состояния. Общий синтаксис для ссылок будет такой.

```
A:псевдокласс { параметр: значение }
```

Различают следующие псевдоклассы, имеющие отношение к ссылкам:

- `active` — активная ссылка;
- `link` — непосещенная ссылка;
- `hover` — состояние ссылки при наведенном на нее курсоре мыши;
- `visited` — посещенная ссылка.

В листинге 3.3 показано использование псевдоклассов для управления цветом ссылок.

Листинг 3.3. Псевдоклассы для ссылок

```
<html>
<head>
<style type="text/css">
  A:link { color: #003366 }
  A:visited { color: #660066 }
  A:hover { color: #800000 }
  A:active { color: #ff0000 }
</style>
</head>
<body>
 <a href=link1.html>Ссылка 1</a> | <a href=link2.html>Ссылка 2</a> | <a
href=link3.html>Ссылка 3</a>
</body>
</html>
```

Псевдокласс `link` обычно не используют, поскольку он определяет то же самое, что и обычный селектор `A`, а именно стиль для непосещенных ссылок. Чтобы не было ненужного повторения, `link`, как правило, опускают.

Ссылки без подчеркивания

Подчеркивание текстовых ссылок уже стало определенным стандартом и сигналом о том, что это не просто текст, а именно ссылка. Это, кстати, является одной из причин, по которой не следует применять подчеркивание к обычному тексту, — пользователи будут считать, что имеют дело со ссыл-

кой. Наряду с использованием подчеркивания благодаря CSS у разработчиков сайтов появился и альтернативный вариант — создание ссылок без подчеркивания. С позиции удобства это не совсем верное решение, поскольку пользователь не может сразу догадаться, что текст, который он видит, является ссылкой. Все ведь уже привыкли — раз используется подчеркивание, значит, это ссылка. Но при правильном применении отсутствие подчеркивания у ссылок может придать сайту определенный эффект. Только надо обязательно дать понять пользователю, что является ссылкой, а что обычным текстом, разграничивая их, например, цветом. Еще можно сделать так, что при наведении курсора ссылка становится подчеркнутой, меняет свой цвет или используется и тот, и другой эффект одновременно.

Чтобы убрать подчеркивание у ссылки, следует для селектора `A` добавить параметр `text-decoration: none` (листинг 3.4).

Листинг 3.4. Отсутствие подчеркивания у ссылок

```
<html>
<head>
<style type="text/css">
 A {
  text-decoration: none      /* Отменяем подчеркивание у ссылки */
}
</style>
</head>
<body>
  <a href=link.html>Ссылка без подчеркивания</a>
</body>
</html>
```

Для псевдоклассов `hover` и `visited` нет необходимости добавлять параметр `text-decoration`, они наследуют свойства селектора `A`.

Подчеркивание ссылок при наведении на них курсора мыши

Чтобы добавить подчеркивание для ссылок, у которых установлен параметр `text-decoration: none`, следует воспользоваться псевдоклассом `A:hover`. Он определяет стиль ссылки, когда на нее наводится курсор мыши. Остается только добавить для псевдокласса параметр `text-decoration: underline` (листинг 3.5).

Листинг 3.5. Подчеркивание ссылок

```
<html>
<head>
<style type="text/css">
 A {
  text-decoration: none              /* Подчеркивание убираем */
}
 A:hover {
  text-decoration: underline         /* При наведении курсора
                                     подчеркивание появляется */
}
</style>
</head>
<body>
  <a href=link.html>Ссылка без подчеркивания</a>
</body>
</html>
```

Еще один пример демонстрирует использование в ссылках одновременно подчеркивания и надчеркивания. При этом тонкие линии будут появляться одновременно над и под ссылкой при наведении на нее курсора. Это достигается применением параметра `text-decoration: underline overline` в селекторе `A:hover` (листинг 3.6).

Листинг 3.6. Использование подчеркивания и надчеркивания в ссылках

```
<html>
<head>
<style type="text/css">
 A { text-decoration: none }
 A:hover { text-decoration: underline overline }
</style>
</head>
<body>
  <a href=link.html>Ссылка без подчеркивания</a>
</body>
</html>
```

При использовании приведенного в данном примере стиля ссылки будут без подчеркивания, но зато сразу с двумя линиями; стоит навести на них курсор, как показано на рис. 3.1.

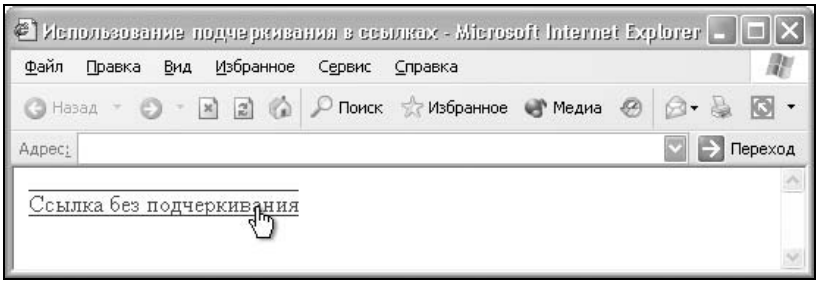


Рис. 3.1. Результат применения подчеркивания и надчеркивания ссылок

Подобное применение подчеркивания в ссылках несколько непривычно, но все равно понятно, что текст служит ссылкой.

Изменение цвета ссылки

Аналогично, с помощью псевдокласса `hover` можно не только добавлять подчеркивание ссылки, но и менять ее цвет через параметр `color`, как показано в листинге 3.7.

Листинг 3.7. Изменение цвета ссылки

```
<html>
<head>
<style type="text/css">
 A {
  text-decoration: none;            /* Скрываем подчеркивание у ссылок */
  color: #0000ff                    /* Исходный цвет ссылки */
}
 A:visited {
  color: #800080                    /* Цвет посещенных ссылок */
}
 A:hover {
  text-decoration: underline;       /* При наведении курсора отображается
                                    подчеркивание */
  color: #ff0000                    /* Цвет ссылки при наведении курсора
                                    */
}
</style>
</head>
<body>
  <a href=link.html>Цветная ссылка</a>
</body>
</html>
```

В примере показано задание ссылок синего цвета (селектор A), посещенных ссылок фиолетового цвета (селектор A:visited) и ссылок красного цвета при наведении на них курсора.

Замечание

При одновременном использовании псевдоклассов visited и hover имеет значение их порядок. Последним в списке стилей должен идти hover, иначе посещенные ссылки не будут изменять свой цвет при наведении на них курсора.

Приведенным способом можно также изменять не только цвет, но и размер ссылки, добавляя параметр font-size к селектору A:hover. Но в этом случае происходит переформатирование окружающего текста, поэтому такой способ акцентирования ссылок использовать не рекомендуется.

Изменение цвета подчеркивания ссылки

При помощи CSS ссылкам можно придать интересную особенность. Цвет ссылки при наведении на нее курсора мыши остается неизменным, зато у нее появляется подчеркивание другого цвета, нежели сама ссылка. Данный прием работает в браузерах Netscape 6, Mozilla, Firefox, Internet Explorer 5.5, Opera 5 и старше.

Чтобы создать подобный эффект, необходимо текст ссылки вложить в контейнер, например тег SPAN, у которого цвет, установленный с помощью стилей, совпадает с цветом ссылок. Цвет линии управляется селектором A:hover, как показано в листинге 3.8.

Листинг 3.8. Создание подчеркивания другого цвета

```
<html>
<head>
<style type="text/css">
 A {
  color: blue                /* Исходный цвет ссылки */
}
 A:hover {
  color: red                 /* Цвет подчеркивания у ссылки при наведении
                             курсора */
}
 .link {
  color: blue                /* Цвет такой же, как у ссылок */
}
</style>
</head>
```

```
<body>
  <a href=link.html><span class=link>Ссылка</span></a>
</body>
</html>
```

В данном примере создается ссылка синего цвета с подчеркиванием, причем цвет линии меняется на красный при наведении на текст курсора мыши.

Декоративное подчеркивание ссылок

Подчеркивание у ссылок можно задать не просто сплошной линией, а, например, пунктирной. Для этого надо воспользоваться параметром `border-bottom`, который создает линию внизу элемента. Указав один из аргументов этого параметра `dashed`, получим пунктирное подчеркивание. В листинге 3.9 показано задание синего пунктира у ссылок красного цвета при наведении на них курсора мыши.

Листинг 3.9. Создание пунктирного подчеркивания для ссылок

```
<html>
<head>
<style type="text/css">
 A {
  color: #ff0000                    /* Цвет ссылок */
}
 A:hover {
  text-decoration: none;            /* Убираем подчеркивание у ссылок */
  border-bottom: 1px dashed blue    /* Добавляем синее пунктирное
                                    подчеркивание */
}
</style>
</head>
<body>
  <a href=link.html>Подчеркнутая ссылка</a>
</body>
</html>
```

При использовании приведенного способа пунктирная линия появляется чуть ниже обычного подчеркивания текста. Поэтому к стилю ссылки следует добавить параметр `text-decoration: none`, чтобы одновременно не получилось две линии (рис. 3.2).

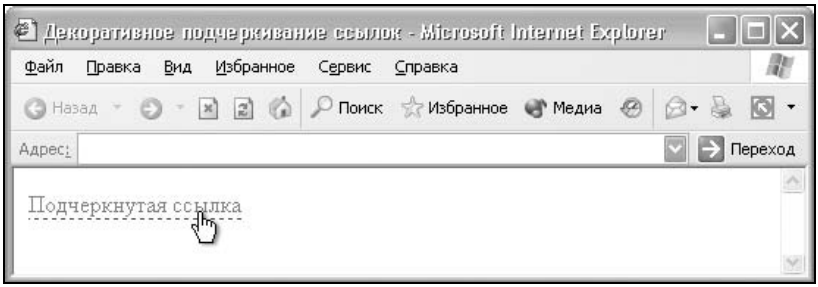


Рис. 3.2. Использование пунктира для выделения ссылки

Не обязательно использовать пунктир, например, для создания двойной линии следует указать параметр `border-bottom: 4px double red`, как показано в листинге 3.10.

Листинг 3.10. Двойное подчеркивание ссылок

```
<html>
<head>
<style type="text/css">
A { color: #0000ff }
A:hover { text-decoration: none; border-bottom: 4px double red }
</style>
</head>
<body>
<a href=link.html>Подчеркнутая ссылка</a>
</body>
</html>
```

Изменяя толщину линии, ее тип и цвет, можно получить множество разнообразных эффектов.

Ссылки разных цветов

Часто возникает необходимость на одной странице одновременно использовать ссылки разных цветов и размеров и применять для каждой области веб-страницы ссылки подходящего типа — одни для меню, другие для текста. Для этого требуется создать два или больше класса со своими параметрами и добавить их к тегу `A`. В листинге 3.11 показано создание трех ссылок разных размеров и цветов. Достаточно поменять значения у соответствующего класса, и цвета у ссылок, где этот класс используется, изменятся автоматически.

Листинг 3.11. Ссылки разных цветов

```
<html>
<head>
<style type="text/css">
  A { font-size: 14px; color: red }
  A:hover { color: green }
  A.link1 { font-size: 12px; color: green }
  A.link1:hover { color: blue }
  A.link2 { font-size: 120%; color: blue }
  A.link2:hover { color: red }
</style>
</head>
<body>
  <p><a href=link1.html>Ссылка 1</a>
  <p><a href=link2.html class=link1>Ссылка 2</a>
  <p><a href=link3.html class=link2>Ссылка 3</a>
</body>
</html>
```

Как видно из данного примера, к любым стилям ссылок применимы псевдоклассы `hover`, а также другие, например `visited` и `active`. Вначале надо указать селектор тега `A`, затем через точку имя класса, после чего через двоеточие добавить требуемый псевдокласс.

Чтобы не писать для каждой ссылки имя класса, что особенно неудобно при большом количестве ссылок, можно воспользоваться контекстными селекторами. Для этого следует определить стиль ссылок, которые будут размещаться внутри элемента, это может быть, например, тег `DIV` или другой подходящий тег (листинг 3.12).

Листинг 3.12. Использование контекстных селекторов

```
<html>
<head>
<style type="text/css">
  A { font-size: 14px; color: red }
  DIV.menu { background: #87bcc9 }
  .menu A { color: #005c6b }
  .menu A:hover { color: #ffff00 }
</style>
</head>
<body>
```

```
  <p><a href=link1.html>Ссылка 1</a>
  <div class=menu>
    <p><a href=link2.html>Ссылка 2</a>
    <p><a href=link3.html>Ссылка 3</a>
  </div>
</body>
</html>
```

В данном примере показано создание класса `menu` для тега `DIV`, внутри которого цвет ссылок будет не красным, как он указан выше, а другим.

Альтернативные способы выделения ссылок

Хотя для выделения текстовых ссылок традиционно применяется подчеркивание, допустимо использование и других способов акцентирования ссылок, которые пригодятся при создании меню. Вот некоторые из них:

- выделение фоновым цветом;
- рамка вокруг ссылки;
- волнистая линия под ссылкой.

Далее указанные методы выделения ссылок рассматриваются более подробно.

Использование фонового цвета

Чтобы добавить к ссылке цветной фон, достаточно воспользоваться параметром `background`, присвоив ему цвет в любом доступном формате. Аналогично можно использовать псевдокласс `hover`, тогда цвет фона под ссылкой будет изменяться при наведении на нее курсора мыши (листинг 3.13).

Листинг 3.13. Фон под ссылкой

```
<html>
<head>
<style type="text/css">
 A {
  background: #fc0;   /* Цвет фона под ссылкой */
  padding: 2px        /* Поля вокруг текста ссылки */
}
 A:hover {
  background: #f73;   /* Цвет фона при наведении на ссылку курсора */
  color: yellow       /* Новый цвет текста */
}
```

```
</style>
</head>
<body>
  <p><a href=link1.html>Ссылка 1</a>
  <p><a href=link2.html>Ссылка 2</a>
  <p><a href=link3.html>Ссылка 3</a>
</body>
</html>
```

Фон под ссылкой точно соответствует области текста, поэтому в примере для селектора `A` добавлен параметр `padding`, создающий поля вокруг текста.

Рамка вокруг ссылки

При использовании рамок со ссылками возможны два варианта. Первый — рамка вокруг ссылок устанавливается заранее и при наведении на нее курсора меняет свой цвет. И второй — рамка отображается, только когда на ссылку наводится курсор.

В листинге 3.14 показано, как изменять цвет рамки, используя атрибут `border`. Подчеркивание текста через параметр `text-decoration` можно убрать или оставить без изменения.

Листинг 3.14. Изменение цвета рамки у ссылок

```
<html>
<head>
<style type="text/css">
 A {
  border: 1px solid blue;           /* Синяя рамка вокруг ссылок */
  padding: 2px;                     /* Поля вокруг текста */
  text-decoration: none             /* Скрываем подчеркивание */
}
 A:hover {
  border: 1px solid red     /* Красная рамка при наведении курсора на
                            ссылку */
}
</style>
</head>
<body>
  <p><a href=link1.html>Ссылка 1</a>
  <p><a href=link2.html>Ссылка 2</a>
  <p><a href=link3.html>Ссылка 3</a>
</body>
</html>
```

Чтобы рамка не "прилипала" к тексту, в примере вокруг него установлены поля с помощью параметра `padding`. Можно также вместе с применением рамки добавить и фон, воспользовавшись свойством `background`.

Если требуется добавить рамку к ссылкам при наведении на них курсора, то следует позаботиться о том, чтобы текст в этом случае не сдвигался. Достичь этого можно двумя способами. Первый способ заключается в создании невидимой рамки вокруг ссылки и последующего изменения цвета рамки с помощью псевдокласса `hover`. Этот способ аналогичен методу, приведенному в листинге 3.10, за исключением того, что первоначальный цвет рамки должен совпадать с цветом фона.

Второй способ добавления рамок вокруг ссылок применяется, когда на вебстранице используется неоднородный фоновый рисунок. В этом случае трудно подобрать исходный цвет рамки, чтобы она не выделялась. Нейтрализовать неизбежный сдвиг текста можно, используя разные отступы от текста до рамки, задаваемые параметром `padding` для селекторов `A` и `A:hover`. Значение для селектора `A:hover` получается вычитанием толщины рамки из аргумента свойства `padding` селектора `A`.

В листинге 3.15 приведены оба описанных способа добавления рамок к ссылкам.

Листинг 3.15. Добавление рамки вокруг ссылки

```
<html>
<head>
<style type="text/css">
/* Первый способ */
 A {
  border: 1px solid white;   /* Белая рамка, совпадающая с цветом фона */
  padding: 1px;              /* Поле вокруг ссылки */
  text-decoration: none      /* Убираем подчеркивание у ссылок */
}
 A:hover {
  border: 1px solid red      /* Красная рамка вокруг ссылки */
}

/* Второй способ */
 A {
  text-decoration: none;     /* Убираем подчеркивание у ссылок */
  padding: 2px               /* Поле вокруг ссылки */
}
 A:hover {
  border: 1px solid red;     /* Красная рамка вокруг ссылки */
  padding: 1px               /* Изменяем величину поля вокруг ссылки */
}
```

```
</style>
</head>
<body>
  <p><a href=link1.html>Ссылка 1</a>
  <p><a href=link2.html>Ссылка 2</a>
  <p><a href=link3.html>Ссылка 3</a>
</body>
</html>
```

На рис. 3.3 показано, как выглядит выделенная ссылка с рамкой при наведении на нее курсора мыши.

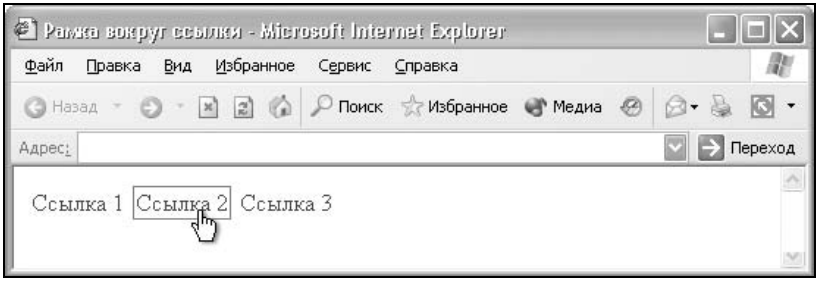


Рис. 3.3. Ссылка с рамкой

Оба приведенных способа одинаково универсальны и хорошо работают как с подчеркнутыми ссылками, так и со ссылками без подчеркивания.

Ссылки можно выделять, не просто добавляя рамку вокруг них при наведении на них курсора мыши, но и меняя стиль границы. Например, используя стилевой атрибут `border-style: outset` для селектора `A`, получим ссылки, напоминающие по виду кнопки (рис. 3.4).

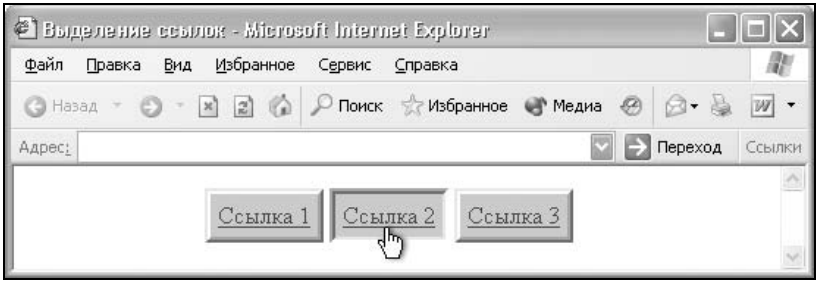


Рис. 3.4. Изменение стиля рамки при наведении на ссылку курсора

Чтобы выделить ссылку, когда на нее наводится курсор, нужно поменять стиль рамки, опять же с помощью атрибута `border-style`; в этом случае ис-

пользуется значение inset, эффект которого напоминает нажатую кнопку (листинг 3.16).

Листинг 3.16. Модификация обрамления ссылки

```
<html>
<head>
<style type="text/css">
 A {
  border-style: outset;       /* Исходный стиль рамки вокруг ссылки */
  background: #ccc;           /* Цвет фона под ссылкой */
  padding: 2px                /* Поля вокруг текста */
}
 A:hover {
  border-style: inset         /* Новый стиль рамки */
}
</style>
</head>
<body>
  <a href=link1.html>Ссылка 1</a>
  <a href=link2.html>Ссылка 2</a>
  <a href=link3.html>Ссылка 3</a>
</body>
</html>
```

Добавление волнистой линии под ссылкой

Можно создать свой собственный вид подчеркивания ссылок, используя для этого графическое изображение. Рисунок подойдет не всякий, он должен обладать рядом свойств:

- он должен быть бесшовным, иными словами, состыковываться сам с собой по горизонтали, это необходимо для получения плавной неразрывной линии;
- высота линии не должна превышать 3 пиксела, иначе часть ее может оказаться "обрезанной".

Высоту линии можно сделать и больше, чем 3 пиксела, если добавить к селектору A параметр padding или padding-bottom.

Пример изображения, удовлетворяющего приведенным условиям, показан на рис. 3.5.

После создания изображения в графическом редакторе следует задать стиль, как показано в листинге 3.17. Основным элементом для создания линии выступает параметр background. Это универсальный атрибут, который опре-

деляет характеристики фонового изображения, в частности путь к графическому файлу, а также смещение фона относительно левого верхнего угла элемента. В данном случае требуется только сдвинуть фоновый рисунок вниз на высоту текста, для этого используется относительная единица `em`, равная высоте шрифта элемента. Кроме того, чтобы рисунок повторялся только по горизонтали, добавляем к параметру `background` аргумент `repeat-x`.

Рис. 3.5. Изображение для создания волнистой линии

Листинг 3.17. Добавление волнистой линии к ссылкам

```
<html>
<head>
<style type="text/css">
 A {
  text-decoration: none;      /* Убираем подчеркивание у ссылок */
  padding: 2px;               /* Поля вокруг текста */
  white-space: nowrap         /* Отменяем переносы строк в ссылках */
}
 A:hover {
  background:
       url(/images/line.gif)         /* Путь к файлу с рисунком */
       0 1.1em                       /* Сдвигаем рисунок вниз под текст */
       repeat-x               /* Повторяем фон только по горизонтали */
}
</style>
</head>
<body>
  <p><a href=link1.html>Ссылка 1</a>
  <p><a href=link2.html>Ссылка 2</a>
  <p><a href=link3.html>Ссылка 3</a>
</body>
</html>
```

Если текст ссылки достаточно длинный и занимает две и более строки, то рисунок отображается только под первой строкой. Чтобы ликвидировать эту особенность, для селектора `A` введен атрибут `white-space` со значением `nowrap`, он отменяет переносы в тексте и отображает его одной строкой. Учтите, что при этом возможно появление горизонтальной полосы прокрутки.

Это, пожалуй, единственный недостаток данного способа, который следует принимать во внимание.

Путь к изображению указывается с помощью ключевого слова `url`, после которого в скобках пишется адрес рисунка. Число `0` в примере означает смещение изображения по горизонтали, а `1.1em` — вниз по вертикали. Вы можете незначительно менять это число в ту или иную сторону, подбирая наилучший вариант в зависимости от используемого рисунка. Окончательный результат использования волнистой линии в ссылках показан на рис. 3.6.

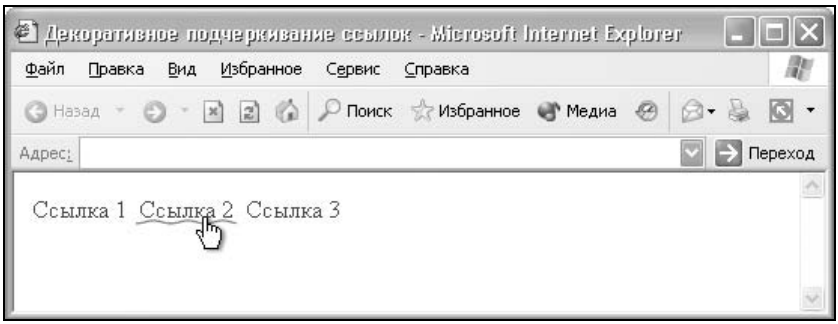


Рис. 3.6. Декоративное подчеркивание ссылок

В данном примере у ссылок убирается подчеркивание (`text-decoration: none`), чтобы оно не мешало восприятию. К тому же классическая линия под текстом плохо сочетается с линией декоративной.

Замечание

Приведенный способ не работает в браузере Opera версии 6 и ниже, поскольку он не применяет свойство `background` к тегу `A`. В браузере Internet Explorer 5 и ниже для ссылок не устанавливаются поля, задаваемые атрибутом `padding`. По этой причине смещение линии вниз следует уменьшить, например, до `0.9em`. В любом случае это значение индивидуально и связано с используемым рисунком.

Интересный эффект можно получить, если использовать в качестве линии анимированное изображение, а не статичную картинку. Код при этом останется прежним, поменяется только исходный рисунок.

Рисунки возле внешних ссылок

Внешней называется такая ссылка, которая указывает и ведет на другой сайт. К сожалению, подобная ссылка никак не отличается от локальных ссылок внутри сайта. Определить, куда она ведет, можно только посмотрев строку

состояния браузера. Но в эту строку заглядывают не все и не всегда. Чтобы пользователи отличали внешние ссылки от обычных, их следует выделять каким-либо способом. Например, поставить возле ссылки маленький рисунок, который показывает, что статус ссылки иной (рис. 3.7).

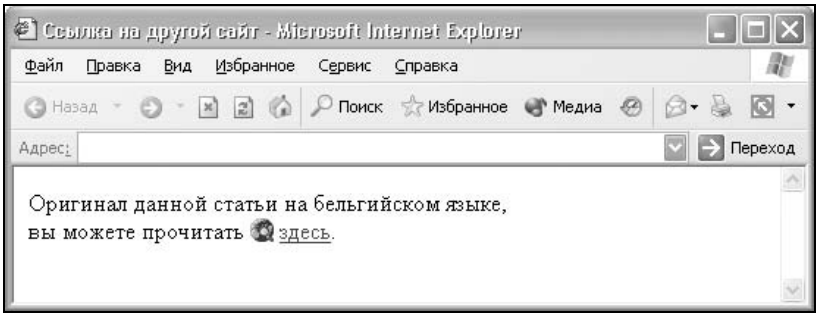


Рис. 3.7. Выделение ссылки с помощью рисунка

Использование рисунков возле внешних ссылок является пока мало распространенной практикой. Но данный способ хорош тем, что оформленная таким образом ссылка однозначно отличается от обычных ссылок внутри сайта, а правильно подобранный рисунок говорит посетителю, что ссылка ведет на другой сайт.

Чтобы автоматизировать процесс добавления рисунка, воспользуемся свойством `display: list-item`. Этот атрибут превращает текст в элемент списка, и теперь к нему можно применять атрибуты, которые предназначены для списков, в том числе и `list-style-image`, устанавливающий пользовательский рисунок в качестве маркера (листинг 3.18).

Листинг 3.18. Ссылки с рисунками

```
<html>
<head>
<style type="text/css">
 A.outer {
  list-style-image: url(/images/check.gif);  /* Путь к файлу
                                              с рисунком */
  display: list-item          /* Устанавливаем ссылку как элемент
                              списка */
 }
</style>
</head>
<body>
  <p><a href=link1.html class=outer>Ссылка с маркером</a>
```

```
  <p><a href=link2.html>Ссылка без маркера</a>
</body>
</html>
```

В примере создается новый класс `outer`, добавление которого к тегу `A` автоматически устанавливает перед ссылкой рисунок. Заметим, что этот метод работает только в браузере Opera, начиная с шестой версии, остальные браузеры его не поддерживают.

Замечание

При использовании свойства `display: list-item` для тега `A` всегда происходит перенос строки до и после ссылки. Иными словами, ссылка отображается только одна в строке. По этой причине ссылку с маркером нельзя вставлять посредине текста.

Ссылки на новое окно

Одни и те же ссылки на сайте могут отличаться друг от друга по своему назначению. Так, ссылка может открываться в новом окне, вести на другой сайт или указывать на якорь на текущей странице. В любом случае, определить, к какому типу относится ссылка, можно только нажав на ссылку, посмотрев исходный код веб-страницы или заглянув в строку состояния браузера при наведении курсора мыши на ссылку. Казалось бы, ссылки разного типа надо и выделять по-разному. Далее перечислены возможные способы оформления внешних ссылок:

1. Цвет.
2. Начертание (жирный, курсивный).
3. Шрифт.
4. Рисунок перед ссылкой.
5. Дополнительная информация.

Способы с 1 по 3 являются неудачными и приведены только для примера. В самом деле, если ссылка выглядит по-другому и отличается от обычных ссылок цветом, насыщенностью или другими параметрами, то это совсем не значит, что она ведет на другой сайт. Скорее пользователи решат, что авторы хотели акцентировать внимание на тексте и таким способом выделили его.

Тем не менее для удобства управления ссылками приведем способ, как изменять стиль ссылки с помощью внутреннего параметра тега `A`. Для этой цели применяется конструкция `A[параметр]` внутри контейнера `STYLE`, где параметр относится к тегу `A`, например, как показано в листинге 3.19.

Листинг 3.19. Выделение ссылок, открывающихся в новом окне

```
<html>
<head>
<style type="text/css">
  A[target=_blank] { font-weight: bold }
</style>
</head>
<body>
  <p><a href=link1.html target=_blank>Ссылка откроется в новом окне</a>
  <p><a href=link2.html>Ссылка откроется в текущем окне</a>
</body>
</html>
```

В данном примере показано, как изменять стиль тега A, для которого добавлен параметр target=_blank, заставляющий ссылку открываться в новом окне.

Замечание

Конструкция A[...] поддерживается браузерами Opera, Netscape, Mozilla, Firefox и не работает в браузере Internet Explorer.

Ссылки во фреймах

Фреймы разделяют окно браузера на отдельные области, расположенные вплотную друг к другу. В каждую из таких областей загружается самостоятельная веб-страница, определяемая тегом FRAME. С помощью фреймов веб-страница делится на два или более документа, которые обычно содержат навигацию по сайту и его содержимое (контент). Механизм фреймов позволяет открывать документ в одном фрейме по ссылке, нажатой в совершенно другом фрейме. Такой подход позволяет продолжать работать с содержимым одного фрейма, в то время как происходит загрузка файла в другой.

Для открытия документа в требуемом фрейме применяется параметр target тега A. В качестве значения target указывается имя фрейма, заданное параметром name тега FRAME.

Предположим, у нас есть документ, состоящий из двух фреймов, названных menu и content, как изображено на рис. 3.8.

Код для создания подобной структуры приведен в листинге 3.20.

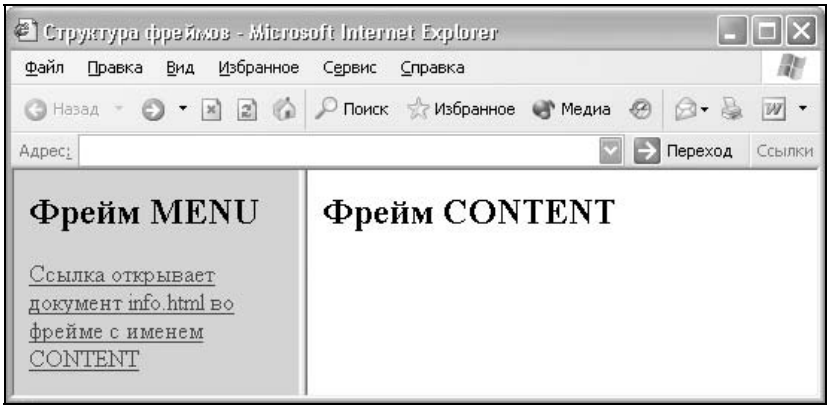


Рис. 3.8. Типичная структура фреймов

Листинг 3.20. Фреймы и их названия

```
<html>
<frameset cols="20%,*">
  <frame src=menu.html name=MENU>
  <frame src=main.html name=CONTENT>
</frameset>
</html>
```

Обычно в левом фрейме находится список ссылок на разделы сайта, а в правом — отображается контент. Чтобы веб-страница открывалась в определенном фреймовом окне, следует использовать код, как показано в листинге 3.21.

Листинг 3.21. Открытие документа в желаемом фрейме

```
<html>
<body>
<a href=info.html target=CONTENT>Ссылка открывает документ info.html
во фрейме с именем CONTENT</a>
</body>
</html>
```

Обратите внимание, что значение атрибута `target` чувствительно к регистру и его следует писать так же, как оно указано в параметре `name`, в данном случае заглавными буквами. Если параметр ссылки `target=CONTENT` будет опущен, документ откроется в текущем фрейме, где находится сама ссылка.

Чтобы одновременно обновить сразу два фрейма и загрузить в них разные документы, придется воспользоваться *скриптом*. Так традиционно называют программу, которая внедряется в тело веб-страницы и выполняет на ней определенные действия. Здесь мы ограничимся рассмотрением скриптов, написанных на языке JavaScript, как наиболее популярном для создания динамических страниц, выполняемых на стороне клиента. В дальнейшем под скриптом мы будем подразумевать скрипт, написанный на языке JavaScript.

Для одновременного обновления двух фреймов по ссылке воспользуемся событием `onClick`, которое отслеживает нажатие на ссылку, и семейством `frames`, которое позволяет обращаться к заданному фрейму в их иерархии. Загрузка требуемого файла происходит с помощью метода `location` (листинг 3.22).

Листинг 3.22. Одновременная загрузка документов в два фрейма

```
<html>
<body>
<a href=menu2.html
onClick="parent.frames.CONTENT.document.location='info.html'">Нажми на
меня, нажми</a>
</body>
</html>
```

В данном примере ссылка используется как обычно, без параметра `target`, а в текущем фрейме откроется документ с именем menu2.html. Одновременно с ним во фрейм, названный `CONTENT`, загрузится файл info.html. Как и в случае с `target`, указывать имя фрейма следует в том же написании, что задается параметром `name`. JavaScript чувствителен к регистру и к любому неправильному написанию.

ГЛАВА 4

Списки

Списком называется взаимосвязанный набор отдельных фраз или предложений, которые начинаются с маркера или цифры. Списки предоставляют возможность упорядочить и систематизировать разные данные и представить их в наглядном и удобном для пользователя виде.

Создание списка

Любой список представляет собой контейнер `UL`, который устанавливает маркированный список, или `OL`, который определяет список нумерованный. Каждый элемент списка должен начинаться с тега `LI`. Если к тегу `UL` или `OL` применяется таблица стилей, то элементы `LI` наследуют эти свойства.

Далее описаны параметры тегов `UL`, `OL` и `LI`, которые позволяют управлять видом списков и нумерацией.

TYPE

Этот параметр устанавливает вид маркера. В зависимости от того, к какому тегу он применяется, модифицируется и набор значений. В табл. 4.1 приведены возможные варианты списков, которые получаются с помощью `type`. Также приводится аналог, позволяющий получить тот же результат с помощью стилей.

Таблица 4.1. *Пример создания списков разных типов*

Тип списка	Код HTML	Пример
Маркированный список с маркерами в виде закрашенного кружка	`<ul type=disc>` `<ul style="list-style: disc">`	• Чебурашка • Крокодил Гена • Шапокляк

Таблица 4.1 *(окончание)*

Тип списка	Код HTML	Пример
Маркированный список с маркерами в виде незакрашенного кружка	`<ul type=circle>` `<ul style="list-style: circle">`	o Чебурашка o Крокодил Гена o Шапокляк
Маркированный список с квадратными маркерами	`<ul type=square>` `<ul style="list-style: square">`	▪ Чебурашка ▪ Крокодил Гена ▪ Шапокляк
Нумерованный список с прописными буквами латинского алфавита	`<ol type=A>` `<ol style="list-style: upper-alpha">`	A. Чебурашка B. Крокодил Гена C. Шапокляк
Нумерованный список со строчными буквами латинского алфавита	`<ol type=a>` `<ol style="list-style: lower-alpha">`	a. Чебурашка b. Крокодил Гена c. Шапокляк
Нумерованный список с римскими цифрами в верхнем регистре	`<ol type=I>` `<ol style="list-style: upper-roman">`	I. Чебурашка II. Крокодил Гена III. Шапокляк
Нумерованный список с римскими цифрами в нижнем регистре	`<ol type=i>` `<ol style="list-style: lower-roman">`	i. Чебурашка ii. Крокодил Гена iii. Шапокляк
Нумерованный список с арабскими цифрами	`<ol type=1>` `<ol style="list-style: decimal">`	1. Чебурашка 2. Крокодил Гена 3. Шапокляк

Параметр `type` может также применяться и к тегу `LI` для того, чтобы менять вид маркера в пределах одного списка.

START

Параметр `start` устанавливает номер, с которого будет начинаться нумерованный список, задаваемый тегом `OL`. При этом не имеет значения, какой тип списка установлен с помощью параметра `type`, аргумент `start` одинаково работает и с римскими, и с арабскими цифрами.

VALUE

Аналогично параметру `start`, параметр `value` устанавливает номер, с которого будет начинаться список, но он применяется только к тегу `LI`. Атрибут `value` применяется для нумерованных списков, когда тег `LI` находится внутри контейнера `OL`.

Пример использования списков показан в листинге 4.1.

Листинг 4.1. Создание различных списков

```
<html>
<body>
Список нумерованный
<ol start=2>
  <li>Второй</li>
  <li>Третий</li>
  <li>Четвертый</li>
</ol>

Список маркированный
<ul type=circle>
  <li>Чебурашка</li>
  <li>Крокодил Гена</li>
  <li>Шапокляк</li>
</ul>
</body>
</html>
```

В примере показано создание нумерованного и маркированного списка с разными параметрами тега `OL` и `UL`.

Маркированные списки

Маркированные списки позволяют разбить большой текст на отдельные блоки. Это привлекает внимание читателя к тексту и повышает его читабельность. С учетом того, что текст воспринимается с экрана монитора хуже, чем с печатного листа, это является весьма полезным приемом.

Для создания маркированного списка используются теги `UL` и `LI` (листинг 4.2).

Листинг 4.2. Создание маркированного списка

```
<html>
<body>
<hr>
<b>Содержание</b>
 <ul>
  <li>Макет с двумя колонками</li>
  <li>Наложение слоев</li>
```

```
  <li>Макет с тремя колонками</li>
 </ul>
<hr>
</body>
</html>
```

С тегом UL связаны следующие особенности:

- в том месте, где встречается тег UL, браузер автоматически добавляет перенос строки, в данном примере по этой причине после текста "Содержание" не стоит тег BR;
- у маркированного текста появляются отступы сверху и снизу;
- маркеры по умолчанию отображаются в виде закрашенного кружка, хотя в браузере Netscape форма маркера скорее напоминает ромб;
- каждый элемент списка сдвигается вправо по отношению к основному тексту.

На рис. 4.1 показан результат данного примера, иллюстрирующий приведенные особенности маркированного списка.

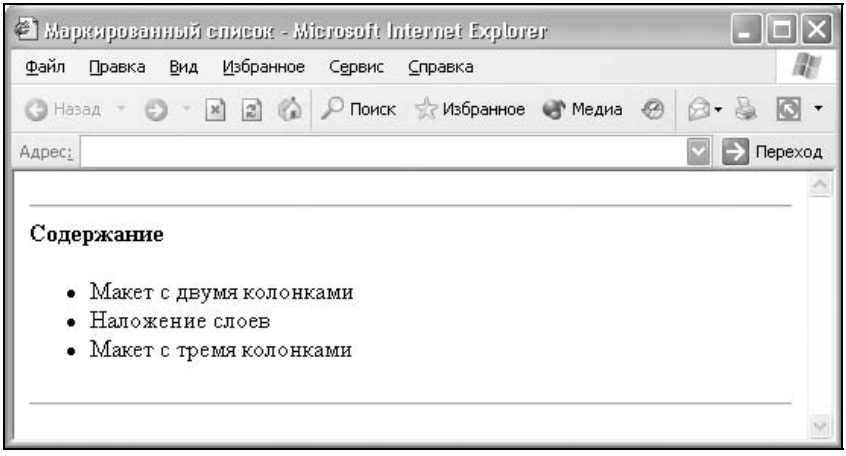


Рис. 4.1. Вид маркированного списка

Изменение вертикальных отступов в списке

Чтобы избавиться от вертикальных отступов до и после маркированного списка, его можно создать вообще без помощи тега UL. В этом случае применяется только тег LI, который создает маркеры перед текстом (листинг 4.3). Кроме того, отсутствует и сдвиг текста от левого края.

Листинг 4.3. Создание маркированного списка без отступов

```
<html>
<body>
<b>Содержание</b>
<hr>
  <li>Математика</li>
  <li>Физика</li>
  <li>Механика</li>
<hr>
</body>
</html>
```

Более удобные и расширенные возможности для управления разными отступами предлагает CSS. Для изменения вертикального отступа от верхнего края списка можно использовать свойство `margin-top`, применяя его к тегу `UL`. Атрибут `margin-bottom` изменяет отступы от нижнего края списка, а `margin-left` от левого края (листинг 4.4).

Листинг 4.4. Изменение отступов в списке

```
<html>
<head>
<style type="text/css">
 UL {
  margin-top: 0.5em;          /* Отступ от верхнего края */
  margin-bottom: 0.5em;       /* Отступ от нижнего края */
  margin-left: 40px           /* Отступ от левого края */
}
</style>
</head>
<body>
<b>Слои</b>
<hr>
 <ul>
  <li>Создание</li>
  <li>Особенности</li>
  <li>Применение</li>
 </ul>
<hr>
</body>
</html>
```

Как показано в данном примере, для указания значений отступов допускается использовать относительные единицы типа `em` или абсолютные, например пикселы (`px`).

Отступы от маркера до текста

Модификация списка также включает изменение расстояния между маркером и текстом, который за ним следует. Для этого существует несколько способов. Самый простой и незатейливый — поставить после тега `LI` один или несколько неразрывных пробелов ` `, как показано в листинге 4.5.

Листинг 4.5. Изменение отступа с помощью неразрывного пробела

```
<html>
<body>
 <ul>
  <li>  Сдвинутый текст</li>
 </ul>
</body>
</html>
```

Опять же, применение стилей позволяет более точно и удобно менять отступ. Кроме того, определяя стиль в одном месте, получаем в итоге единый вид оформления всех списков и текстов.

Отступ от маркера до текста управляется параметром `padding-left`, который применяется к селектору `LI`, как показано в листинге 4.6.

Листинг 4.6. Изменение отступа с помощью стилей

```
<html>
<head>
<style type="text/css">
 LI {
  padding-left: 20px         /* Отступ от маркера до текста */
}
</style>
</head>
<body>
<b>Списки</b>
<hr>
 <ul>
  <li>Маркированные</li>
```

```
  <li>Нумерованные</li>
  <li>Многоуровневые</li>
 </ul>
<hr>
</body>
</html>
```

Как и в случае с вертикальными отступами, регулировать расстояние с помощью стилей можно путем использования как относительных, так и абсолютных единиц измерения.

Вид маркера

Маркеры могут принимать один из трех видов: закрашенный кружок (по умолчанию), незакрашенный кружок и квадрат. Для выбора типа маркера используется параметр `type=...` тега `UL`. Вместо многоточия подставляется одно из трех значений, приведенное в табл. 4.1. Использование параметра `type` тега `UL` показано в листинге 4.7.

Листинг 4.7. Изменение вида маркера с помощью параметра `type`

```
<html>
<body>
 <ul type=square>
   <li>Пример использования квадратного маркера</li>
 </ul>
</body>
</html>
```

Рассмотрим также вариант применения стилей для управления видом маркера. При этом для селектора `UL` или `LI` используется параметр `list-style` или `list-style-type` (листинг 4.8). В качестве аргументов применяются те же значения, что и для параметра `type`.

Листинг 4.8. Изменение вида маркера с помощью стилей

```
<html>
<head>
<style type="text/css">
 LI {
  list-style: square         /* Квадрат в качестве маркера */
}
```

```
</style>
</head>
<body>
<hr>
 <ul>
  <li>Шаб-Нигурах</li>
  <li>Нъярлототеп</li>
  <li>Ктулху</li>
 </ul>
<hr>
</body>
</html>
```

В примере показано создание маркированного списка, где в качестве значка маркера используется небольшой однотонный квадрат.

Хотя количество значений параметра `type` тега `UL` ограничено тремя, это не значит, что в нашем распоряжении всего три вида маркера. Существует множество спецсимволов, которые с успехом могут выступать в качестве значка маркера. "Прикрутить" их непосредственно к тегу `LI` не получится, поэтому придется действовать в обход. Для этого прячем маркеры списка с помощью стилевого свойства `list-style-type: none` и в тексте после каждого тега `LI` добавляем свой собственный символ. В листинге 4.9 в качестве такого маркера выступает символ `&raquo;`, который в браузере отображается как двойная стрелка вправо.

Листинг 4.9. Маркеры, созданные с помощью спецсимволов

```
<html>
<head>
<style type="text/css">
 LI {
  list-style-type: none;      /* Скрываем маркеры списка */
  text-indent: -1em           /* Сдвигаем элементы списка влево */
}
</style>
</head>
<body>
<hr>
 <ul>
  <li>&raquo; Пики &#8212; &spades;</li>
  <li>&raquo; Трефы &#8212; &clubs;</li>
  <li>&raquo; Бубны &#8212; &diams;</li>
```

```
  <li>&raquo; Червы &#8212; &hearts;</li>
 </ul>
<hr>
</body>
</html>
```

Поскольку использование свойства `list-style-type` со значением `none` не убирает маркеры совсем, а только скрывает их от просмотра, то список получается смещенным вправо. Чтобы избавиться от этой особенности, в примере добавляется атрибут `text-indent` с отрицательным значением. Его задача — переместить текст левее на один символ.

Кроме символа `&raquo;`, выступающего в качестве маркера, в данном примере нашли применение и другие спецсимволы. Так, символ с кодом `&#8212;` устанавливает длинное тире, а остальные используются для создания привычных нам обозначений карт (рис. 4.2).

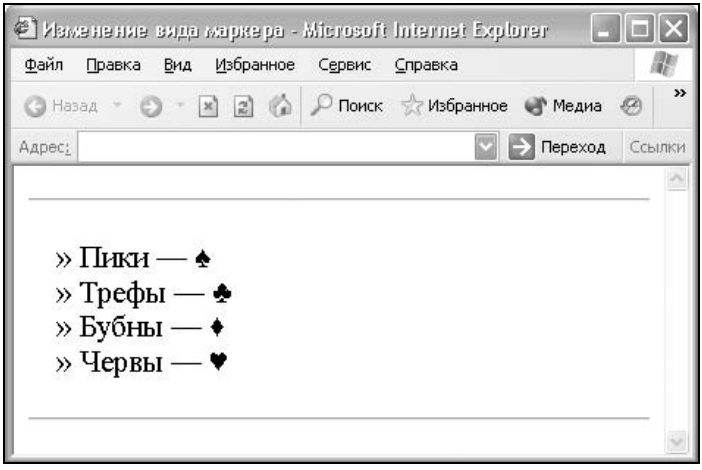


Рис. 4.2. Вид списка с использованием собственных маркеров

С помощью псевдокласса `before` можно автоматизировать процесс добавления своих маркеров к элементам списка. В задачу этого псевдокласса входит определение содержимого и его вида для элемента, к которому он применяется. В этом его дополняет свойство `content`, добавляющее требуемый текст к списку. К сожалению, использование спецсимволов, как было показано ранее, в этом случае неприемлемо и придется воспользоваться шестнадцатеричными значениями символов. В листинге 4.10 показано добавление к списку маркера в виде двойной стрелки вправо (`\BB`) и пробела после него (`\20`). Цвет значка маркера изменяется на красный через атрибут `color`.

Листинг 4.10. Использование псевдокласса `before`

```
<html>
<head>
<style type="text/css">
 LI {
  list-style-type: none;      /* Скрываем маркеры списка */
  text-indent: -1em           /* Сдвигаем элементы списка влево */
}
 LI:before {
  color: red;                 /* Маркеры красного цвета */
  content: "\BB \20"          /* Добавляем маркер в шестнадцатеричном
                              формате*/
}
</style>
</head>
<body>
<ul>
  <li>Пики &#8212; &spades;</li>
  <li>Трефы &#8212; &clubs;</li>
  <li>Бубны &#8212; &diams;</li>
  <li>Червы &#8212; &hearts;</li>
 </ul>
</body>
</html>
```

Указанный способ создания маркеров работает в браузерах Netscape 6, Mozilla, Firefox, Opera и не действует в браузере Internet Explorer.

Список с рисованными маркерами

Стили позволяют установить в качестве маркера любое подходящее изображение через свойство `list-style-image`. В качестве значения используется относительный или абсолютный путь к графическому файлу, как показано в листинге 4.11.

Листинг 4.11. Использование изображения в качестве маркера

```
<html>
<head>
<style type="text/css">
 UL {
  list-style-image: url(images/check.gif)
}
```

```
</style>
</head>
<body>
<ul>
  <li>Заголовок должен быть короче трех строк.
  <li>При названии разделов используйте уже устоявшиеся термины,
такие как гостевая книга, чат, ссылка, главная страница и другие.
  <li style="list-style-image: none">Перед использованием специального
термина или слова решите, будет ли оно понятно читателю.
<ul>
</body>
</html>
```

Рисунок лучше всего выбирать небольшого размера, чтобы не превращать элементы списка в подрисуночные подписи. Для восстановления первоначального вида маркеров отдельных пунктов списка используйте значение `none`, как показано в данном примере.

На рис. 4.3 показан результат действия примера по использованию в качестве маркеров небольших картинок.

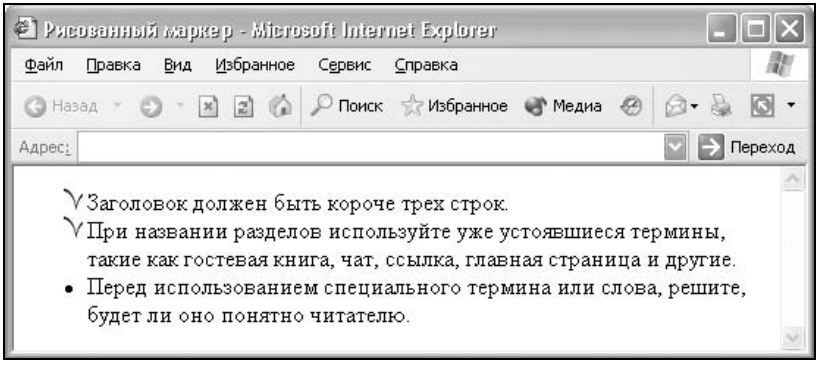


Рис. 4.3. Рисунок в качестве маркера

Маркеры разного цвета

Цвет маркеров совпадает с цветом текста на веб-странице. Однако интересный эффект в плане оформления и привлечения внимания читателя состоит в том, чтобы цвет текста списка отличался по цвету от маркеров перед ним. Из-за иерархического наследования свойств стилей элементов `UL` и `LI` напрямую установить цвет маркера не удается. Тем не менее можно пойти на хитрость и выделить текст внутри контейнера `LI` с помощью тега `SPAN`, а для него поменять цвет текста (листинг 4.12).

Листинг 4.12. Маркеры и текст списка разного цвета

```
<html>
<head>
<style type="text/css">
 LI { list-style: square;
  color: red                /* Цвет маркера */
}
 LI SPAN {
  color: black              /* Цвет текста в списке */
}
</style>
</head>
<body>
<b>Основные цвета</b>
<hr>
 <ul>
  <li><span>Красный</span></li>
  <li><span>Зеленый</span></li>
  <li><span>Синий</span></li>
 </ul>
<hr>
</body>
</html>
```

В данном примере цвет маркера списка, установленный через селектор `LI`, задан красным через свойство `color`, а для тега `SPAN` — черным. Конструкция `LI SPAN` позволяет определить стиль только для тега `SPAN`, который располагается внутри контейнера `LI`. Такой способ записи называется контекстными селекторами. Он позволяет задать стиль для тегов, которые вложены один внутрь другого. Во всех остальных случаях стиль применяться не будет.

Еще один способ использования маркеров, отличающихся по цвету от текста, — создание небольшого графического изображения и установка его в качестве маркера списка, как было показано в листинге 4.11. Однако при изменении размера шрифта в браузере ширина и высота графических маркеров остается постоянной, в то время как обычные маркеры изменяются вместе с текстом.

Положение текста и маркера

Существует два способа размещения маркера относительно текста: маркер выносится за границу элементов списка или обтекается текстом (рис. 4.4).

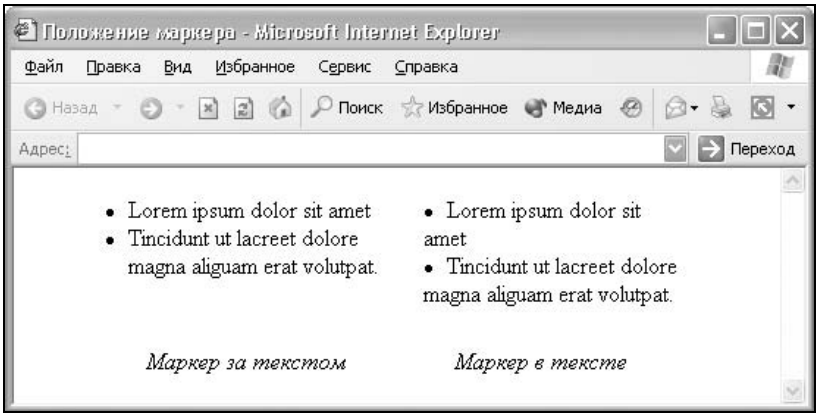


Рис. 4.4. Размещение маркера относительно текста

Чтобы управлять положением маркера, применяется свойство CSS `list-style-position`. Этот параметр имеет два значения: `outside` — маркеры размещаются за пределами текстового блока и `inside` — маркеры являются частью текстового блока и отображаются в элементе списка (листинг 4.13).

Листинг 4.13. Изменение положения маркеров

```
<html>
<head>
<style type="text/css">
 UL { list-style-position: outside }
</style>
</head>
<body>
<hr>
 <ul>
  <li>Перед началом работы проверьте наличие оборудования, входящего
      в комплект ЭВМ.</li>
  <li>При отсутствии одного или нескольких периферийных устройств
      следует сразу же обратиться к техническому персоналу ВЦ.</li>
  <li>После осмотра визуальными методами своего рабочего места можно
     осторожно включить питание ЭВМ.</li>
 </ul>
<hr>
</body>
</html>
```

При использовании аргумента `outside` в браузерах Internet Explorer и Opera замечена следующая особенность. Если добавить к тегу `UL` параметр `margin-`

left, который регулирует отступ левого края текста, с отрицательным или нулевым значением, то в некоторых случаях маркеры не будут отображаться. Это связано с тем, что отступ применяется к самому тексту без учета того, что слева от него находятся маркеры. Поэтому сдвиг может привести к тому, что маркеры списка окажутся за пределами видимости веб-страницы или таблицы, в которой располагается список.

Нумерованные списки

Нумерованные списки представляют собой набор элементов с их порядковыми номерами. Вид и тип нумерации зависит от параметров тега OL, который и применяется для создания списка. В качестве нумерующих элементов могут выступать следующие значения:

- арабские цифры (1, 2, 3, ...);
- прописные латинские буквы (A, B, C, ...);
- строчные латинские буквы (a, b, c, ...);
- римские цифры в верхнем регистре (I, II, III, ...);
- римские цифры в нижнем регистре (i, ii, iii, ...).

С практической точки зрения, принципы отображения элементов маркированного списка могут аналогичным способом применяться и к нумерованному списку. Но учитывая, что мы имеем дело с перечислением, существуют некоторые особенности, о которых и пойдет речь далее.

Нумерация списка

Допускается начинать список с любого номера; для этой цели применяется параметр start тега OL или value тега LI. В качестве значения параметра задается любое целое положительное число. При этом неважно, какой тип нумерации установлен, даже если в качестве списка используются латинские буквы. Если одновременно для списка применяются параметры start и value, то последний имеет большее преимущество, и нумерация отображается с числа, указанного аргументом value, как показано в листинге 4.14.

Листинг 4.14. Изменение нумерации списка

```
<html>
<body>
<hr>
 <ol type=I start=4>
  <li>Следует тщательно позаботиться о своем рабочем месте.</li>
  <li>Освещение в помещении отрегулировать таким образом, чтобы источник
света находился сбоку или сзади оператора.</li>
```

```
  <li value=7>Во избежание медицинских осложнений стул рекомендуется
   выбирать с мягким сидением.</li>
 </ol>
<hr>
</body>
</html>
```

Первый элемент списка в данном примере будет начинаться с римской цифры IV, поскольку указан параметр `start=4`, затем идет номер V, а последний элемент следует не по порядку и назначается номером VII.

Многоуровневые списки

Многоуровневые списки предназначены для организации сложной иерархической структуры текста, обычно таких документов, как юридические или технические. Реально на веб-странице нельзя автоматически ввести многоуровневую нумерацию типа использования подпунктов 1.1 или 2.1.3. Поэтому приходится вводить числа отдельно или упрощать отображение списка. Так в листинге 4.15 пункты и подпункты списка обозначаются числами.

Листинг 4.15. Создание многоуровневого списка

```
<html>
<body>
<hr>
 <ol>
  <li>Пункт 1</li>
  <ol>
    <li>Подпункт 1.1</li>
    <li>Подпункт 1.2</li>
  </ol>
  <li>Пункт 2</li>
  <ol>
    <li>Подпункт 2.1</li>
    <li>Подпункт 2.2</li>
  </ol>
 </ol>
<hr>
</body>
</html>
```

Результат действия данного примера приведен на рис. 4.5. Обратите внимание, что у подпунктов имеется отступ слева, но нет вертикальных отступов сверху и снизу списка.

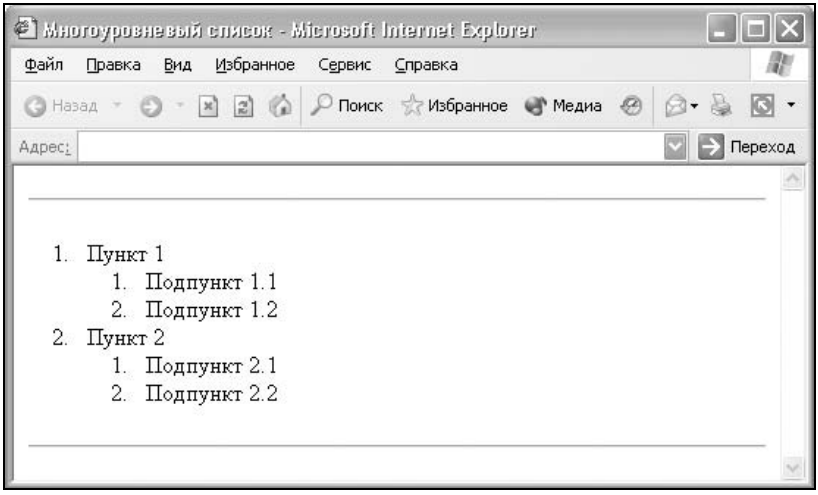


Рис. 4.5. Вид многоуровневого списка

Альтернативно можно использовать в качестве подпунктов латинские или римские буквы. Чтобы установить этот параметр, имеет смысл воспользоваться стилями для сокращения кода и повышения удобства редактирования (листинг 4.16).

Листинг 4.16. Представление многоуровневого списка

```
<html>
<head>
<style type="text/css">
 OL LI { font-weight: bold }              /* Выделение пунктов */
 OL OL { list-style: lower-alpha }        /* Тип нумерации подпунктов */
 OL OL LI { font-weight: normal }         /* Оформление подпунктов */
</style>
</head>
<body>
<hr>
 <ol>
  <li>Пункт 1</li>
  <ol>
    <li>Подпункт a</li>
    <li>Подпункт b</li>
  </ol>
  <li>Пункт 2</li>
  <ol>
    <li>Подпункт a</li>
```

```
    <li>Подпункт b</li>
  </ol>
 </ol>
<hr>
</body>
</html>
```

Применение стилей также позволяет по-разному выделять и оформлять пункты и подпункты (рис. 4.6).

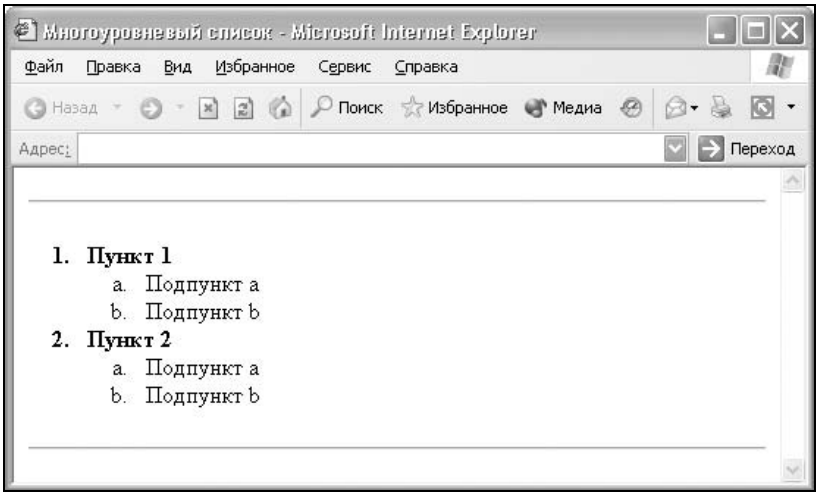


Рис. 4.6. Выделение пунктов списка

Можно ввести специальный класс и включать его в тег `OL` или `LI` или воспользоваться приведенным в примере способом, который основан на применении контекстных селекторов. Следует учитывать, что стиль селектора `OL` наследуется селектором `LI`, поэтому для пунктов и подпунктов приходится задавать разное оформление, чтобы стиль не перекочевывал с одного элемента списка на другой.

ГЛАВА 5

Линии и рамки

Линии — прекрасный декоративный элемент, который находит применение на сайте во множестве разных случаев. С помощью линий и рамок можно выделять смысловые элементы, привлекая к ним внимание читателей, отделять один материал от другого и применять их для оформления веб-страниц.

Создание линий

Для создания линий применяется несколько способов, один из них основан на использовании стилевого атрибута `border` и его производных. Этот параметр позволяет одновременно установить толщину, стиль и цвет рамки вокруг элемента. Значения могут идти в любом порядке, разделяемые пробелами; браузер сам определит, какое из них соответствует нужному атрибуту (листинг 5.1). Для установки линии на определенной стороне элемента воспользуйтесь параметрами `border-top`, `border-bottom`, `border-left`, `border-right`, которые задают линию сверху, снизу, слева и справа соответственно.

Листинг 5.1. Использование атрибута `border`

```
<html>
<body>
<div style="border: 2px solid black; background: #fc3; padding: 10px">
...
</div>
</body>
</html>
```

В данном примере вокруг прямоугольной области отображается сплошная рамка черного цвета толщиной 2 пиксела. Первый аргумент в данном

случае определяет толщину, второй — тип линии, а третий — ее цвет. Для управления видом рамки существует восемь значений, приведенных на рис. 5.1.

Рис. 5.1. Стили рамок

Замечание

Браузер Internet Explorer версии 4 и 5 не применяет свойство `border` к встроенным элементам типа тега `SPAN`.

Обычно атрибут `border` применяется к тегу `DIV`, который является блочным элементом и предназначен для выделения фрагмента документа с целью изменения вида его содержимого. Как правило, такой вид блока управляется с помощью стилей. Чтобы не описывать каждый раз стиль внутри тега, можно вынести описания стилей во внешнюю таблицу стилей, а для тега добавить параметр `class` или `id` с именем селектора. В дальнейшем, упоминая теги `DIV` или `SPAN`, к которым применяются стили, мы будем использовать термин *слой*.

Горизонтальные линии

Горизонтальные линии хорошо использовать для отделения одного блока текста от другого. Небольшой по размеру текст, сверху и снизу которого располагаются горизонтальные линии, привлекает больше внимания читателя, чем обычный текст.

Для создания линии применяется несколько способов: с помощью тега `HR`, через рисунки, таблицы и стили.

Использование тега *HR*

С помощью тега `HR` можно нарисовать горизонтальную линию, вид которой зависит от используемых параметров, а также браузера. Тег `HR` относится к блочным элементам, поэтому линия всегда начинается с новой строки, а после нее все элементы отображаются на следующей строке. Благодаря множеству параметров тега `HR` видом линии, созданной через этот тег, легко управлять. Если еще подключить мощь стилей, то добавление линии в документ превращается в простое занятие. В листинге 5.2 показано создание линий разной толщины и ширины.

Листинг 5.2. Создание линий с помощью тега HR

```
<html>
<body>
  <hr noshade color=red size=4 width=75% align=right>
  <hr noshade color=red size=2 width=60% align=right>
  <hr noshade color=red size=1 width=50% align=right>
</body>
</html>
```

Атрибут `size` определяет толщину линии в пикселах, а `width` — ее ширину в пикселах или процентах. Результат использования тега `HR` всегда выравнивается по центру окна браузера, и чтобы изменить положение линии, применяется параметр `align` со значением `left` или `right`, как показано в данном примере.

По умолчанию линия отрисовывается трехмерной, что не всегда соответствует дизайну сайта. Поэтому когда требуется создать сплошную линию, к тегу `HR` следует добавить параметр `noshade`. В то же время использование атрибута `color` запрещает трехмерные эффекты, как если бы был добавлен параметр `noshade`. Заметим при этом, что атрибут `color` работает только в браузере Internet Explorer, а в остальных браузерах никак не влияет на способ отображения линии.

Горизонтальная линия может применяться для разделения абзацев или логических блоков. В этом случае удобнее воспользоваться стилями, чтобы все линии были однотипны, а их видом было проще управлять. Далее перечислены параметры CSS, которые регулируют характеристики линии, созданной через тег `HR`:

- `width` — определяет ширину линии и измеряется как в пикселах, так и в процентах от ширины окна браузера;
- `height` — устанавливает высоту линии, обычно измеряется в пикселах;
- `text-align` — выравнивание линии по краю;
- `color` — изменяет цвет линии в браузере Internet Explorer;
- `background-color` — задает цвет линии в браузерах Opera, Netscape, Mozilla, Firefox.

Как видно из описания, цвет линии меняется для различных браузеров. К тому же Opera добавляет темную рамку вокруг линии. От нее можно избавиться, если добавить атрибут `border: 0px solid #ff0000`, где последний аргумент совпадает с цветом линии. Хотя толщина в данном случае нулевая, тем не менее стоит указать ее цвет. Как это не удивительно, но именно та-

кая конструкция и позволяет получить желаемый результат. В конечном виде код для создания линии приведен в листинге 5.3.

Листинг 5.3. Стиль линии для различных браузеров

```
<html>
<head>
<style type="text/css">
 BODY {
  text-align: center  /* Выравниваем текст на веб-странице по центру */
}
 HR {
  border: 0px solid red; /* Убираем ненужную рамку для браузера Opera */
  background-color: red; /* Цвет линии для браузера Opera и Netscape */
  color: red;                /* Цвет линии для браузера IE */
  height: 3px;               /* Высота линии */
  width: 300px;              /* Ширина линии */
  text-align: center         /* Выравнивание по центру */
}
</style>
</head>
<body>
  Пол и Пропилен
<hr>
  Наносят ответный удар
</body>
</html>
```

Таким образом, для создания сплошной линии приходится использовать сразу два параметра: `color` и `background-color`. Один из них предназначен для браузера Internet Explorer, а второй для Netscape и Opera. Кроме того, Opera добавляет вокруг линии темную рамку, которую можно убрать с помощью свойства `border`. В итоге линии в разных браузерах получаются не совсем идентичными по своему виду, но тем не менее близкими, что и требовалось получить.

Применение рисунков

В качестве горизонтальных линий можно использовать любой подходящий рисунок, растянутый до нужной ширины. На рис. 5.2 показана нарисованная буква H, ширина которой задана в размере 100 % от ширины окна браузера.

Обратите внимание, что при таком растягивании рисунка в нем однозначно появляются искажения. Следовательно, чем абстрактнее рисунок, тем луч-

ше: никто не поймет, что на нем изображено. В качестве единиц измерения можно устанавливать не только проценты, но и пикселы, как показано в листинге 5.4.

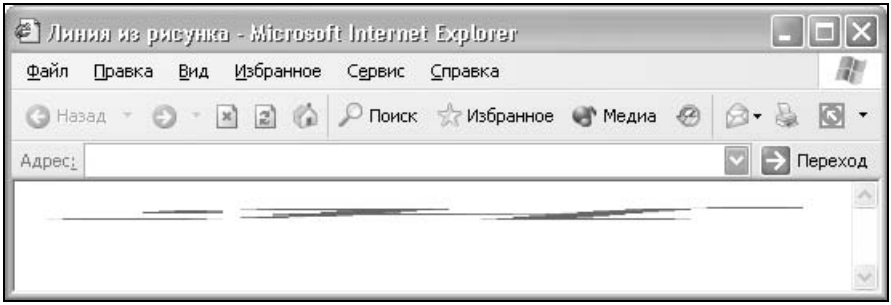


Рис. 5.2. Использование рисунка для создания линии

Листинг 5.4. Рисунок в качестве горизонтальной линии

```
<html>
<body>
Пример линии, сделанной из рисунка 100% ширины.<br>
<img src=line.gif width=100% height=10><br>

Пример линии, сделанной из рисунка фиксированной ширины 400 пикселов.<br>
<img src=line.gif width=400 height=10>
</body>
</html>
```

Высота рисунка также устанавливается произвольной, обычно в пикселах.

Использование таблиц

Что делать, когда требуется не сплошная линия, а декоративная? Здесь на помощь придут таблицы. Вначале готовим подходящий рисунок в графическом редакторе, при этом он обязательно должен состыковываться сам с собой по горизонтали. Затем создаем таблицу, состоящую из одной ячейки, и вставляем в нее наш фоновый рисунок. Хитрость состоит в том, чтобы в ячейке кроме фона был еще один рисунок, иначе в браузере Netscape 4.*x* мы просто ничего не увидим. В качестве такого рисунка традиционно используется прозрачный GIF-файл размером 1 × 1 пиксел. В листинге 5.5 он называется spacer.gif. Следует отметить, что старшие версии браузера Netscape не требуют использования *распорки* (так обычно называется подобный рисунок).

Листинг 5.5. Создание горизонтальных линий с помощью таблиц

```
<html>
<body>
<table border=0 background=dot.gif width=100% height=2 cellspacing=0
cellpadding=0>
 <tr>
  <td><img src=spacer.gif width=1 height=2></td>
 </tr>
</table>
</body>
</html>
```

Вставка фонового рисунка в таблицу происходит через параметр `background` тега `TD`, где в качестве аргумента используется имя файла.

Применение стилей

Еще один прием создания горизонтальных линий основан на использовании стилей. Причем можно добавлять как декоративные линии, основанные на рисунках, так и применять встроенные в стиль типы. Например, вместо таблицы, приведенной в предыдущем примере, можно воспользоваться тегом `DIV`, предварительно описав его стиль, как показано в листинге 5.6.

Листинг 5.6. Декоративная линия, созданная с помощью стилей

```
<html>
<head>
<style type="text/css">
 #line {
  width: 100%;                        /* Ширина линии */
  background-image: url(dot.gif);     /* Рисунок для линии */
  background-repeat: repeat-x;        /* Повторять фон по горизонтали */
  margin-top: 10px;                   /* Отступ сверху */
  margin-bottom: 10px;                /* Отступ снизу */
}
</style>
</head>
<body>
  <div id=line> </div>
</body>
</html>
```

Отступы сверху и снизу линии нужны лишь для удобства, и их всегда можно убрать при необходимости. Код стиля можно сократить, воспользовавшись универсальными параметрами CSS. Так, вместо атрибута `background-repeat`, который устанавливает способ повторения фона, можно использовать `background`, который позволяет сразу описать и путь к фоновому изображению, и способ его повтора через пробел, а `margin` — отступы сразу со всех сторон, как показано далее.

```
background: url(dot.gif) repeat-x
margin: 10px 0px 10px 2px
```

В контейнер `DIV` обязательно следует поместить неразделяемый пробел ` ` или невидимый рисунок. Дело в том, что браузеры, кроме Internet Explorer, не отображают фоновый рисунок у "пустых" тегов.

Так получаются линии из рисунков, а вот чтобы обойтись без них, можно использовать параметр `border-top`, который задает линию сверху блочного тега, или `border-bottom`, который создает линию снизу от блока. Это универсальные свойства, они одновременно определяют толщину, стиль и цвет линии, которые описываются через пробел (листинг 5.7).

Листинг 5.7. Горизонтальная линия снизу от текста

```
<html>
<body>
 <p style="border-bottom: 1px solid #800000; padding-bottom: 7px">
  "Дима, у тебя есть что-нибудь похожее на транплюкатор?".<br>
  "Что значит похожее?!" - восклицает в ответ Дима. - "У меня есть самый
настоящий транплюкатор!"<br>
  С этими словами лезет на балкон, долго громыхает там железяками и
достает оттуда такую штуку, при виде которой сразу понимаешь - Он.
 </p>
</body>
</html>
```

Первый параметр определяет толщину линии в пикселах, второй — стиль линии, возможные значения и их вид приведены на рис. 5.1, а последний параметр задает цвет линии, в данном случае — темно-красный.

Замечание

К блочным тегам относятся таблица (тег `TABLE`), параграф (тег `P`), слой (тег `DIV`), линия (тег `HR`), заголовки (теги `H1`, ..., `H6`) и некоторые другие менее распространенные теги.

Результат выполнения примера показан на рис. 5.3.

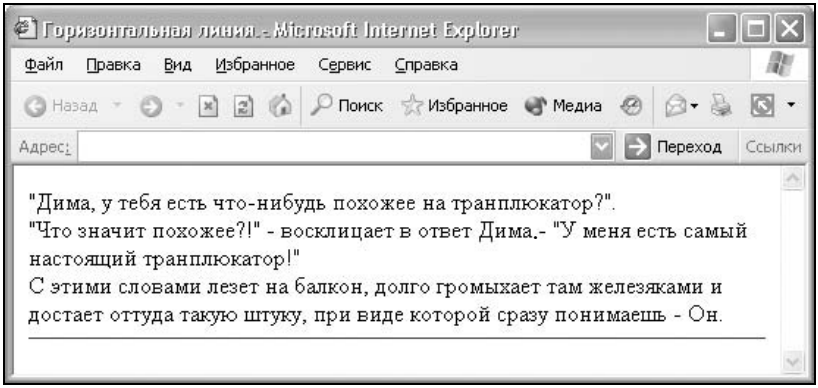


Рис. 5.3. Горизонтальная линия под текстом

Чтобы не связывать линию с текстом, можно воспользоваться способом, который описан в листинге 5.6. А именно пишем код `<div id=line> </div>`, где `line` служит селектором со следующим стилем (листинг 5.8).

Листинг 5.8. Горизонтальная линия, созданная с помощью стилей

```
<html>
<head>
<style type="text/css">
 #line {
  width: 400px;                    /* Ширина линии */
  border-top: 1px solid red;       /* Вид линии */
  position: relative;
  left: 50%;
  margin-left: -200px
}
</style>
</head>
<body>
  <div id=line align=center> </div>
</body>
</html>
```

Последние три стилевых параметра у селектора `line` предназначены для выравнивания линии по центру. Более подробно об этом читайте в *главе 8*, посвященной выравниванию элементов.

Линия с рисунком

Добавление небольших изображений к горизонтальным или вертикальным линиям расширяет возможности по их оформлению, а также делает такие линии уникальными по своему виду. Так, на рис. 5.4 приведена пунктирная линия с текстом и рисунком ножниц.

Рис. 5.4. Вид линии с изображением

Имеется несколько способов совмещения линий разного типа и рисунка, например через фоновое изображение или позиционирование.

Классический метод предполагает, что линия вначале рисуется в графическом редакторе, а затем вставляется как повторяющееся фоновое изображение. Для этого случая требуется подготовить две картинки (рис. 5.5). Одна из них содержит изображение ножниц, а вторая — фрагмент линии. Рамка вокруг линии, естественно, не рисуется, она приведена лишь для примера, чтобы показать размеры рисунка. Высота картинок должна быть одинакова, а ширина может различаться.

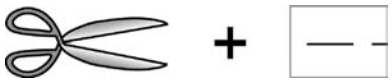


Рис. 5.5. Изображения, необходимые для создания линии на рис. 5.4

Далее создаем новый стилевой класс с именем `line`, где указываем параметр `background`, он задает фоновый рисунок горизонтальной линии для `DIV`. Сам рисунок добавляется через тег `IMG`; но чтобы он располагался по левому или правому краю, требуется поместить его внутрь контейнера `SPAN` и задать его выравнивание через свойство `float`. Значение `left` определяет рисунок на линии слева, как показано на рис. 5.4, а `right` — справа.

Рядом с линией можно добавить надпись, размер и шрифт которой также задается через стили. Если текст и линия накладываются друг на друга, то следует добавить атрибут `padding-top`; его задача заключается в том, чтобы опустить надпись ниже линии. Однако он же приводит к тому, что рисунок также сдвигается от своего первоначального положения. Вернуть его на место поможет использование свойства `margin-top` с отрицательным значением в стиле тега `SPAN` (листинг 5.9).

Листинг 5.9. Использование фонового рисунка

```
<html>
<head>
```

```
<style type="text/css">
 .line {
  background:
       url(line.gif)          /* Путь к графическому файлу с линией */
       repeat-x;              /* Повторяем фон по горизонтали */
  text-align: center;         /* Выравнивание надписи по центру */
  padding-top: 10px           /* Отступ для текста сверху */
}
</style>
</head>
<body>
<div class=line><span style="float: left; margin-top: -10px; width:
52px"><img src=scissors.gif width=52 height=22></span>Линия отреза</div>
</body>
</html>
```

В браузере Opera рисунок ножниц немного смещается по горизонтали вправо. Это легко устранить, если в стиле тега `SPAN`, внутри которого находится изображение, указать ширину через свойство `width`, значение которого должно совпадать с шириной рисунка. Тот же результат получается, если к стилю `SPAN` добавить свойство `text-align` со значением `left`.

Следующий способ связан с использованием свойства `border`. Для тега `DIV` задаем стилевой атрибут `border-top`, в качестве значения которого указываются желаемая толщина, стиль и цвет линии. Тегу `SPAN`, расположенному внутри контейнера `DIV`, устанавливаем относительное позиционирование через `position: relative` и смещаем его содержимое вверх, используя атрибут `top` с отрицательным значением. В свою очередь, картинка размещается внутри тега `SPAN` и сдвигается вместе с ним (листинг 5.10).

Листинг 5.10. Позиционирование рисунка и линии

```
<html>
<head>
<style type="text/css">
 .line {
  text-align: center;                /* Выравниваем текст по центру */
  font-family: Arial, sans-serif;    /* Рубленый шрифт для надписи */
  font-size: 12px;                   /* Размер шрифта надписи */
  border-top: 1px dashed black       /* Параметры линии */
}
</style>
</head>
```

```
<body>
<br><br>
<div class=line><span style="position: relative; top: -12px; float: left;
text-align: left"><img src=scissors.gif width=52 height=22></span>Линия
отреза</div>
</body>
</html>
```

Результат применения свойства `border` и позиционирования для создания линии показан на рис. 5.6.

Рис. 5.6. Вид линии, полученной с помощью атрибута `border`

Примите во внимание, что при изменении положения слоя за счет позиционирования место, которое первоначально занимал слой, остается. Для рисунков с большой высотой это может привести к тому, что расстояние между линией и следующим за ней содержимым получится слишком большим.

Поскольку свойство `border` и его производные добавляют границу только по краям элемента, то без позиционирования рисунка никак не обойтись, иначе он будет находиться выше или ниже линии. Однако указанную особенность можно использовать в своих целях, например, как показано на рис. 5.7.

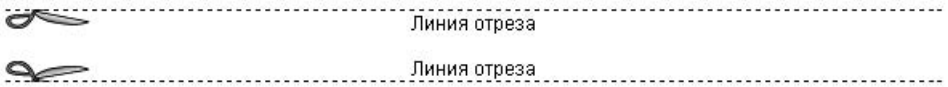


Рис. 5.7. Добавление рисунка ножниц в качестве фона

Разумеется, подойдет не всякое изображение, но для ножниц, которые часто изображают как на рис. 5.7, такой подход оправдан.

В зависимости от того, снизу или сверху от линии требуется добавить картинку, применяют атрибут `border-bottom` или `bottom-top`. Само изображение устанавливается как фоновое с помощью свойства `background`. В качестве значений этого параметра указывается путь к графическому файлу, а также его положение относительно левого верхнего угла блока. Первое число указывает координату по горизонтали, а второе — по вертикали (листинг 5.11). Значение зависит от изображения и высоты рисунка и подбирается в каждом случае индивидуально.

Листинг 5.11. Отображение половины рисунка возле линии

```
<html>
<head>
<style type="text/css">
 .line1, .line2 {
  text-align: center;        /* Выравниваем надпись по центру */
  font-family: sans-serif;   /* Рубленый шрифт для надписи */
  font-size: 12px;           /* Размер шрифта надписи */
  height: 13px               /* Высота отображения рисунка с ножницами */
}
 .line1 {
  background:
       url(scissors.gif)     /* Путь к файлу с рисунком ножниц */
       no-repeat             /* Отображаем фон без повторений */
       0px -11px;            /* Смещаем рисунок с ножницами вверх */
  border-top: 1px dashed black      /* Добавляем линию сверху ножниц */
}
 .line2 {
  background: url(scissors.gif)
       no-repeat
       0px 3px;              /* Сдвигаем рисунок с ножницами вниз */
  border-bottom: 1px dashed black   /* Добавляем линию сверху ножниц */
}
</style>
</head>
<body>
  <p>Подпись снизу от линии.</p>
  <div class=line1>Линия отреза</div>

  <p>Подпись сверху от линии.</p>
  <div class=line2>Линия отреза</div>
</body>
</html>
```

В примере используется группирование селекторов, когда их имена в стиле перечисляются через запятую. Группирование предназначено для присвоения общих совпадающих параметров сразу для нескольких элементов. При этом сокращается объем кода, а текст со стилями становится более читабельным, особенно при большом количестве различных селекторов.

Вертикальные линии

Помещенная возле текста вертикальная линия привлекает к нему внимание читателя, позволяя тем самым выделять нужные абзацы или блоки текста. Вертикальные линии могут также служить для разделения колонок текста, чтобы взгляд читателя не перескакивал с одной колонки на другую.

Для создания подобных линий применяются таблицы и стили. Далее рассматриваются оба способа.

Создание вертикальных линий через таблицы

Благодаря тому, что ячейку таблицы можно залить любым подходящим цветом, таблицы давно применяются для создания различных линий. Вначале создаем таблицу, одна из ячеек которой имеет ширину 1—2 пиксела. Параметры `cellpadding` и `border` должны быть равны нулю, а `cellspacing` определяет расстояние от текста до линии. Затем в этой ячейке указываем фон нужного цвета: `<td width=1 bgcolor=#ff0000>`, где `width` в данном случае задает ширину линии, а `bgcolor` — ее цвет (листинг 5.12).

Листинг 5.12. Линия, созданная с помощью таблицы

```
<html>
<body>
<table width=600 cellpadding=0 cellspacing=12 border=0 align=center>
<tr>
<td width=1 bgcolor=#ff0000><img src=spacer.gif width=4 height=1></td>
<td>Когда субъект деятельности по участию в конкурсе осуществляет
выполнение поставленной задачи, возникает острая необходимость передать
информацию об этом факте обыденной реальности главному субординатору,
находящемуся на вершине иерархической лестницы конкурса, и
продемонстрировать ему предмет материального мира, обретенный
в результате выполнения одной из поставленных задач.</td>
</tr>
</table>
</body>
</html>
```

Чтобы такая линия была видна в некоторых старых браузерах, в ячейку нужно что-нибудь поместить. Неразделяемый пробел ` ` для этой цели не очень подходит, он может сделать линию слишком толстой, что нам не нужно. Идеальное решение — прозрачный рисунок размером 1 × 1 пиксел, который в примере называется spacer.gif. Заметим, что в современных браузерах подобный прием использовать необязательно, линия будет отображаться указанной толщины и без дополнительных рисунков.

Благодаря использованию параметра `background` тега `TD` можно создавать декоративные вертикальные линии как слева, так и справа от текста. Опять же вначале требуется подготовить фоновый рисунок, который состыковывается сам с собой по вертикали, а затем добавить его в таблицу, как показано в листинге 5.13.

Листинг 5.13. Декоративная вертикальная линия

```
<html>
<body>
<table width=500 cellpadding=0 cellspacing=12 border=0 align=center>
<tr>
<td width=15 background=vline.gif><img src=spacer.gif width=15
height=1></td>
<td>Каждый из рефлексирующих субъектов виртуального мира может находиться
в двух основных состояниях: стационарном и мобильном, при котором они или
сводят передвижение в трехмерном Евклидовом пространстве с декартовой
системой координат к абстрактному минимуму, или ограничиваются
формальными перемещениями массы в некоторой части пространственного
континуума.</td>
<td width=15 background= vline.gif><img src=spacer.gif width=15
height=1></td>
</tr>
</table>
</body>
</html>
```

Результат создания декоративной линии представлен на рис. 5.8.

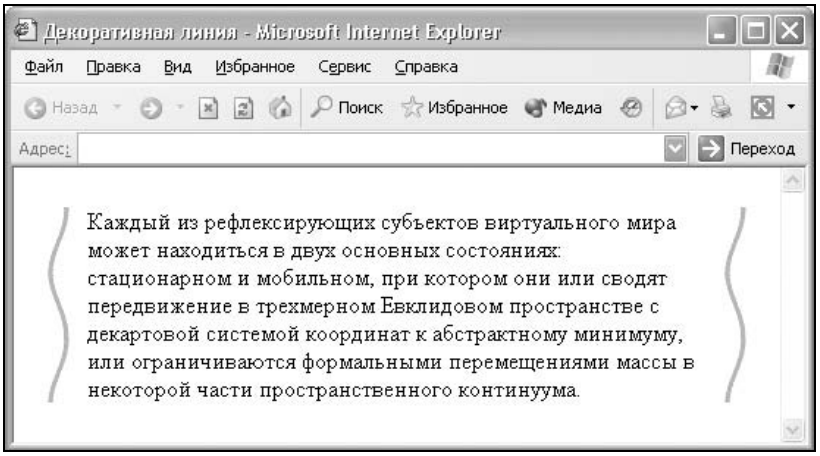


Рис. 5.8. Декоративная вертикальная линия

Использование таблиц для создания вертикальных линий в некоторых случаях доставляет неудобство. Это связано с тем, что ячейки таблицы имеют свойство менять свои размеры в определенных ситуациях, например, когда ширина одних ячеек задана в процентах, а других — в пикселах. Это может привести к тому, что фоновый рисунок будет повторяться не только по вертикали, но и по горизонтали, а это станет причиной появления второй линии. Чтобы этого не произошло, задавайте размеры у всех ячеек таблицы. Хотя в примере ширина средней ячейки не указана, текст внутри нее словно "распирает" ячейку, за счет чего устанавливаются исходные размеры, и линии отображаются корректно. Еще один способ борьбы с возможным варьированием ширины ячеек заключается в том, чтобы справа от линии оставить пустое пространство шириной 20—40 пикселов (рис. 5.9). Цвет фона рисунка линии при этом должен совпадать с фоном веб-страницы.

Рис. 5.9. Изображение для создания декоративной вертикальной линии

Рамка вокруг изображения приведена для получения представления о размерах рисунка и для использования на сайте не нужна.

Создание вертикальных линий с помощью стилей

Вертикальные линии через стили создаются ранее указанным способом, который приведен в листинге 5.8. В данном случае необходимо воспользоваться свойствами `border-left` или `border-right`; они создают линии справа или слева от блочного элемента (листинг 5.14).

Листинг 5.14. Вертикальная линия

```
<html>
<head>
<style type="text/css">
 #vline {
  border-left: 1px solid red;
  margin-left: 30px;                /* Отступ от края окна до линии */
  padding-left: 7px                 /* Поле от линии до текста */
}
</style>
</head>
```

```
<body>
<div id=vline>
История о том, как происходила борьба со злобным демоном Флюром, как
нашли место, где он должен был возродиться, победили его, и почему
из этого ничего не получилось.
</div>
</body>
</html>
```

В данном примере слева от текста добавляется вертикальная линия красного цвета. Высота такой линии привязана к блоку текста и зависит от него. С помощью параметров `padding` и `margin` можно регулировать размеры отступов от линии до текста, что позволяет изменять вид текста.

Воспользовавшись этим способом, можно создать декоративную линию из рисунка. В этом случае необходимо применять свойство `background`, которое одновременно задает адрес изображения и определяет, по какой оси будет повторяться фоновая картинка (листинг 5.15).

Листинг 5.15. Декоративная линия, созданная с помощью стилей

```
<html>
<head>
<style type="text/css">
 #vline {
  background: url(check.gif) repeat-y;      /* Фоновый рисунок */
  margin-left: 40px;                /* Отступ от края окна до линии */
  padding-left: 20px;               /* Поле от рисунка до текста */
}
</style>
</head>
<body>
<div id=vline>
История о том, зачем понадобилось искать могилу мальчика, как ее нашли,
выкопали, и что из этого получилось.
</div>
</body>
</html>
```

Аргументы параметра `background` перечисляются через пробел, в данном случае `url` указывает путь к графическому файлу, а `repeat-y` говорит браузеру, что фоновый рисунок следует повторять только вниз по вертикальной оси. Поскольку рисунок несколько шире, чем линия в один пиксел, то отступы необходимо задавать больше. В любом случае подбор параметров сле-

дует производить "вручную", исходя из размеров изображения и намерений разработчика.

В данном примере показан способ, когда декоративная линия располагается слева от текста. Можно линию сделать и справа, опять же обратившись к универсальному свойству `background`. У него есть замечательный аргумент, который задает положение левого верхнего угла фонового изображения, как показано в листинге 5.16.

Листинг 5.16. Декоративная линия справа от текста

```
<html>
<head>
<style type="text/css">
 #vline {
  background: url(check.gif) repeat-y right;  /* Фон с правого края */
  padding-right: 20px;                        /* Отступ справа */
}
</style>
</head>
<body>
<div id=vline>
...
</div>
</body>
</html>
```

К атрибуту `background` следует добавить аргумент `right`, тогда фон будет расположен у правого края блока. А отступ от рисунка до текста теперь управляется параметром `padding-right`. Опять же величина отступа зависит от используемого рисунка и устанавливается индивидуально.

К сожалению, приведенным способом нельзя установить декоративную линию одновременно справа и слева от блока, поскольку фоновый рисунок может располагаться только с одной стороны.

Создание рамок

Информация, обведенная рамкой, позволяет отделить один материал на веб-странице от другого и привлечь внимание читателя. Кроме того, рамка вносит элегантность в оформление сайта.

Для создания рамок применяется несколько способов, большая часть из которых использует свойства таблиц. Стили дополняют возможности таблиц

и позволяют создать рамки разных видов простыми средствами. Далее приводятся распространенные приемы получения рамок, которые можно встретить повсеместно на разных сайтах.

Использование параметра таблицы *border*

Самый простой способ создания рамки заключается в использовании параметров таблицы `border`, который определяет толщину рамки, и `bordercolor`, который задает ее цвет. Рамки, созданные таким образом, отличаются по своему виду в разных браузерах (рис. 5.10).

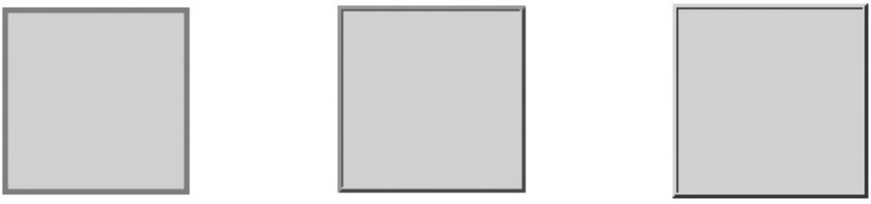


Рис. 5.10. Рамка, полученная с помощью параметра `border`:
а — Internet Explorer; *б* — Netscape; *в* — Opera

Браузер Netscape имитирует трехмерность рамки (рис. 5.10, *б*), Opera вообще оставляет ее цвет без изменения (рис. 5.10, *в*), а Internet Explorer делает рамку сплошной (рис. 5.10, *а*). В браузерах Mozilla и Firefox вид рамки будет аналогичным Netscape.

В листинге 5.17 приведен код для создания рамки с помощью параметров таблицы.

Листинг 5.17. Использование параметра `border`

```
<html>
<body>
<table border=2 bordercolor=#ff0000 width=100 height=100 cellpadding=6
cellspacing=0 bgcolor=#e0e0e0>
<tr>
<td>Содержимое</td>
</tr>
</table>
</body>
</html>
```

Параметр `border` определяет толщину рамки, `bordercolor` — ее цвет, а `cellpadding` задает расстояние от рамки до содержимого таблицы.

Использование параметра таблицы *cellspacing* и *bgcolor*

Чтобы браузеры отображали рамки одинаково, можно использовать способ их создания, связанный со свойствами таблицы `bgcolor` и `cellspacing`. Через параметр `bgcolor` вся таблица "закрашивается" цветом, совпадающим с желаемым цветом рамки. Внутренней ячейке нужно задать другой цвет фона, например белый. Рамка при этом образуется за счет того, что размер ячейки меньше размера самой таблицы, из-за этой разницы и получается видимая граница. Расстояние между ячейками управляется параметром таблицы `cellspacing`, который и уменьшает размеры ячейки на требуемую величину (листинг 5.18).

Листинг 5.18. Использование параметра `bgcolor`

```
<html>
<body>
<table cellpadding=6 cellspacing=1 border=0 width=100 height=100
bgcolor=#000000>
 <tr>
  <td bgcolor=#ffffff>
   <table>
    <tr>
     <td>Содержимое</td>
    </tr>
   </table>
  </td>
 </tr>
</table>
</body>
</html>
```

Толщину рамки можно варьировать, изменяя значение параметра `cellspacing`, а отступы от края границы до содержимого — параметром `cellpadding`.

Вложенные таблицы

Еще один простой и универсальный способ построения рамки, одинаково работающий в разных браузерах, основан на наложении двух таблиц друг на друга. Если мы возьмем прямоугольник, например из красной бумаги, и сверху на него наложим еще один прямоугольник белого цвета чуть меньшего размера, то увидим красную рамку. Только вместо листа бумаги мы

используем тег TABLE, а цвет таблицы задаем параметром bgcolor (листинг 5.19).

Листинг 5.19. Использование вложенных таблиц

```
<html>
<body bgcolor=#ffffff>
<table width=300 height=300 border=0 cellspacing=0 cellpadding=1
bgcolor=#ff0000>
 <tr>
  <td><table border=0 cellspacing=0 cellpadding=10 bgcolor=#ffffff
width=300 height=300>
 <tr>
  <td align=center>Содержимое</td>
 </tr>
  </table>
  </td>
 </tr>
</table>
</body>
</html>
```

В верхней таблице устанавливаем ширину и высоту таблицы по желанию, параметром bgcolor задаем цвет рамки, значение cellspacing присваиваем равным нулю, а параметр cellpadding управляет толщиной границы. Чем больше значение этого параметра, тем толще будет и рамка.

Для внутренней таблицы надо обязательно задать ее цвет, отличный от цвета внешней таблицы и совпадающий с цветом фона веб-страницы (в этом примере задан белый цвет). Параметры cellspacing и cellpadding (в данном случае не имеет значения, какой из них использовать) определяют расстояние от границы рамки до содержимого таблицы.

Ширина и высота внутренней таблицы обязательно должны совпадать с шириной и высотой внешней таблицы.

Замечание

Перенос строки в тексте автоматически добавляет пробел. Если рамка в некоторых местах получается толще, чем задумано, следует убрать лишние пробелы между тегами и тем самым сократить количество строк.

Заливка ячеек таблицы цветом

Использование цвета фона ячеек таблицы — один из универсальных и распространенных способов создания рамки. Вначале создаем таблицу, со-

стоящую из девяти ячеек (рис. 5.11), и заливаем крайние ячейки нужным цветом (рис. 5.12).

Рис. 5.11. Исходная таблица для создания рамки

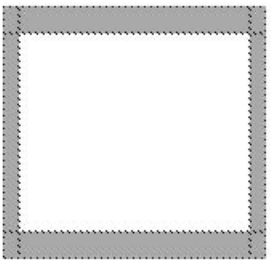


Рис. 5.12. Ячейки с установленным фоновым цветом

Толщина рамки определяется шириной и высотой ячеек. Желательно поместить внутрь этих ячеек таблицы распорку — прозрачный рисунок размером 1 × 1 пиксел, как уже упоминалось ранее, это необходимо для некоторых старых браузеров, иначе для них этот способ работать не будет (листинг 5.20).

Листинг 5.20. Создание рамки заливкой цветом ячеек таблицы

```
<html>
<body>
<table width=200 border=0 height=200 cellspacing=0 cellpadding=0>
<tr bgcolor=#ff6633>
<td width=2 height=2><img src=spacer.gif width=2 height=2></td>
<td height=2><img src=spacer.gif width=2 height=2></td>
<td width=2 height=2><img src=spacer.gif width=2 height=2></td>
</tr>
<tr>
<td width=2 bgcolor=#ff6633><img src=spacer.gif width=2 height=2></td>
<td align=center style="padding: 7px">Содержимое</td>
<td width=2 bgcolor=#ff6633><img src=spacer.gif width=2 height=2></td>
</tr>
<tr bgcolor=#ff6633>
<td width=2 height=2><img src=spacer.gif width=2 height=2></td>
<td height=2><img src=spacer.gif width=2 height=2></td>
<td width=2 height=2><img src=spacer.gif width=2 height=2></td>
</tr>
</table>
</body>
</html>
```

В данном примере создается красная рамка толщиной 2 пиксела, такая толщина смотрится наиболее элегантно. Параметры таблицы `cellspacing` и `cellpadding` приравниваются к нулю, чтобы рамка оставалась нужной толщины и без зазоров. Если поместить внутрь такой таблицы текст, он будет плотно прилегать к рамке, что некрасиво и плохо читается. Чтобы добавить отступы, можно создать для этой цели дополнительные ячейки, использовать вложенную таблицу или стили, как показано в примере.

Приведенный способ чувствителен к разным параметрам, поэтому в случае неправильной отрисовки рамки нужно проверить следующее:

- если размер таблицы задается в пикселах, надо задать фиксированные размеры всех ячеек;
- когда ширина таблицы указана в процентах, внутри рамки нужно поместить текст, который "распирает" внутреннюю ячейку до нужной ширины; при отсутствии текста или другого содержимого внутри таблицы с рамкой ширина некоторых ячеек окажется больше желаемой;
- в ячейках, где проходит рамка, не должно быть никаких символов неразделяемого пробела (` `), которые так любят автоматически добавлять некоторые редакторы веб-страниц;
- перенос строки автоматически добавляет в текст пробел, поэтому текст между открывающим и закрывающим тегом `<td>` и `</td>` следует писать в одну строку, чтобы не было зазоров между ячейками.

Декоративная рамка

Для создания декоративной рамки ее необходимо предварительно нарисовать в каком-нибудь графическом редакторе. На рис. 5.13, например, показана рамка со скругленными углами. Затем это изображение разрезаем на 9 частей, помеченных на рис. 5.14 линиями и номерами. Разрезать изображение можно в программе ImageReady, Photoshop и т. д.

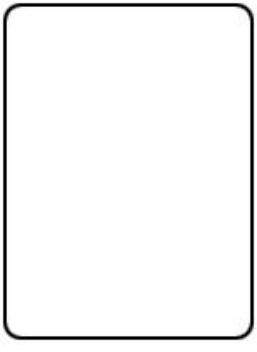


Рис. 5.13. Нарисованный в редакторе прямоугольник с закругленными рамками

Рис. 5.14. Разрезанный на элементы рисунок

После разрезания рисунка на части получим 8 фрагментов, приведенных в табл. 5.1.

Таблица 5.1. *Изображения, необходимые для создания рамки, и их описание*

Рисунок	Положение	Имя файла
	Левый верхний угол	1.gif
	Верхняя горизонтальная линия	2.gif
	Правый верхний угол	3.gif
	Левая вертикальная линия	4.gif
	Правая вертикальная линия	6.gif
	Левый нижний угол	7.gif
	Нижняя горизонтальная линия	8.gif
	Правый нижний угол	9.gif

Пятого фрагмента рамки не будет (5.gif), поскольку его заменит содержимое рамки. Теперь создаем таблицу размером 3×3 ячеек, как показано на рис. 5.11, и в ее ячейках размещаем созданные рисунки (листинг 5.21).

Листинг 5.21. Создание декоративной рамки

```
<html>
<body>
<table width=400 border=0 cellspacing=0 cellpadding=0>
<tr>
<td height=13><img src=1.gif width=12 height=13></td>
<td background=2.gif height=13><img src=1x1.gif height=13></td>
<td height=13><img src=3.gif width=14 height=13></td>
</tr>
<tr>
<td background=4.gif width=12> </td>
<td align=center>Содержимое</td>
<td background=6.gif width=12> </td>
</tr>
<tr>
<td height=13><img src=7.gif width=12 height=13></td>
<td background=8.gif><img src=1x1.gif height=13 width=1></td>
<td height=13><img src=9.gif width=14 height=13></td>
```

```
</tr>
</table>
</body>
</html>
```

В таблице параметры `border`, `cellspacing` и `cellpadding` должны быть равны нулю, иначе линии не будут состыковываться между собой.

Уголки вставляем как обычные рисунки с помощью тега `IMG`, а вот горизонтальные и вертикальные линии следует задавать с помощью фона ячейки таблицы параметром `background`. Это позволит сделать нашу рамку масштабируемой, и размер ее будет изменяться в зависимости от содержимого.

Чтобы линии плотно прилегали друг к другу и между ними не было пробелов, надо обязательно указывать все размеры ячеек и рисунков.

Приведенный способ позволяет создавать рамки практически любого вида.

Использование стилей для создания рамки

С помощью стилей рамку можно применить к любому блочному тегу, например параграфу (тег `P`), таблице (тег `TABLE`) и слою (тег `DIV`). Стили позволяют проще и удобнее создавать рамку, чем при использовании таблиц, и предоставляют разные виды рамок, которые показаны на рис. 5.1. Изменяя значения стилевых параметров, можно получить самые разнообразные рамки. Так, в листинге 5.22 показано создание двойной рамки вокруг текста.

Листинг 5.22. Двойная рамка вокруг текста

```
<html>
<head>
<style type="text/css">
 P {
  border: double 4px black;          /* Двойная рамка */
  background: #fc0;                  /* Цвет фона под текстом */
  padding: 5px                       /* Поля вокруг текста */
}
</style>
</head>
<body>
<p>При работе на вычислительной технике необходимо сесть так, чтобы руки
с предплечьями образовывали прямой угол, глаза поставить на расстояние
30-40 см от рабочей поверхности монитора. </p>
</body>
</html>
```

В данном примере используется двойная рамка, толщина которой складывается из толщины линий и расстояния между ними. Чтобы рамка не соприкасалась с текстом, задан отступ от рамки до содержимого параметром `padding`.

Интересный эффект можно получить, если одновременно установить пунктирную границу и фон у блочного элемента (рис. 5.15).

Рис. 5.15. Способ оформления текста в виде пунктирной рамки

Внешняя граница рамки всегда проходит по внешней границе блока, иными словами, рамка никогда не превышает заданные размеры элемента; если цвет фона и рамки совпадает, мы увидим лишь сплошной прямоугольник. Это хорошо заметно на рис. 5.16, который представляет собой увеличенный фрагмент границы.

Рис. 5.16. Граница у элемента всегда проходит внутри него

В связи с указанной особенностью поведения рамки, цвет границы должен совпадать с цветом фона веб-страницы, а сама рамка должна быть пунктирной. Это одновременно достигается с помощью атрибута `border`, как показано в листинге 5.23.

Листинг 5.23. Пунктирное обрамление текста

```
<html>
<head>
<style type="text/css">
 .borderLayer {
  border: 3px dotted white;          /* Параметры рамки */
  background: maroon;                /* Цвет фона */
  padding: 7px;                      /* Поля вокруг текста */
  color: white                       /* Цвет текста */
}
</style>
</head>
```

```
<body bgcolor=white>
 <div class=borderLayer>
  Lorem ipsum dolor sit amet...
 </div>
</body>
</html>
```

Пунктир в примере задается через аргумент `dotted` параметра `border`, а цвет блока с помощью свойства `background`. Толщину рамки рекомендуется установить размером 2—4 пиксела, тогда она будет более выразительной. Можно также поэкспериментировать и с разными значениями, задающими вид рамки.

Чтобы изменить вид границы, иногда недостаточно просто указать стиль рамки, ведь в данном случае мы имеем дело с инвертированным цветом. Поэтому можно воспользоваться вложенными один в другой элементами, и для внешнего указать желаемый вид рамки. Подобный подход расширяет возможности по изменению вида обрамления и позволяет получить границы, как, например, показано на рис. 5.17.

Рис. 5.17. Использование пунктира для обрамления

Создать границу, которая приведена на рис. 5.17, помогут два элемента `DIV` (листинг 5.24). Первый, назовем его `borderLayer`, задает характеристики рамки — ее толщину, цвет и стиль. Второй, с именем `bgLayer`, определяет цвет фона, текста и ширину полей вокруг текста. Цвет рамки и фона при этом должны совпадать.

Листинг 5.24. Изменение вида границы с помощью стилей

```
<html>
<head>
<style type="text/css">
 .borderLayer {
  border: 3px dashed maroon          /* Параметры рамки */
}
 .bgLayer {
  background: maroon;                /* Цвет фона */
  padding: 7px;                      /* Поля вокруг текста */
  color: white                       /* Цвет текста */
}
```

```
</style>
</head>
<body>
<div class=borderLayer>
 <div class=bgLayer>
  Lorem ipsum dolor sit amet...
 </div>
</div>
</body>
</html>
```

В данном примере для задания вида рамки используется стиль `dashed` параметра `border`, толщина 3 пиксела и цвет `maroon`, который на русском языке можно охарактеризовать как красно-коричневый.

Прием, основанный на вложенных тегах, нашел свое применение и при создании широких рамок вокруг текста. Такая рамка отличается от обычной не просто шириной (этого можно добиться с помощью обычного атрибута `border`), а некоторыми характеристиками — параметрами внутренней и внешней рамки, расстоянием и цветом между ними (рис. 5.18).

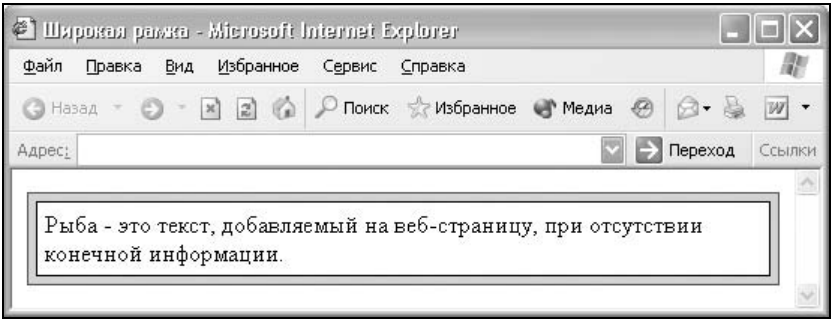


Рис. 5.18. Вид широкой рамки с границами разного цвета

Для создания указанной рамки вначале устанавливается слой с заданной границей с помощью тега `DIV`, а затем внутрь него помещается еще один слой, но уже с другой рамкой. Поскольку цвет фона по умолчанию принимается прозрачным, то для внутреннего тега `DIV` надо обязательно добавить атрибут `background`, как показано в листинге 5.25.

Листинг 5.25. Создание рамки с помощью вложенных тегов `DIV`

```
<html>
<head>
<style type="text/css">
```

```
 .border1 {
  border: solid 1px #634f36;          /* Параметры внешней рамки */
  background: #dad4c4;                /* Цвет поля между границами */
  padding: 5px                        /* Расстояние между рамками */
}
 .border2 {
  border: 1px solid black;            /* Параметры внутренней рамки */
  background: white;                  /* Цвет фона под текстом */
  padding: 5px                        /* Поля вокруг текста */
}
</style>
</head>
<body>
<div class=border1>
 <div class=border2>
Рыба - это текст, добавляемый на веб-страницу, при отсутствии конечной
информации.
 </div>
</div>
</body>
</html>
```

В данном примере используется два селектора: первый, с именем `border1`, задает характеристики внешней границы, цвет фона между рамками, а также расстояние между ними через атрибут `padding`; второй селектор, названный `border2`, определяет параметры внутренней рамки, цвет фона под содержимым и поля вокруг него.

Рамки и плавающие фреймы

Плавающим фреймом называется элемент веб-страницы, создаваемый с помощью тега `IFRAME`, который разрешает встраивать на страницу содержимое других файлов. Такая возможность позволяет хранить определенные данные в одном месте и вставлять их при необходимости в любое число документов. Отдельные преимущества использования плавающих фреймов состоят в следующем:

- файл, загружаемый во фрейм, помещается в кэш браузера — место на локальном компьютере для хранения информации, к которой часто осуществляется доступ, — и таким образом последующее обращение к файлу происходит быстрее;
- допускаются ссылки на HTML-документы, изображения, серверные программы не только на своем сайте, но и обращение к файлам, расположенным на других сайтах по абсолютному адресу;

- плавающие фреймы легко встраиваются в содержимое веб-страницы благодаря наличию параметров, регулирующих ширину, высоту и способ выравнивания;
- редактирование повторяющихся данных одновременно на множестве веб-страниц упрощается за счет того, что хранить их можно в одном файле, а в документ вставлять с помощью плавающих фреймов.

Вместе с тем, несмотря на перечисленные достоинства, плавающие фреймы имеют и другие особенности, которые стоит принимать во внимание. Так, кроме браузера Internet Explorer, тег `IFRAME` понимают браузеры Opera 6, Netscape 7, Mozilla, Firefox и некоторые другие. Номер версии последних здесь не приводится, поскольку в этих браузерах изначально установлена поддержка современных стандартов, в том числе и плавающих фреймов.

Еще одна особенность связана с отображением рамок вокруг фрейма. По умолчанию всегда создается рамка с эффектом трехмерности. Чтобы ее скрыть, применяется параметр `frameborder` тега `IFRAME`. Причем в качестве значения допускается использовать `0` или `no`. Браузеры одинаково понимают любой из аргументов, так что применяйте их по своему усмотрению, например, как показано в листинге 5.26.

Листинг 5.26. Создание рамки вокруг плавающего фрейма

```
<html>
<body>
<iframe src=sample.html width=600 height=100 align=center frameborder=0
style="border: 2px solid black">
</iframe>
</body>
</html>
```

В данном примере показано создание рамки вокруг фрейма с помощью атрибута `border`. Указывать параметр `frameborder` в таком случае обязательно, иначе исходная трехмерная рамка совместится с рамкой, установленной с помощью стилей. На рис. 5.19 показан первоначальный вид границы у плавающего фрейма, а рис. 5.20 демонстрирует результат примера, приведенного в листинге 5.26.

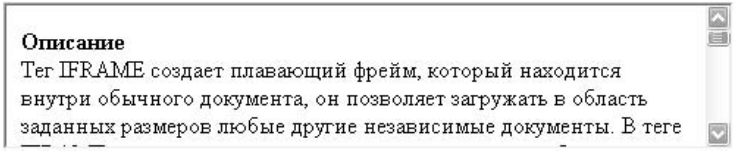


Рис. 5.19. Трехмерная рамка у плавающего фрейма, заданная по умолчанию

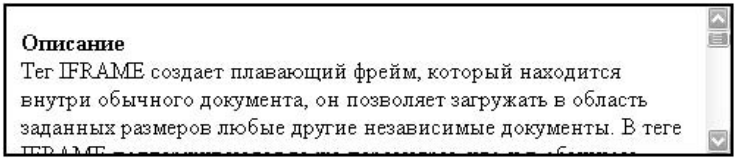


Рис. 5.20. Рамка вокруг фрейма, созданная с помощью стилей

Причины задания собственных оригинальных рамок могут быть следующими: необходимость придерживаться выбранного оформления для всех элементов веб-страницы, выпадение эффекта трехмерности из общего дизайна сайта, изменение цвета рамки и т. д.

Рамки с тенью

Тень придает объем элементу и привлекает к нему внимание посетителей сайта, позволяя расставлять акценты на нужные области веб-страницы. Способ создания тени с помощью таблицы и графических изображений был описан в *главе 2*, а здесь мы рассмотрим альтернативные методы.

Создание тени с помощью атрибута *border*

Относительно простой подход к созданию тени заключается в ее имитации с помощью установки линии серого цвета на двух сторонах прямоугольной области (рис. 5.21).

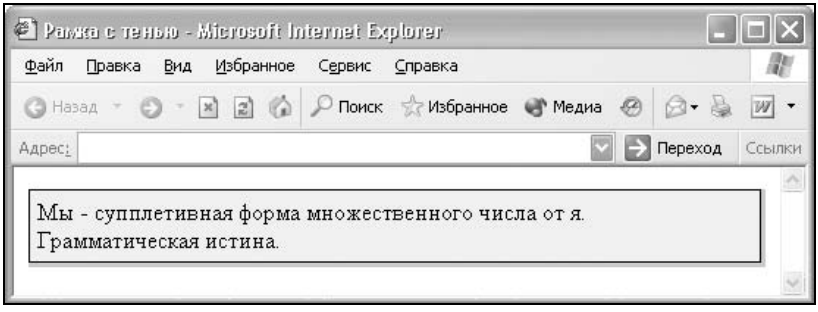


Рис. 5.21. Вид тени, созданной с помощью атрибута `border`

Как видно из данного рисунка, полученная тень в полном смысле тенью не является, а лишь имитирует ее вид. Однако во многих случаях подобного результата вполне достаточно.

Чтобы добавить такую тень, применим свойство `border` и его производные. Одновременно установить серую линию справа и снизу вместе с рамкой не получится, поэтому воспользуемся двумя вложенными слоями. Для первого,

с именем `shadowLayer`, добавим линию серого цвета толщиной 3 пиксела справа и снизу от блока (листинг 5.27). Эта линия будет имитировать тень. Второй слой, `borderLayer`, вложенный внутрь первого, определяет стиль рамки, цвет фона и полей вокруг текста.

Листинг 5.27. Добавление тени с помощью линий

```
<html>
<head>
<style type="text/css">
 .shadowLayer {
  border-right: 3px solid #ccc;      /* Параметры тени справа */
  border-bottom: 3px solid #ccc      /* Параметры тени снизу */
}
 .borderLayer {
  border: 1px solid black;           /* Параметры рамки */
  background: #fc0;                  /* Цвет фона слоя */
  padding: 5px                       /* Поля вокруг текста */
}
</style>
</head>
<body>
<div class=shadowLayer>
 <div class=borderLayer>
  Мы - супплетивная форма множественного числа от я. Грамматическая
истина.
 </div>
</div>
</body>
</html>
```

Если рамка не нужна, то можно обойтись лишь одним тегом `DIV`, но в этом случае требуется обеспечить достаточный контраст между цветом тени и фона.

Применение слоев

Тень не обязательно делать размером во всю сторону блока, иногда достаточно ее лишь обозначить, как показано на рис. 5.22. Такой способ использования тени придает выразительность рамке, но при этом она становится трехмерной лишь отчасти, из-за этого она не входит в конфликт с другими плоскими объектами на веб-странице.

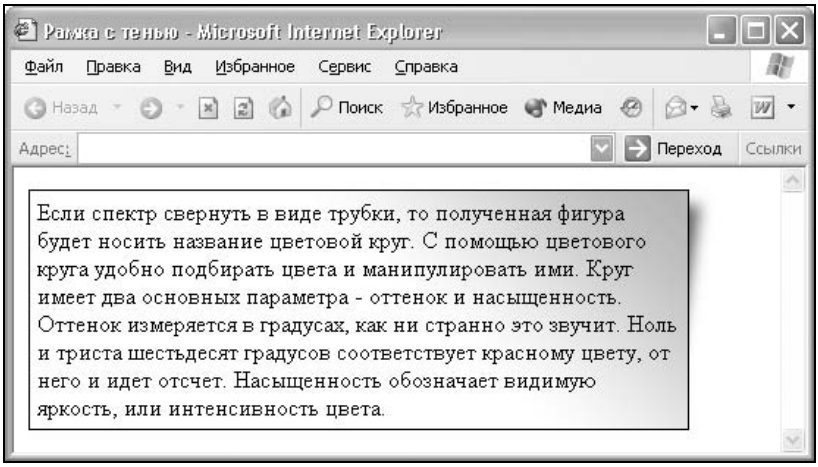


Рис. 5.22. Тень от уголка рамки

Для создания подобной рамки понадобятся две картинки. Первое изображение будет представлять тень (рис. 5.23), а второе, необязательное, используется в качестве фонового рисунка в слое (рис. 5.24).

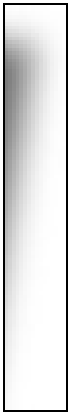


Рис. 5.23. Изображение тени для создания рамки

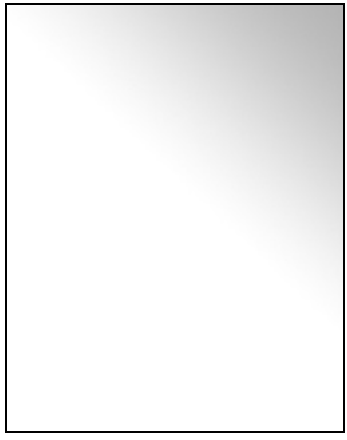


Рис. 5.24. Изображение для использования в качестве фона

Управление видом рамки происходит с помощью двух слоев: `bgLayer` — устанавливает фоновый рисунок, ширину слоя, поля вокруг текста и параметры границы, а `shadowLayer` — добавляет рисунок тени. Чтобы тень располагалась справа от границы, используется параметр `float` со значением `left`. В этом случае слой будет выравниваться по левому краю и обтекаться с остальных сторон (листинг 5.28). По этой причине в самом конце кода жела-

тельно добавить тег BR с атрибутом стиля clear: both, иначе текст после рамки может "заползти" на эту же строку. В таком написании этот тег устанавливает перенос строки и одновременно отменяет выравнивание элементов.

Листинг 5.28. Добавление тени с помощью стилей и слоев

```
<html>
<head>
<style type="text/css">
 .bgLayer {
  border: 1px solid black;   /* Параметры рамки */
  background:
       url(/images/bg1.gif) /* Путь к фоновому изображению */
       right                /* Фон в правом верхнем углу слоя */
       no-repeat;           /* Не повторять фон */
  width: 450px;             /* Ширина слоя в пикселах */
  padding: 5px;             /* Поля вокруг содержимого */
  float: left               /* Обтекание с правого края */
}
 .shadowLayer {
  background:
       url(/images/bg2.gif) /* Путь к рисунку с тенью */
       no-repeat;           /* Не повторять фон */
  width: 16px;              /* Ширина рисунка с тенью */
  height: 112px;            /* Высота рисунка с тенью */
  float: left               /* Обтекание с правого края */
}
</style>
</head>
<body>
 <div class=bgLayer>
  Если спектр свернуть в виде трубки, то полученная фигура будет носить
название цветовой круг. С помощью цветового круга удобно подбирать цвета
и манипулировать ими.
</div>
<div class=shadowLayer> </div>
<br style="clear: both">
</body>
</html>
```

В данном примере рисунок с тенью добавляется через свойство background, того же результата можно добиться и без использования стилей путем вставки изображения через тег IMG.

Чтобы фоновый рисунок не сдвигался, ему назначается жестко заданное положение и отменяется повторение. Все это проводится через все тот же универсальный атрибут `background`.

Обратите внимание, что указанный способ добавления тени действует для слоя фиксированной ширины, заданной в пикселах. Если требуется создать подобную рамку для ширины, установленной в процентах, удобнее использовать таблицу.

Использование таблицы

Таблицы с невидимой границей удобны тем, что позволяют легко выстраивать в нужном порядке элементы веб-страницы. Подобная таблица образует своеобразный каркас, называемый *модульная сетка*, который определяет, где будут располагаться разные элементы. Для создания рамки с тенью понадобится таблица с двумя колонками, в левую колонку помещается текст, а правая предназначена для рисунка тени. Чтобы изображение плотно прилегало к границе, параметры `border`, `cellpadding` и `cellspacing` тега `TABLE` следует обнулить. Поля вокруг текста, характеристики рамки и фон можно эффективно добавить и с помощью таблицы, как показано в листинге 5.29.

Листинг 5.29. Добавление тени с помощью таблицы

```
<html>
<head>
<style type="text/css">
 TD.bg {
  border: 1px solid black;
  background: url(/images/bg.gif) right no-repeat;
  padding: 5px
}
</style>
</head>
<body>
<table width=100% cellpadding=0 cellspacing=0 border=0>
 <tr>
  <td class=bg>
  Цветовой круг имеет два основных параметра - оттенок и насыщенность.
Оттенок измеряется в градусах, как ни странно это звучит. Ноль и триста
шестьдесят градусов соответствует красному цвету, от него и идет отсчет.
  </td>
  <td width=16 valign=top><img src=/images/shadow.gif width=16
height=112></td>
 </tr>
```

```
</table>
</body>
</html>
```

Вид рамки с тенью, полученной с помощью таблицы, показан на рис. 5.22. В этом смысле она ничем не отличается от рамки, созданной через слои. Однако для таблицы характерно то, что ее ширину можно легко задавать как в процентах, так и пикселах.

Учтите, что по умолчанию вертикальное выравнивание в ячейке таблицы происходит по центру, поэтому для ячейки с изображением тени обязательно следует добавить параметр `valign=top`.

Замечание

Высота слоя или таблицы устанавливается автоматически исходя из объема содержимого. Если такая высота окажется меньше высоты рисунка с тенью, то тень по вертикали будет выходить за пределы рамки, что смотрится крайне неэстетично. Учитывайте этот момент и подбирайте высоту изображения, исходя из высоты содержимого рамки или делая тень достаточно небольшой, около 50—60 пикселов.

Рамки и заголовки

Для удобства представления информации ее желательно не только поместить в рамку, но и установить для нее заголовок. Оформление таких рамок зависит от их предназначения, общего дизайна страницы и изысков разработчиков. Далее показаны несколько способов создания и отображения рамки с заголовком, которую мы будем называть *панелью*.

Простая панель

Наиболее доступный вариант создания простой панели заключается в использовании таблицы. В этом случае создается таблица нужной ширины с двумя ячейками, одна из которых выступает в качестве заголовка, а в другую помещается содержимое. Манипулируя параметрами таблицы или их эквивалентами CSS, можно получить вариант, по виду напоминающий стандартное окно Windows (рис. 5.25).

В листинге 5.30 показано создание панели с использованием таблицы. Для удобства изменения оформления большинство параметров перенесено в стили. С этой целью добавлено три селектора: `panel` управляет шириной панели и задает характеристики рамки; `title` формирует вид заголовка, а `TD` устанавливает поля вокруг содержимого ячеек.

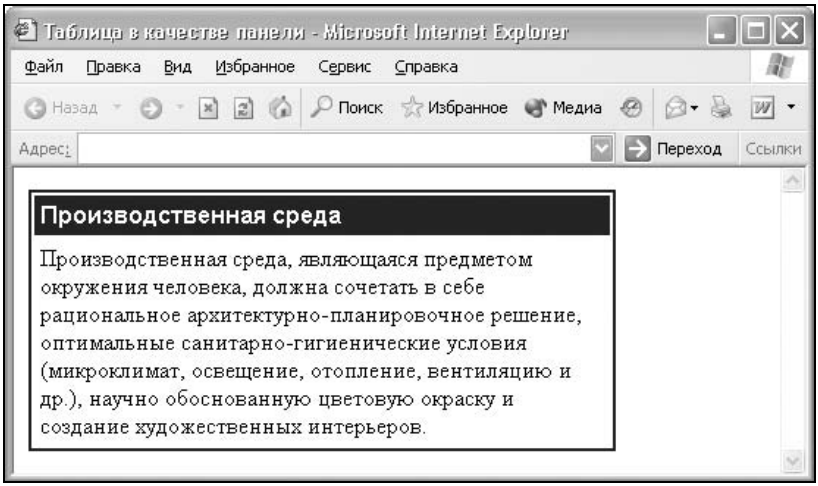


Рис. 5.25. Вид панели, созданной с помощью таблицы

Листинг 5.30. Применение таблицы и стилей для управления видом панели

```
<html>
<head>
<style type="text/css">
 TABLE.panel {
  width: 400px;                      /* Ширина таблицы в пикселах */
  border: 2px solid navy             /* Параметры рамки */
}
 TD.title {
  background: navy;                  /* Цвет фона под заголовком */
  color: white;                      /* Цвет заголовка */
  font-family: Arial, sans-serif;    /* Шрифт для текста заголовка */
  font-weight: bold                  /* Жирное начертание */
}
 TABLE.panel TD {
  padding: 4px                       /* Поля вокруг текста */
}
</style>
</head>
<body>
<table cellspacing=2 class=panel>
 <tr>
  <td class=title>Заголовок панели</td>
 </tr>
```

```
 <tr>
  <td>Содержимое панели</td>
 </tr>
</table>
</body>
</html>
```

Желательно использовать одинаковый цвет рамки и фона заголовка, так они смотрятся более цельно. В данном примере рамка установлена темно-синего цвета с помощью зарезервированного имени `navy`. Параметр `cellspacing` тега `TABLE` оставлен не случайно, изменение его значения влияет на вид отображения заголовка. На рис. 5.26 показан результат, полученный при разных значениях `cellspacing`.

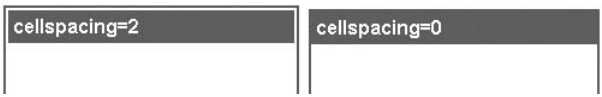


Рис. 5.26. Вид заголовка панели при разных значениях атрибута `cellspacing`

Для создания панели не обязательно применять таблицу, можно воспользоваться и слоями. Подход с применением слоев для верстки различных элементов считается современным и более прогрессивным. На самом деле, структура кода получается почти идентичной, только вместо тегов `TABLE` и `TD` применяется `DIV` (листинг 5.31).

Листинг 5.31. Создание панели с помощью слоев

```
<html>
<head>
<style type="text/css">
 .panel {
  width: 400px; border: 2px solid navy
}
 .title {
  background: navy; color: white;
  font-family: Arial, sans-serif; font-weight: bold
}
 .title, .content { padding: 4px }
</style>
</head>
<body>
 <div class=panel>
```

```
  <div class=title>
   Заголовок панели
  </div>
  <div class=content>
   Содержимое панели
  </div>
 </div>
</body>
</html>
```

Несмотря на то, что оба подхода приводят к одному результату и в них активно применяются стили, использование таблицы противоречит ортодоксальному взгляду на HTML, который заключается в том, что таблицы следует применять только для представления табличных данных, а никак не для верстки. В этом смысле слои придерживаются идеологии разделения содержимого и оформления. Суть в следующем: желательно писать такой код HTML, в который добавляются только блочные элементы со ссылкой на таблицу стилей, а также изображения и текст. Само оформление в виде набора цветов, шрифтов, положения и других характеристик задается через стили. В итоге мы получаем простой, быстро загружаемый код, видом которого можно удобно и централизованно управлять с помощью стилей.

Панель с фиксированным заголовком

При использовании в панели длинного списка, ее высота становится неудобно большой, из-за чего неоправданно возрастает и общая высота вебстраницы. В таком случае размеры панели можно установить фиксированными, а для просмотра содержимого воспользоваться полосой прокрутки, как показано на рис. 5.27.

Рис. 5.27. Вид панели с вертикальной полосой прокрутки

Основное свойство CSS, которое понадобится для управления видом содержания, — `overflow`. В его задачу входит способ отображения блочного элемента, особенно если его размеры заведомо меньше высоты или ширины содержимого. Для этого свойство `overflow` имеет следующие аргументы:

- `visible` — отображается все содержимое элемента, даже за пределами установленной высоты и ширины; это значение определено по умолчанию;
- `hidden` — отображается только область внутри элемента, остальное будет обрезано;
- `scroll` — всегда добавляются полосы прокрутки;
- `auto` — полосы прокрутки добавляются только при необходимости, если содержимое не помещается в заданную область.

Таким образом, добавляя `overflow: auto` к слою, мы гарантируем, что информацию всегда можно будет удобно просмотреть. Если объем данных меньше размера слоя, то полосы прокрутки отображаться не будут, в противном случае, они помогают пролистать содержимое.

Итак, чтобы получить панель с фиксированным заголовком, понадобится три слоя. Первый из них выступает в роли контейнера и определяет внешний вид панели, в частности цвет фона и рамки. Остальные два слоя устанавливают стилевые атрибуты для заголовка и содержания (листинг 5.32).

Листинг 5.32. Создание панели с фиксированным заголовком

```
<html>
<head>
<style type="text/css">
 .panel {
  width: 200px;                    /* Ширина панели */
  background: #d8ecd4;             /* Цвет фона панели */
  border: 1px solid #546852        /* Параметры рамки вокруг */
}
 .title {
  background: #7a9879;             /* Цвет фона под заголовком */
  padding: 4px;                    /* Поля вокруг текста заголовка */
  color: #ffffee;                  /* Цвет заголовка */
  font-family: Arial, sans-serif;  /* Параметры шрифта */
  font-weight: bold                /* Жирное начертание */
}
 .content {
  padding: 4px;                    /* Поля вокруг содержимого панели */
  overflow: auto;                  /* Добавить полосы прокрутки */
  height: 200px                    /* Высота панели без заголовка */
}
```

```
</style>
</head>
<body>
 <div class=panel>
  <div class=title>
   Заголовок панели
  </div>
  <div class=content>
   Содержимое панели
  </div>
 </div>
</body>
</html>
```

Поскольку свойство `overflow` применяется к слою с именем `content`, у которого, к тому же, установлена заданная высота, то полосы прокрутки будут отображаться только для содержимого этого слоя. Заголовок панели, заданный слоем `title`, независимо от объема информации всегда остается на месте.

Панель с графическим заголовком

Использование рисунков расширяет возможности по оформлению панелей и рамок, позволяя получить самые разнообразные варианты. Как правило, применяемые для этой цели графические файлы малы по объему, поэтому они загружаются достаточно быстро, не создавая проблем пользователю с ожиданием отображения рамки.

Вначале рассмотрим создание заголовка с закругленными уголками, как показано на рис. 5.28.

Первоначально следует изобразить вид панели в графическом редакторе. Уголки не обязательно делать закругленными, это зависит от вкуса и предпочтения разработчика. Главное, чтобы цвет заголовка был однотонным. После чего разрезаем рисунок с заголовком на две части, как показано на рис. 5.29.

На рис. 5.29, *а* изображена левая часть заголовка панели. Поскольку обычно заранее неизвестно, какой объем текста предполагается поместить в заголовок, то его высота может довольно сильно варьировать. Поэтому высоту картинки лучше взять с запасом. Ширина изображения зависит от того, где и как предполагается применять панель. Если ширина панели будет фиксирована и задана в пикселах, то такую же величину имеет смысл взять и для изображения заголовка. В том случае, когда ширина панели задается в пикселах и зависит от размеров окна браузера и разрешения монитора, ширину

картинки для заголовка лучше установить большего размера, в пределах 1000—1500 пикселов. Бояться, что возникнет горизонтальная полоса прокрутки, в данном случае не нужно, рисунок будет использоваться как фоновый, а это значит, что на размер окна браузера он не повлияет. Наоборот, если ширина рисунка будет меньше ширины окна, то возможно появление "дыр" в заголовке.

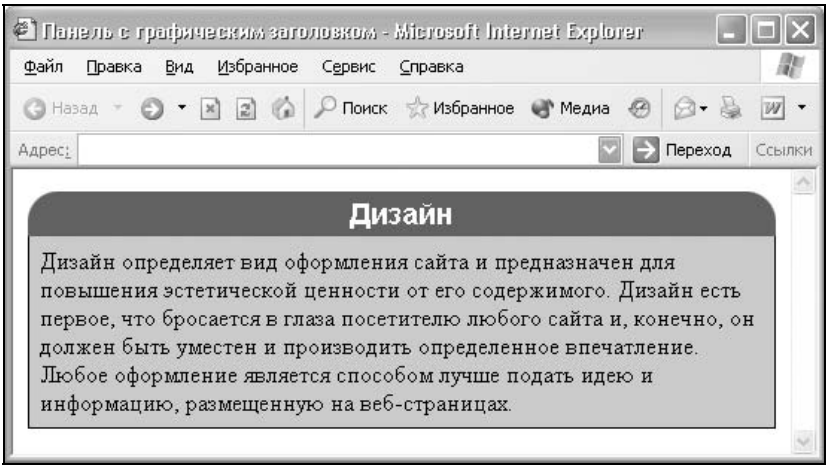


Рис. 5.28. Пример панели с заголовком, созданным с помощью изображений

Рис. 5.29. Подготовленные изображения для создания заголовка: *а* — уголок слева; *б* — уголок справа

На рис. 5.29, *б* представлена правая часть заголовка панели. Высота этого рисунка совпадает с высотой левого изображения, а ширина всегда будет одинаковой.

Теперь остается установить картинки в качестве фона для двух вложенных элементов. Для этого создаем слой с именем `panel` и для него устанавливаем параметр `background` с изображением, показанным на рис. 5.29, *а*, положение которого задано в левом верхнем углу без повторений. Внутрь этого

слоя помещаем заголовок H1 и для него тоже определяем фон, как на рис. 5.29, *б*, только выровненный по верхнему правому углу. Остается определить стиль текста заголовка и содержимого, а также при желании к панели добавить рамку (листинг 5.33).

Листинг 5.33. Создание панели с графическим заголовком

```
<html>
<head>
<style type="text/css">
 .panel {
  background:
       url(top-left.gif)     /* Путь к файлу с левым уголком */
       left top              /* Размещаем фон в левом верхнем углу */
       no-repeat;            /* Отменяем повторение фона */
  width: 320px               /* Ширина панели в пикселах */
}
 H1.panelTitle {
  background:
       url(top-right.gif)    /* Путь к файлу с правым уголком */
       right top             /* Размещаем фон в правом верхнем углу */
       no-repeat;            /* Отменяем повторение фона */
  font-family: Arial, sans-serif;   /* Рубленый шрифт для заголовка */
  font-size: 120%;           /* Размер текста заголовка */
  color: #ffffee;            /* Цвет текста в заголовке */
  text-align: center;        /* Выравниваем заголовок по центру */
  margin: 0px;               /* Убираем для тега H1 отступы */
  padding: 4px 7px           /* Добавляем поля вокруг заголовка */
}
 .panelBody {
  padding: 7px;              /* Поля вокруг текста */
  border: 1px solid black;   /* Параметры рамки */
  border-top: none;          /* Убираем верхнюю границу */
  background: #ccc           /* Цвет фона */
}
</style>
</head>
<body>
 <div class=panel>
  <h1 class=panelTitle>Заголовок</h1>
  <div class=panelBody>
   Содержимое
  </div>
```

```
 </div>
</body>
</html>
```

В данном примере показано создание панели с четко заданной шириной. Однако этот же код можно применять и в случае использования процентной записи. При этом поменяется только значение атрибута `width`.

Для тега `H1` характерно автоматическое добавление отступов снизу и сверху от текста. При использовании этого тега в панели подобное нежелательно, поэтому отступы обнуляются при помощи атрибута `margin`. Также можно вообще отказаться от использования `H1` и заменить его элементом `DIV`. Стиль при этом сохранится, а вот атрибут `margin` можно будет убрать. С другой стороны, поисковые системы при анализе веб-страницы повышают рейтинг слов и фраз, которые встречаются внутри тега `H1`.

Верхняя линия у рамки убирается с помощью значения `none` свойства `border-top`. Аналогично при желании можно скрыть границы и с других краев элемента.

Чтобы текст заголовка не наплывал на закругление рамки, рекомендуется добавить вокруг него поля через свойство `padding`. Для этой же цели поля устанавливаются и для текста содержимого панели, иначе он будет соприкасаться с рамкой вокруг.

Текст заголовков можно подготовить в графическом редакторе и сохранить в виде изображения. В этом случае доступно все многообразие шрифтов, установленных на компьютере, а также разные эффекты и фильтры, применяемые к графике. Однако при активном использовании панелей с разными текстами заголовков придется заранее готовить множество рисунков. К тому же если требуется поменять текст, то снова понадобится обращаться к редактору. Тем не менее применение изображений для заголовков позволяет создавать панели, которые смотрятся выигрышно и при этом достаточно быстро загружаются, например, как показано на рис. 5.30.

Приведенная панель создается следующим образом. Вначале делаем набросок панели в графическом редакторе, а затем вырезаем фрагмент рисунка с текстом заголовка (рис. 5.31, *а*) и верхнюю линию (рис. 5.31, *б*).

Все границы в панели, кроме верхней линии, устанавливаются через свойство `border`. Поскольку верхняя граница имеет изгибы, то для ее создания и понадобятся рисунки. Заголовок панели вставляется как обычный рисунок с помощью тега `IMG`, а линия — в виде фона для слоя с именем `panelTitle`. Чтобы в тексте заголовка фоновая линия не прослеживалась, следует убрать любую прозрачность в GIF-файле с текстом. Высота изображений с линией и текстом заголовка должна быть одинаковой, тогда произойдет точная состыковка между ними. Из-за того, что рамка применяется к слою с именем `panelBody`, а линия с помощью графики отображается только сверху, правая

сторона заголовка остается вообще без границы. Это легко исправляется добавлением линии справа для слоя `panelTitle`. Только убедитесь, что характеристики контуров во всех случаях совпадают (листинг 5.34). Также проверьте, чтобы цвет фона под основным текстом был такой же, как и в применяемых рисунках.

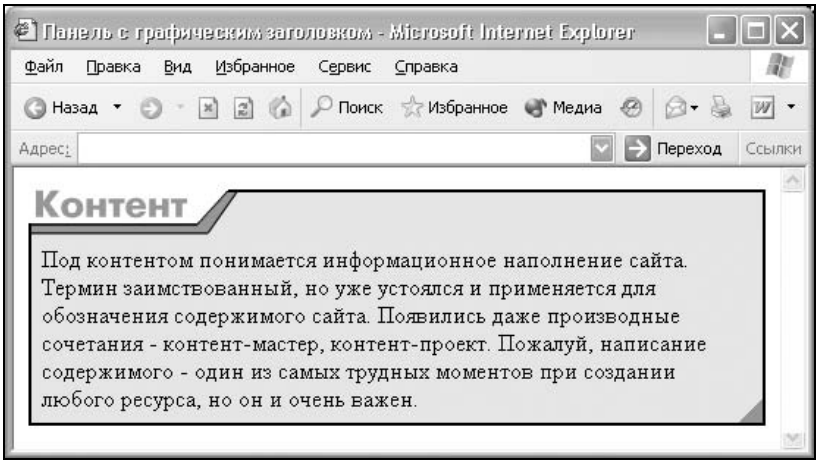


Рис. 5.30. Текст заголовка в виде рисунка

а *б* *в*

Рис. 5.31. Изображения, необходимые для создания панели:
а — фрагмент рисунка с текстом; *б* — верхняя линия; *в* — правый нижний уголок

Листинг 5.34. Панель с текстом в виде рисунка

```
<html>
<head>
<style type="text/css">
 .panel {
  width: 320px                              /* Общая ширина панели */
}
 .panelTitle {
  background:
       url(top-bg.gif)                /* Путь к файлу с верхней линией */
       repeat-x;                /* Повторять фон только по горизонтали */
  border-right: 2px solid black      /* Линия справа */
}
```

```
 .panelBody {
  padding: 7px;              /* Поля вокруг основного текста */
  border: 2px solid black;   /* Параметры границы */
  border-top: none;          /* Верхнюю линию не рисуем */
  background:
      url(bottom-bg.gif)     /* Путь к файлу с уголком */
      right bottom           /* Помещаем изображение в правый нижний угол */
      no-repeat;             /* Отменяем повторение фона */
  background-color: #e6e6e6  /* Цвет фона под текстом */
}
</style>
</head>
<body>
 <div class=panel>
  <div class=panelTitle><img src=title.gif width=143 height=31></div>
  <div class=panelBody>
  Содержимое
  </div>
 </div>
</body>
</html>
```

В правом нижнем углу панели добавлен небольшой треугольник (рис. 5.31, *в*) для красоты. Этот рисунок устанавливается как фоновый для слоя `panelBody`, при этом через стили отменяется его повторение и жестко задается положение картинки. Если в этом изображении нет нужды, просто удалите атрибут `background` у селектора `panelBody`.

ГЛАВА 6

Таблицы

Благодаря универсальности таблиц и большому числу параметров, управляющих их видом, таблицы надолго стали определенным стандартом для верстки веб-страниц. Таблица с невидимой границей представляет собой словно модульную сетку, в блоках которой удобно размещать элементы веб-страницы. Тем не менее это не совсем правильный подход, ведь каждый объект HTML определен для своих собственных целей и если он используется не по назначению, причем повсеместно, это значит, что альтернатив нет. Так оно и было долгое время, пока на смену таблицам при верстке сайтов не пришли слои. Это не значит, что слои теперь используются сплошь и рядом, но наметилась четкая тенденция — таблицы применяются для размещения табличных данных, а слои — для верстки и оформления.

Создание таблиц

Таблица состоит из строк и столбцов ячеек, которые могут содержать текст и рисунки. Обычно таблицы используются для упорядочения и представления данных, однако возможности таблиц этим не ограничиваются. С помощью таблиц удобно верстать макеты страниц, расположив фрагменты текста и изображений нужным образом.

Для добавления таблицы на веб-страницу используется тег `TABLE`. Этот элемент служит контейнером для элементов, определяющих содержимое таблицы. Любая таблица состоит из строк и ячеек, которые задаются с помощью тегов `TR` и `TD` соответственно. Таблица должна содержать хотя бы одну ячейку (листинг 6.1). Вместо тега `TD` допускается использовать тег `TH`. Текст в такой ячейке обычно отображается браузером жирным шрифтом и выравнивается по центру. В остальном разницы между ячейками, созданными через теги `TD` и `TH`, нет.

Листинг 6.1. Создание таблицы

```
<html>
<body>
<table border=1 width=100% cellpadding=5>
 <tr>
  <th>Ячейка 1</td>
  <th>Ячейка 2</td>
 </tr>
 <tr>
  <td>Ячейка 3</td>
  <td>Ячейка 4</td>
 </tr>
</table>
</body>
</html>
```

Порядок расположения ячеек и их вид показан на рис. 6.1.

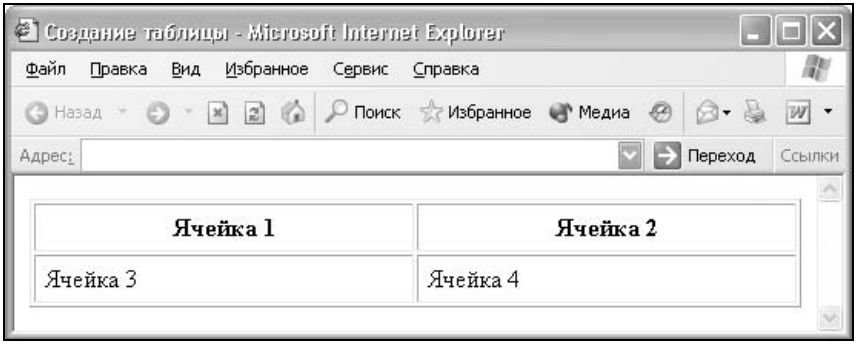


Рис. 6.1. Результат создания таблицы с четырьмя ячейками

Тот факт, что таблицы применяются достаточно часто и не только для отображения табличных данных, обязан не только их гибкости и универсальности, но и обилию параметров тегов `TABLE`, `TR` и `TD`. Далее перечислены параметры тега `TABLE`.

- `align`. Задает выравнивание таблицы по краю окна браузера. Допустимые значения: `left` — выравнивание таблицы по левому краю, `center` — по центру и `right` — по правому краю.
- `background`. Определяет изображение, которое будет использоваться в качестве фонового рисунка таблицы. В отличие от обычных изображений для фона не устанавливаются ширина и высота, и он всегда отображает-

ся в натуральную величину с масштабом 100 %. Если рисунок по размеру меньше ширины или высоты таблицы, то картинка повторяется по горизонтали вправо и вниз, выстраиваясь как мозаика. По этой причине на месте стыка фоновых картинок могут возникнуть видимые перепады, заметные для посетителей сайта. При выборе фонового рисунка убедитесь, что обеспечен достаточный контраст между ним и содержимым таблицы. Для фона можно использовать анимированные изображения в формате GIF, но они отвлекают внимание читателей. В качестве аргумента принимается любой допустимый адрес изображения — как относительный, так и абсолютный.

- `bgcolor`. Устанавливает цвет фона таблицы. Можно использовать этот параметр совместно с `background`, подобрав цвет фона, близкий к фоновому рисунку. В таком случае еще до загрузки графического файла таблица будет с цветным фоном.
- `border`. Устанавливает толщину рамки в пикселах. По умолчанию рамка изображается трехмерной, но если используется параметр `bordercolor` тега `TABLE`, то вид рамки меняется в зависимости от браузера (см. рис. 5.10). Браузер Internet Explorer делает рамку одноцветной, Netscape имитирует трехмерность рамки, а браузер Opera вообще оставляет цвет рамки без изменений. Когда в теге `TABLE` используется параметр `border` без аргументов (`<table border>`), браузер отображает рамку толщиной один пиксел. При совместном использовании параметров `border`, `bordercolor` и `cellspacing` можно получить двойную рамку.
- `bordercolor`. Устанавливает цвет рамки таблицы. Рамка обычно рисуется как трехмерная, добавление параметров `bordercolor` и `border` к тегу `TABLE` создают однотонную линию.
- `cellpadding`. Определяет расстояние между границей ячейки и ее содержимым. Этот параметр добавляет пустое пространство к ячейке, увеличивая тем самым ее размеры. Без параметра `cellpadding` текст в таблице "налипает" на рамку, ухудшая тем самым его восприятие. Добавление `cellpadding` позволяет улучшить читабельность текста. При отсутствии границ этот параметр не имеет особого значения, но он может помочь, когда требуется установить пустой промежуток между ячейками.
- `cellspacing`. Задает расстояние между внешними границами ячеек. Если установлен параметр `border`, то толщина границы принимается в расчет и входит в общее значение.
- `cols`. Параметр `cols` указывает количество столбцов в таблице, помогая браузеру в подготовке к ее отображению. Без этого параметра таблица будет показана только после того, как все ее содержимое будет загружено в браузер и проанализировано. Использование параметра `cols` позволяет несколько ускорить отображение содержимого таблицы.

- height. Устанавливает высоту таблицы. В спецификации HTML 4 этого параметра нет, однако браузеры в большинстве случаев понимают его, если он установлен у тега TABLE. Если высота таблицы не указана, то берется суммарное значение параметра height всех тегов TD и TH, в противном случае высота вычисляется на основе содержимого таблицы.
- rules. Сообщает браузеру, где отображать границы между ячейками. По умолчанию рамка рисуется вокруг каждой ячейки, образуя тем самым сетку. В дополнение можно указать отображение линий между колонками (значение cols), строками (rows) или группами (groups), которые определяются наличием тегов THEAD, TFOOT, TBODY, COLGROUP или COL. Толщина границы и ее цвет указывается с помощью параметров border и bordercolor.
- width. Задает ширину таблицы. Если общая ширина содержимого превышает указанную ширину таблицы, то браузер будет пытаться "втиснуться" в заданные размеры за счет форматирования текста. В случае когда это невозможно, например, в таблице находятся изображения, параметр width будет проигнорирован, и новая ширина таблицы будет вычислена на основе ее содержимого. Как и в случае с высотой, если ширина явно не указана, то она будет вычисляться на основе содержимого таблицы.

Каждая ячейка таблицы, задаваемая через тег TD, в свою очередь, тоже имеет свои параметры, часть из которых совпадает с параметрами тега TABLE.

- align. Задает выравнивание содержимого ячейки по горизонтали. Возможные значения: left — выравнивание по левому краю, center — по центру и right — по правому краю.
- background. Определяет изображение, которое будет использоваться в качестве фонового рисунка ячейки таблицы. Если фон уже установлен для всей таблицы в теге TABLE, то в ячейке он будет накладываться поверх основного, закрывая его. В некоторых случаях нижний фон может проглядывать сквозь верхний, например, если установлен параметр cellspacing, отличный от нуля, или в качестве фонового изображения используется формат GIF с прозрачными участками.
- bgcolor. Устанавливает цвет фона ячейки. Используя этот параметр совместно с атрибутом bgcolor тега TABLE, можно получить разнообразные цветовые эффекты в таблице.
- bordercolor. Устанавливает цвет рамки вокруг ячейки. Рамка отображается, когда установлен параметр border с ненулевым значением тега TABLE.
- colspan. Устанавливает число ячеек, которые должны быть объединены по горизонтали. Этот параметр имеет смысл для таблиц, состоящих из нескольких строк. Например, как для таблицы, показанной на рис. 6.2.

В данной таблице содержатся две строки и две колонки, причем верхние горизонтальные ячейки объединены с помощью параметра `colspan`.

Рис. 6.2. Пример таблицы, в которой используется горизонтальное объединение ячеек

- `height`. Браузер сам устанавливает высоту таблицы и ее ячеек исходя из их содержимого. Однако при использовании параметра `height` высота ячеек будет изменена. Здесь возможны два варианта. Если значение `height` меньше, чем содержимое ячейки, то этот параметр будет проигнорирован. В случае, когда установлена высота ячейки, превышающая ее содержимое, добавляется пустое пространство по вертикали.
- `nowrap`. Добавление параметра `nowrap` к тегу `TD` заставляет текст внутри ячейки отображаться без переносов, одной сплошной строкой. Неправильное использование этого атрибута может привести к тому, что таблица будет слишком широкой и не поместится целиком на веб-страницу. Как следствие, появится горизонтальная полоса прокрутки. В любом случае, пользоваться подобной таблицей будет неудобно, поэтому применение параметра `nowrap` осуждается в спецификации HTML 4.
- `rowspan`. Устанавливает число ячеек, которые должны быть объединены по вертикали. Этот параметр имеет смысл для таблиц, состоящих из нескольких строк. Например, как для таблицы, показанной на рис. 6.3. В ней содержатся две строки и две колонки, левые вертикальные ячейки объединены с помощью параметра `rowspan`.

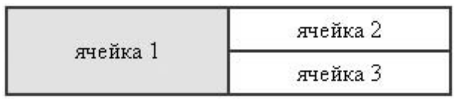


Рис. 6.3. Пример таблицы, в которой применяется вертикальное объединение ячеек

- `valign`. Устанавливает вертикальное выравнивание содержимого ячейки. По умолчанию контент ячейки располагается по ее вертикали в центре. Возможные значения: `top` — выравнивание по верхнему краю строки, `middle` — выравнивание посередине, `bottom` — выравнивание по нижнему краю, `baseline` — выравнивание по базовой линии, при этом происходит привязка содержимого ячейки к одной линии.
- `width`. Задает ширину ячейки. Суммарное значение ширины всех ячеек может превышать общую ширину таблицы только в том случае, если содержимое ячейки превышает указанную ширину.

Особенности таблиц

У каждого параметра таблицы есть свое значение, установленное по умолчанию. Из этого следует, что если какой-то атрибут пропущен, то неявно он все равно присутствует, причем с некоторым значением. Поэтому вид таблицы может оказаться совсем другим, чем предполагал разработчик. Чтобы понимать, что можно ожидать от таблиц, следует знать их явные и неявные особенности, которые перечислены далее.

- Одну таблицу допускается помещать внутрь ячейки другой таблицы. Это требуется для представления сложных данных или в том случае, когда одна таблица выступает в роли модульной сетки, а вторая, внутри нее, в роли обычной таблицы.
- Размеры таблицы изначально не установлены и вычисляются на основе содержимого ячеек. Например, общая ширина определяется автоматически исходя из суммарной ширины содержимого ячеек плюс ширина границ между ячейками, поля вокруг содержимого, устанавливаемые через параметр `cellpadding`, и расстояние между ячейками, которое определяется значением `cellspacing`.
- Если для таблицы задана ее ширина в процентах или пикселах, то содержимое таблицы подстраивается под указанные размеры. Так, браузер автоматически добавляет переносы строк в текст, чтобы он полностью поместился в ячейку, и при этом ширина таблицы осталась без изменений. Бывает, что ширину содержимого ячейки невозможно изменить, как это, например, происходит с рисунками. В этом случае ширина таблицы увеличивается несмотря на указанные размеры.
- Рамка таблицы в случае добавления параметра `border` к тегу `TABLE` изначально отображается как трехмерная. Присоединение параметра `bordercolor` превращает рамку в однотонную, ликвидируя тем самым эффект трехмерности.
- Пока таблица не загрузится полностью, ее содержимое не начнет отображаться. Дело в том, что браузер, прежде чем показать содержимое таблицы, должен вычислить необходимые размеры ячеек, их ширину и высоту. А для этого нужно знать, что в этих ячейках находится. Поэтому браузер и ожидает, пока загрузится все, что находится в ячейках, и только потом отображает таблицу.

Заголовок таблицы

При большом количестве таблиц на странице каждой из них удобно задать заголовок, содержащий название таблицы и ее описание. Для этой цели в HTML существует специальный тег `CAPTION`, который устанавливает текст

и его положение относительно таблицы. Проще всего размещать текст по центру таблицы сверху или снизу от нее; в остальных случаях браузеры поразному интерпретируют параметры тега CAPTION, поэтому результат получается неодинаковый. По умолчанию заголовок помещается сверху таблицы по центру, его ширина не превышает ширины таблицы, и в случае длинного текста он автоматически переносится на новую строку. Для изменения положения заголовка существует параметр align, который может принимать следующие значения:

- left — выравнивает заголовок по левому краю таблицы. Аргумент работает только в браузерах Internet Explorer и Opera 7, Netscape пренебрегает этим параметром и устанавливает текст по центру;
- right — в браузере Internet Explorer и Opera 7 располагает заголовок сверху таблицы и выравнивает его по правому краю таблицы. В браузере Netscape параметр игнорируется, а Opera версии 6 и ниже отображает заголовок справа от таблицы;
- center — заголовок располагается сверху таблицы по ее центру. Такое расположение задано в браузерах по умолчанию;
- top — результат аналогичен действию параметра center, но в отличие от него входит в спецификацию HTML 4 и понимается всеми браузерами;
- bottom — заголовок размещается внизу таблицы по ее центру.

Как видно из описания значений параметра align, получить универсальный код, одинаково работающий в разных браузерах, в случае расположения заголовка по правому или левому краю довольно проблематично. Тогда нам на помощь придут стили, в частности атрибут text-align. Его задача — заставить текст выравниваться по нужному краю. В листинге 6.2 показано, как установить заголовок сверху таблицы и выровнять его по ее правому краю. Обратите внимание, что тег CAPTION находится внутри контейнера TABLE, это его стандартное местоположение.

Листинг 6.2. Создание заголовка таблицы с помощью тега CAPTION

```
<html>
<head>
<style type="text/css">
 CAPTION {
  text-align: right;          /* Выравнивание по правому краю */
  font-style: italic          /* Курсивный текст */
}
</style>
</head>
<body>
```

```
<table width=70% border=1 cellpadding=4 cellspacing=0 align=center>
 <caption>Изменение добычи ресурсов по годам</caption>
  <tr>
    <th> </th>
    <th>2003</th><th>2004</th><th>2005</th>
  </tr>
  <tr>
    <td>Нефть</td>
    <td>43</td><td>51</td><td>79</td>
  </tr>
  <tr>
    <td>Золото</td>
    <td>29</td><td>34</td><td>48</td>
  </tr>
  <tr>
    <td>Дерево</td>
    <td>38</td><td>57</td><td>36</td>
  </tr>
</table>
</body>
</html>
```

На рис. 6.4 показан результат данного примера. Заметьте, что заголовок выравнивается не строго по правому краю таблицы, поскольку на него, как и на содержимое ячеек, действует параметр `cellpadding`. Можно представить, что заголовок — это еще одна ячейка таблицы, на которую распространяются характеристики `cellpadding` и `cellspacing`.

Рис. 6.4. Вид заголовка таблицы

Удобство использования тега CAPTION состоит в том, что заголовок, созданный с его помощью, оказывается привязанным к таблице и не выходит за условные рамки, ограниченные ее шириной. Тем не менее такого же результата можно добиться и с помощью стилей, как показано в листинге 6.3.

Листинг 6.3. Создание заголовка таблицы с помощью стилей

```
<html>
<head>
<style type="text/css">
 .caption {
  margin: 0px 15%;           /* Отступы сверху и сбоку от текста */
  padding-bottom: 4px;       /* Поле под текстом */
  text-align: right;         /* Выравнивание по правому краю */
  font-style: italic         /* Курсивный текст */
}
</style>
</head>
<body>
<p align=center class=caption>Изменение добычи ресурсов по годам</p>
<table width=70% border=1 cellpadding=4 cellspacing=0 align=center>
 ...
</table>
</body>
</html>
```

В этом примере создается новый класс с именем caption, который применяется к параграфу (тегу P). В данном случае таблица выравнивается по центру веб-страницы, поэтому то же самое должно произойти и с параграфом. Для этой цели добавляются отступы слева и справа через параметр margin. Ширину при этом явно не указываем, она получается вычитанием из общей ширины окна (100 %) удвоенного отступа слева (в примере 15 %). Таким образом, выходит, что ширина текстового блока совпадает с шириной таблицы. В случае использования пикселов в качестве единиц измерения рекомендуется обратиться к *главе 8* или воспользоваться тегом CAPTION. Расстояние от таблицы до текста в примере регулируется значением padding-bottom.

Выравнивание таблиц

Для задания выравнивания таблицы по центру веб-страницы или по одному из ее краев предназначен параметр align тега TABLE. Результат будет заметен только в том случае, если ширина таблицы не занимает всю доступную об-

ласть, другими словами, меньше 100 %. На самом деле `align` не только устанавливает выравнивание, но и заставляет текст обтекать таблицу с других сторон аналогично поведению тега `IMG`. Но поскольку тег `TABLE` не имеет параметра `hspace`, задающего поле по горизонтали, то эту роль необходимо переложить на стили, в частности атрибут `margin`. В листинге 6.4 показано выравнивание таблицы по правому краю и ее обтекание текстом.

Листинг 6.4. Выравнивание таблицы по правому краю

```
<html>
<body>
 <table width=200 bgcolor=#c0c0c0 cellspacing=0 cellpadding=5 border=1
bordercolor=black align=right style="margin: 10px">
  <tr><td>Содержимое таблицы</td></tr>
 </table>
 Обтекающий таблицу текст...
</body>
</html>
```

В этом примере создается таблица с фоном серого цвета и выравниванием по правому краю. Для создания отступов от таблицы до текста используется параметр `margin` со значением 10 пикселов.

Чтобы запретить обтекание таблицы при ее выравнивании по одному из краев, проще всего добавить после таблицы тег `BR` с заданным стилем `clear: both`. Эта команда запрещает обтекание как с левого, так и с правого края (листинг 6.5).

Листинг 6.5. Отмена обтекания таблицы

```
<html>
<body>
 <table width=200 bgcolor=#c0c0c0 align=right>
  <tr><td>Содержимое таблицы</td></tr>
 </table>
 <br style="clear: both">
 <p>Текст...</p>
</body>
</html>
```

Выравнивание таблицы по центру выполняется аналогично, через параметр `align=center` тега `TABLE`, но в этом случае никакого обтекания не происходит, и текст после таблицы начинается всегда с новой строки.

Если применяется несколько таблиц на одной странице, то имеет смысл задать для всех них выравнивание через стили. Для этого к стилю таблицы следует добавить отступы через свойство `margin`, как показано в листинге 6.6.

Листинг 6.6. Выравнивание таблицы с помощью свойства `margin`

```
<html>
<head>
<style type="text/css">
 TABLE {
  margin: auto        /* Автоматические отступы вокруг таблицы */
}
</style>
</head>
<body>
 <table width=200 bgcolor=#c0c0c0 border=1 bordercolor=black>
  <tr><td>...</td></tr>
 </table>
</body>
</html>
```

Следует отметить, что данный пример не работает должным образом в браузере Internet Explorer, поэтому для него следует использовать свойство `text-align`, применяя его к селектору `BODY` (листинг 6.7).

Листинг 6.7. Выравнивание таблицы через атрибут `text-align`

```
<html>
<head>
<style type="text/css">
 BODY {
  text-align: center  /* Выравнивание таблицы в Internet Explorer */
}
 TABLE {
  margin: auto     /* Выравнивание таблицы в браузерах Opera и Netscape */
}
 P {
  text-align: left    /* Выравнивание текста по левому краю */
}
</style>
</head>
```

```
<body>
 <table width=200 bgcolor=#c0c0c0 border=1 bordercolor=black>
  <tr><td>...</td></tr>
 </table>
 <p>Текст</p>
</body>
</html>
```

Добавление свойства `text-align: center` к селектору `BODY` заставляет выравниваться по центру все содержимое веб-страницы, включая таблицы, поэтому для текста требуется установить дополнительное выравнивание по нужному краю. Одновременное использование атрибутов `margin` и `text-align` позволяет получить универсальный код, корректно работающий в разных браузерах.

Таблицы и рамки

Чтобы четко отделить содержимое одной ячейки от другой, к ячейкам добавляются границы. За их создание отвечает параметр `border` тега `TABLE`, который определяет толщину рамки, и `bordercolor`, устанавливающий ее цвет. Как было показано в *главе 5* (см. рис. 5.10), рамки, созданные с помощью свойств таблицы, получаются разными по своему виду в каждом браузере. Чтобы этого избежать, рекомендуется воспользоваться стилевым атрибутом `border`, применяя его к ячейкам таблицы (тегу `TD` или `TH`). Однако и здесь нас подстерегают подводные камни. Поскольку рамка создается для каждой ячейки, то в местах соприкосновения ячеек получается граница удвоенной толщины. Для устранения указанной особенности существует несколько путей. Самый простой — воспользоваться свойством `border-collapse` со значением `collapse`. В его задачу входит отслеживать соприкосновение линий и вместо двойной границы изображать одинарную. Этот атрибут достаточно добавить к тегу `TABLE`, а дальше он все сделает самостоятельно (листинг 6.8).

Листинг 6.8. Применение свойства `border-collapse` при создании рамок таблицы

```
<html>
<head>
<style type="text/css">
TABLE {
border-collapse: collapse    /* Линия между ячейками
                                отображается как одна */
}
```

```
TD {
border: 2px solid green;      /* Параметры рамки вокруг ячеек таблицы */
text-align: center            /* Выравниваем текст по центру */
}
</style>
</head>
<body>
<table width=200 cellspacing=0 cellpadding=2>
<tr><td>0</td><td>X</td><td>X</td></tr>
<tr><td>0</td><td>0</td><td>X</td></tr>
<tr><td>X</td><td>X</td><td>0</td></tr>
</table>
</body>
</html>
```

Разница между границами таблицы при добавлении параметра `border-collapse`, а также без него, представлена на рис. 6.5.

0	X	X
0	0	X
X	X	0

а

0	X	X
0	0	X
X	X	0

б

Рис. 6.5. Вид таблицы при использовании атрибута `border-collapase`: *а* — атрибут не установлен; *б* — атрибут установлен

На рис. 6.5, *а* показана рамка таблицы, используемая по умолчанию. Обратите внимание, что внутри таблицы все линии имеют удвоенную толщину. Добавление параметра `border-collapse` убирает эту особенность, и толщина всех линий становится одинаковой (рис. 6.5, *б*).

Для создания однотипных линий внутри таблицы можно пойти и другим путем. Следует добавить для селектора `TD` рамку, но отменить линию справа и снизу, задавая соответствующим атрибутам значение `none`. Тогда при состыковке ячеек их границы не будут накладываться друг на друга по той причине, что линия будет только одна. Однако при этом методе формирования границ нет линий снизу и справа у самой таблицы. Добавляя параметры `border-right` и `border-bottom` к селектору `TABLE`, получим в итоге нужную рамку (листинг 6.9). Для однотипности следует позаботиться о том, чтобы стиль, толщина и цвет границы во всех случаях совпадали.

Листинг 6.9. Создание рамки у таблицы с помощью атрибута `border`

```
<html>
<head>
```

```
<style type="text/css">
 TABLE {
  border-right: 2px solid green;    /* Граница у таблицы справа */
  border-bottom: 2px solid green;   /* Граница у таблицы снизу */
}
 TD {
  border: 2px solid green;   /* Параметры рамки вокруг ячеек таблицы */
  border-right: none;        /* Убираем линию справа */
  border-bottom: none;       /* Убираем линию снизу */
  text-align: center         /* Выравниваем текст по центру */
}
</style>
</head>
<body>
<table width=200 cellspacing=0 cellpadding=2>
  <tr><td>0</td><td>X</td><td>X</td></tr>
  <tr><td>0</td><td>0</td><td>X</td></tr>
  <tr><td>X</td><td>X</td><td>0</td></tr>
</table>
</body>
</html>
```

У этого способа возможны вариации, например, для селектора `TD` добавить границу только справа и снизу, а у `TABLE`, наоборот, добавить атрибут `border`, но линию справа и снизу убрать. В любом случае представленный результат будет один.

Простой и оригинальный вид таблицы можно получить, если цвет границ сделать совпадающим с цветом фона. Но чтобы линии были видны, обязательно следует сделать заливку фона у тега `TD` или `TABLE`. Тогда ячейки таблицы получаются словно рассеченные резцом между собой (рис. 6.6).

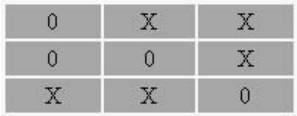


Рис. 6.6. Границы внутри таблицы белого цвета

Данный пример представлен в листинге 6.10.

Листинг 6.10. Использование невидимых границ в таблице

```
<html>
<head>
```

```
<style type="text/css">
 BODY {
  background: white          /* Цвет фона веб-страницы */
}
 TABLE {
  border-collapse: collapse  /* Линия между ячейками
                             отображается как одна */
}
 TD {
  border: 2px solid white;   /* Параметры рамки вокруг ячеек таблицы */
  background: #ccc;          /* Цвет фона ячеек */
  text-align: center         /* Выравниваем текст по центру */
}
</style>
</head>
<body>
<table width=200 cellspacing=0 cellpadding=2>
  <tr><td>0</td><td>X</td><td>X</td></tr>
  <tr><td>0</td><td>0</td><td>X</td></tr>
  <tr><td>X</td><td>X</td><td>0</td></tr>
</table>
</body>
</html>
```

В примере цвет фона веб-страницы вводится через свойство `background`, добавленное к селектору `BODY`. Хотя белый цвет установлен по умолчанию, его иногда желательно задавать, чтобы пользователь не указал свой цвет фона через настройки браузера. Такой же цвет должен быть и у линий таблицы, в этом случае они явно не выделяются и лишь разделяют ячейки между собой.

Цвет фона ячеек также определяется атрибутом `background`, применяемым к селектору `TD`.

Таблицы с фиксированным заголовком

При ограничении места на веб-странице можно зафиксировать высоту таблицы и добавить полосы прокрутки, с помощью которых прокручивается ее содержимое. Кроме того, если верхнюю строку таблицы закрепить на одном месте, то становится удобно просматривать содержимое длинной таблицы (рис. 6.7).

Чтобы создать конструкцию, подобную приведенной на рисунке, потребуется две таблицы и два контейнера `DIV`. Первый `DIV`, назовем его `container`, не

обязателен, но требуется для того, чтобы добавить рамку вокруг области и установить общую ширину. Второй DIV, с именем scroll с помещенной внутрь него таблицей, задает полосы прокрутки и высоту отображения данных. Разделение одной таблицы на две необходимо, поскольку внутрь тега TABLE вставить DIV не получится, можно сделать только наоборот. Для наглядности вложенность тегов представлена на рис. 6.8.

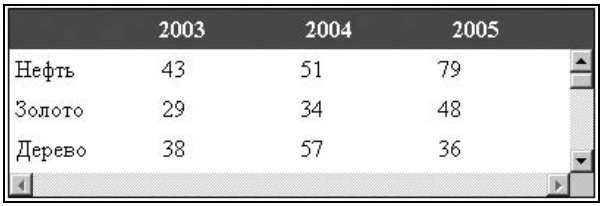


Рис. 6.7. Вид таблицы с фиксированным заголовком

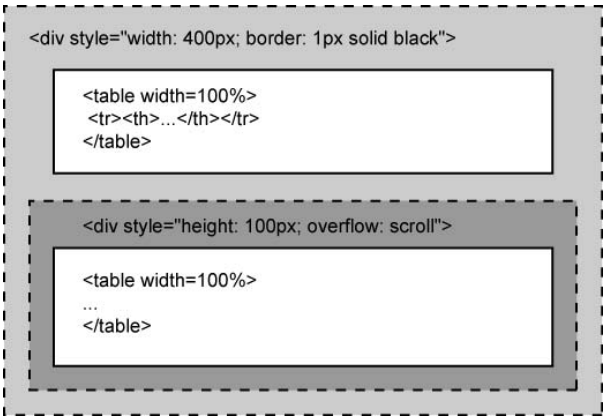


Рис. 6.8. Порядок тегов для создания таблицы с заголовком

Окончательный полученный код приведен в листинге 6.11.

Листинг 6.11. Создание таблицы с фиксированным заголовком

```
<html>
<head>
<style type="text/css">
 TH {
  background: #666;          /* Фон заголовка */
  color: white;              /* Цвет текста */
  text-align: left           /* Выравнивание текста заголовка */
}
```

```
 DIV.container {
  border: 1px solid black;   /* Рамка вокруг таблицы */
  width: 400px               /* Ширина таблицы в пикселах */
}
 DIV.scroll {
  height: 100px;             /* Высота области прокрутки */
  overflow: scroll           /* Добавляем полосы прокрутки */
}
</style>
</head>
<body>
<div class=container>
<table width=100% border=0 cellspacing=0 cellpadding=4>
  <tr>
    <th width=100> </th>
    <th width=100>2003</th>
    <th width=100>2004</th>
    <th width=100>2005</th>
  </tr>
</table>
<div class=scroll>
 <table width=100% border=0 cellspacing=0 cellpadding=4>
  <tr>
    <td width=100>Нефть</td>
    <td width=100>43</td>
    <td width=100>51</td>
    <td width=100>79</td>
  </tr>
  <tr>
    <td>Золото</td><td>29</td><td>34</td><td>48</td>
  </tr>
  <tr>
    <td>Дерево</td><td>38</td><td>57</td><td>36</td>
  </tr>
 </table>
</div>
</div>
</body>
</html>
```

Так как две таблицы — заголовок и содержимое — создаются независимо, то для совмещения колонок требуется задать ширину каждой колонки с помощью параметра `width`.

Ускорение загрузки таблиц

Использование таблицы, в которой хранится большой объем информации, замедляет отображение табличных данных на веб-странице. Это связано с тем, что браузер, прежде чем показать содержимое таблицы, должен вычислить необходимые размеры ячеек, их ширину и высоту. А для этого необходимо знать, что же в этих ячейках находится. Вот браузер и ожидает, пока загрузится все, что помещено в ячейках, и только потом форматирует и отображает таблицу. Этот процесс можно ускорить, если придерживаться следующих принципов.

- Указывайте ширину ячеек с помощью параметра `width` тега `TD` или через стили. Ширину не обязательно задавать для каждой ячейки, достаточно это сделать только для первой строки таблицы. Остальные ячейки будут подстраиваться под заданный размер.
- Применяйте тег `COL`, который определяет ширину и другие атрибуты одной или нескольких колонок таблицы. При наличии этого тега браузер начинает показывать содержимое таблицы, не дожидаясь ее полной загрузки. Тег `COL` можно использовать совместно с тегом `COLGROUP`, который задает группу колонок, обладающих общими параметрами.
- Добавление параметра `cols` тега `TABLE` устанавливает число колонок в таблице и позволяет браузеру производить меньше вычислений при ее загрузке. Как следствие, происходит небольшое ускорение отображения ячеек. Этот параметр указывать не обязательно при наличии тегов `COL` или `COLGROUP`.
- Используйте стилевой параметр `table-layout`, при установленном значении `fixed` он повышает производительность построения таблиц. Ширина колонок в этом случае определяется либо с помощью тега `COL`, либо вычисляется на основе первой строки. Если данные о форматировании первой строки таблицы по каким-либо причинам получить невозможно, таблица делится на колонки равной ширины.

В листинге 6.12 показан пример таблицы, в которой для указания ширины и цвета фона сразу у нескольких колонок используются теги `COL` и `COLGROUP`. Наряду с ними добавлен атрибут `table-layout: fixed`, он говорит браузеру о том, что размеры колонок таблицы фиксированы и определены заранее. В этом случае отображение таблицы начинается сразу же с ее первых строк. Следует заметить, что параметр `bgcolor` тега `COL` поддерживается только браузером Internet Explorer, а остальными браузерами будет проигнорирован.

Листинг 6.12. Использование в таблице колонок фиксированной ширины

```
<html>
<body>
```

```
<table width=550 cellpadding=2 cellspacing=0 border=1 bordercolor=black
rules=groups style="table-layout: fixed">
<colgroup width=100>
<colgroup span=9 align=center width=50>
 <col span=5 bgcolor=#c0c0c0>
 <col span=4 bgcolor=#ffffdd>
</colgroup>
  <tr>
   <td> </td><td>1995</td><td>1996</td><td>1997</td>
   <td>1998</td><td>1999</td><td>2000</td><td>2001</td>
   <td>2002</td><td>2003</td>
  </tr>
  <tr>
   <td>Нефть</td><td>5</td><td>7</td><td>2</td><td>8</td>
   <td>3</td><td>34</td><td>62</td><td>74</td><td>57</td>
  </tr>
  <tr>
   <td>Золото</td><td>3</td> <td>6</td><td>4</td><td>6</td>
   <td>4</td><td>69</td><td>72</td><td>56</td><td>47</td>
  </tr>
  <tr>
  <td>Дерево</td><td>5</td><td>8</td><td>3</td><td>4</td>
   <td>7</td><td>73</td><td>79</td><td>34</td><td>86</td>
  </tr>
 </table>
</body>
</html>
```

Обычно закрывать тег COLGROUP не требуется, но если он выступает как контейнер для элементов COL, тогда следует добавить тег </COLGROUP> в конце группы.

Разница между свойствами тегов COLGROUP и COL не очень велика и состоит в следующем. COLGROUP позволяет объединять колонки в определенные группы, также при добавлении к тегу TABLE параметра rules=groups браузер будет рисовать линию только между колонками, созданными с помощью COLGROUP. В остальных случаях поведение колонок, назначенных через элементы COLGROUP и COL, идентично.

Выделение ячеек таблицы курсором мыши

Выделение строки таблицы при наведении на нее курсора мыши, или подсветка строк, как еще называют этот эффект, позволяет весьма наглядно пометить ячейки таблицы. Выделение происходит динамически, при наве-

дении курсора на строку меняется цвет ее фона, а когда курсор уводится прочь, фон снова восстанавливается на первоначальный. На рис. 6.9 продемонстрировано, как изменяется стиль горизонтальных ячеек, когда курсор мыши наведен на одну из них.

Рис. 6.9. Подсветка строк таблицы

Стандартные средства HTML и CSS для создания подобного эффекта не годятся, поэтому нам придется обратиться к скриптам. Алгоритм основан на использовании CSS и DOM (Document Object Model, объектная модель документа) для доступа и изменения свойств ячеек.

Вначале следует задать стиль таблицы. Чтобы эффект работал только для определенных таблиц, им лучше присвоить свое собственное имя, например `ruler`. Это делать не обязательно, если все таблицы предстоит оформить одинаково.

Поскольку заголовок таблицы тоже является строкой, то, чтобы он не менял цвет фона при наведении на него курсора мыши, ему следует присвоить свое собственное имя, например `header`. Для выделенных строк также указывается свой стиль, назовем его `line`, в нем можно определить цвет фона, текста и другие желаемые характеристики.

Со стилем закончили и теперь переходим к скрипту. В нашей функции `tableRuler()` вначале определяется, поддерживает ли браузер метод `getElementById`; именно с помощью него можно узнать, будет работать скрипт в браузере или нет. Хотя существуют и другие подходы для изменения свойств ячеек, доступ к объектам через `getElementById` соответствует современным стандартам и поддерживается всеми последними версиями браузеров.

После этого в цикле идет проверка всех таблиц на веб-странице, и дальнейшая работа происходит только с теми таблицами, у которых в теге `TABLE` установлен `class=ruler`. Это сделано для того, чтобы отсечь любые другие

таблицы, для которых применение подсветки не желательно. Для выбранных таблиц снова заводится цикл, в котором просматриваются уже все строки (тег TR). Отбрасываем строку TR class=header (она служит заголовком таблицы, и ее обрабатывать не надо), для остальных отслеживаем события мыши onmouseover и onmouseout. Первое событие отвечает за наведение курсора мыши на строку таблицы, в этом случае имя класса стиля меняется на line. Событие onmouseout выполняется в случае вывода курсора из строки, при этом ее стиль восстанавливается.

Сам скрипт описывается внутри контейнера SCRIPT. Окончательный код для создания подсветки строк таблицы приведен в листинге 6.13.

Листинг 6.13. Создание подсветки строк таблицы

```
<html>
<head>
<style type="text/css">
 TABLE.ruler {
  width: 100%;                /* Ширина таблицы в процентах */
  border: 1px solid navy      /* Рамка вокруг таблицы */
}
 TD {
  padding: 4px                /* Поля вокруг содержимого ячеек */
}
 TR.line {                    /* Стиль выделенной строки */
  background: #fc0;           /* Фон под выделенной строкой */
  color: #333                 /* Цвет текста выделенной строки */
}
  .header {                   /* Стиль заголовка */
  background: navy;           /* Цвет фона заголовка */
  color: white                /* Цвет текста в заголовке */
}
</style>
<script language="JavaScript">
function tableRuler() {

// Проверяем, поддерживает ли текущий браузер DOM
  if (document.getElementById) {
    tables = document.getElementsByTagName('table')
// Пробегаемся по всем таблицам на странице
    for (i=0;i<tables.length;i++) {

// Работаем только с теми таблицами,
// у которых установлен класс с именем ruler
      if (tables[i].className == 'ruler') {
        trs = tables[i].getElementsByTagName('tr')
```

```
// Пробегаемся по всем строкам выбранной таблицы
        for (j=0;j<trs.length;j++) {

// Для заголовка таблицы цвет не меняем
          if (trs[j].className != 'header') {

// В остальных случаях изменяем имя стиля строки TR на line
            trs[j].onmouseover = function() { this.className = 'line'; ↳
                                              return false }
            trs[j].onmouseout = function() { this.className = ''; ↳
                                              return false }
          }
        }
      }
    }
  }
}
</script>
</head>
<body onLoad="tableRuler()">
 <table class=ruler>
  <tr class=header>
    <th> </th><th>2003</th><th>2004</th><th>2005</th>
  </tr>
  <tr>
    <td>Нефть</td><td>43</td><td>51</td><td>79</td>
  </tr>
  <tr>
    <td>Золото</td><td>29</td><td>34</td><td>48</td>
  </tr>
  <tr>
    <td>Дерево</td><td>38</td><td>57</td><td>36</td>
  </tr>
 </table>
</body>
</html>
```

В браузере Opera выделенная строка таблицы имеет не прямоугольную форму, а содержит выступы в местах, где встречается текст. Чтобы обойти эту особенность, уберите свойство `padding` из стилей и добавьте к тегу `TABLE` параметр `cellpadding` с тем же значением.

Шаблоны таблиц

Для наглядного представления информации к таблицам рекомендуется применять дизайн, а не оставлять их вид по умолчанию. В этом случае таблицы будут хорошо вписываться в общий вид веб-страницы, и с ними будет приятно работать. Далее представлены некоторые таблицы, оформление которых задается с помощью стилей. Названия у них условные и приведены лишь для удобства.

Простая таблица

Характеризуется горизонтальными линиями, выделяющими заголовок таблицы и ее нижний край (рис. 6.10).

	2003	2004	2005
Нефть	43	51	79
Золото	29	34	48
Дерево	38	57	36

Рис. 6.10. Вид простой таблицы

Поскольку содержимое ячейки, созданной с помощью тега TH, по умолчанию выравнивается по центру, в примере для таких ячеек установлено выравнивание по левому краю (листинг 6.14).

Листинг 6.14. Использование горизонтальных линий в таблице

```
<html>
<head>
<style type="text/css">
 TABLE {
  border-top: 1px solid black;       /* Линия вверху таблицы */
  border-bottom: 1px solid black     /* Линия внизу таблицы */
}
 TH {
  border-bottom: 1px solid black;    /* Линия внизу заголовка */
  text-align: left                   /* Выравнивание по левому краю */
}
</style>
</head>
<body>
 <table width=300 cellspacing=0 cellpadding=4>
```

```
  <tr><th> </th><th>2003</th><th>2004</th><th>2005</th></tr>
  <tr><td>Нефть</td><td>43</td><td>51</td><td>79</td></tr>
  <tr><td>Золото</td><td>29</td><td>34</td><td>48</td></tr>
  <tr><td>Дерево</td><td>38</td><td>57</td><td>36</td></tr>
 </table>
</body>
</html>
```

Простая таблица с заголовком

В этой таблице заголовок выделен путем использования выворотки — белый текст на темном фоне. Вокруг самой таблицы добавлена тонкая рамка (рис. 6.11).

	2003	2004	2005
Нефть	43	51	79
Золото	29	34	48
Дерево	38	57	36

Рис. 6.11. Вид простой таблицы с заголовком

Выравнивание содержимого ячеек происходит по центру путем применения атрибута `text-align: center`, заданного для селектора `TABLE`. Обратите внимание, что при таком использовании `text-align` выравнивается не сама таблица, а лишь ее контент. Для крайних левых ячеек, где выравнивание по центру не требуется, создается новый класс с именем `title`, в котором указаны требуемые параметры (листинг 6.15).

Листинг 6.15. Выделение ячеек с помощью фона

```
<html>
<head>
<style type="text/css">
 TABLE {
  border: 1px solid black;          /* Рамка вокруг таблицы */
  text-align: center          /* Выравнивание текста ячеек по центру */
}
 TH {
  background: black;          /* Цвет фона заголовка */
  color: white                /* Цвет текста в заголовке */
}
```

```
 TD.title {
  text-align: left           /* Выравнивание по левому краю */
}
</style>
</head>
<body>
 <table width=300 border=0 cellspacing=0 cellpadding=4>
  <tr><th> </th><th>2003</th><th>2004</th><th>2005</th></tr>
  <tr><td class=title>Нефть</td><td>43</td><td>51</td><td>79</td></tr>
  <tr><td class=title>Золото</td><td>29</td><td>34</td><td>48</td></tr>
  <tr><td class=title>Дерево</td><td>38</td><td>57</td><td>36</td></tr>
 </table>
</body>
</html>
```

Таблица с цветным заголовком

Этот вариант таблицы похож на предыдущий, но левая колонка выделена с помощью фона, и строки таблицы отделены друг от друга пунктирной линией (рис. 6.12).

	2003	2004	2005
Нефть	43	51	79
Золото	29	34	48
Дерево	38	57	36

Рис. 6.12. Вид таблицы с цветным заголовком

Для создания пунктирной линии внизу ячеек у селектора `TD` добавляется атрибут `border-bottom` со значением `dashed` (листинг 6.16). Использование линии того же цвета, что и фон левой колонки, дает разный вид в браузерах. Происходит это за счет того, что Internet Explorer промежутки в пунктирной линии заполняет белым цветом, а остальные браузеры оставляют их прозрачными. Применение свойства `border-collapse` необходимо, чтобы пунктирная линия в последних ячейках не совместилась с нижней границей таблицы.

Листинг 6.16. Добавление пунктирной линии между строками

```
<html>
<head>
<style type="text/css">
```

```
 TABLE {
  border: 1px solid black;   /* Рамка вокруг таблицы */
  text-align: center;        /* Выравнивание текста ячеек по центру */
  border-collapse: collapse  /* Линия между ячейками
                             отображается как одна */
}
 TH {
  background: darkblue;              /* Цвет фона заголовка */
  border-bottom: 1px solid white;    /* Линия под заголовком */
  color: white                       /* Цвет текста заголовка */
}
 TD {
  border-bottom: 1px dashed #ccc     /* Пунктирная линия под ячейками */
}
 TD.title {
  text-align: left;          /* Выравнивание по левому краю */
  font-weight: bold;         /* Жирное начертание */
  background: #ccc           /* Цвет фона левой колонки */
}
</style>
</head>
<body>
<table width=300 border=0 cellspacing=0 cellpadding=4>
  <tr><th> </th><th>2003</th><th>2004</th><th>2005</th></tr>
  <tr><td class=title>Нефть</td><td>43</td><td>51</td><td>79</td></tr>
  <tr><td class=title>Золото</td><td>29</td><td>34</td><td>48</td></tr>
  <tr><td class=title>Дерево</td><td>38</td><td>57</td><td>36</td></tr>
</table>
</body>
</html>
```

Таблица с чередующимися строками

Для удобного представления строчных данных имеет смысл делать строки таблицы чередующимися, чтобы цвет фона четных и нечетных строк различался (рис. 6.13). Кроме цвета фона, можно изменять и другие характеристики, например цвет текста.

Для изменения цвета фона у определенных строк следует ввести новый класс и применять его для четных или нечетных строк. В листинге 6.17 этот класс называется `even`, и он добавляется к тегу `TR` для всех строк через одну. Цвет фона остальных строк устанавливается в селекторе `BODY`. Это сделано с учетом иерархии объектов в таблице. Наибольший приоритет имеет тег `TD`, затем идет `TR` и только после этого `TABLE`. Это значит, что если задать цвет

фона для ячейки, то поменять его путем добавления свойства `background` к селектору `TR` или `TABLE` не удастся. Но допустимо обратное, как это показано в данном примере. А именно, устанавливая цвет фона ячеек через селектор `TABLE`, можно легко поменять цвет всей строки, если присвоить тегу `TR` класс `even`, в котором определяется свой параметр `background`.

	2003	2004	2005
Нефть	43	51	79
Золото	29	34	48
Дерево	38	57	36
Камни	17	26	92

Рис. 6.13. Таблица с разными четными и нечетными строками

Листинг 6.17. Изменение стиля строк

```
<html>
<head>
<style type="text/css">
 TABLE {
  border-bottom: 2px solid black;    /* Линия внизу таблицы */
  background: oldlace;               /* Цвет фона нечетных строк */
  border-collapse: collapse          /* Линия между ячейками
                                    отображается как одна */
}
 TH {
  background: firebrick;             /* Цвет фона заголовка */
  border-bottom: 2px solid black;    /* Линия под заголовком */
  color: white                       /* Цвет текста заголовка */
}
 TD {
  text-align: center;         /* Выравнивание текста ячеек по центру */
  border-bottom: 1px dotted firebrick      /* Пунктирная линия
                                          между строками */
}
 TD.title {
  text-align: left;           /* Выравнивание текста по левому краю */
  font-style: italic;         /* Курсивный текст */
  font-weight: bold;          /* Жирное начертание */
}
 TR.even {
  background: white;          /* Цвет фона у четных строк */
```

```
  color: firebrick           /* Цвет текста четных строк */
}
</style>
</head>
<body>
<table width=300 border=0 cellspacing=0 cellpadding=4>
  <tr><th> </th><th>2003</th><th>2004</th><th>2005</th></tr>
  <tr><td class=title>Нефть</td><td>43</td><td>51</td><td>79</td></tr>
  <tr class=even>
    <td class=title>Золото</td><td>29</td><td>34</td><td>48</td></tr>
  <tr><td class=title>Дерево</td><td>38</td><td>57</td><td>36</td></tr>
  <tr class=even>
    <td class=title>Камни</td><td>17</td><td>26</td><td>92</td></tr>
</table>
</body>
</html>
```

Таблица с выделенными колонками

Колонки удобно выделять цветом фона или вертикальными линиями, чтобы взгляд читателя скользил по ним сверху вниз, не перескакивая на соседний раздел (рис. 6.14). При этом цвет четных и нечетных колонок может различаться.

	2003	2004	2005
Нефть	43	51	79
Золото	29	34	48
Дерево	38	57	36

Рис. 6.14. Вид таблицы с выделенными колонками

Для создания таблицы с выделенными колонками существует несколько способов. Самый простой — это добавить сразу после `TABLE` несколько тегов `COL` по числу колонок и для них установить свой собственный стиль (листинг 6.18). Класс `title` предназначен для оформления самой левой колонки с заголовками строк, `odd` — для всех следующих нечетных колонок, а `even` — для четных.

Листинг 6.18. Использование тега `COL` для оформления колонок

```
<html>
<head>
```

```
<style type="text/css">
 TABLE {
  border: 1px solid black           /* Рамка вокруг таблицы */
}
 TH {
  text-align: left;                 /* Выравнивание по левому краю */
  border-bottom: 4px double black   /* Двойная рамка под заголовком */
}
 COL.title {
  background: #f0f0f0               /* Цвет фона левой колонки */
}
 COL.odd {
  background: #fec688               /* Цвет фона нечетных колонок */
}
 COL.even {
  background: #fbdebc               /* Цвет фона четных колонок */
}
</style>
</head>
<body>
 <table width=300 cellspacing=0 cellpadding=4>
  <col class=title>
  <col class=odd>
  <col class=even>
  <col class=odd>
  <tr><th> </th><th>2003</th><th>2004</th><th>2005</th></tr>
  <tr><td>Нефть</td><td>43</td><td>51</td><td>79</td></tr>
  <tr><td>Золото</td><td>29</td><td>34</td><td>48</td></tr>
  <tr><td>Дерево</td><td>38</td><td>57</td><td>36</td></tr>
 </table>
</body>
</html>
```

Хотя код в данном примере получился простым и удобным, следует заметить, что браузер Netscape (а также Mozilla и Firefox) не отображает стиль, в частности цвет фона у тега `COL`. Поэтому в этих браузерах колонки у таблицы никак не будут выделены. Чтобы исправить положение и создать универсальный код, можно пойти другим путем, например, отказаться от фоновой заливки и установить только вертикальные границы (рис. 6.15).

Вертикальные линии между колонками устанавливаются автоматически, если добавить параметр `rules=cols` к тегу `TABLE` (листинг 6.19). Цвет и толщину линии можно поменять только в браузере Internet Explorer, используя

параметры border и bordercolor того же тега TABLE. Остальные браузеры по умолчанию делают линии черного цвета и шириной в один пиксел.

	2003	2004	2005
Нефть	43	51	79
Золото	29	34	48
Дерево	38	57	36

Рис. 6.15. Использование разделительной вертикальной линии

Листинг 6.19. Использование параметра rules

```
<html>
<head>
<style type="text/css">
 TABLE {
  border: 1px solid black           /* Рамка вокруг таблицы */
}
 TH {
  text-align: left;                 /* Выравнивание по левому краю */
  background: #ddd;                 /* Цвет фона заголовка */
  border-bottom: 4px double black   /* Рамка внизу заголовка */
}
 .title {
  background: #bce18d;              /* Цвет фона левой колонки */
  font-style: italic                /* Курсивное начертание */
}
</style>
</head>
<body>
 <table width=300 cellspacing=0 cellpadding=4 rules=cols>
  <tr><th
class=title> </th><th>2003</th><th>2004</th><th>2005</th></tr>
  <tr><td class=title>Нефть</td><td>43</td><td>51</td><td>79</td></tr>
  <tr><td class=title>Золото</td><td>29</td><td>34</td><td>48</td></tr>
  <tr><td class=title>Дерево</td><td>38</td><td>57</td><td>36</td></tr>
 </table>
</body>
</html>
```

Вид таблиц с линиями, полученными с помощью параметра rules, незначительно отличается в разных браузерах, но в целом отображается корректно.

Вертикальные линии, а также цвет фона колонок можно изменять с помощью стилей, если создать отдельные классы и применять их к определенным ячейкам (рис. 6.16). В этом случае нам доступны все средства стилей, используемых для оформления, допустимо изменять толщину, цвет и стиль границ, фоновую картинку, цвет фона и т. д.

	2003	2004	2005
Нефть	43	51	79
Золото	29	34	48
Дерево	38	57	36

Рис. 6.16. Вид таблицы с цветными колонками

В листинге 6.20 показано добавление двух классов: `odd` предназначен для нечетных колонок, идущих сразу после колонки с заголовками строк, а `even` — для четных колонок. Слева от каждой колонки устанавливается пунктирная линия, чтобы разделение содержимого ячеек происходило не только по цвету, но и с помощью границы между ними.

Листинг 6.20. Выделение колонок таблицы разным цветом

```
<html>
<head>
<style type="text/css">
 TABLE {
  border: 1px solid black           /* Рамка вокруг таблицы */
}
 TH {
  text-align: left;          /* Выравнивание текста по левому краю */
  background: chocolate;     /* Цвет фона заголовка */
  color: bisque;             /* Цвет текста заголовка */
  border-bottom: 2px solid black    /* Граница внизу заголовка */
}
 .odd {
  background: #fec688;              /* Цвет фона нечетных колонок */
  border-left: 1px dashed chocolate        /* Линия слева от колонки */
}
 .even {
  background: #fbdebc;              /* Цвет фона четных колонок */
  border-left: 1px dashed chocolate        /* Линия слева от колонки */
}
```

```
</style>
</head>
<body>
 <table width=300 cellspacing=0 cellpadding=4>
  <tr><th> </th><th>2003</th><th>2004</th><th>2005</th></tr>
  <tr><td>Нефть</td><td class=odd>43</td><td class=even>51</td><td
class=odd>79</td></tr>
  <tr><td>Золото</td><td class=odd>29</td><td class=even>34</td><td
class=odd>48</td></tr>
  <tr><td>Дерево</td><td class=odd>38</td><td class=even>57</td><td
class=odd>36</td></tr>
 </table>
</body>
</html>
```

Поскольку формирование таблицы происходит по строкам, то для выделения колонок приходится указывать нужный класс для каждой ячейки колонки.

ГЛАВА 7

Формы

Форма предназначена для обмена данными между пользователем и сервером. Область применения форм не ограничена отправкой данных на сервер, с помощью клиентских скриптов можно получить доступ к любому элементу формы, изменять его и применять по своему усмотрению.

Для отправки формы на сервер используется кнопка `SUBMIT`, такого же эффекта можно добиться, если нажать клавишу <Enter> в пределах формы.

Когда форма отправляется на сервер, управление данными передается *CGI-программе*, заданной параметром `action` тега `FORM`. Аббревиатурой CGI (Common Gateway Interface, общий шлюзовый интерфейс) обозначается протокол, с помощью которого программы взаимодействуют с веб-сервером. С помощью CGI на сервере можно выполнять программы на любом языке программирования и результат их действия выводить в виде веб-страницы. Наиболее популярны следующие языки — Perl, PHP, C. Программа получает данные, введенные пользователем в форме, и дальше производит с ними некоторые манипуляции, которые заложены разработчиком.

Перед отправкой данных браузер подготавливает информацию в виде пары "имя=значение", где имя определяется параметром `name` тега `INPUT` или другим допустимым в форме, а значение введено пользователем или установлено в поле формы по умолчанию. Если для отправки данных используется метод `GET`, то адресная строка может принимать следующий вид:

```
http://www.htmlbook.ru/cgi-bin/handler.cgi?
nick=%C2%E0%ED%FF+%D8%E0%EF%EE%F7%EA%E8%ED&page=5
```

Параметры перечисляются после вопросительного знака, который следует за адресом CGI-программы, и разделяются между собой символом амперсанда (`&`). Нелатинские символы преобразуются в шестнадцатеричное представление (в форме `%HH`, где `HH` — шестнадцатеричный код для значения ASCII-символа), пробел заменяется на плюс (+).

Добавление формы

Для указания браузеру, где начинается и заканчивается форма, используется тег FORM. Между открывающим и закрывающим тегами <FORM> и </FORM> можно помещать любые необходимые теги HTML (листинг 7.1). Это позволяет добавить элементы формы в ячейки таблицы для их форматирования, а также использовать изображения. В документе допускается наличие любого количества форм, но одновременно на сервер может быть отправлена только одна форма. По этой причине формы не должны быть вложены одна в другую.

Листинг 7.1. Применение формы

```
<html>
<body>
<form action=/cgi-bin/handler.cgi>
<b>Как по вашему мнению расшифровывается аббревиатура "ОС"?</b><br>
 <input type=radio name=answer value=a1>Офицерский состав<br>
 <input type=radio name=answer value=a2>Операционная система<br>
 <input type=radio name=answer value=a3>Большой полосатый мух
 <input type=submit value="Проверить">
</form>
</body>
</html>
```

В любой форме содержится несколько параметров:

1. Элементы формы, которые представляют собой стандартные поля для ввода информации.
2. Кнопка отправки данных формы на сервер.
3. Адрес программы на веб-сервере, которая будет обрабатывать содержимое данных формы.

Если данные формы не планируется отправлять на сервер, что возможно в случае использования клиентских программ (скриптов), то указывать адрес не обязательно.

Возможные параметры тега FORM перечислены далее.

- action. Указывает обработчика, к которому обращаются данные формы при их отправке на сервер. В качестве обработчика может выступать CGI-программа или HTML-документ, который включает в себя серверные сценарии (например Parser). После выполнения обработчиком действий с данными формы он возвращает новый HTML-документ. Если параметр action отсутствует, то после отправки формы текущая страница

перезагружается, возвращая все элементы формы к их значениям по умолчанию. В качестве обработчика можно указать адрес электронной почты, начиная его с ключевого слова mailto, как показано в листинге 7.2. При отправке формы будет запущена почтовая программа, установленная по умолчанию. В целях безопасности в браузере предусмотрено, что незаметно отправить по почте информацию, введенную в форме, невозможно. Для корректной интерпретации данных используйте параметр enctype="text/plain" в теге FORM.

Листинг 7.2. Отправка формы по почте

```
<html>
<body>
<form action=mailto:vlad@htmlbook.ru enctype="text/plain">
...
</form>
</body>
</html>
```

- enctype. Устанавливает тип для данных, отправляемых вместе с формой. Обычно не требуется определять значение параметра enctype, данные вполне правильно интерпретируются на стороне сервера. Однако если используется поле для отправки файла (INPUT type=file), то следует задать параметр enctype как multipart/form-data. Допускается устанавливать сразу несколько значений, разделяя их запятыми.
- method. Указывает метод, который сообщает серверу о цели запроса. Различают два основных метода: GET и POST. Существуют и другие методы, но они пока мало используются.
  - GET. Этот метод является одним из самых распространенных и предназначен для получения требуемой информации и передачи данных в адресной строке. Пары "имя=значение" присоединяются в этом случае к адресу после вопросительного знака и разделяются между собой амперсандом (символ &). Удобство применения метода GET заключается в том, что адрес со всеми параметрами можно использовать неоднократно, сохранив его, например, в папке **Избранное** браузера, а также менять значения параметров прямо в адресной строке.
  - POST. Метод POST посылает на сервер данные в запросе браузера. Это позволяет отправлять большее количество данных, чем доступно методу GET, поскольку у него установлено ограничение в 4 Кб. Большие объемы данных используются в форумах, почтовых службах, при заполнении базы данных и т. д.

- `target`. После того, как обработчик формы получает данные, он возвращает результат в виде HTML-документа. Вы можете определить окно, в которое будет загружаться итоговая веб-страница. В качестве аргумента используется имя окна или фрейма, заданное параметром `name`. Если установлено несуществующее имя, то будет открыто новое окно. В качестве зарезервированных имен используются следующие:
  - `_blank` — результат выполнения запроса к обработчику формы загружается в новом окне браузера;
  - `_self` — загружает страницу, возвращаемую обработчиком формы в текущее окно;
  - `_parent` — загружает страницу во фрейм-родитель, если фреймов нет, то этот параметр работает как `_self`;
  - `_top` — отменяет все фреймы, если они имеются, и загружает страницу в полном окне браузера, в противном случае этот параметр работает как `_self`.

Если параметр `target` не установлен, возвращаемый результат отображается в текущем окне.

Элементы форм

Форма представляет собой лишь контейнер для размещения объектов, которые дублируют элементы интерфейса операционной системы: кнопки, поле со списком, переключатели, флажки и т. д. На рис. 7.1 представлены различные элементы форм, которые можно добавить на веб-страницу.

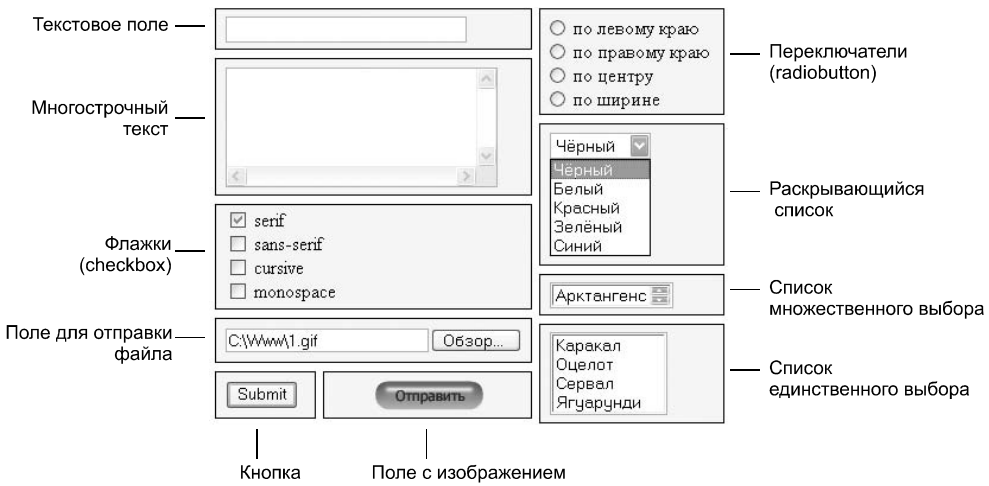


Рис. 7.1. Элементы форм и их названия

Следует констатировать, что для обозначения элементов интерфейса существует несколько разных подходов. Это связано с тем, что приходится подыскивать русскоязычный термин, который бы наиболее точно характеризовал объект. Как правило, получается либо длинно и неудобно, либо не совсем правильно. Альтернативный вариант — использовать оригинальные названия в латинской транскрипции, давая в скобках перевод. Существует и другой подход, который заключается в том, чтобы писать англоязычные термины в русской транскрипции. Такое написание тоже имеет право на существование, поскольку в последнее время оно часто встречается в технической литературе, вдобавок оно ближе тем людям, которые по своей профессии часто упоминают такие термины. Однако чтобы не возникало путаницы, для обозначения элементов форм в дальнейшем мы будем придерживаться выражений, указанных на рис. 7.1.

Текстовое поле

Текстовое поле предназначено для ввода символов с клавиатуры. Различают три элемента формы, которые используются для этой цели, — однострочное текстовое поле, поле для ввода пароля и многострочное текстовое поле.

Однострочное текстовое поле

Такое поле предназначено для ввода пользователем строки текста. Размер поля можно ограничить по ширине, но это делается больше для удобства дизайна, чтобы элемент можно было вместить в отведенное для него место. Текст при ограничении ширины поля можно писать как обычно, но при наборе введенные ранее символы скрываются. Чтобы их посмотреть, придется перемещать курсор с помощью клавиатуры.

Синтаксис создания текстового поля следующий:

```
<input type=text параметры>
```

Параметры поля перечислены в табл. 7.1.

Таблица 7.1. *Описание параметров текстового поля*

Параметр	Описание
`maxlength`	Максимальное количество символов, разрешенных при наборе текста. Если этот параметр опустить, то число вводимых символов не ограничено
`name`	Имя поля. Предназначено для того, чтобы обработчик формы мог идентифицировать поле

Таблица 7.1 *(окончание)*

Параметр	Описание
size	Ширина поля. Физический размер зависит от настроек операционной системы и выбранного браузера
value	Начальный текст, содержащийся в поле

Пример использования текстового поля приведен в листинге 7.3.

Листинг 7.3. Создание однострочного текстового поля

```
<html>
<body>
 <form action=/cgi-bin/handler.cgi>
  <b>Как ваше имя?</b><br>
  <input type=text maxlength=25 size=20>
  <input type=submit value=OK>
 </form>
</body>
</html>
```

Ширина текстового поля — величина нестабильная и в разных браузерах может меняться в небольших пределах. Для формы, которая располагается в колонке ограниченной ширины, подобные изменения приводят к нарушению исходного макета. В этом случае лучше вообще отказаться от использования параметра size и заменить его стилями. К тому же CSS позволяет изменять цвет фона, тип шрифта и рамки вокруг поля, как показано в листинге 7.4.

Листинг 7.4. Изменение вида текстового поля с помощью стилей

```
<html>
<head>
<style type="text/css">
 .textfield {
  background: #f0f0f0;              /* Цвет фона под текстом */
  color: red;                       /* Цвет символов */
  border: 2px solid black;          /* Рамка вокруг поля */
  width: 50%;                       /* Ширина поля в процентах */
  font-family: Arial, sans-serif;   /* Тип шрифта */
  font-size: 90%                    /* Размер шрифта */
}
```

```
</style>
</head>
<body>
  <form action=/cgi-bin/handler.cgi>
  <b>Как ваше имя?</b><br>
  <input type=text maxlength=25 class=textfield>
  <input type=submit value=OK>
 </form>
</body>
</html>
```

В данном примере создается новый класс, который связывается с тегом `INPUT`. Ширина поля указана в процентах, но можно воспользоваться и другими единицами измерения, например пикселами, если требуется жестко задать размер поля. Также по желанию можно сменить тип шрифта, которым набирается текст, через атрибуты `font-family` и `font-size`, и задать новый цвет символов с помощью свойства `color`.

Обычно текстовое поле отображается на веб-странице слегка утопленным за счет использования псевдотрехмерных эффектов. Добавление атрибута `border` создает вокруг поля рамку желаемого цвета и толщины, но при этом указанный эффект теряется (рис. 7.2).

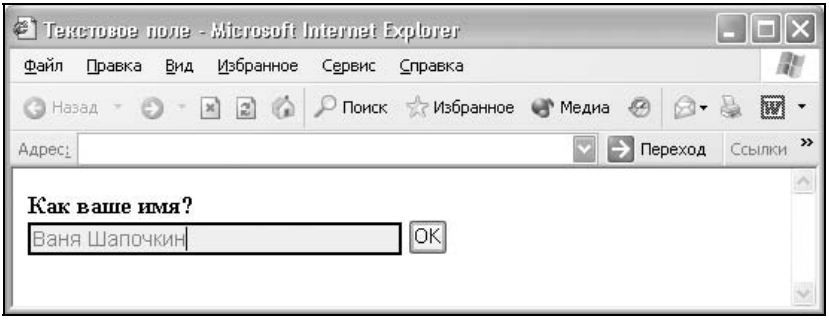


Рис. 7.2. Вид текстового поля при использовании стилей

Заметим, что вид поля, представленный на рис. 7.2, будет таким только при использовании значения `solid` атрибута `border`. Изменяя этот параметр, в частности толщину рамки и ее тип, можно получить самые разнообразные виды текстовых полей, показанных на рис. 7.3.

Параметры стилей, которые были использованы для получения подобных рамок, приведены на рис. 7.3 в соответствующих текстовых полях. Учтите, что хотя рамки корректно отображаются всеми современными браузерами, по виду они могут незначительно отличаться друг от друга.

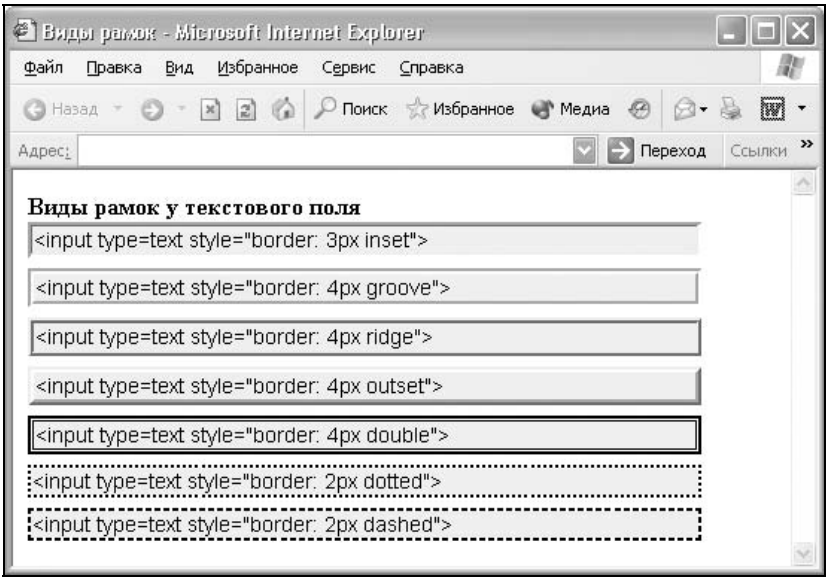


Рис. 7.3. Текстовое поле с разными рамками

Поле для ввода пароля

Поле для ввода пароля — обычное текстовое поле, вводимый текст в котором отображается звездочками. Такая особенность предназначена для того, чтобы никто не подглядел вводимый пароль.

Синтаксис создания поля для ввода пароля следующий:

```
<input type=password параметры>
```

Параметры поля для пароля аналогичны параметрам текстового поля и приведены в табл. 7.1.

Поле для пароля нашло широкое применение на сайтах для авторизации пользователей и разграничения доступа к разделам сайта, где требуется подтвердить свои полномочия. В листинге 7.5 показано, как создавать подобные поля.

Листинг 7.5. Использование поля для ввода пароля

```
<html>
<body>
 <form action=/cgi-bin/handler.cgi>
  <b>Логин:</b> <input type=text maxlength=25 size=20 name=text><br>
  <b>Пароль:</b> <input type=password maxlength=15 size=20 name=pass>
```

```
  <p><input type=submit value=OK>
 </form>
</body>
</html>
```

В разных браузерах и операционных системах отображаемые при вводе символы могут выглядеть по-разному. Обычно введенный текст заменяется звездочками, но под Windows XP используются другие символы (рис. 7.4).

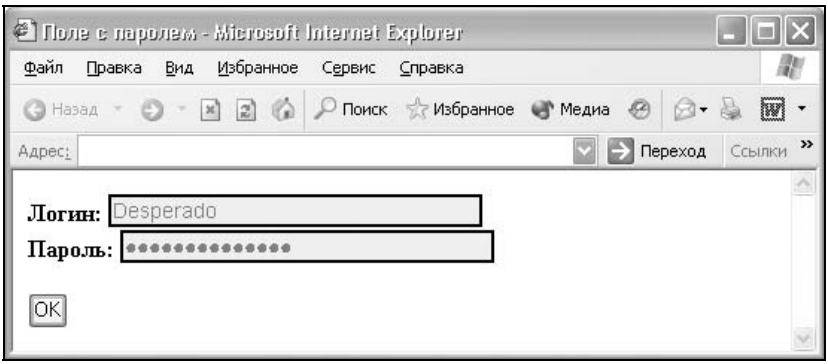


Рис. 7.4. Вид поля для ввода пароля в браузере Internet Explorer под Windows XP

Поскольку поле для пароля в некотором роде является разновидностью текстового поля, то к нему применимы те же параметры и свойства стилей. Поэтому на рис. 7.3 поля отображаются с рамкой и вводимыми символами одного цвета.

Замечание

Хотя вводимый текст и не отображается на экране, на сервер введенная информация передается в открытом виде без шифрования. Поэтому использование этого поля не обеспечивает безопасности данных и их можно перехватить.

Многострочное текстовое поле

Для создания области, в которую можно вводить несколько строк текста, предназначен тег TEXTAREA. В отличие от тега INPUT в текстовом поле допустимо делать переносы строк, они сохраняются при отправке данных на сервер.

Синтаксис создания поля для ввода многострочного текста следующий:

```
<textarea параметры>...</textarea>
```

Параметры поля отличаются от однострочного варианта и перечислены в табл. 7.2.

Таблица 7.2. *Описание параметров тега* `TEXTAREA`

Параметр	Описание
`cols`	Ширина текстового поля, которая определяется числом символов моноширинного шрифта. Иными словами, ширина задается количеством близстоящих букв одинаковой ширины по горизонтали. Если размер шрифта изменяется с помощью стилей, ширина также соответственно меняется
`disabled`	Блокирует доступ и изменение текстового поля. В таком случае оно отображается серым цветом и недоступным для активации пользователем. Кроме того, такое поле не может получить фокус путем нажатия на клавишу <Tab>, с помощью мыши или другим способом. Тем не менее такое состояние поля можно менять с помощью скриптов
`name`	Определяет уникальное имя элемента `TEXTAREA`. Как правило, это имя применяется при отправке данных на сервер или для доступа к полю через скрипты. В качестве имени используется набор символов, включая числа и буквы. JavaScript чувствителен к регистру, поэтому при обращении к элементу по имени соблюдайте ту же форму написания, что и в параметре `name`
`readonly`	Когда к тегу `TEXTAREA` добавляется параметр `readonly`, текстовое поле недоступно для изменения пользователем, в том числе в него не допускается вводить новый текст или модифицировать существующий. Вдобавок, поле "только для чтения" не может получить фокус
`rows`	Высота текстового поля, которая определяется количеством отображаемых строк без прокрутки содержимого. Если размер шрифта изменяется с помощью стилей, высота поля также соответственно меняется
`wrap`	Параметр `wrap` говорит браузеру о том, как осуществлять перенос текста в поле `TEXTAREA` и в каком виде отправлять данные на сервер. Если этот параметр отсутствует, текст в поле набирается одной строкой, а когда число введенных символов превышает ширину области, появляется горизонтальная полоса прокрутки. Нажатие клавиши <Enter> переносит текст на новую строку, и курсор устанавливается у левого края поля

Между тегами `<TEXTAREA>` и `</TEXTAREA>` можно поместить любой текст, который будет отображаться внутри поля. На рис. 7.5 показан вид текстового поля в зависимости от заданных параметров.

При оформлении многострочного поля применяются те же стилевые параметры, что и для однострочного текста. А именно можно изменять ширину (атрибут `width`), высоту (параметр `height`), границу (параметр `border`), цвет текста и фона (`color` и `background` соответственно). Пример создания текстового поля с разными характеристиками приведен в листинге 7.6.

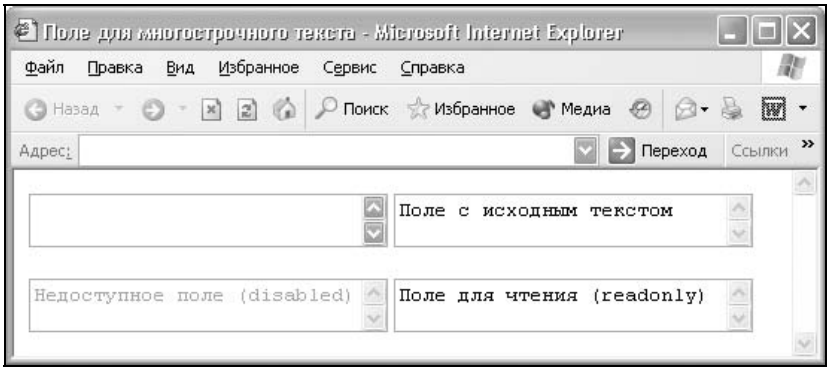


Рис. 7.5. Типы поля для ввода текста

Листинг 7.6. Использование текстового поля

```
<html>
<head>
<style type="text/css">
TEXTAREA {
  background: #f0f0f0;              /* Цвет фона под текстом */
  color: #008080;                   /* Цвет символов */
  padding: 5px;                     /* Поля вокруг текста */
  border: 1px solid black;          /* Черная рамка вокруг поля */
  width: 50%                        /* Ширина поля в процентах */
}
</style>
</head>
<body>
 <form action=/cgi-bin/handler.cgi>
  <b>Введите ваш отзыв:</b><br>
  <textarea rows=10>
  </textarea>
  <p><input type=submit value="Отправить">
 </form>
</body>
</html>
```

Как уже отмечалось, добавление рамки к полю через свойство `border` отменяет трехмерные эффекты, используемые по умолчанию. Кроме того, вокруг текста при использовании рамок убираются поля, поэтому требуется их установить с помощью атрибута `padding`, как показано в данном примере.

Рисунки в текстовом поле

Изображение возле текстового поля обычно добавляется для привлечения внимания пользователя и дизайнерского оформления. Стили разрешают установить рисунок не просто рядом с полем, но прямо в поле для ввода текста (рис. 7.6).

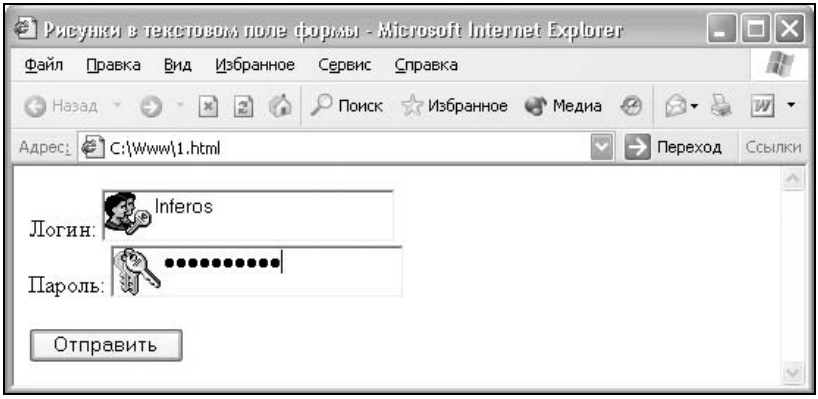


Рис. 7.6. Вид текстового поля с фоновым рисунком

Вначале готовим необходимые изображения, а затем добавляем их к тегу `INPUT` в виде фона, используя атрибут `background`. Исходный рисунок можно уменьшить в графическом редакторе или, наоборот, увеличить высоту поля, подгоняя его под высоту изображения. С этой целью применяется свойство `height`, как показано в листинге 7.7. В качестве аргумента параметра `background` необходимо использовать `no-repeat`, тогда рисунок будет отображаться только один раз и не станет повторяться, как это задано для фона по умолчанию. Чтобы не писать текст поверх рисунка, к стилю тега `INPUT` следует добавить атрибут `padding-left`. Он обеспечивает набор текста правее рисунка, а его значение зависит от ширины изображения.

Листинг 7.7. Добавление изображения в текстовое поле

```
<html>
<head>
<style type="text/css">
 INPUT.enter {
  height: 36px;              /* Высота поля */
  width: 200px;              /* Ширина поля */
  padding-left: 34px;        /* Отступ слева от края до текста */
}
```

```
</style>
</head>
<body>
<form action=/cgi-bin/handler.cgi>
Логин: <input type=text class=enter style="background: url(login.gif)
no-repeat"><br>
Пароль: <input type=password class=enter style="background: url(pass.gif)
no-repeat">
<p><input type=submit value="Отправить">
</form>
</body>
</html>
```

Из-за того, что текстовое поле изображается утопленным за счет трехмерной рамки, реальная высота области несколько меньше указанной высоты. Так, в данном примере используются рисунки высотой 32 пиксела. Если установить такое же значение и для поля, то изображения окажутся обрезанными снизу. Чтобы этого не произошло, высота поля в примере задана больше. С той же целью можно установить другой вид рамки, используя стилевое свойство `border`.

Кнопки

Кнопки являются одним из самых понятных и интуитивных элементов интерфейса. По их виду сразу становится понятно, что единственное действие, которое с ними можно производить, — это нажимать на них. За счет этой особенности кнопки часто применяются в формах, особенно при их отправке и очистке.

Кнопку на веб-странице можно создать двумя способами — с помощью тега `INPUT` и тега `BUTTON`. Рассмотрим вначале добавление кнопки через `INPUT` и его синтаксис.

```
<input type=button параметры>
```

Основных параметров всего два — это `name` и `value`. Атрибут `name` задает имя кнопки и предназначен для того, чтобы обработчик формы мог идентифицировать это поле. Параметр `name` может быть опущен, в таком случае значение кнопки не передается на сервер. Значение кнопки и надпись на ней одновременно устанавливаются с помощью параметра `value`, как показано в листинге 7.8.

Листинг 7.8. Создание кнопки

```
<html>
<body>
```

```
 <form action=/cgi-bin/handler.cgi>
  <input type=button name=press value="  Нажми меня нежно  ">
 </form>
</body>
</html>
```

В надписи на кнопке можно ставить пробелы в любом количестве, за счет них можно регулировать ее ширину.

Второй способ создания кнопки основан на использовании тега BUTTON. Он по своему действию напоминает результат, получаемый с помощью тега INPUT. В отличие от этого тега, BUTTON предлагает расширенные возможности по созданию кнопок. Например, на подобной кнопке можно размещать любые элементы HTML, в том числе изображения и таблицы. Используя стили, можно определить вид кнопки путем изменения шрифта, цвета фона, размеров и других параметров. На рис. 7.7 показаны разные виды кнопок, полученные с помощью указанного тега.

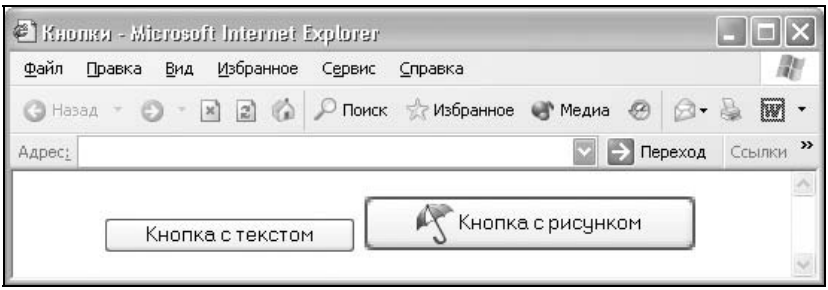


Рис. 7.7. Разные виды кнопок

Теоретически, тег BUTTON должен располагаться внутри формы, устанавливаемой элементом FORM. Тем не менее браузеры не выводят сообщение об ошибке и корректно работают с тегом BUTTON, если он встречается самостоятельно. Однако если результат нажатия на кнопку необходимо отправить на сервер, то помещать BUTTON в контейнер FORM обязательно. В листинге 7.9 показан пример создания кнопки с текстом и рисунком.

Листинг 7.9. Кнопки, созданные с помощью тега BUTTON

```
<html>
<body>
<p align=center>
<button>Кнопка с текстом</button>
<button><img src=umbrella.gif width=25 height=32 align=absmiddle>
Кнопка с рисунком</button>
```

```
</p>
</body>
</html>
```

В этом примере показано создание обычной кнопки с текстом, а также кнопки с одновременным использованием текста и рисунка. Чтобы рисунок и текст были выровнены по одной оси, добавлен параметр `absmiddle` для тега `IMG`.

Размер кнопки зависит от содержимого контейнера `BUTTON`, но пробелы игнорируются и простым увеличением их количества, как в случае использования `INPUT`, ширину кнопки изменить не удастся. Поэтому размеры кнопки можно менять с помощью рисунка или добавляя неразделяемый пробел ` `. Также свою роль могут сослужить и стили, как будет показано далее.

Кнопка *SUBMIT*

Для отправки данных на сервер предназначена специальная кнопка `SUBMIT`. Ее вид ничем не отличается от других кнопок, но при нажатии на нее происходит выполнение серверной программы, указанной параметром `action` тега `FORM`. Эта программа, называемая еще обработчиком формы, получает данные, введенные пользователем в полях формы, производит с ними необходимые манипуляции, после чего возвращает результат в виде HTML-документа. Что именно делает обработчик, зависит от автора сайта; так, подобная технология применяется при создании опросов, форумов, гостевых книг, тестов и многих других вещей.

Синтаксис создания кнопки `SUBMIT` зависит от используемого тега.

```
<input type=submit параметры>
<button type=submit>Надпись на кнопке</button>
```

Параметры кнопки `SUBMIT` такие же, как и у простых кнопок (листинг 7.10).

Листинг 7.10. Кнопка для отправки данных на сервер

```
<html>
<body>
 <form action=/cgi-bin/handler.cgi method=GET>
  ...
  <p align=center><input type=submit></p>
 </form>
</body>
</html>
```

Параметр name для этого типа кнопки может быть опущен. Если значение параметра value не указывать, на кнопке автоматически появится надпись "Подача запроса" для русской версии браузера Internet Explorer, "Отправить запрос" для русской версии Firefox или "Submit Query" для Netscape.

Кнопка *RESET*

При нажатии на кнопку RESET данные формы возвращаются в первоначальное значение. Как правило, эту кнопку применяют для очистки введенной в полях формы информации. Но для больших форм от использования кнопки RESET лучше вообще отказаться, чтобы по ошибке на нее не нажать, ведь тогда придется заполнять форму заново.

Синтаксис создания указанной кнопки прост и похож на другие кнопки.

```
<input type=reset параметры>
<button type=reset>Надпись на кнопке</button>
```

В листинге 7.11 показана форма с одним текстовым полем, которое уже содержит предварительно введенный текст с помощью параметра value тега INPUT. После изменения текста и нажатия на кнопку "Очистить", значение поля будет восстановлено, и в нем снова появится надпись "Введите текст".

Листинг 7.11. Кнопка для очистки формы

```
<html>
<body>
 <form action=/cgi-bin/handler.cgi>
<input type=text value="Введите текст">
<p><input type=submit value="Отправить"> <input type=reset
value="Очистить">
</form>
</body>
</html>
```

Значение кнопки RESET никогда не пересылается на сервер, причем если надпись на кнопке опустить, иными словами, не задавать параметр value, по умолчанию будет добавлен текст "Сброс" или "Reset".

Цветные кнопки

Вид и цвет кнопок зависит от операционной системы и браузера. Тем не менее можно изменить цвет кнопок по своему усмотрению, воспользовавшись стилями. Для этого требуется только добавить к кнопке атрибут

background, как показано в листинге 7.12. Дополнительно можно поменять параметры шрифта, в частности цвет текста.

Листинг 7.12. Изменение цвета кнопки

```
<html>
<head>
<style type="text/css">
 .colorButton {
  background: #ff0000;              /* Цвет самой кнопки */
  color: #000;                      /* Цвет текста на кнопке */
  font-family: Tahoma, sans-serif;  /* Тип шрифта */
  font-size: 15px                   /* Размер текста */
}
</style>
</head>
<body>
 <form>
  <input type=button class=colorButton value="Красная кнопка">
 </form>
</body>
</html>
```

Приведенная в данном примере манипуляция с цветом одновременно меняет и форму кнопки. На рис. 7.8 показаны кнопки, используемые в браузере по умолчанию, и вид, который они приобретают после применения стилей.

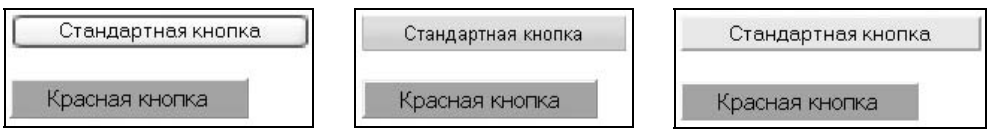


Рис. 7.8. Изменение вида кнопок в разных браузерах:
а — Internet Explorer 6; *б* — Opera 7; *в* — Netscape 7

Заметьте, что каждый браузер показывает кнопку по-своему, различается буквально все — размер шрифта, ширина и форма кнопки. При добавлении стилей вид кнопки становится менее притязательным, но зато отображается одинаково в разных браузерах.

Кроме однотонного цвета на кнопке можно использовать и фоновый рисунок, это позволит сделать заливку кнопки градиентом, как показано на рис. 7.9.

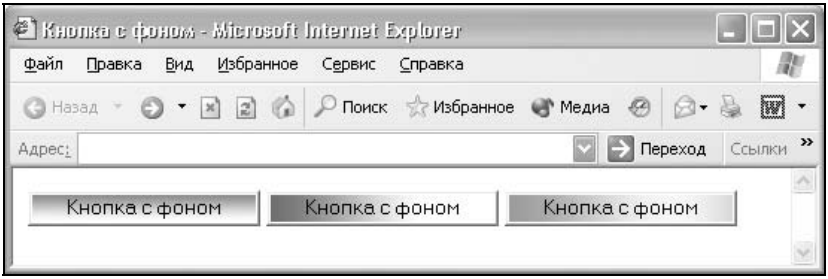


Рис. 7.9. Варианты кнопок с фоновым рисунком

Вначале следует подготовить фоновое изображение в графическом редакторе. Поскольку ширина кнопки зависит от текста на ней, то размеры рисунка лучше сделать заведомо большего размера. После того как картинка будет готова, опять воспользуемся параметром `background`, но в качестве его значения укажем путь к графическому файлу через `url`, как показано в листинге 7.13.

Листинг 7.13. Создание фонового рисунка на кнопке

```
<html>
<body>
 <form>
  <input type=button style="background: url(/images/bgbutton.gif)"
value="Кнопка с фоном">
 </form>
</body>
</html>
```

В этом примере в качестве фонового рисунка для кнопки используется файл с именем bgbutton.gif. Как и в случае цветных кнопок, добавление стиля отменяет особенности браузеров и делает кнопку прямоугольной.

Переключатели

Переключатели (radiobutton) используют, когда необходимо выбрать один-единственный вариант из нескольких предложенных. Создаются они следующим образом:

```
<input type=radio name=имя параметры>
```

У этого поля три основных параметра: `name`, `value` и `checked`. Имя (`name`) однозначно идентифицирует поле, вдобавок, поскольку переключатели являются групповыми элементами, то имя у всех элементов группы должно быть

одинаковым. Такой подход однозначно устанавливает принадлежность поля к определенной группе. Параметр `value` задает, какое значение будет отправлено на сервер. Здесь каждый элемент должен иметь свое уникальное значение, чтобы можно было идентифицировать, какой пункт был выбран пользователем. Для предварительного выделения пункта списка применяется параметр `checked`. По определению, набор переключателей может иметь только один выделенный пункт, поэтому добавление `checked` сразу к нескольким полям не приведет к какому бы то ни было выдающемуся результату. В любом случае будет отмечен элемент, находящийся в коде HTML последним.

По части оформления переключатели не дают особо разгуляться фантазии. С помощью стилей допустимо устанавливать рамку вокруг каждого переключателя, менять цвет фона под ним, а также изменять расстояние между самими элементами и текстом рядом с ними (листинг 7.14).

Листинг 7.14. Использование переключателей

```
<html>
<head>
<style>
 .radiobutton {
  color: red;                          /* Цвет маркера */
  background: #ccc;                    /* Цвет фона под переключателем */
  border: 1px dashed black;            /* Параметры рамки */
  margin: 2px                          /* Отступы вокруг элемента */
}
</style>
</head>
<body>
 <form action=/cgi-bin/handler.cgi>
  <b>Какое у вас состояние разума?</b><br>
  <input name=dzen type=radio class=radiobutton> Не дзен<br>
  <input name=dzen type=radio class=radiobutton> Дзен<br>
  <input name=dzen type=radio class=radiobutton> Полный дзен
  <p><input type=submit value="Выбрать">
 </form>
</body>
</html>
```

Браузеры по-разному интерпретируют параметры, приведенные в данном примере, и веб-страница в их интерпретации выглядит неодинаково (рис. 7.10).

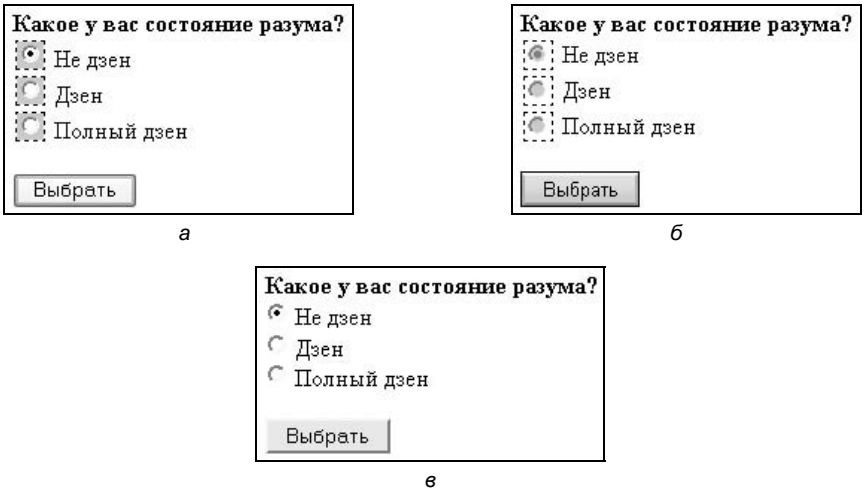


Рис. 7.10. Отображение переключателей в разных браузерах:
а — Internet Explorer 6; *б* — Opera 7; *в* — Netscape 7

Браузер Netscape (Firefox) (рис. 7.10, *в*) вообще не реализует результат действия стилей и полностью их игнорирует. Opera (рис. 7.10, *б*) применяет параметры `color` и `background` только к маркеру, кружок в данном примере отображается серым цветом, а точка, показывающая выделение текущего элемента, — красным. Internet Explorer (рис. 7.10, *а*) вообще никак не использует атрибут `color`, а фоновый цвет применяется к прямоугольнику, ограниченному рамкой.

Обычно чтобы активизировать переключатель, надо щелкнуть на нем мышью. С помощью тега `LABEL` можно сделать активным и текст рядом с элементом, тогда при нажатии курсором мыши на тексте появляется возможность выбирать желаемый пункт.

Для связывания элемента формы и текста в теге `LABEL` используется параметр `for`, аргументом для которого служит идентификатор элемента, задаваемый атрибутом `id` тега `INPUT`, как показано в листинге 7.15.

Листинг 7.15. Использование тега `LABEL`

```
<html>
<body>
 <form>
  <input type=radio name=psi id=radio1>
                              <label for=radio1>Импритинг</label><br>
  <input type=radio name=psi id=radio2><label for=radio2>Реимпритинг</label>
 </form>
```

```
</body>
</html>
```

Имя идентификатора (id) и значение параметра for обязательно должны совпадать. Именно в этом случае и осуществляется связь между текстом и элементом формы. Причем не имеет значения, насколько близко они располагаются между собой на веб-странице. При нажатии на текст в окне браузера происходит переход к элементу формы. Исключением является браузер Opera, который игнорирует эту особенность, и переход при активизации текста не реализует.

Польза от применения тега LABEL не особенно велика, но следует учитывать, что в операционных системах активация переключателей и флажков происходит при щелчке на тексте возле них. А следовательно, такой подход более привычен и понятен пользователям сайтов.

Флажки

Флажки (checkbox) используют, когда необходимо выбрать два или более варианта из предложенного списка. Если требуется выбрать лишь один вариант, то для этого следует предпочесть переключатели (radiobutton).

Флажок создается следующим образом:

```
<input type=checkbox параметры>
```

Параметры флажков идентичны переключателям, а именно: name задает имя поля, value — его значение, а checked устанавливает флажок как помеченный. При этом каждый флажок, входящий в группу, рассматривается как независимый, поэтому имена и значения у них должны различаться (листинг 7.16).

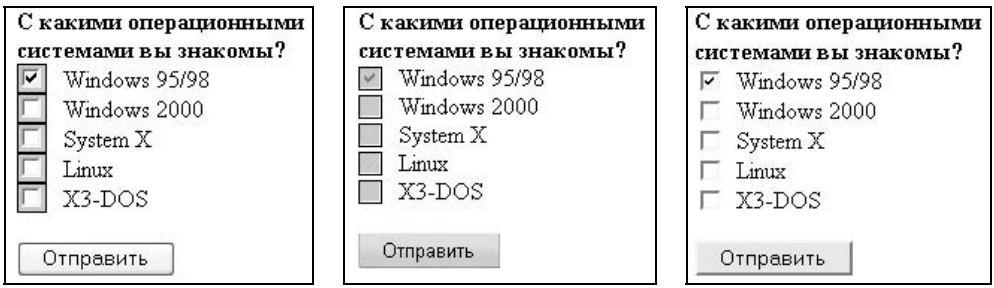
Как и в случае переключателей, с помощью стилей можно изменять вид и положение флажков. Так, в данном примере задан цвет фона, параметры рамки и расстояние между флажком и текстом.

Листинг 7.16. Использование флажков

```
<html>
<head>
<style type="text/css">
 .col {
  background: #ccc;          /* Цвет фона флажка */
  color: red;                /* Цвет галочки в браузере Opera */
  border: 1px solid black;   /* Рамка вокруг флажка */
  margin-right: 7px          /* Отступ между текстом и флажком */
}
```

```
</style>
</head>
<body>
 <form action=/cgi-bin/handler.cgi>
  <b>С какими операционными системами вы знакомы?</b><br>
  <input class=col type=checkbox name=option1 value=a1 checked> Windows
95/98<br>
  <input class=col type=checkbox name=option2 value=a2> Windows 2000<br>
  <input class=col type=checkbox name=option3 value=a3> System X<br>
  <input class=col type=checkbox name=option4 value=a4> Linux<br>
  <input class=col type=checkbox name=option5 value=a5> X3-DOS
  <p><input type=submit value="Отправить">
 </form>
</body>
</html>
```

Следует констатировать, что и здесь браузеры совершенно по-разному отображают флажки, вид которых управляется через CSS (рис. 7.11).



а *б* *в*

Рис. 7.11. Вид флажков в разных браузерах:
а — Internet Explorer 6; *б* — Opera 7; *в* — Netscape 7

Единственный параметр в данном примере, который реализует браузер Netscape (Firefox) (рис. 7.11, *в*), — это `margin-left`, регулирующий расстояние от флажка до текста справа. Следует учесть еще пробел перед текстом, он тоже вносит свою лепту в определение дистанции между текстом и флажком. Остальные параметры браузер Netscape игнорирует. Opera (рис. 7.11, *б*) при использовании атрибута `background` полностью заливает флажок заданным цветом, а Internet Explorer (рис. 7.11, *а*) оставляет белый квадрат, в который ставится галочка. Кроме того, цвет самой галочки умеет менять только Opera через параметр `color`.

Тег `LABEL`, как и в случае переключателей, может применяться и для флажков. Это позволяет устанавливать или снимать галочку путем нажатия на тексте возле флажка.

Поле со списком

Поле со списком, называемое еще *ниспадающее меню*, — один из гибких и удобных элементов формы. В зависимости от настроек в списке можно выбирать одно или несколько значений. Преимущество списка состоит в его компактности: он может занимать всего одну строку, а чтобы просмотреть весь список, нужно на него нажать. Однако это является и недостатком, ведь пользователю сразу не виден весь выбор.

Поле со списком создается следующим образом:

```
<select параметры>
 <option параметры>Пункт 1</option>
 <option>Пункт 2</option>
 <option>Пункт 3</option>
</select>
```

Тег `SELECT` позволяет создать элемент интерфейса в виде раскрывающегося списка, а также список с одним или множественным выбором, как показано на рис. 7.12. Конечный вид зависит от использования параметра `size` тега `SELECT`, который устанавливает высоту списка. Ширина списка определяется самым широким текстом, указанным в теге `OPTION`, а также может изменяться с помощью стилей. Каждый пункт создается через тег `OPTION`, который должен быть вложен в контейнер `SELECT`. Если планируется отправлять данные списка на сервер, то следует поместить элемент `SELECT` внутрь формы. Это также необходимо, когда к данным списка идет обращение через скрипты.

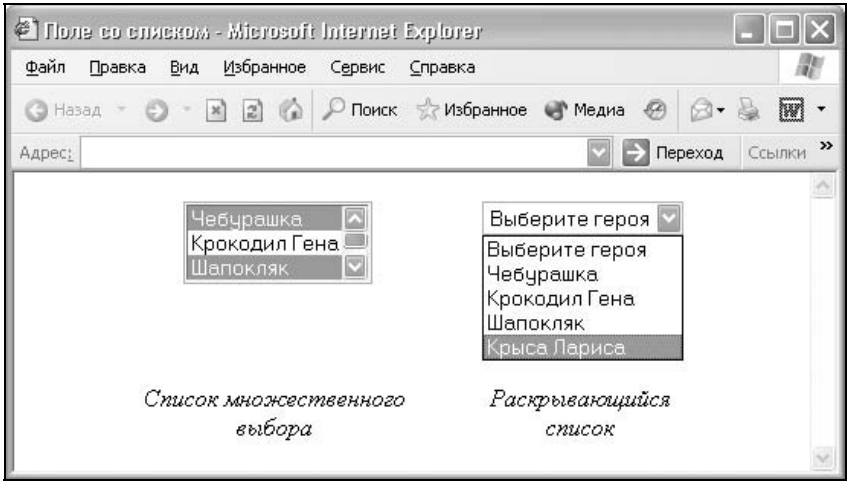


Рис. 7.12. Разные виды списков

Рассмотрим параметры тега SELECT, с помощью которых можно изменять вид и представление списка.

- multiple. Наличие параметра multiple дает команду браузеру отображать содержимое элемента SELECT как список множественного выбора. Конечный вид списка зависит от используемого параметра size. Если он отсутствует, то высота списка равна количеству пунктов; если значение size меньше числа пунктов, то появляется вертикальная полоса прокрутки. Когда size=1, список превращается в "крутилку", как показано на рис. 7.13, но выбирать с помощью нее одновременно несколько пунктов списка становится неудобно. Для выбора нескольких значений списка применяются клавиши <Ctrl> и <Shift> совместно с курсором мыши.

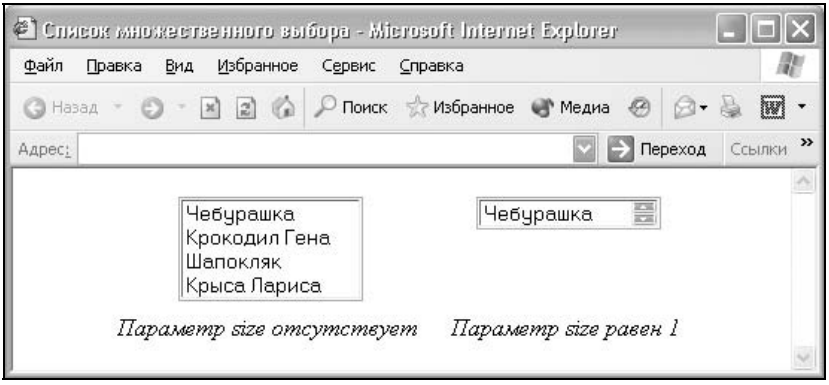


Рис. 7.13. Вид списков множественного выбора

- name. Определяет уникальное имя элемента SELECT. Как правило, это имя используется для доступа к данным через скрипты или для получения выбранного значения серверным обработчиком.
- size. Устанавливает высоту списка. Если значение параметра size равно единице, то список превращается в раскрывающийся. При добавлении параметра multiple к тегу SELECT при size=1 список отображается как "крутилка". Во всех остальных случаях получается список с одним или множественным выбором. Значение по умолчанию зависит от параметра multiple. Если он присутствует, то размер списка равен количеству элементов. Когда параметра multiple нет, значение параметра size равно 1.

Тег OPTION также имеет параметры, влияющие на вид списка, они представлены далее.

- selected. Делает текущий пункт списка выделенным. Если у тега SELECT добавлен параметр multiple, то можно выделять более одного пункта.
- value. Определяет значение пункта списка, которое будет отправлено на сервер. На сервер отправляется пара "имя=значение", где имя задается

параметром name тега SELECT, а значение — параметром value выделенных пунктов списка. Значение может как совпадать с текстом пункта, так и быть самостоятельным.

Результат применения списка представлен в листинге 7.17.

Листинг 7.17. Использование списка

```
<html>
<body>
 <form action=/cgi-bin/handler.cgi>
 <b>Выбери персонажа</b><br>
  <select name=hero>
   <option value=s1>Чебурашка</option>
   <option value=s2 selected>Крокодил Гена</option>
   <option value=s3>Шапокляк</option>
   <option value=s3>Крыса Лариса</option>
  </select>
  <input type=submit value="Отправить">
 </form>
</body>
</html>
```

В этом примере создается список из четырех пунктов с именем hero, причем второй пункт из них предварительно выделен через параметр selected тега OPTION.

Скрытое поле

Скрытое поле не отображается на странице и прячет свое содержимое от пользователя. Посетитель не может ничего в него ввести или напечатать. Цель создания скрытых полей состоит в передаче технической информации на сервер. В большинстве случаев это необходимо для передачи данных формы от страницы к странице.

Синтаксис создания скрытого поля следующий:

```
<input type=hidden name=... value=...>
```

В данном случае параметры означают следующее:

- name — имя поля; оно позволяет программе идентифицировать его;
- value — значение поля, определяющее, какая информация будет отправлена на сервер.

Пример использования скрытых полей приведен в листинге 7.18.

Листинг 7.18. Использование скрытого поля

```
<html>
<body>
<form action=/cgi-bin/handler.cgi method=POST>
<b>Напишите любимое слово (никакие данные не будут передаваться на
сервер!):</b><br>
 <input type=text size=25 name=word>
 <input type=hidden name=UserName value=Vasya>
 <input type=hidden name=password value=pupkin><br>
 <input type=submit value=OK>
</form>
</body>
</html>
```

В данном примере показано создание двух скрытых полей, одно из них носит имя `UserName` и получает значение `Vasya`, а второе именуется `password` со значением `pupkin`. В результате отправки формы обработчику, указанному в параметре `action`, программа может легко прочитать эти данные и интерпретировать их по усмотрению разработчика.

Поле с изображением

Поля с изображениями аналогичны по действию кнопке `SUBMIT`, но представляют собой рисунок. Это расширяет возможности дизайнерских изысков по оформлению формы. Когда пользователь нажимает на рисунок, данные формы отправляются на сервер и обрабатываются программой, заданной параметром `action` тега `FORM`.

Изображение в форме создается следующим образом:

```
<input type=image параметры>
```

Хотя по своему назначению указанное поле похоже на кнопку `SUBMIT`, его параметры совпадают с параметрами тега `IMG`, который добавляет изображение на веб-страницу (листинг 7.19). Не будем повторять параметры, кого это интересует, могут посмотреть их в *главе 2*.

Листинг 7.19. Кнопка с изображением

```
<html>
<body>
 <form action=/cgi-bin/handler.cgi>
  <b>Введите ваше имя:</b><br>
```

```
  <input type=text size=35>
  <input type=image align=absmiddle src=/images/imgbutton.gif width=91
                                                          height=25>
 </form>
</body>
</html>
```

Результат данного примера показан на рис. 7.14.

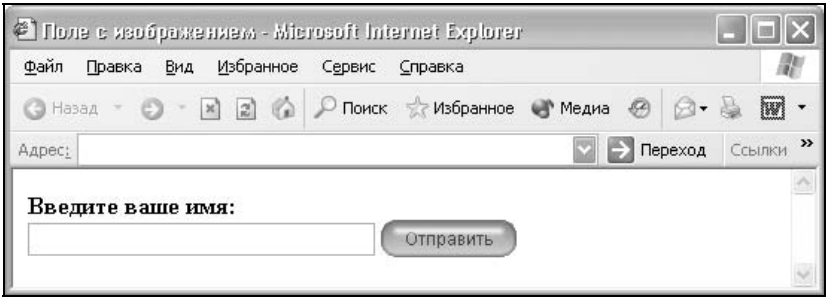


Рис. 7.14. Вид поля с изображением

Перед использованием указанного поля требуется подготовить изображение в графическом редакторе.

Отправка файла

Для того чтобы можно было отправить на сервер файл, используется специальное поле, которое задается параметром `type=file` тега `INPUT`. Такой элемент формы отображается как текстовое поле, рядом с которым располагается кнопка **Обзор** (**Browse...** для английской версии Netscape или **Choose** в браузере Opera). При нажатии на эту кнопку открывается окно для выбора файла, в котором пользователь может указать нужный файл (рис. 7.15).

Создание поля для отправки файла показано в листинге 7.20.

Листинг 7.20. Создание поля для отправки файла

```
<html>
<body>
<form enctype="multipart/form-data" method=POST>
<b>Укажите файл для отправки на сервер:</b><br>
<input type=file size=30>
</form>
</body>
</html>
```

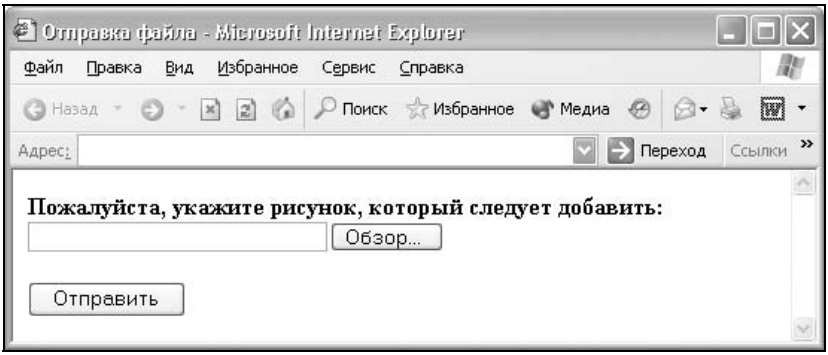


Рис. 7.15. Вид поля для отправки файла на сервер

Параметр формы `enctype="multipart/form-data"` нужен для корректной передачи файла. Если его не указать, то будет передан лишь путь к файлу. Дополнительные параметры те же, что и для текстового поля.

Поскольку графические файлы занимают относительно большой объем данных, их следует отправлять на сервер с помощью метода `POST`, как показано в данном примере.

Группирование элементов формы

При создании сложной формы не обойтись без визуального отделения одного логического блока от другого. Этого можно добиться, применяя внутри тега `FORM` сочетания тегов и стилей. Например, элементы формы можно выделить, если использовать для них фоновый цвет или рамку, задавая их через CSS. Кроме того, существует и другой подход, который состоит в применении тега `FIELDSET`. Этот контейнер группирует элементы формы, отображая вокруг них рамку (листинг 7.21).

Листинг 7.21. Использование тега `FIELDSET`

```
<html>
<body>
<form>
 <fieldset>
  <b>Работа со временем</b><br>
  <input type=checkbox value=t1> Создание пунктуальности (никогда
                                 не будете никуда опаздывать).<br>
  <input type=checkbox value=t2> Излечение от пунктуальности (никогда
                                 никуда не будете торопиться).<br>
  <input type=checkbox value=t3> Изменение восприятия времени.
 </fieldset>
```

```
</form>
</body>
</html>
```

Результат выполнения данного примера показан на рис. 7.16.

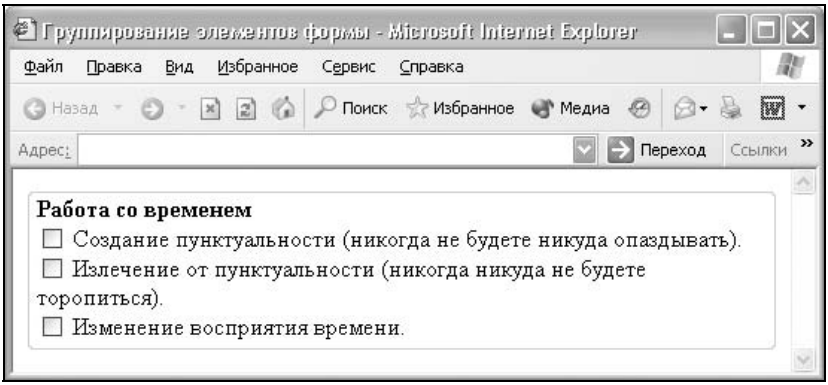


Рис. 7.16. Вид рамки, создаваемой с помощью тега FIELDSET

Чтобы добавить к рамке специальный заголовок, применяется контейнер LEGEND, который должен располагаться в теге FIELDSET. Внутри тега LEGEND допустимо использовать текст и теги форматирования, типа B, I, SUP, SUB, а также стили (листинг 7.22).

Листинг 7.22. Использование тега LEGEND

```
<html>
<body>
<form>
 <fieldset>
  <legend style="font-weight: bold">Изменение убеждений</legend>
  <input type=checkbox value=t1> Изменение религиозной веры (на выбор:
                                 буддизм, конфуцианство, индуизм).<br>
  <input type=checkbox value=t2> Изменение веры в непогрешимость
                                                 любимой партии.<br>
  <input type=checkbox value=t3> Убеждение в том, что инопланетяне
                                                      существуют.<br>
  <input type=checkbox value=t4> Выбор политического строя, как самого
        лучшего в своем роде (на выбор: феодализм, социализм, коммунизм,
                               капитализм).<br>
  <input type=checkbox value=t5> Повышение веры в собственные
                                                      способности.<br>
 </fieldset>
```

```
</form>
</body>
</html>
```

Любые пробелы внутри контейнера LEGEND будут проигнорированы, если вы все же хотите их добавить, применяйте символ неразделяемого пробела .

При использовании тегов FIELDSET и LEGEND учтите, что результат работы в разных браузерах будет несколько различаться, как показано на рис. 7.17—7.19 (на примере листинга 7.22).

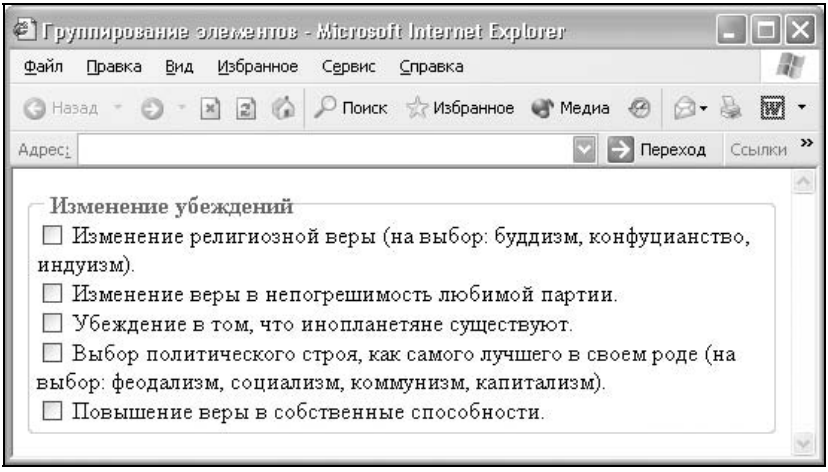


Рис. 7.17. Отображение в браузере Internet Explorer под Windows XP

Рис. 7.18. Отображение в браузере Opera 7

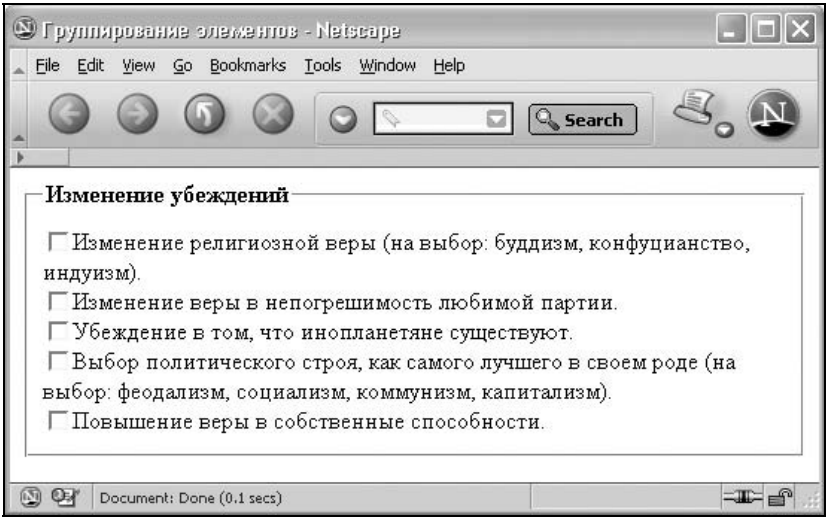


Рис. 7.19. Отображение в браузере Netscape 7

В браузере Internet Explorer рамка получается светлее, чем в других браузерах и с закругленными углами. Такой вид характерен только для операционной системы Windows XP, для Windows 98/2000 рамка будет прямоугольной безо всяких закруглений. Для остальных браузеров цвет рамки по умолчанию черный, а цвет заголовка совпадает с цветом текста. Кроме того, Opera "забывает" о полях вокруг текста, из-за этого текст "налипает" на рамку. Цвет рамки группы поменять не удастся, а вот отступы вокруг текста и цвет заголовка изменяются с помощью стилей (листинг 7.23).

Листинг 7.23. Изменение вида заголовка группы

```
<html>
<body>
<form>
<fieldset style="padding: 10px">
  <legend style="color: red">Пробуждение обаяния</legend>
...
</fieldset>
</form>
</body>
</html>
```

В этом примере отступы устанавливаются с помощью параметра `padding`, а цвет заголовка через атрибут `color`.

ГЛАВА 8

Выравнивание элементов

Выравнивая различные элементы веб-страницы по невидимой линии, мы тем самым зрительно связываем их друг с другом. Такие элементы, как текст и графические изображения, с помощью выравнивания можно объединить в одну группу, показывая связь между ними. Подобное логическое объединение вроде бы разных элементов придает дизайну сайта профессиональный и законченный вид, а читателю позволяет легче ориентироваться в документе и быстрее находить нужную информацию.

Особенность веб-страниц состоит в том, что ширина окна браузера может варьироваться в различных пределах и зависит от размера и разрешения монитора, настроек операционной системы и некоторых других факторов. Это приводит к тому, что методы выравнивания элементов как по горизонтали, так и по вертикали отличаются от методов, применяемых в текстовых редакторах или системах верстки, и должны учитывать указанную особенность. В идеале, веб-страница, в частности выровненные элементы, должна отображаться корректно и без ошибок при любой ширине окна браузера. Для этого существует несколько универсальных методов, основанных на использовании таблиц, стилей и слоев, которые и рассмотрены далее.

Выравнивание рисунков по горизонтали

Тег `IMG`, добавляющий изображение в документ, хорош еще и тем, что у него есть параметр `align`, который выравнивает рисунок по нужному краю. Если на странице после изображения идет текст, он в этом случае будет обтекать рисунок по его правому краю. То же правило относится не только к тексту, но и к другим элементам веб-страницы. Таким образом, когда нужно выровнять два изображения, находящихся на одной горизонтальной линии, — один по левому, а другой по правому краю — следует одновременно задать выравнивание у двух тегов `IMG`, как показано в листинге 8.1.

Листинг 8.1. Выравнивание двух рисунков по сторонам окна

```
<html>
<body>
<img src=left_title.gif width=100 height=100 align=left>
<img src=right_title.gif width=100 height=100 align=right>
<h1 align=center>Заголовок</h1>
</body>
</html>
```

Добавление параметра `align` сразу к двум тегам `IMG`, как показано в данном примере, открывает любопытную особенность. Хотя текст заголовка (тег `H1`) и находится в коде страницы в самом низу, он будет отображаться строго между картинками, как показано на рис. 8.1.

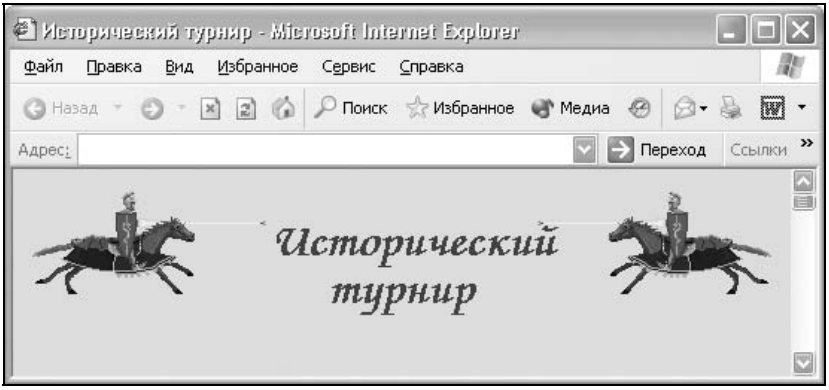


Рис. 8.1. Размещение текста по центру между изображениями

То же правило относится и к тексту после изображений: вместо того, чтобы располагаться строго после рисунков, текст будет обтекать их. Чтобы избавиться от подобной особенности, добавляется несколько тегов переносов строки `BR`, но это вряд ли можно назвать универсальным методом. Лучше обратиться к стилям и воспользоваться свойством `clear`; оно отменяет действие выравнивания. Стиль можно добавить к тому же тегу `BR`, поместив его после всех изображений, как показано в листинге 8.2. Кроме того, если уж речь зашла о стилях, допустимо заменить и параметр `align` конструкцией `style="float:left"`, где `left` указывает край, по которому происходит выравнивание.

Листинг 8.2. Отмена выравнивания изображений

```
<html>
<body>
```

```
<img src=left_title.gif width=100 height=100 style="float: left">
<img src=right_title.gif width=100 height=100 style="float: right">
<br style="clear: both">
...
</body>
</html>
```

Значение `both` параметра `clear` отменяет обтекание элемента одновременно с правого и левого края. Его рекомендуется устанавливать, когда требуется снять обтекание с двух сторон.

Приведенный способ выравнивания универсален, прост и работает во всех браузерах, но при этом имеет некоторые ограничения. В частности, выравнивание по вертикали всегда происходит только по верхнему краю. Поэтому можно обратиться к таблицам, они позволяют задавать положение рисунков одновременно по горизонтали и вертикали.

Так, если требуется изменить форматирование двух изображений, создаем таблицу, состоящую из двух колонок. Выравнивание внутри ячейки по горизонтали происходит с помощью параметра `align` тега `TD`, а по вертикали — с помощью параметра `valign`, как показано в листинге 8.3.

Листинг 8.3. Использование таблицы для выравнивания

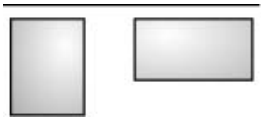
```
<html>
<body>
<table width=100% border=0 cellspacing=0 cellpadding=0>
 <tr>
  <td valign=bottom><img src=sample1.gif width=50 height=50></td>
  <td align=right valign=bottom><img src=sample2.gif width=50
height=86></td>
 </tr>
</table>
</body>
</html>
```

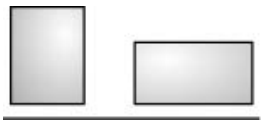
В данном примере показано, как выравнивать рисунки по нижнему краю, для этого используется значение `bottom` параметра `valign`. Обратите также внимание, что атрибут `align=left` не приводится, это значение для ячейки задано по умолчанию. Возможные положения изображений и соответствующие коды HTML приведены в табл. 8.1.

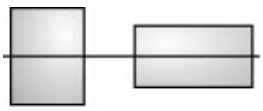
Разумеется, не обязательно одновременно выравнивать изображения по одной линии. Ничто не мешает задать для содержимого одной ячейки положение по верхнему краю, а для другой — по нижнему.

Таблица 8.1. *Способы выравнивания рисунков с помощью таблицы*

Выравнивание	Код

	`<td valign=top>`
	`<img src=sample1.gif width=50 height=50></td>`
	`<td align=right valign=bottom>`
	`<img src=sample2.gif width=50 height=86></td>`

	`<td valign=bottom>`
	`<img src=sample1.gif width=50 height=50></td>`
	`<td align=right valign=bottom>`
	`<img src=sample2.gif width=50 height=86></td>`

	`<td>`
	`<img src=sample1.gif width=50 height=50></td>`
	`<td align=right>`
	`<img src=sample2.gif width=50 height=86></td>`

Таблицы пригодятся и в случае, когда двум объектам на веб-странице надо одновременно задать выравнивание по центру и по правому или левому краю. Такое размещение часто используется при публикации формул с их нумерацией. Понятно, что HTML не позволяет создавать сложные формулы, поэтому приходится готовить их в другой программе и вставлять в документ как изображение (рис. 8.2).

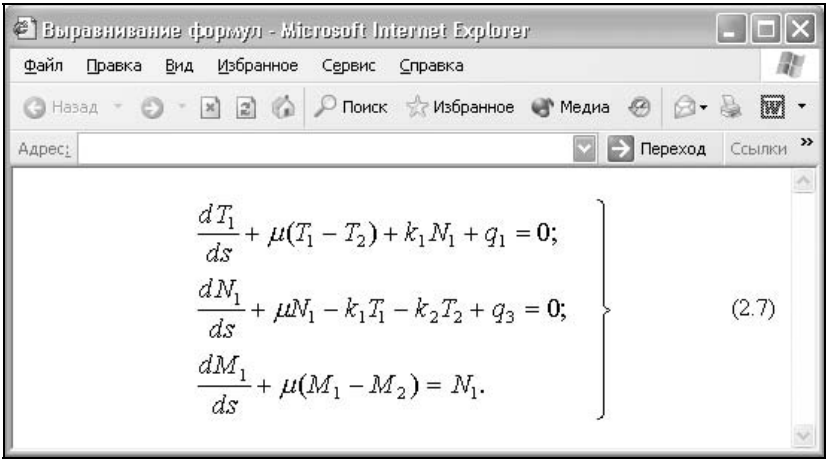


Рис. 8.2. Выравнивание формулы с ее номером

Для такого размещения элементов, как показано на рис. 8.2, требуется создать таблицу с тремя ячейками. Крайние из них должны иметь одинаковый

размер, в средней ячейке выравнивание делается по центру, а в правой — по правому краю (листинг 8.4).

Листинг 8.4. Выравнивание по центру и по краю

```
<html>
<body>
 <table width=100% border=0 cellspacing=0 cellpadding=0>
  <tr>
   <td width=10%> </td>
   <td align=center><img src=formula.gif width=289 height=158></td>
   <td width=10% align=right>(2.7)</td>
  </tr>
 </table>
</body>
</html>
```

Левая ячейка хоть и пуста, нужна для того, чтобы точно позиционировать формулу по центру, без нее произойдет небольшой сдвиг центрального рисунка влево. При необходимости в эту ячейку также можно поместить и текст.

Центрирование по вертикали

Одним из способов показать посетителю направленность и название сайта является использование сплэш-страницы. Это первая страница, на которой, как правило, расположен рисунок или flash-заставка, выражающие главную идею сайта. Изображение также является и ссылкой на остальные разделы сайта, своеобразным входом в его разделы. Подобный рисунок принято располагать по центру окна браузера, и независимо от изменения размеров окна выровненный по центру объект всегда остается на своем месте (рис. 8.3).

Центрирование элементов на веб-странице не ограничивается изображениями и применяется при добавлении различных подсказок или рекламных сообщений.

Самый простой и один из старых способов центрирования по вертикали состоит в использовании таблиц. Вначале создаем таблицу с шириной и высотой равной 100 %. По умолчанию все, что помещается в ячейку таблицы, будет автоматически располагаться по центру вертикали, поэтому никаких дополнительных параметров указывать не нужно. Что касается выравнивания по горизонтали, для этого к тегу `TD` следует добавить аргумент `align=center`, иначе изображение в таблице будет размещаться по левому краю (листинг 8.5).

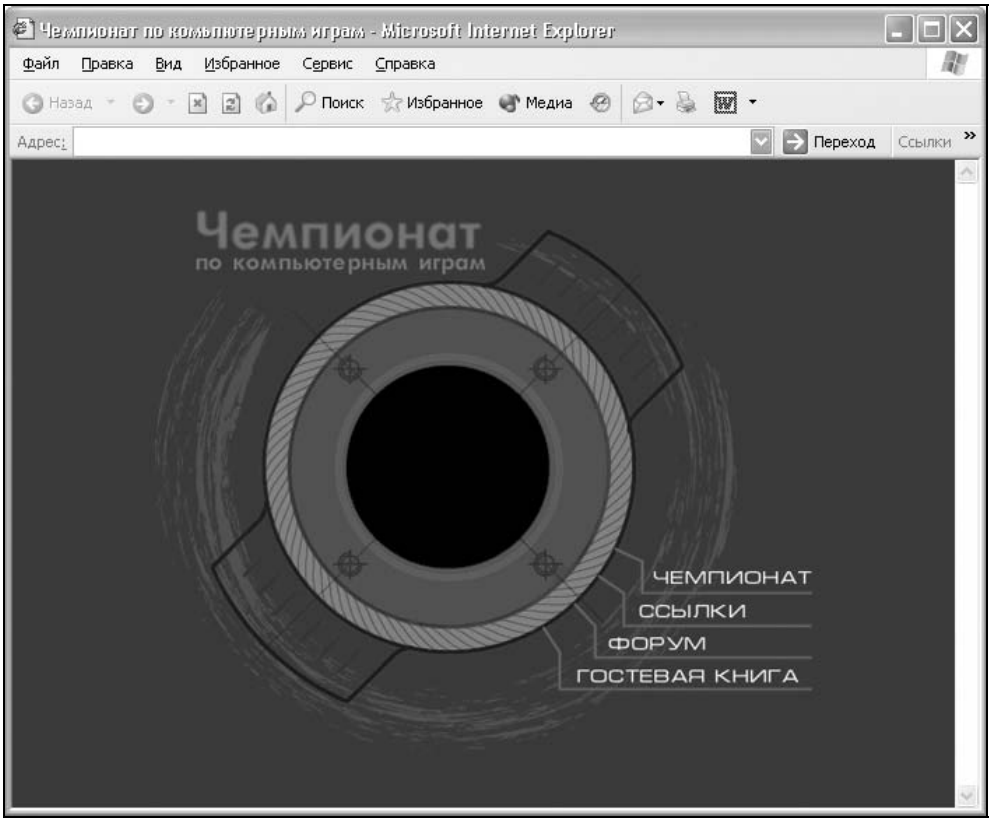


Рис. 8.3. Рисунок, выровненный по центру веб-страницы

Листинг 8.5. Центрирование по вертикали с помощью таблицы

```
<html>
<head>
<style type="text/css">
 BODY { margin: 0px; padding: 0px }
</style>
</head>
<body>
<table width=100% height=100%>
 <tr>
  <td align=center>
   <img src=images/title.gif width=414 height=100 alt="Рецепты HTML">
  </td>
 </tr>
```

```
</table>
</body>
</html>
```

По умолчанию в таблице рамка не отображается, поэтому параметр `border` тега `TABLE`, задающий толщину границы, указывать не обязательно. Ширина таблицы задается аргументом `width`, а ее высота — через параметр `height`. После того, как значения у них будут установлены на 100 %, как бы не изменялись размеры окна браузера, рисунок в ячейке таблицы будет находиться строго по центру.

При указанном способе размещения картинки не должно возникать горизонтальных или вертикальных полос прокрутки за исключением случая, когда размер окна браузера заметно меньше размеров изображения. Тем не менее в некоторых браузерах возникает эффект, при котором полосы прокрутки все же добавляются. Для этого случая и добавлен стиль к селектору `BODY`, который убирает отступы на веб-странице. Более подробно об отступах читайте в *главе 9*.

Современный подход к верстке веб-страниц предполагает активное использование слоев вместо таблиц. *Слои* представляют собой элементы веб-страницы, видом и положением которых управляют с помощью стилей, что позволяет накладывать слои друг на друга, а также динамически менять параметры. Это позволяет создавать в документе разные эффекты, такие как выпадающие меню, игры, разворачивающиеся баннеры, плавающие окна и т. д.

Рассмотрим альтернативное решение приведенной задачи, которое реализуется с помощью слоев.

Чтобы слой точно располагался по центру веб-страницы, воспользуемся *абсолютным позиционированием*. Для этого в стилях элемента требуется указать параметр `position: absolute`. При абсолютном позиционировании отсчет координат всегда ведется от левого верхнего угла окна браузера, а не относительно родительского элемента, как обычно.

Чтобы слой размещался по центру веб-страницы, зададим ему положение 50 % слева и сверху через свойства `left` и `top`. Поскольку отсчет координат ведется от левого верхнего угла слоя, то слой придется сдвинуть от полученного положения на половину его ширины влево и на половину высоты вверх. Эта манипуляция осуществляется параметрами `margin-left` и `margin-top`, как показано в листинге 8.6.

Листинг 8.6. Центрирование по вертикали с помощью стилей

```
<html>
<head>
```

```
<style type="text/css">
 #center {
  position: absolute;        /* Абсолютное позиционирование */
  left: 50%;                 /* Положение от левого края */
  top: 50%;                  /* Положение от правого края */
  margin-left: -207px;       /* Смещение картинки влево */
  margin-top: -50px          /* Смещение картинки вверх */
}
</style>
</head>
<body>
  <img id=center src=sample.gif width=414 height=100 alt="Рецепты HTML">
</body>
</html>
```

В данном примере привязка изображения к стилям происходит через идентификатор `center`. Ширина рисунка составляет 414 пикселов, разделив его пополам, получим число 207, которое выступает в качестве значения параметра `margin-left`. Отрицательные координаты получаются вследствие того, что сдвиг происходит в сторону, противоположную новой оси координат. Для наглядности свойства стилей приведены на рис. 8.4.

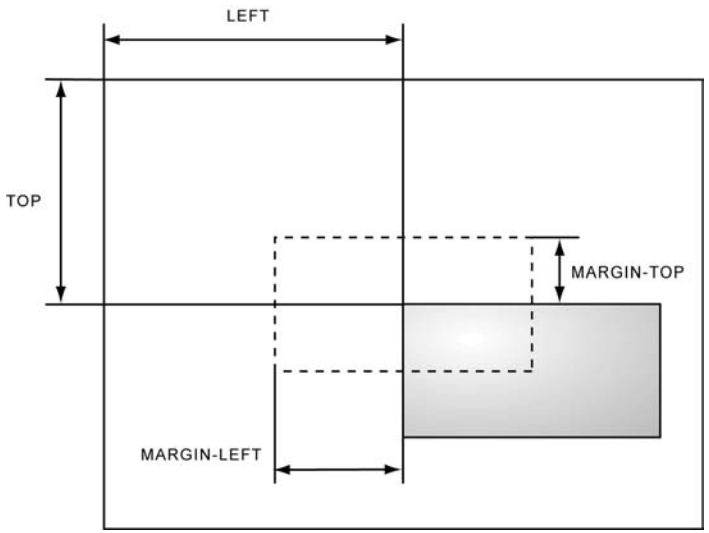


Рис. 8.4. Параметры CSS для центрирования рисунка

Удобство применения стилей и слоев заключается в том, что они позволяют располагать один элемент веб-страницы поверх другого. Например, слои пригодятся для создания наложения, когда изображение отображается по-

верх уже существующего контента типа текста. Такой подход используется для создания разных эффектов на веб-странице: всплывающих окон, рекламы, контекстных меню и т. д.

На рис. 8.5 показан пример того, как слой выравнивается по центру окна и одновременно располагается поверх текста.

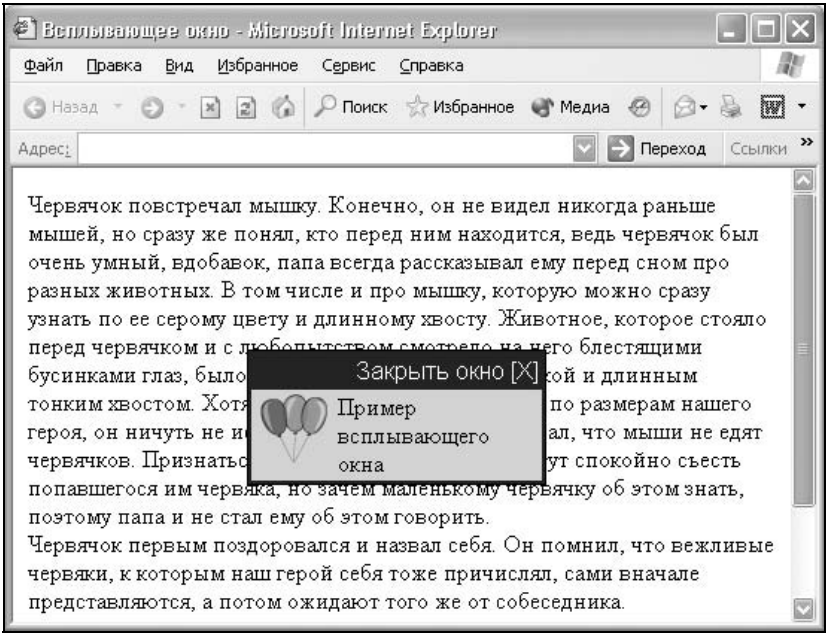


Рис. 8.5. Центрирование слоя

Чтобы реализовать приведенную схему, создадим слой с именем `popup` и через параметр `position` установим для него абсолютное позиционирование. Теперь следует заняться простейшими расчетами. Берем ширину слоя в процентах и вычитаем ее из 100 % (общая ширина страницы), после чего делим пополам. Полученное число будет значением параметра `left`, который выражает положение от левого края. Аналогично поступаем и с высотой, после чего присваиваем результат параметру `top` (листинг 8.7).

Листинг 8.7. Центрирование слоя на веб-странице

```
<html>
<head>
<style type="text/css">
 #popup {
  position: absolute;        /* Абсолютно заданные координаты */
```

```
  top: 40%;                  /* Расстояние от верхнего края */
  left: 30%;                 /* Расстояние от левого края */
  width: 40%;                /* Ширина слоя */
  height: 20%;               /* Высота слоя */
  background: #fc0;          /* Цвет фона */
}
</style>
</head>
<body>
<div id=popup>
  <table width=100% height=100% border=1 bordercolor=navy cellpadding=2
cellspacing=0>
  <tr>
   <td height=10 align=right bgcolor=navy style="font-family: sans-serif;
color: white">Закрыть окно [X]</td>
  </tr>
  <tr>
   <td><img src=ball.gif align=left>Пример всплывающего окна</td>
  </tr>
  </table>
</div>
...
</body>
</html>
```

За счет того, что у слоя задается абсолютное позиционирование, он помещается поверх остальных элементов веб-страницы, поэтому никаких дополнительных параметров вводить не требуется.

При указанном способе выравнивания иногда можно заметить, что по высоте слой располагается не совсем по центру. Это связано с тем, что высота слоя, хотя и задается фиксированной через параметр `height`, в некоторых случаях может и превышать указанную величину. Например, подобный эффект наблюдается, если внутри слоя располагается изображение, как показано на рис. 8.5. Оно выступает в качестве своеобразной распорки и не дает слою уменьшаться по высоте. За счет этого нижний отступ от слоя становится меньше, чем верхний, который жестко задан атрибутом `top` и не изменяет своего значения.

Выравнивание слоя по центру

Основное отличие веб-страницы от листа бумаги заключается в их размерах. Если лист имеет заданную фиксированную ширину и высоту, то по отношению к веб-сайту такого сказать нельзя. Веб-документ отображается в ок-

не браузера и может изменять свои размеры в зависимости от операционной системы, типа монитора, установленного разрешения и т. д. Использование выравнивания позволяет проигнорировать указанную особенность и располагать элемент у края окна или по его центру.

Когда речь идет об использовании слоев, то в нашем распоряжении имеется несколько способов, чтобы задать выравнивание, — с помощью отступов, параметра `text-align`, параметра `align` тега `DIV`, а также через абсолютное позиционирование.

Использование отступов

Применение отступов для центрирования слоя по горизонтали основано на свойствах CSS `margin-left` и `margin-right`, как показано в листинге 8.8.

Листинг 8.8. Размещения слоя по центру через отступы

```
<html>
<head>
<style type="text/css">
 #centerLayer {
  margin-left: 30%;          /* Отступ слева */
  margin-right: 30%;         /* Отступ справа */
  background: #fc0;          /* Цвет фона слоя */
  padding: 10px              /* Расстояние от границы до содержимого */
}
</style>
</head>
<body>
<div id=centerLayer>
  ...
</div>
</body>
</html>
```

В данном примере показано размещение слоя шириной 40 % по центру. Хотя сама ширина напрямую не задается, она определяется значением атрибутов `margin-left` и `margin-right`. Эти параметры устанавливают отступ слева и справа; чтобы слой располагался посередине, их значения должны быть равны.

Недочетом приведенного метода является необходимость задавать ширину слоя в процентах. В случае, когда ширина указывается в пикселах, вышеописанный прием не работает, ведь нам не известна общая ширина окна в пикселах, а значит, и определить отступы не представляется возможным.

Можно, конечно, воспользоваться языком JavaScript, но его лучше приберечь для более сложных случаев.

Применение параметра *text-align*

Следующий способ более универсален и не зависит от того, какие единицы измерения используются для установки ширины. Основная работа в этом случае выпадает на долю параметра `text-align`. Его задача состоит в выравнивании элемента по нужному краю (листинг 8.9).

Листинг 8.9. Выравнивание слоя с помощью параметра `text-align`

```
<html>
<head>
<style type="text/css">
 BODY {
  text-align: center  /* Выравниваем содержимое веб-страницы по центру */
}
 #centerLayer {
  width: 400px;              /* Ширина слоя в пикселах */
  margin-left: auto;         /* Отступ слева (для браузера Opera) */
  margin-right: auto;        /* Отступ справа (для браузера Opera)*/
  background: #fc0;          /* Цвет фона */
  padding: 10px;             /* Поля вокруг текста */
  text-align: left           /* Выравнивание содержимого по левому краю */
}
</style>
</head>
<body>
<div id=centerLayer>
  ...
</div>
</body>
</html>
```

Выравнивание по центру производится с помощью атрибута `text-align: center`, который устанавливается для селектора `BODY`. Этот параметр воздействует не только на слои, но и на их содержимое, поэтому для стиля самого слоя необходимо также использовать `text-align`, но уже с другим значением. Обычно используется значение `left` (как в примере), которое задает выравнивание по левому краю, и `justify`, определяющее выравнивание по ширине. По правому краю, как правило, текст не выравнивается из-за того, что читать его становится неудобно.

Ширина слоя определяется параметром `width`, его значение можно указывать в процентах или пикселах. Атрибуты `margin-left` и `margin-right` со значением `auto` используются для браузеров Opera, Netscape и Firefox, которые обязательно требуют их наличия.

Параметр *align* тега *DIV*

Данный способ размещения по центру вообще не требует использования стилей и связан с параметром `align` тега `DIV`. Указывая значение `center`, заставляем содержимое слоя выравниваться по его центру. Поэтому необходимо создать два слоя, один из которых будет служить контейнером для другого, как показано в листинге 8.10.

Листинг 8.10. Выравнивание слоя с помощью параметра `align`

```
<html>
<head>
<style type="text/css">
 #centerLayer {
  width: 400px;              /* Ширина слоя в пикселах */
  background: #fc0;          /* Цвет фона */
  padding: 10px;             /* Поля вокруг текста */
  text-align: left           /* Ширина слоя в пикселах */
}
</style>
</head>
<body>
<div align=center>
 <div id=centerLayer>
  ...
 </div>
</div>
</body>
</html>
```

Опять же, как и в случае использования параметра `text-align`, размещаться по центру будет и текст внутри слоя. Поэтому следует принудительно задать ему необходимое выравнивание через стили.

Абсолютное позиционирование слоя

Выравнивание слоя по центру происходит аналогично тому, как было показано ранее в листинге 8.6. Вначале следует указать ширину и высоту слоя с помощью параметров `width` и `height`. Размеры можно задавать в пикселах,

процентах или других единицах. Ширину, например, можно определить в процентах, а высоту в пикселах. Из-за этой особенности предлагаемый метод размещения по центру является наиболее универсальным.

Следующий шаг — задаем абсолютное позиционирование слоя через аргумент `position: absolute`. Положение слоя следует определить как 50 % по горизонтали и по вертикали с помощью свойств `left` и `top`. Эти значения остаются неизменными независимо от используемых единиц измерения.

Так как координаты слоя определяются от его левого верхнего угла, для точного выравнивания следует добавить параметры `margin-left` и `margin-top` с отрицательными значениями. Их величина должна быть равна половине ширины слоя (для `margin-left`) и высоты (для `margin-top`).

Чтобы высота слоя не менялась из-за его контента, включен параметр `overflow: auto`. Он добавляет полосы прокрутки, если в них возникнет необходимость, высота при этом остается неизменной (листинг 8.11). Учтите, что такая запись не работает в браузере Opera 6 и ниже.

Листинг 8.11. Выравнивание по центру слоя с шириной, заданной в пикселах

```
<html>
<head>
<style>
 #centerLayer {
  width: 400px;                    /* Ширина слоя в пикселах */
  height: 300px;                   /* Высота слоя в пикселах */
  position: absolute;              /* Абсолютное позиционирование */
  left: 50%;                       /* Положение слоя от левого края */
  top: 50%;                        /* Положение слоя от верхнего края */
  margin-left: -200px;             /* Отступ слева */
  margin-top: -150px;              /* Отступ сверху */
  background: #fc0;                /* Цвет фона */
  border: solid 1px black;         /* Параметры рамки */
  padding: 10px;                   /* Поля вокруг текста */
  overflow: auto                   /* Добавление полосы прокрутки */
}
</style>
</head>

<body>
<div id=centerLayer>
  ...
</div>
</body>
</html>
```

В случае использования процентной записи стиль меняется незначительно. Надо так же поделить ширину и высоту пополам и добавить полученные значения в качестве аргументов к свойствам `margin-left` и `margin-top` (листинг 8.12).

Листинг 8.12. Выравнивание по центру слоя с шириной, заданной в процентах

```
<html>
<head>
<style>
 #centerLayer {
  width: 40%;                    /* Ширина слоя в процентах */
  height: 30%;                   /* Высота слоя в процентах */
  position: absolute;            /* Абсолютное позиционирование */
  left: 50%;                     /* Положение слоя от левого края */
  top: 50%;                      /* Положение слоя от верхнего края */
  margin-left: -20%;             /* Отступ слева */
  margin-top: -15%;              /* Отступ сверху */
}
</style>
</head>

<body>
<div id=centerLayer>
  ...
</div>
</body>
</html>
```

Ширина и высота слоя напрямую связана с отступами слева и сверху; если требуется установить значение одного из параметров в процентах, соответственно поменяется и запись другого параметра. Как показано в данном примере, ширина слоя установлена в 20 %, следовательно, и для свойства `margin-left` также надо применить проценты, в данном случае −20 %.

Указанная особенность позволяет применять любые единицы измерения, а не ограничиваться только одной формой записи, что делает код подходящим практически для всех случаев.

Манипуляции с текстом

Чтобы придать выразительность тексту и повысить его восприятие, можно отказаться от традиционной схемы расположения заголовков и абзацев и попробовать сместить их относительно друг друга, как показано на рис. 8.6.

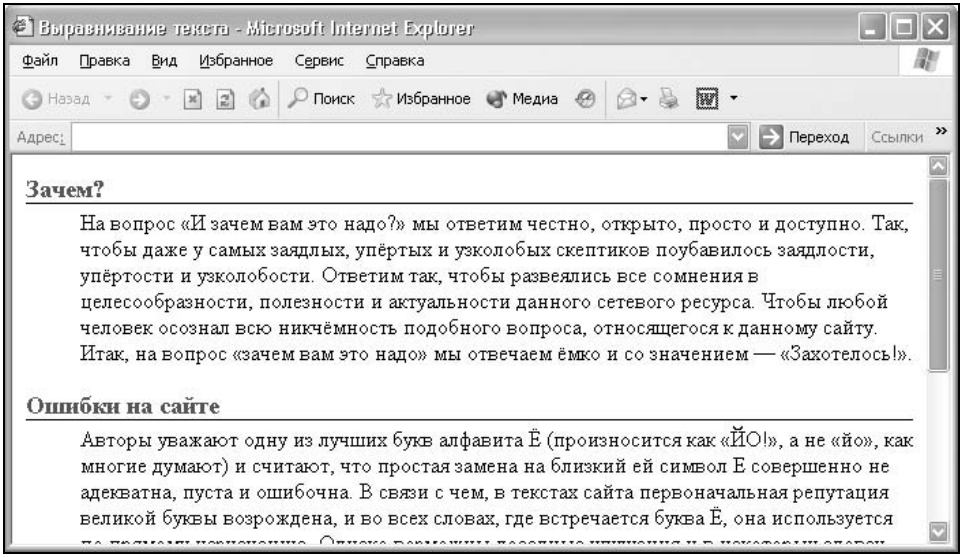


Рис. 8.6. Сдвиг заголовков относительно линии абзацев

Для создания сдвига текста от вертикальной линии воспользуемся свойством `margin-left`, присваивая его стилю параграфа `P`. Сам заголовок текста, формируемый тегом `H2`, остается на своем месте, и требуется лишь задать ему стиль оформления, как показано в листинге 8.13.

Листинг 8.13. Сдвиг текста относительно вертикальной линии

```
<html>
<head>
<style type="text/css">
 H2 {
  font-size: 110%;           /* Размер шрифта */
  color: #09599d;            /* Цвет заголовка */
  font-weight: bold;         /* Жирное начертание */
  margin-bottom: 0.2em;      /* Отступ между заголовком и текстом
                                под ним */
  border-bottom: 1px solid black    /* Линия под заголовком */
}
 P {
  padding-left: 40px;        /* Отступ слева */
  margin-top: 0px            /* Отступа сверху нет */
}
</style>
</head>
```

```
<body>
<h2>Заголовок</h2>
<p>Текст абзаца...</p>
</body>
</html>
```

Как известно, при использовании блочного тега `P` сверху и снизу параграфа автоматически добавляются отступы. Их величину можно регулировать, если воспользоваться свойством `margin-top` или `margon-bottom`. В данном случае вертикальные отступы между заголовком и текстом под ним управляются с помощью селектора `H2`, как показано в этом примере. Разумеется, аналогичный результат можно получить, сделав наоборот: обнулив значение `margin-bottom` у селектора `H2` и добавив `margin-top: 0.2em` к селектору `P`.

При форматировании текста, как показано на рис. 8.6, его организация становится понятна с первого взгляда. Сразу видно, какой текст является заголовком блока, и абзацы четко отделены друг от друга. Но можно еще поэкспериментировать с текстом и вообще вынести заголовки за пределы блока, разместив их слева от абзаца, как показано на рис. 8.7.

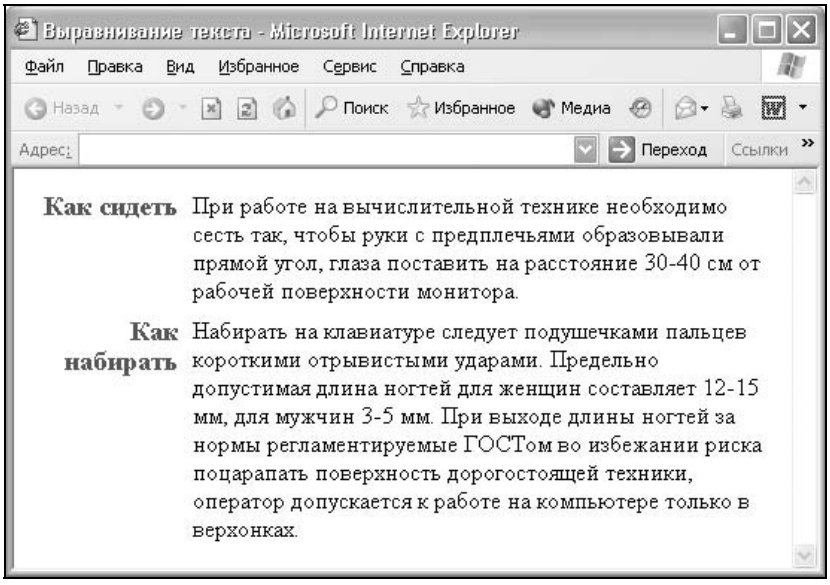


Рис. 8.7. Размещение заголовков слева от текста

Чтобы реализовать приведенную схему, есть два способа, основанных на разных системах верстки, — с помощью стилей и таблиц. И тот и другой способ имеют свои недочеты, поэтому говорить о том, что один из них лучше, вряд ли имеет смысл. Рассмотрим вначале выравнивание текстовых

блоков с помощью стилей. Для этого воспользуемся свойством `float`, применяя его к заголовку и параграфу. Этот параметр CSS определяет, по какой стороне будет выравниваться элемент, при этом остальные элементы будут обтекать его с других сторон. Добавление этого параметра к стилю текста и заголовка позволяет получить требуемый результат, заголовок будет располагаться слева от текста (листинг 8.14).

Листинг 8.14. Выравнивание заголовка слева от текста

```
<html>
<head>
<style type="text/css">
 H2 {
  font-size: 110%;           /* Размер шрифта */
  color: #09599d;            /* Цвет заголовка */
  float: left;               /* Обтекание слева */
  width: 20%;                /* Ширина заголовка в процентах */
  text-align: right;         /* Выравнивание по правому краю */
  margin-top: 0px            /* Отступа сверху нет */
}
 P {
  padding-left: 10px;        /* Расстояние между заголовком и текстом */
  margin-top: 0px;           /* Отступа сверху нет */
  margin-bottom: 0.5em;      /* Отступ снизу */
  float: left;               /* Обтекание слева */
  width: 78%                 /* Ширина текста в процентах */
}
</style>
</head>
<body>
 <h2>Заголовок</h2>
 <p>Текст...</p>
</body>
</html>
```

Заметим, что приведенный метод работает лишь в том случае, если после заголовка идет всего один параграф. Добавление нового тега `P` приведет к тому, что текст окажется сдвинутым влево относительно других блоков. Так что придется после контейнера `H2` ставить только один параграф. В случае необходимости разделить текст на абзацы воспользуйтесь тегом переноса строки `BR`. Обратите также внимание, что ширина основного текста установлена в 78 %, а не 80 %, как логично можно предположить, отняв от 100 число 20. Такое число (78 %) необходимо для корректного отображения до-

кумента в браузере Netscape (Firefox). Это связано с особенностью данного браузера включать значение `padding` в общую ширину слоя.

Указанных недочетов лишены таблицы, применяемые в качестве невидимой сетки, в ячейках которых помещаются нужные текстовые элементы. Однако и таблица в качестве основы для выравнивания текста не является панацеей и имеет свои минусы. Так, конечный код получается довольно громоздким, и править его довольно неудобно. Впрочем, для тех, кто пользуется HTML-редактором для верстки веб-страниц, вряд ли подобная особенность является проблемой. Другим недостатком таблиц является тот факт, что пока таблица не загрузится полностью, ее содержимое не будет отображаться в браузере. Опять же это существенно для текстов большого объема, в этом случае пользователю придется ожидать полной загрузки таблицы, прежде чем приступить к чтению. Понятие "большой объем", конечно же, относительно и зависит от способа подключения к Интернету, скорости канала и других факторов.

Таким образом, получается, что независимо от способа верстки недостатки есть в любом случае, поэтому какой способ предпочесть, зависит от разработчика.

Рассмотрим способ выравнивания заголовков и текста с помощью таблиц. Для этого создаем таблицу, состоящую из двух колонок, в левую ячейку помещаем заголовок, а в правую — собственно параграфы текста. При этом обязательно следует задать выравнивание во всех ячейках по верхнему краю параметром `valign=top` тега `TD`, и для ячейки с заголовками установить выравнивание по правому краю (листинг 8.15). Можно, конечно, и не сближать между собой таким способом текст в ячейках и оставить выравнивание по горизонтали заданным по умолчанию, но тогда связь между заголовком и текстом становится менее выраженной.

Листинг 8.15. Выравнивание текста с помощью таблиц

```
<html>
<head>
<style type="text/css">

H2 {
  font-size: 110%;           /* Размер шрифта */
  color: #09599d;            /* Цвет заголовка */
  margin: 0px                /* Убираем отступы вокруг заголовка */
}
 P {
  margin-top: 0px;           /* Отступа сверху нет */
  margin-bottom: 0.5em       /* Отступ снизу */
}
```

```
</style>
</head>
<body>
 <table width=100% border=0 cellspacing=0 cellpadding=4>
  <tr valign=top>
    <td width=20% align=right><h2>Заголовок</h2></td>
    <td><p>Текст...</p></td>
  </tr>
  <tr valign=top>
    <td align=right><h2>Заголовок</h2></td>
    <td><p>Текст...</p></td>
  </tr>
 </table>
</body>
</html>
```

Ширину колонок, как показано в этом примере, достаточно задать в первых ячейках, остальные будут подстраиваться под них. Расстояние между колонками в данном случае определяется значением параметра `cellpadding` тега `TABLE`, также можно воспользоваться и стилями, указав для параграфа отступ с помощью свойства `margin-left`.

Указанный способ применения таблиц пригодится и для выравнивания элементов форм, например, как показано на рис. 8.8.

Чтобы текст возле текстовых полей был выровнен по левому краю, а сами элементы формы — по правому, потребуется таблица с невидимой границей и двумя колонками. В левой колонке будет размещаться текст, а в правой текстовые поля, как показано в листинге 8.16.

Листинг 8.16. Выравнивание элементов форм с помощью таблицы

```
<html>
<body>
<form method=POST action=/inc/ac.php target=popmsg name=comment>
 <table width=100% border=0 cellspacing=0 cellpadding=4>
  <tr>
    <td align=right>Имя</td>
    <td><input type=text name=name maxlength=50 size=20></td>
  </tr>
  <tr>
    <td align=right>E-mail</td>
    <td><input type=text name=email maxlength=50 size=20></td>
  </tr>
```

```
  <tr>
    <td align=right valign=top>Комментарий</td>
    <td><textarea name=text cols=45 rows=10></textarea></td>
  </tr>
  <tr>
    <td> </td>
    <td><input type=submit value="Добавить комментарий"></td>
  </tr>
 </table>
</form>
</body>
</html>
```

В данном примере в тех ячейках, где требуется задать выравнивание по правому краю, добавлен параметр `align=right`.

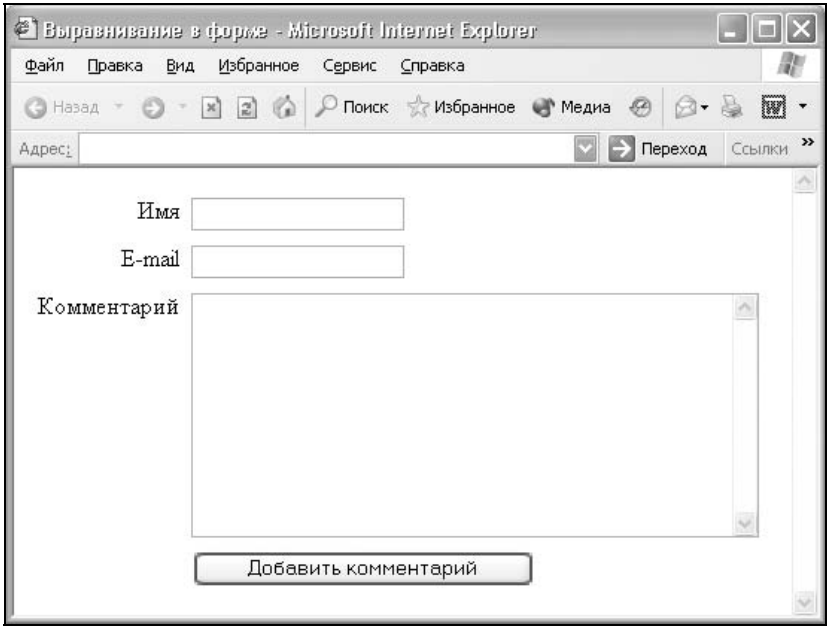


Рис. 8.8. Выравнивание элементов формы

Таким образом, таблицы как способ выравнивания различных элементов списывать со счета еще рано, несмотря на возможности CSS. В некоторых случаях, как видно из приведенных примеров, таблицы дают простой и удобный код.

Выравнивание текста внутри слоя

В отличие от ячеек таблиц, в которых можно задавать выравнивание как по вертикали, так и по горизонтали, со слоями дело обстоит несколько иначе. Если для горизонтального выравнивания контента слоя существует атрибут `text-align`, то для выравнивания вертикального контента прямого метода "в лоб" нет.

Выравнивание по верхнему краю

В данном случае все зависит от того, где именно требуется расположить текст по вертикали. Так, по верхнему краю содержимое слоя выравнивается по умолчанию, и задавать какие-то особые стилевые параметры не требуется. В табл. 8.2 показаны возможные способы расположения текста возле верхней границы слоя.

Таблица 8.2. *Способы выравнивания текста внутри слоя по верхнему краю*

Выравнивание	Код

текст	`<div style="height: 300px; width: 400px; background: #fc0; padding: 7px">`
	`текст`
	`</div>`

текст	`<div style="height: 300px; width: 400px; background: #fc0; padding: 7px; text-align: center">`
	`текст`
	`</div>`

текст	`<div style="height: 300px; width: 400px; background: #fc0; padding: 7px; text-align: right">`
	`текст`
	`</div>`

В табл. 8.2 показано создание слоя с помощью тега `DIV` фиксированной ширины и высоты, которые задаются соответственно параметрами `width` и `height`. Цвет фона прямоугольника определяется атрибутом `align`, а расстояние между текстом и границей слоя — через свойство `padding`. Разница между примерами в таблице заключается в значении параметра `text-align`, устанавливающего горизонтальное выравнивание текста внутри слоя. Если этот параметр не задан, то выравнивание текста происходит по левому краю, что и продемонстрировано в первом примере.

Выравнивание по центру слоя

Чтобы задать выравнивание посередине слоя, можно воспользоваться двумя способами. Первый способ похож на тот, что приведен в листинге 8.6. А именно используются свойства `left` и `top` со значением 50 %, которые смещают левый верхний угол содержимого слоя точно в его центр. Затем происходит корректировка положения с помощью задания отступов `margin-left` и `margin-top` с отрицательными аргументами. Применяя подобный рецепт к тексту, получим код, приведенный в листинге 8.17.

Листинг 8.17. Выравнивание текста по центру слоя через позиционирование

```
<html>
<head>
<style type="text/css">
 #layer {
  height: 300px;              /* Высота слоя в пикселах */
  width: 400px;               /* Ширина слоя в пикселах */
  background: #fc0;           /* Цвет фона */
  border: 1px solid black;    /* Вид рамки вокруг */
  padding: 7px;               /* Поля вокруг содержимого */
  text-align: center          /* Выравнивание текста по центру */
}
 #text {
  position: relative;         /* Относительное позиционирование */
  top: 50%;                   /* Положение от верхнего края */
  margin-top: -0.5em          /* Сдвиг вверх на половину высоты текста */
}
</style>
</head>
<body>
<div id=layer>
 <div id=text>
  текст
 </div>
</div>
</body>
</html>
```

В данном примере используется два селектора стиля. Первый, с именем `layer`, определяет вид слоя — цвет фона, параметры рамки, а также его высоту и ширину. Внутрь него вкладываем еще один слой с именем `text`, который и задает положение нашего текста. Чтобы поместить текст посередине слоя, требуется установить *относительное позиционирование* с помощью

атрибута `position: relative`. При таком позиционировании отсчет координат ведется от левого верхнего угла родительского элемента, в данном случае в качестве него выступает слой `layer`. Теперь остается сдвинуть текст вниз на середину, воспользовавшись параметром `top` и задав у него значение 50 %. Поскольку отсчет начала координат ведется от левого верхнего угла текстового блока, то в случае одной строки ее необходимо сместить вверх, присвоив отрицательное значение в `0.5em` параметру `margin-top`.

Сразу заметим, что приведенный код хорошо себя показывает только для текста в одну-две строки. Основных недостатков, которые не позволяют назвать код универсальным, два. Во-первых, точно определить высоту, на которую следует сдвигать текст, в случае использования нескольких строк, довольно затруднительно. Особенно если ширина слоя указана в процентах, при этом число строк в зависимости от изменения размеров окна браузера может варьироваться в широких пределах. И, во-вторых, превышение текстовым блоком высоты слоя приводит к тому, что строки начинают выходить за пределы цветного прямоугольника. Казалось бы, что надо просто убрать параметр `height`, устанавливающий четко заданную высоту слоя. Но этот вариант помогает только для браузеров Opera и Netscape, а Internet Explorer продолжает отображать текст ниже слоя (рис. 8.9).

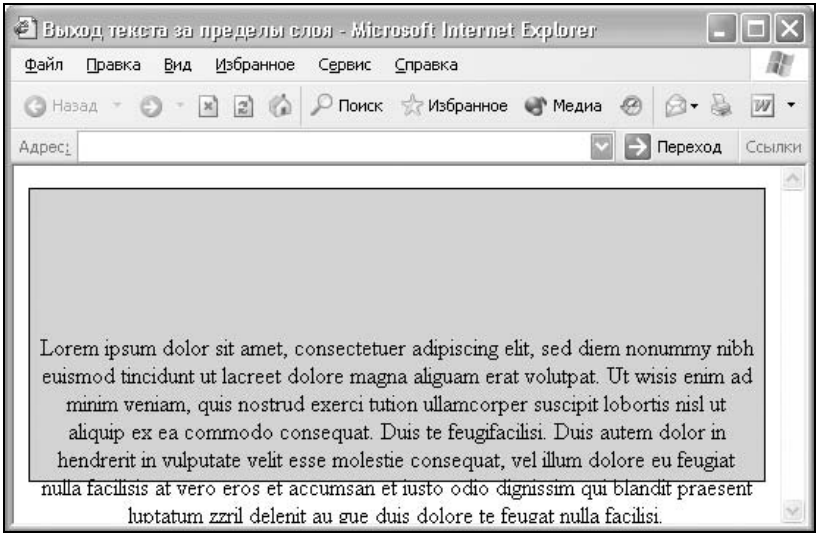


Рис. 8.9. Выход содержимого слоя за его пределы

Другой способ выравнивания текста по центру слоя состоит в применении отступов сверху и снизу от содержимого слоя. Для этой цели можно воспользоваться свойствами `padding-top` и `padding-bottom`, которые определяют расстояние от верхнего и нижнего края текста до границы слоя соответственно. Кроме них существует и универсальный параметр `padding`, устанав-

ливающий поля вокруг текста сразу в четырех направлениях. Если число аргументов равно двум, как показано в листинге 8.18, то первый задает отступ одновременно сверху и снизу по вертикали, а второй — по горизонтали.

Листинг 8.18. Выравнивание текста внутри слоя с помощью полей

```
<html>
<head>
<style type="text/css">
 #layer {
  width: 400px;                    /* Высота слоя в пикселах */
  background: #fc0;                /* Цвет фона */
  border: 1px solid black;         /* Вид рамки вокруг */
  padding: 150px 7px;              /* Поля вокруг содержимого */
  text-align: center               /* Выравнивание текста по центру */
}
</style>
</head>
<body>
<div id=layer>
  Текст
</div>
</body>
</html>
```

Высоту слоя в данном случае указывать не обязательно, она будет формироваться автоматически и складывается из высоты текста и вертикальных полей. В определенном смысле это является и недочетом приведенного способа выравнивания, поскольку жестко установить высоту слоя довольно проблематично.

Выравнивание по нижнему краю

Аналогично с помощью отступов можно выравнивать текст и по нижнему краю слоя. При этом можно ограничиться только одним параметром — `padding-top`, регулирующим значение отступа сверху от текста, как показано в листинге 8.19.

Листинг 8.19. Выравнивание текста по нижнему краю

```
<html>
<head>
<style type="text/css">
```

```
 BODY {
  margin: 0px; padding: 0px
}
 #title {
  width: 100%;                        /* Высота слоя в процентах */
  background: #f90;                   /* Цвет фона слоя */
  padding-top: 20px;                  /* Поле сверху от текста */
  font-family: Arial, sans-serif;     /* Рубленый шрифт текста */
  font-size: 50px;                    /* Размер шрифта */
  color: white;                       /* Белый цвет текста */
  font-weight: bold;                  /* Жирное начертание */
  text-align: center                  /* Выравнивание текста по центру */
}
</style>
</head>
<body>
<div id=title>
 <span style="position: relative; top: 12px">Пример заголовка</span>
</div>
</body>
</html>
```

В данном примере показано создание заголовка, выровненного по нижнему краю цветного прямоугольника, характеристики которого определяются с помощью селектора `title`. Чтобы не было отступов от края браузера, добавлены свойства `margin` и `padding` с нулевыми значениями для селектора `BODY`. Более подробную информацию о том, почему используется именно такой подход для создания отступов, смотрите в *главе 9*. Результат примера из листинга 8.19 продемонстрирован на рис. 8.10.

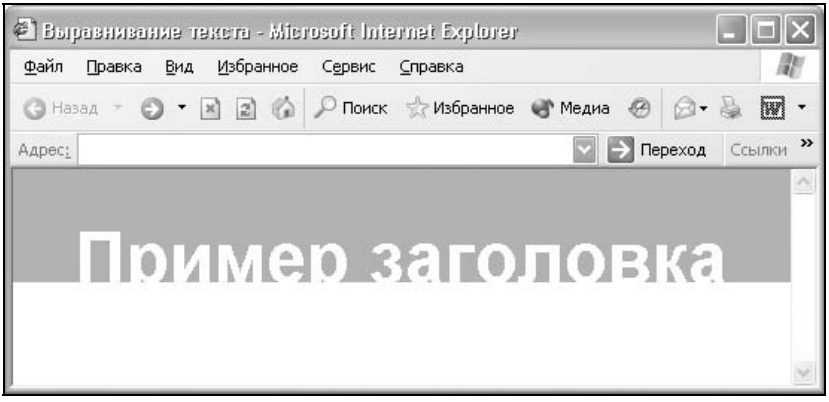


Рис. 8.10. Выравнивание текста по нижнему краю слоя

При любом способе выравнивания между текстом и границей слоя будет небольшой промежуток, размер которого зависит от шрифта. Чтобы избавиться от промежутка, в примере текст смещается вниз путем задания у текста относительного позиционирования и параметра `top`. Как видно на рис. 8.10, хвостик у буквы "р" пропадает, но оформление текста становится более выразительным. Изменяя цвет текста и фона, а также перемещая текст относительно базовой линии, можно получить множество интересных эффектов.

ГЛАВА 9

Отступы и поля

Любое пустое пространство вокруг элемента невольно притягивает к нему взгляд, а для текста еще и обеспечивает его нормальное восприятие. Не зря во всех книгах существуют внешние и внутренние поля, благодаря которым текст легче воспринимается при чтении. Пустой промежуток вокруг элемента не только выделяет его на веб-странице, но и позволяет отделить один элемент от другого. Использование отступов не ограничивается текстом или созданием пустых полей, как и большинство других техник; с помощью отступов можно выравнивать элементы, задавать колонки, верстать документы и многое другое.

Создание отступов

Отступом мы будем называть пространство от внешней границы текущего элемента до внутренней границы его родительского элемента. Например, текст на рис. 9.1, помещенный внутрь слоя, по отношению к нему выступает как *дочерний элемент*, а слой, соответственно, для текста будет родителем. В этом случае отступ будет отсчитываться от рамки слоя до края текста. Отступы не обязательно должны быть равными со всех сторон, по желанию разработчика их значение можно изменять по своему усмотрению.

Если у элемента нет родителя, тогда отступом будет расстояние от границы элемента до края окна браузера.

Для управления значениями отступов вокруг элемента предназначен стилевой атрибут `margin`. Это универсальный параметр, и в зависимости от числа значений он устанавливает отступы со всех сторон элемента или для каждой его стороны отдельно. Например, указав одно-единственное значение, получим в итоге отступы вокруг элемента. Если указаны два аргумента, то первый устанавливает горизонтальные отступы, а второй — вертикальные.

Три аргумента задают вначале отступ от верхнего края элемента, затем горизонтальные отступы, а в последнюю очередь — отступ снизу. Четыре значения поочередно определяют отступы со всех сторон элемента, начиная с верхнего края и продолжая по часовой стрелке.

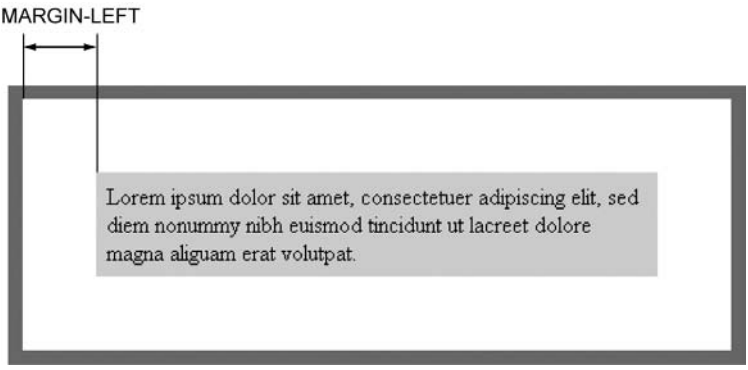


Рис. 9.1. Отступ от левого края элемента

В листинге 9.1 показано использование параметра `margin` с разными значениями.

Листинг 9.1. Применение свойства `margin`

```
<html>
<body>
<div style="width: 100%; margin: 15px; background: navy">
 ...
</div>
<table width=100% style="margin: 20px; background: #f90">
<tr>
 <td>...</td>
</tr>
</table>
<div style="width: 100%; margin: 20px 0px; background: olive">
 ...
</div>
</body>
</html>
```

В данном примере показано создание трех блочных элементов, ширина у них установлена в 100 %, а отступы заданы разными с помощью `margin`. На рис. 9.2 представлен результат отображения полученной веб-страницы.

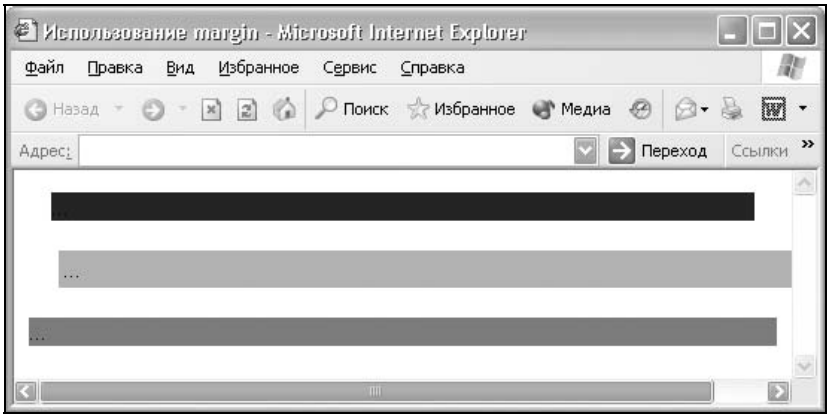


Рис. 9.2. Вид отступов для разных элементов

Как видно из рисунка, отступы изменяют вид разных элементов. В частности, на рис. 9.2 для таблицы, которая показана второй сверху, отступы увеличивают ее общую ширину, но поскольку ее ширина задана как 100 %, это приводит к появлению горизонтальной полосы прокрутки и выходу таблицы за видимые границы окна. Слои, построенные с помощью тега `DIV`, ведут себя более правильно в этом отношении, но все же неодинаково в разных браузерах. Так, браузер Netscape (Mozilla и Firefox) прибавляет к ширине слоя величину горизонтальных отступов, а Internet Explorer и Opera оставляют ширину неизменной. По этой причине нужно быть внимательным при добавлении отступов к элементам. Как устранить указанную особенность и заставить браузеры отображать код без горизонтальной полосы прокрутки, читайте далее в *разд. "Поля у блочных элементов"* данной главы.

Для задания отступов с разных сторон элемента существуют производные от свойства `margin` — `margin-left`, `margin-right`, `margin-top` и `margin-bottom`. Они управляют величиной отступа слева, справа, сверху и снизу элемента соответственно (листинг 9.2).

Листинг 9.2. Изменение отдельных отступов у элемента

```
<html>
<body>
<div style="margin-left: 30%; margin-top: 10%; width: 300px; height:
50px; background: #f90">
 ...
</div>
</body>
</html>
```

В данном примере показана установка отступов для слоя сверху и слева. Значения при этом задаются в процентах.

Отступы на веб-странице

Горизонтальные и вертикальные отступы от края окна браузера до содержимого веб-страницы встроены в браузер по умолчанию. Тем не менее можно изменить значение этих параметров, делая отступы больше или меньше по желанию. Так, например, можно установить отступ от верхнего края окна до заголовка страницы или вообще убрать его.

Отступы задаются одновременным использованием стилевых параметров `margin` и `padding`. Наличие двух атрибутов вместо одного обусловлено требованиями разных браузеров, `margin` — Internet Explorer и Netscape, а `padding` — Opera. Совмещение этих параметров гарантирует, что веб-страница будет одинаково отображаться в этих браузерах (листинг 9.3).

Листинг 9.3. Изменение отступов у веб-страницы

```
<html>
<head>
<style type="text/css">
 BODY {
  margin: 0px;    /* Отступ для браузеров Internet Explorer и Netscape */
  padding: 0px       /* Отступ для браузера Opera */
}
</style>
</head>
<body>
<table width=100% border=0 bgcolor=navy>
 <tr><td>
  ...
 </td></tr>
</table>
</body>
</html>
```

В данном примере таблица на веб-странице будет отображаться "вплотную", без промежутков между границей таблицы и окном браузера.

При желании отступы в окне браузера на отдельных сторонах можно добавить с помощью параметров `margin-top`, `margin-bottom`, `margin-right` и `margin-left`. Прежде чем использовать указанные атрибуты, следует задать нулевое значение параметрам `margin` и `padding` (листинг 9.4).

Листинг 9.4. Изменение верхнего отступа от края браузера

```
<html>
<head>
<style type="text/css">
 BODY {
  margin: 0px; padding: 0px;        /* Вначале убираем отступы */
  margin-top: 10px                  /* Затем задаем отступ сверху */
}
</style>
</head>
<body>
 ...
</body>
</html>
```

В этом примере показано добавление верхнего отступа для содержимого веб-страницы через свойство `margin-top`. Величина остальных отступов обнуляется.

Существует и другой, устаревший подход, основанный на использовании параметров тега `BODY`. Его хитрость заключается в том, что браузеры Internet Explorer и Netscape имеют разные параметры для указания значения отступов. Для браузера Internet Explorer в теге `BODY` следует указать параметр `leftmargin` для горизонтального отступа и `topmargin` — для вертикального, а в Netscape те же функции выполняют параметры `marginwidth` и `marginheight`. Объединяя все параметры воедино, получим универсальный код, который будет работать во всех браузерах одинаково (листинг 9.5).

Листинг 9.5. Изменение отступов у веб-страницы через параметры тега `BODY`

```
<html>
<body leftmargin=0 topmargin=0 marginwidth=0 marginheight=0>
 ...
</body>
</html>
```

Приведенный в данном примере метод считается устаревшим из-за активного распространения стилей и преимуществ, которые они в себе несут. В частности, стиль можно хранить в отдельном файле и только делать ссылку на него из веб-документов. В этом случае не придется каждый раз задавать множество параметров для одного тега `BODY`.

Отступы в форме

При использовании формы вокруг нее сверху и снизу автоматически добавляются отступы. Эта особенность не всегда является необходимой и подчас только вредит, особенно если форма располагается внутри таблицы. Чтобы в этом случае избавиться от отступов, сделайте наоборот — поместите таблицу внутрь формы (листинг 9.6).

Листинг 9.6. Таблицы и формы

```
<html>
<body>
 <form>
  <table>
   <tr><td>
    <input type=text size=10>
   </td></tr>
  </table>
 </form>
</body>
</html>
```

Представленное в данном примере решение является лишь частичным, все равно отступы в форме остаются, хотя и не явно. Правильный способ изменения отступов состоит в использовании стилей, надо добавить параметр `margin` с нулевым значением к тегу `FORM` (листинг 9.7).

Листинг 9.7. Использование стилей для установки отступов в форме

```
<html>
<body>
 <form style="margin: 0px">
  ...
 </form>
</body>
</html>
```

Использование свойства `margin` в данном случае позволяет удобно и универсально задать отступы в форме или убрать их совсем.

Создание полей

Полем мы будем называть расстояние от внутреннего края рамки элемента до воображаемого прямоугольника, ограничивающего его содержимое (рис. 9.3).

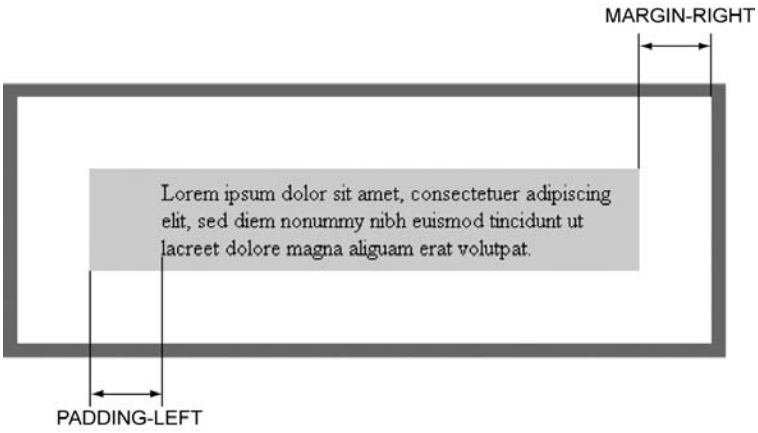


Рис. 9.3. Поле слева от текста

Разница между полями и отступами заключается в том, что отступы устанавливают пустое пространство вокруг элемента, а поля — внутри. Это продемонстрировано на рис. 9.3, где показан отступ от текстового блока до рамки родительского элемента и поле слева от текста до края границы фона под ним.

Для изменения указанной характеристики предназначен стилевой атрибут `padding`. Он позволяет задать поля для всех или выборочных сторон элемента. Этот параметр действует аналогично `margin`, поэтому итоговый результат зависит от числа аргументов. Для указания полей на разных сторонах элемента можно воспользоваться свойствами `padding-left`, `padding-right`, `padding-top` и `padding-bottom`, они управляют величиной полей слева, справа, сверху и снизу соответственно.

Основное предназначение полей — создать пустое пространство вокруг содержимого блочного элемента, например текста, чтобы он не прилегал плотно к границе элемента. Использование полей повышает читабельность текста и улучшает внешний вид страницы. В листинге 9.8 показано использование полей для оформления текста.

Листинг 9.8. Использование параметра `padding`

```
<html>
<body>
<div style="padding: 20px; width: 100%; background: #fc0; border:
                                                        1px solid black">
 ...
</div>
```

```
<div style="padding: 10px; padding-left: 50px; margin-top: 10px;
                                        width: 100%; background: #fc0">
 ...
</div>
</body>
</html>
```

В данном примере создается два слоя с разными характеристиками. В первом слое вокруг текста по вертикали и горизонтали задается одинаковое поле со значением 20 пикселов с помощью параметра `padding`. Во втором слое поле слева увеличено через свойство `padding-left`. Обратите внимание, что этот атрибут идет после описания `padding`, в данном случае менять их местами нельзя, поскольку `padding` устанавливает поля для всех сторон сразу. Результат примера показан на рис. 9.4.

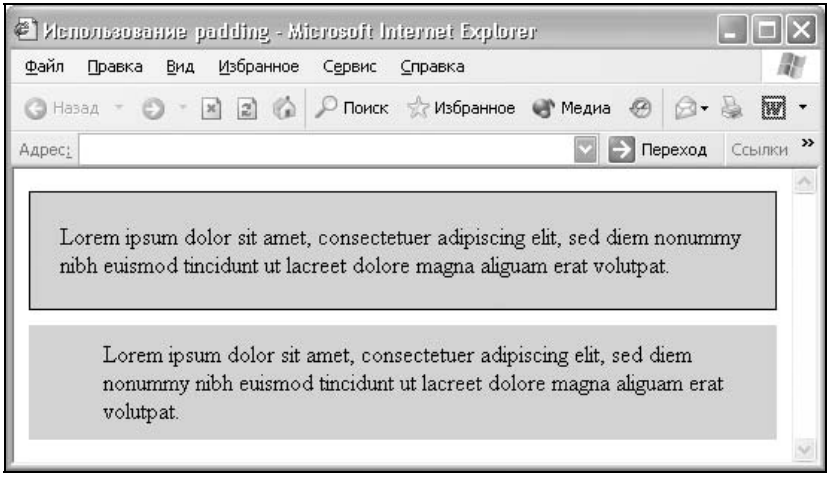


Рис. 9.4. Поля вокруг текста

Расстояние между слоями по вертикали определяется параметром `margin-top`, и здесь, наконец, нашлось применение отступам.

Поля у блочных элементов

Браузеры по-разному интерпретируют параметр `padding`, поэтому они неодинаково отображают результат на веб-странице. Чтобы понять особенности и различия браузеров при использовании полей, создадим прямоугольную область с помощью тега `DIV` и через `padding` установим у нее поля (листинг 9.9).

Листинг 9.9. Поля для тега `DIV`

```
<html>
<body>
<div style="padding: 10px; width: 100%; background: #c0c0c0">
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.</p>
</div>
</body>
</html>
```

Параметр `padding`, если у него указано только одно значение, устанавливает одинаковую величину поля со всех сторон. Однако, как показано на рис. 9.5, 9.6 и 9.7, результат действия данного примера в основных браузерах будет различаться.

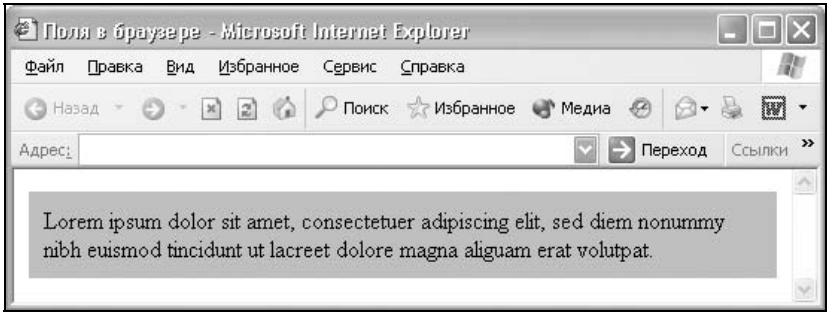


Рис. 9.5. Поля в браузере Internet Explorer 6

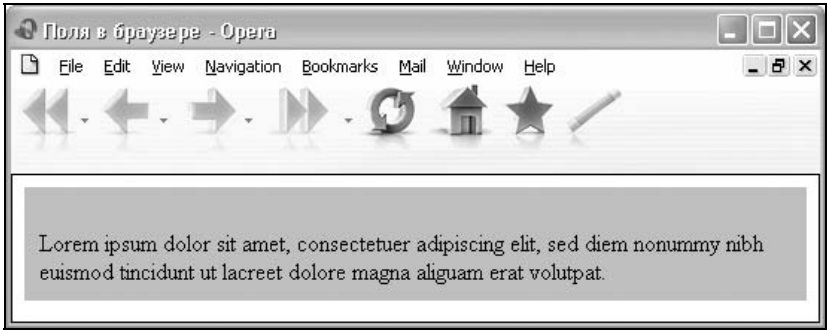


Рис. 9.6. Поля в браузере Opera 7

Internet Explorer, сохраняя заданную ширину блока неизменной, управляет величиной полей от края до текста, как, собственно, и должно быть. Брау-

зер Netscape (Firefox) увеличивает размер блока, добавляя к его ширине, заданной параметром `width`, еще и величину полей. Заметим, что схожий результат с Netscape показывает и браузер Opera 6, но, как видно на рис. 9.6, в последующих версиях подход разработчиков этого браузера был изменен. Вертикальные отступы тоже очень сильно разнятся, показывая совершенно противоречивые результаты.

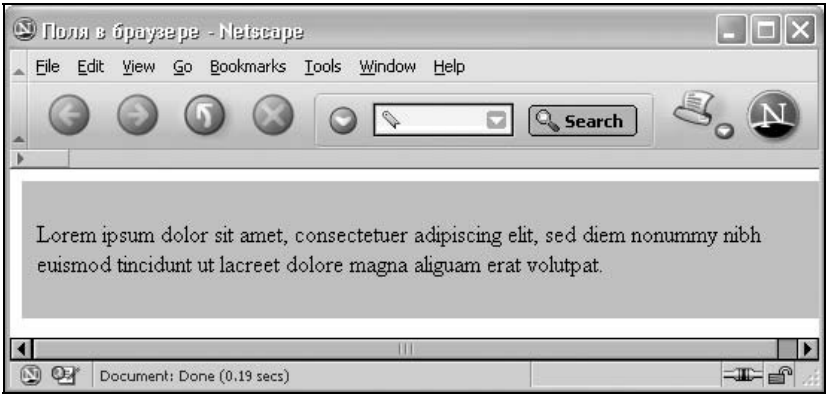


Рис. 9.7. Поля в браузере Netscape 7

Чтобы добиться корректного результата в разных браузерах, поля следует устанавливать не у блочного элемента, а у дочернего, который помещен внутри него. Для листинга 9.7 это будет тег параграфа `P`. Оставляя ширину и цвет фона для тега `DIV` неизменным, величину полей переносим в стиль параграфа (листинг 9.10).

Листинг 9.10. Поля, отображаемые одинаково в разных браузерах

```
<html>
<body>
<div style="width: 100%; background: #c0c0c0">
<p style="padding: 10px">Lorem ipsum dolor sit amet, consectetuer
adipiscing elit, sed diem nonummy nibh euismod tincidunt ut lacreet
dolore magna aliguam erat volutpat.</p>
</div>
</body>
</html>
```

Хотя данное решение не идеально, поскольку требует добавления еще одного блока внутрь уже существующего, оно дает нужный результат с наименьшими затратами.

Использование отступов и полей

Область применения отступов достаточно широка и не ограничена их прямым назначением. Например, как было показано в *главе 8*, отступы активно используются для выравнивания содержимого по центру веб-страницы. Кроме того, существуют и другие задачи, решить которые отступы и поля могут быстрее и эффективнее, чем "стандартные" методы типа использования таблиц.

Подытожим то, что уже было сказано по поводу применения полей и отступов, и перечислим области, где они находят непосредственное применение.

- Пустое пространство вокруг текста. Оно необходимо в том случае, когда текст находится внутри цветной области или помещается в рамку. В этом случае поля требуются, чтобы создать "воздух" вокруг содержимого, облегчающий чтение и восприятие информации, а также отодвинуть границу от текста.
- Смещение левой или правой границы элемента.
- Выравнивание элементов по центру области или окна браузера.
- Добавление горизонтальных или вертикальных промежутков между повторяющимися элементами, например абзацами текста.
- Размещение текста или других элементов по одному из краев слоя. В отличие от таблиц содержимое слоя нельзя напрямую выравнивать по его нижнему краю или по центру. Приходится использовать поля, чтобы указать, где в слое должен располагаться текст.
- Создание колонок — блоков текста, расположенных рядом друг с другом по горизонтали.

Основная особенность атрибутов `padding` и `margin` связана с тем, что они достаточно редко применяются самостоятельно. Наилучший эффект от их использования получается только совместно с другими свойствами стилей или элементами веб-страницы, где результат добавления отступов и полей становится заметен. Далее показаны некоторые примеры, в которых они играют ключевую роль.

Поля вокруг текста

Обычно поля для текстового блока указываются, когда вокруг него установлена рамка. В этом случае текст не соприкасается с границей и между ним и рамкой всегда остается дистанция. В то же время и саму рамку можно отодвинуть от края блока, установив между ними отступы (рис. 9.8).

Чтобы регулировать расстояние между текстом и рамкой, рамкой и краем блока, потребуется два слоя. Первый, `block1`, задает общий цвет фона и промежуток между линиями рамки и краем блока. Второй слой, `block2`,

задает поля вокруг текста, отделяя их, таким образом, от рамки (листинг 9.11).

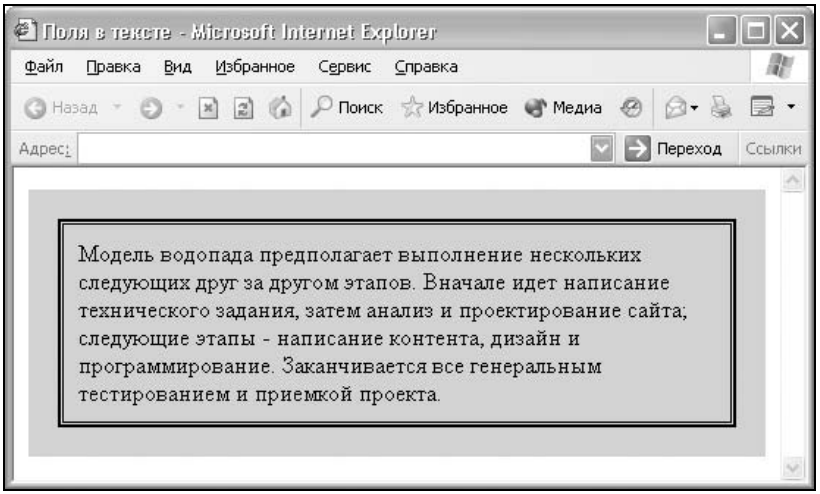


Рис. 9.8. Использование полей для управления рамкой

Листинг 9.11. Управление размерами полей от текста до рамки

```
<html>
<head>
<style type="text/css">
 .block1 {
  background: #fc0;          /* Цвет фона */
  padding: 20px              /* Расстояние от края до рамки */
}
 .block2 {
  padding: 10px;             /* Расстояние от рамки до текста */
  border: 4px double black   /* Параметры рамки */
}
</style>
<body>
 <div class=block1>
  <div class=block2>
    Содержимое
  </div>
 </div>
</body>
</html>
```

Для изменения расстояния от края до рамки теоретически можно использовать атрибут `margin` для класса `block2` вместо `padding` в классе `block1`. Несмотря на то, что логически это должно привести к одному результату, во всех браузерах цвет фона не отображается сверху и снизу рамки. Браузеры по какой-то странной причине непременно хотят видеть поля для внешнего слоя, поэтому, добавив свойство `padding` со значением 1 пиксел, мы получим желаемый вид. В листинге 9.12 показан стиль, который следует задавать с учетом поведения браузеров. Остальной код останется без изменения и приведен в листинге 9.11.

Листинг 9.12. Использование отступов вокруг текста

```
<style type="text/css">
 .block1 {
  background: #fc0;          /* Цвет фона */
  padding: 1px               /* Поля должны быть обязательно! */
}
 .block2 {
  margin: 20px;              /* Расстояние от рамки до края блока */
  padding: 10px;             /* Расстояние от рамки до текста */
  border: 4px double black   /* Параметры рамки */
}
</style>
```

Таким образом, для управления расстоянием между вложенными слоями лучше всего указывать поля, а при использовании отступов следует принимать во внимание особенности браузеров и проверять в них полученный результат.

Отступы в тексте

Следует понимать, что для некоторых текстовых элементов вроде заголовков `H1`... `H6` и параграфов `P` отступы, в частности вертикальные, добавляются автоматически. В HTML это предусмотрено, чтобы создать отбивку между абзацами или заголовком и основным текстом. Подобные настройки по умолчанию не всегда удачны, поэтому их можно поменять, задавая величину требуемых отступов. Рассмотрим добавление цитат. В HTML для этой цели предусмотрен специальный тег `BLOCKQUOTE`. Текст, обозначенный этим тегом, традиционно отображается как выровненный блок с отступами слева и справа (примерно по 40 пикселов), а также с отбивкой сверху и снизу (рис. 9.9).

Линии сверху и снизу добавлены отдельно, через тег `HR`, чтобы были отчетливо видны отступы вокруг текста. Изменить их величину можно через

margin, задавая величину этого параметра для селектора BLOCKQUOTE. Чтобы цитата выделялась на фоне остального текста, добавим также линию сверху и снизу (листинг 9.13).

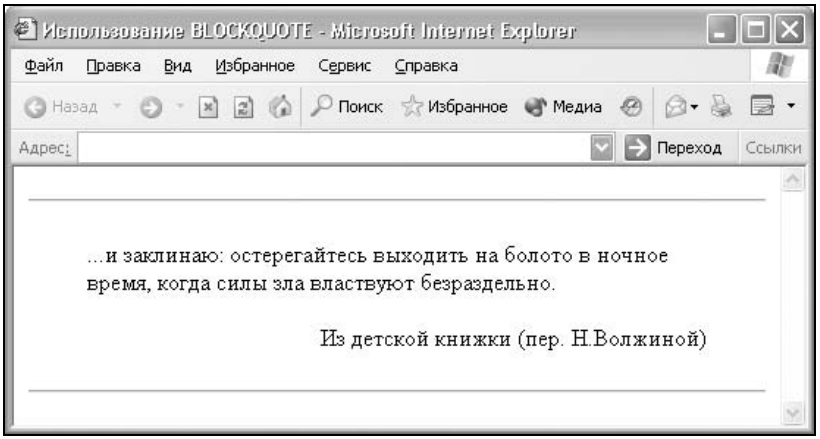


Рис. 9.9. Вид текста, оформленного с помощью тега BLOCKQUOTE

Листинг 9.13. Оформление цитат с помощью отступов

```
<html>
<head>
<style type="text/css">
 BLOCKQUOTE {
  margin:
       10px                          /* Отступ сверху и снизу от линии */
       40px;                         /* Отступ справа и слева от линии */
  padding:
       5px            /* Вертикальное расстояние от цитаты до линии */
       0px;           /* Горизонтальное расстояние от цитаты до линии */
  font-style: italic;                /* Курсивное начертание */
  border-top: 2px solid crimson;     /* Параметры линии сверху от цитаты */
  border-bottom: 2px solid crimson  /* Параметры линии снизу */
}
</style>
<body>
<blockquote>
...и заклинаю: остерегайтесь выходить на болото в ночное время, когда
силы зла властвуют безраздельно.
<p align=right>Из детской книжки (пер. Н.Волжиной)
```

```
</blockquote>
</body>
</html>
```

Для управления положением цитаты и линий сверху и снизу от нее используется аргумент `margin`. Он одновременно "сжимает" блок текста по горизонтали, используя отступ справа и слева, и указывает расстояние от линии до основного текста. Чтобы горизонтальная линия не соприкасалась слишком тесно с текстом цитаты, вводится свойство `padding`, которое регулирует дистанцию между ними (рис. 9.10).

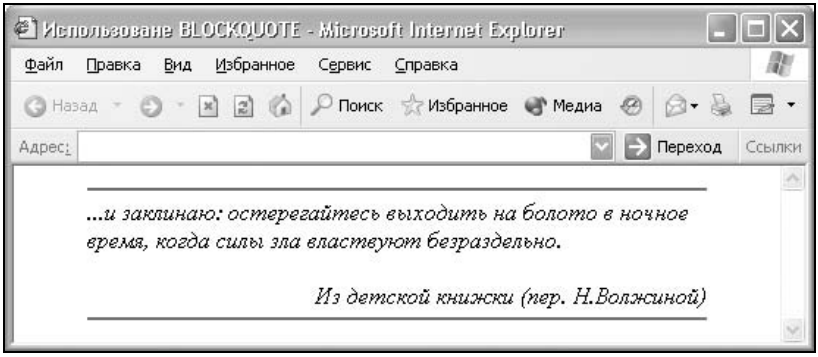


Рис. 9.10. Вид текста цитаты, оформленной через стили

Аналогичным образом отступы можно добавлять или, наоборот, убирать и у других блочных текстовых элементов.

Создание двух колонок

Арсенал средств для создания колонок с помощью стилей достаточно обширен. Сюда входит применение плавающих элементов, абсолютного и относительного позиционирования, а также отступов. Каждый способ имеет свои особенности и сферу применения, поэтому выделить какой-то один из них проблематично. Тем не менее колонки, созданные с использованием отступов, имеют следующие преимущества:

- ширина колонок может быть задана как в пикселах, так и в процентах, таким образом занимая всю ширину веб-страницы или лишь указанную область;
- при изменении размеров окна браузера верстка не "разваливается", как это часто происходит с колонками, установленными с помощью свойства `float`;
- колонки можно задать равной высоты несмотря на объем их содержимого.

Для создания двух колонок понадобится три слоя. Первый, с именем `column`, выступает в роли основного контейнера, он задает общую ширину макета и цвет фона у левой колонки. Ширина определяется атрибутом `width` и может быть указана как в пикселах, так и в процентах. Следующий слой, `col1`, определяет параметры левой колонки, в частности цвет текста и ширину, которая также может быть задана в любых доступных для CSS единицах. Чтобы колонки располагались по горизонтали, для этого слоя следует установить атрибут `float: left`. Элементы `DIV` являются блочными и по умолчанию располагаются вертикально друг под другом. Использование `float`, как показано в листинге 9.14, обеспечивает их положение по горизонтали.

Последний слой, с именем `col2`, задает параметры оформления правой колонки — цвет фона и текста. Кроме того, здесь обязательно следует привести параметр `margin-left`, задающий смещение левой границы элемента вправо. Значение у него должно быть таким же, как и ширина слоя `col1`. Иными словами, если ширина левой колонки установлена в процентах, также в процентах следует задавать отступ. Это условие касается и других единиц. Ширину для этой колонки указывать не нужно, она будет вычислена автоматически исходя из уже заданных размеров.

Листинг 9.14. Создание двух колонок с помощью `margin-left`

```
<html>
<head>
<style type="text/css">
 .column {
  background: #800000;        /* Цвет фона левой колонки */
  width: 100%                 /* Суммарная ширина всех колонок */
}
 .col1 {
  width: 200px;               /* Ширина левой колонки */
  float: left;                /* Обтекание с правого края */
  color: white                /* Цвет текста в левой колонке */
}
 .col2 {
  background: #e0e0e0;        /* Цвет фона правой колонки */
  margin-left: 200px          /* Сдвиг правого края на ширину
                              первой колонки */
}
 P {
  padding: 7px;               /* Поля вокруг текста */
  margin: 0px                 /* Отменяем отступы у параграфа */
}
```

```
</style>
</head>
<body>
 <div class=column>
  <div class=col1>
    <p>Содержимое левой колонки</p>
  </div>
  <div class=col2>
    <p>Содержимое правой колонки</p>
  </div>
 </div>
</body>
</html>
```

Пример колонок, созданных с использованием отступа слева, показан на рис. 9.11.

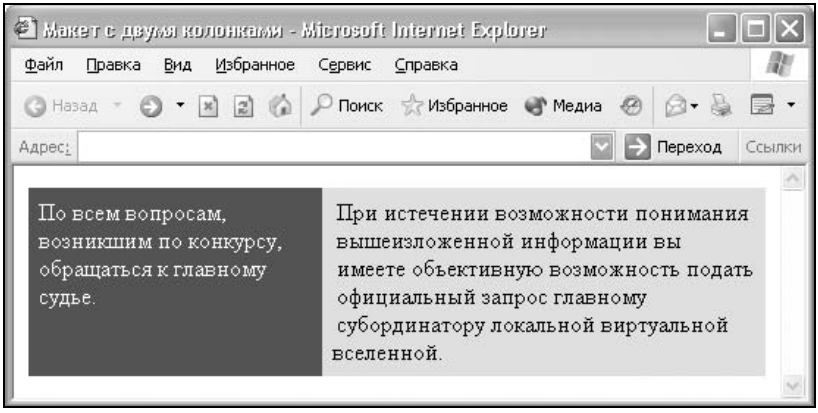


Рис. 9.11. Макет с двумя колонками

При создании колонок указанным методом нужно принять во внимание следующие замечания:

- добавление полей (параметр `padding`) непосредственно к слоям может привести к ненужному выходу текста за пределы колонок в браузере Firefox. Чтобы этого не произошло, поля устанавливаются для параграфа (тег `P`), который располагается внутри каждого слоя;
- из-за того, что сверху и снизу от параграфа автоматически добавляются отступы, в Internet Explorer внизу правой колонки появляется горизонтальная полоса, а в браузере Opera весь макет незначительно смещается вниз. Для устранения этих особенностей и предназначено свойство `margin: 0px`, которое применяется к селектору `P`;

- в браузере Internet Explorer 6 текст в правой колонке связан с содержимым левой колонки и по какой-то странной причине немного сдвигается вправо. Обратите внимание, что на рис. 9.11 последнее слово "вселенной" находится в правильной позиции, как она задана полями, а остальной текст располагается несколько правее;
- к сожалению, приведенный способ имеет один недостаток, который не позволяет использовать его как универсальный метод. Колонки будут отображаться корректно и одинаковой высоты только в том случае, если содержимое правой колонки (слой `col2`) превышает по объему содержимое левой колонки (слой `col1`). Другими словами, правая колонка всегда должна быть больше по высоте.

Когда высота левой колонки превышает высоту правой, браузеры поразному реагируют на этот факт. Internet Explorer и Opera увеличивают высоту слоя, что приводит к появлению полосы внизу правой колонки (рис. 9.12, *а*). Браузеры Netscape и Firefox не превышают высоту, но зато отображают текст поверх фона (рис. 9.12, *б*).

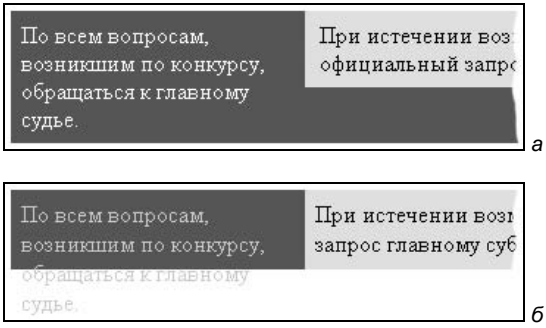


Рис. 9.12. Реакция браузеров на превышение заданной высоты слоя: *а* — Internet Explorer и Opera; *б* — Netscape и Firefox

Из-за указанной особенности при верстке слоями колонки одинаковой высоты применяются редко. Как вариант — можно не указывать цвет фона для левой колонки, тогда превышение ее содержимым высоты будет не так заметно.

ГЛАВА 10

Элементы оформления

Различного рода "украшательства" сайта свойственны скорее разработчикам-новичкам, открывшим для себя новые возможности и старающимся воплотить полученные знания на практике. Понятно, что можно легко обойтись без многих элементов дизайна, сведя все содержимое к простому тексту с иллюстрациями. Такой подход, именуемый *академический дизайн*, характеризуется отсутствием излишеств и даже аскетизмом. Хотя он имеет право на существование, но он постепенно уходит в прошлое. Ведь любой дизайн, в первую очередь, предназначен не столько для оформления документа, сколько для лучшего донесения информации до читателя. В этом смысле многие возможности HTML и CSS по оформлению различных элементов, если их правильно применить, помогают достичь данной цели.

Здесь мы не будем рассматривать целесообразность применения того или иного приема. В любом случае решать этот вопрос предстоит разработчику исходя из направленности сайта, его аудитории, удобства и собственных предпочтений.

Полосы прокрутки

Полосы прокрутки позволяют просматривать содержимое блочного элемента, если оно целиком в него не помещается и выходит за область заданных размеров. Это особенно удобно, когда по дизайну веб-страницы требуется использовать блок текста фиксированной высоты или ширины.

Чтобы добавить полосы прокрутки, используется свойство `overflow`, которое принимает одно из четырех значений:

- `visible` — отображается все содержимое элемента. Браузеры Netscape и Firefox оставляют размеры блока неизменными, но его содержимое при указании этого параметра выходит за пределы области. Остальные брау-

зеры в этом случае игнорируют заданные размеры и отображают информацию, словно никакого параметра `overflow` нет;

- `hidden` — отображается только область внутри элемента, остальное будет обрезано;
- `scroll` — всегда добавляются полосы прокрутки, независимо от размеров содержимого элемента;
- `auto` — полосы прокрутки добавляются только при необходимости.

При отображении полос прокрутки обязательно следует указывать размер блока с помощью атрибутов `width` (ширина) или `height` (высота). Приведенный далее пример демонстрирует создание текстового блока с полосой прокрутки (листинг 10.1).

Листинг 10.1. Полосы прокрутки для отдельного элемента

```
<html>
<head>
<style type="text/css">
 .hero {
  overflow: scroll;          /* Показываем полосы прокрутки */
  width: 400px;              /* Ширина блока */
  height: 150px;             /* Высота блока */
  border: solid 1px black    /* Параметры рамки */
}
</style>
</head>
<body>
 <div class=hero>
  <h2>Герои</h2>
  <p>Чебурашка</p>
  <p>Крокодил Гена</p>
  <p>Шапокляк</p>
  <p>Крыса Лариса</p>
 </div>
</body>
</html>
```

В данном примере текст помещается в контейнер `DIV`, для которого задается ширина и высота с помощью стилей, а также всегда отображаются полосы прокрутки. Результат примера показан на рис. 10.1.

Обратите внимание, что горизонтальная полоса прокрутки отображается всегда, хотя и является неактивной в данном случае. Браузер Internet Explo-

rer позволяет управлять отдельно вертикальной и горизонтальной полосой прокрутки с помощью свойств `overflow-y` и `overflow-x` соответственно. Так, можно запретить показ горизонтальной прокрутки (скроллинга), применяя `overflow-x: hidden` к селектору `BODY`, как показано в листинге 10.2.

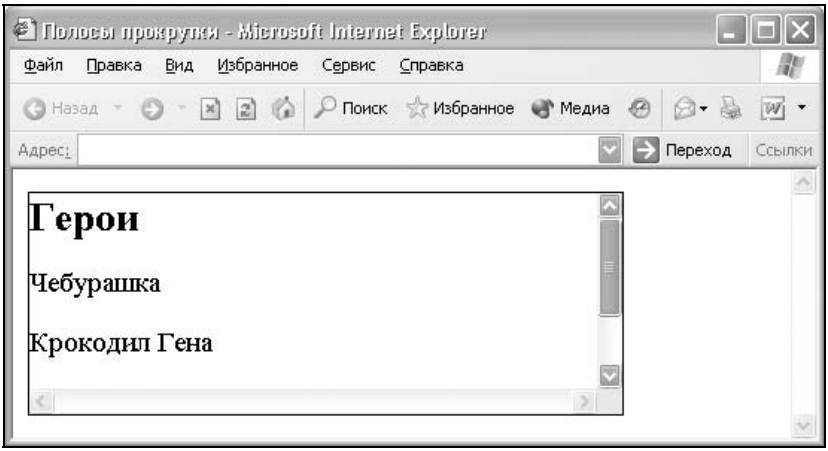


Рис. 10.1. Полосы прокрутки для блока текста

Листинг 10.2. Запрет горизонтальной полосы прокрутки

```
<html>
<head>
<style type="text/css">
BODY { overflow-x: hidden }
</style>
</head>
<body>
...
</body>
</html>
```

Заметьте, что данный пример будет работать только в одном браузере, Internet Explorer, остальные браузеры покажут полосы прокрутки как обычно.

Замечание

Перед тем как убрать полосы прокрутки с веб-страницы, подумайте, действительно ли вам это так необходимо. Отсутствие возможности прокрутки содержимого веб-страницы создает трудности посетителям сайта при просмотре информации.

Если вы попали в ситуацию, когда полос прокрутки по какой-либо причине на экране нет, а информация тем не менее есть, но при этом на экран не помещается, то прокрутить веб-страницу можно следующим способом. Нажимаем кнопку мыши, когда ее курсор находится в любом месте веб-страницы, и, не отпуская, перемещаемся вниз. Так происходит выделение содержимого и одновременно его прокрутка. Но этот метод, мягко говоря, не очень удобен, и рекомендовать его можно лишь в крайних случаях.

Полосы прокрутки во фреймах

Для управления отображением полос прокрутки во фреймах используется параметр `scrolling` тега `FRAME`. Он может принимать два основных значения: `yes` — всегда вызывает появление полос прокрутки независимо от объема информации и `no` — запрещает их появление (листинг 10.3).

Листинг 10.3. Запрет полосы прокрутки во фреймах

```
<html>
<frameset cols=200,*>
 <frame src=menu.html name=MENU noresize scrolling=no>
 <frame src=content.html name=CONTENT>
</frameset>
</html>
```

Как показано в этом примере, в левом фрейме с именем `MENU` не будет полосы прокрутки. В соседнем с ним фрейме, хотя параметр `scrolling` и не указан, полосы прокрутки будут видны, т. к. эта возможность установлена по умолчанию.

Новые окна

Чтобы удалить полосы прокрутки из новых окон, возможностей HTML будет недостаточно. Универсальный подход требует использования языка JavaScript для создания нового окна, при этом применяется метод `open`, который имеет следующий синтаксис:

```
window.open("URL", "имя окна", "параметры")
```

URL представляет собой путь к HTML-документу, который следует открыть в новом окне. Имя представляет собой условное название открываемого окна, его можно опустить. Параметры для управления видом окна перечислены в табл. 10.1.

Таблица 10.1. *Отображение элементов окна браузера*

Параметр	Значение	Описание
directories	yes | no или 1 | 0	Отображает панель ссылок
location	yes | no или 1 | 0	Отображает адресную строку
menubar	yes | no или 1 | 0	Отображает меню
resizable	yes | no или 1 | 0	Определяет, может ли пользователь самостоятельно изменять размеры окна
scrollbars	yes | no или 1 | 0	Показывает полосу прокрутки
status	yes | no или 1 | 0	Отображает строку состояния
toolbar	yes | no или 1 | 0	Показывает кнопки на панели инструментов
left	пикселы	Задает положение левого верхнего угла по горизонтали
top	пикселы	Устанавливает положение левого верхнего угла по вертикали
height	пикселы	Высота окна
width	пикселы	Ширина окна

Перечисленные параметры могут идти в любом порядке через запятую. Размер окна не должен быть меньше, чем 100 × 100 пикселов, и не допускается размещать его за пределами экрана. Включение или выключение параметра можно определять через указание значения yes или no (1 или 0). Например, center=yes идентично center=1.

Из этого списка нас интересует параметр scrollbars, именно он убирает полосы прокрутки (листинг 10.4).

Листинг 10.4. Создание нового окна без полос прокрутки

```
<html>
<head>
<script language="JavaScript">
function tipsWindow() {
 window.open("tips.html", "TIP", "width=400, height=300, status=0,
menubar=0, location=0, resizable=0, directories=0, toolbar=0,
scrollbar=0");
}
</script>
</head>
```

```
<body onLoad="tipsWindow()">
 ...
</body>
</html>
```

В данном примере показано создание нового окна с размерами 400×300 пикселов без различных элементов навигации, в том числе без полос прокрутки. Для выполнения нашей функции `tipsWindow()` используется событие `onLoad`, которое наступает при загрузке текущего документа. В новом окне открывается файл с именем tips.html.

Изменение цвета полос прокрутки

Браузер Internet Explorer позволяет видоизменять цвет полос прокрутки или отдельных элементов на веб-странице. Для этой цели введено несколько параметров CSS, которые управляют цветом составляющих элементов полосы прокрутки:

- `scrollbar-face-color` — цвет ползунка и кнопок;
- `scrollbar-track-color` — цвет дорожки ползунка;
- `scrollbar-darkshadow-color` — цвет нижней границы тени ползунка и кнопок;
- `scrollbar-shadow-color` — цвет тени ползунка и кнопок;
- `scrollbar-highlight-color` — цвет подсветки, создающий эффект объемности;
- `scrollbar-base-color` — цвет основных элементов ползунка: самого ползунка, кнопок со стрелками, дорожки для ползунка;
- `scrollbar-3dlight-color` — цвет верхней и левой части ползунка и кнопок;
- `scrollbar-arrow-color` — цвет стрелок на кнопках.

В листинге 10.5 показано, как изменить цвет полосы прокрутки для всей страницы целиком и для текстового поля формы.

Листинг 10.5. Цветные полосы прокрутки

```
<html>
<head>
<style type="text/css">
 BODY {
  background: #053f38;              /* Цвет фона страницы */
  color: #ffd595;                   /* Цвет текста */
```

```
  scrollbar-face-color: #053f38;     /* Цвет полосы прокрутки */
  scrollbar-arrow-color: white       /* Цвет стрелок */
}
 TEXTAREA {
  scrollbar-arrow-color: orange;     /* Цвет стрелок у текстового поля */
  height: 100px
}
</style>
</head>
<body>
<form action=/cgi-bin/handler.cgi>
<textarea>

</textarea>
<p><input type=submit value="Отправить">
</form>
</body>
</html>
```

В данном примере показана установка цвета полосы прокрутки аналогично фоновому цвету веб-страницы. Это визуально расширяет рабочую область окна браузера и вписывает полосы прокрутки в дизайн сайта.

Поскольку текстовое поле является дочерним элементом тега BODY, то тег TEXTAREA наследует его свойства полос прокрутки. Поэтому повторять параметры для изменения цвета полос не обязательно за исключением случая, когда вы хотите переопределить некоторые параметры. Так, в данном примере цвет стрелок у текстового поля меняется с белого, который установлен для всей страницы целиком, на оранжевый.

При изменении цвета указанным способом меняется и вид ползунка и кнопок. На рис. 10.2 представлен вид полос прокрутки в браузере Internet Explorer 6 под Windows, установленный по умолчанию (рис. 10.2, *а*), и в случае добавления стиля (рис. 10.2, *б*).

Цвет и вид полос прокрутки может быть модифицирован не только для элементов форм или всей страницы целиком, но и для блока текста, показанного на рис. 10.1. Для этого случая код страницы приведен в листинге 10.6.

Листинг 10.6. Изменение цвета полос прокрутки для блока текста

```
<html>
<head>
<style type="text/css">
```

```
 .hero {
  overflow: scroll;                  /* Показываем полосы прокрутки */
  overflow-x: hidden;                /* Скрываем горизонтальную полосу */
  padding: 7px;                      /* Поля вокруг содержимого */
  width: 400px;                      /* Ширина блока */
  height: 150px;                     /* Высота блока */
  border: solid 1px black;           /* Параметры рамки */
  scrollbar-face-color: green;       /* Цвет полосы прокрутки */
  scrollbar-arrow-color: white       /* Цвет стрелок */
}
</style>
</head>
<body>
 <div class=hero>
 ...
 </div>
</body>
</html>
```

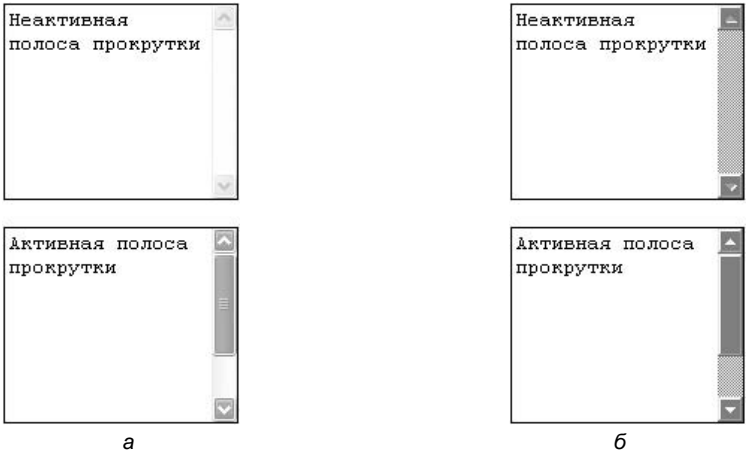


Рис. 10.2. Вид полос прокрутки до и после изменения их цвета: *а* — по умолчанию; *б* — при добавлении стиля

Большинство браузеров проигнорирует некоторые параметры в данном примере, вроде установки нового цвета полос прокрутки и скрытия горизонтальной полосы, поскольку они предназначены только для браузера Internet Explorer. В связи с этим рекомендуется протестировать сайт в разных браузерах на предмет выявления несовпадений. Страница не обязательно должна выглядеть в них идентично, главное добиться отсутствия ошибок и недочетов.

Вид курсора мыши

С помощью стилей можно переопределить вид курсора мыши и выбрать один из пятнадцати доступных вариантов. Однако такая возможность поддерживается не всеми браузерами: иногда мы увидим обычный курсор, словно его вид не был изменен.

Прежде чем изменить вид курсора, решите, будет ли он использоваться к месту. Многих пользователей подобные перемены могут ввести в заблуждение, когда, например, вместо традиционной "руки", появляющейся при наведении на ссылку, возникает нечто другое. В большинстве случаев лучше оставить вид курсора по умолчанию.

В табл. 10.2 приведены возможные типы курсоров мыши, которые заложены в стили.

Таблица 10.2. *Вид курсора и его значение*

Вид	Значение	Вид	Значение
	default		ne-resize
	crosshair		e-resize
	pointer		se-resize
	move		s-resize
	text		sw-resize
	wait		w-resize
	help		nw-resize
	n-resize		

Учтите, что вид курсора в браузере может несколько отличаться от приведенных в табл. 10.2 курсоров. Это связано с выбором операционной системы и ее настройками.

Синтаксис изменения типа курсора очень прост. Следует определить селектор стиля и в нем использовать параметр `cursor` с одним из значений, описанных в таблице. В листинге 10.7 показано, как можно переопределить вид курсора при наведении его на разные ссылки.

Листинг 10.7. Изменение курсора мыши при наведении его на ссылку

```
<html>
<head>
<style type="text/css">
 A.movelink { cursor: move }
 A.helplink { cursor: help }
</style>
</head>
<body>
 <p><a href=link1.html class=movelink>ПЕРЕМЕСТИТЕ ЭТОТ ТЕКСТ</a>
 <p><a href=link2.html class=helplink>СПРАВКА</a>
</body>
</html>
```

В данном примере показано создание двух классов. Первый класс, с именем `movelink`, превращает курсор мыши в стрелку для перемещения при наведении курсора на первую ссылку. Как только курсор будет выведен за пределы этой ссылки, он примет свой первоначальный вид. Второй класс, с именем `helplink`, предназначен для изменения вида курсора мыши при наведении в область ссылки с надписью "Справка". В этом случае курсор примет вид стрелки со знаком вопроса.

Если вы желаете переопределить курсор мыши для всей веб-страницы целиком, а не только для ссылок, воспользуйтесь селектором `BODY`, как показано в листинге 10.8.

Листинг 10.8. Изменение вида курсора мыши для всей веб-страницы

```
<html>
<head>
<style type="text/css">
 BODY { cursor: crosshair }
</style>
</head>
<body>
 ...
</body>
</html>
```

В этом примере показано, как задать курсор в виде перекрестья, которое будет отображаться в пределах всей страницы за исключением ссылок, где курсор будет принимать традиционный вид.

Также можно задать разный вид курсора для отдельных областей веб-страницы, используя теги DIV или SPAN. В этом случае вначале определяется класс и его стиль, а затем он применяется к тегу, например SPAN. Такой подход позволяет описать стиль один-единственный раз, а затем применять его в любом нужном месте (листинг 10.9).

Листинг 10.9. Изменение вида курсора мыши для разных областей веб-страницы

```
<html>
<head>
<style type="text/css">
 .cross { cursor: crosshair }
</style>
</head>
<body>
 <div class=cross>На этом тексте курсор мыши примет вид
перекрестья.</div>
</body>
</html>
```

В пределах блочного элемента DIV курсор в данном примере будет изменять свой вид и принимать первоначальный вид при выходе за пределы текстового блока.

Только браузер Internet Explorer позволяет установить курсор оригинального вида. Предварительно требуется подготовить файл в формате CUR для статичных курсоров или ANI для анимированных (набор готовых курсоров для операционной системы Windows можно найти в папке Windows\Cursors). После этого файл подключается с помощью параметра cursor с атрибутом url(), где в скобках указывается путь к файлу с курсором (листинг 10.10).

Листинг 10.10. Установка собственного курсора мыши на веб-странице

```
<html>
<head>
<style type="text/css">
 BODY { cursor: url(dinosaur.ani) }
</style>
</head>
<body>
 ...
</body>
</html>
```

На рис. 10.3 показан пример страницы с курсором, установленным через стили.

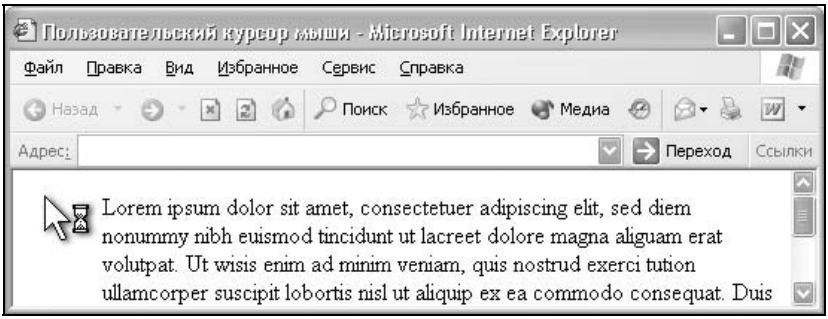


Рис. 10.3. Документ с пользовательским курсором мыши

Учтите, что подобные курсоры отображаются не во всех браузерах, к тому же пользователю приходится ждать загрузки дополнительного файла. Хотя, справедливости ради, следует отметить, что объем файлов с курсором достаточно небольшой. Хуже другое — пользователи привыкают к стандартному виду курсора, они не сразу могут понять, что означает новая форма курсора, и привыкнуть к ней. С другой стороны, удачное использование курсора может оживить сайт и придать ему нужную направленность и стилистику. В любом случае, использовать собственные курсоры или нет, является прерогативой разработчиков.

Ссылка на иконку сайта

Когда веб-страница сохраняется в разделе браузера **Избранное**, возле имени сайта отображается также и небольшой рисунок, часто называемый *иконкой*. Браузер добавляет свои собственные встроенные иконки размером 16 × 16 пикселов. Тем не менее можно использовать оригинальный рисунок, тогда при сохранении адреса сайта в **Избранном** будет отображаться индивидуальный логотип. Для этого требуется создать свою собственную иконку размером 16 × 16 пикселов и сохранить ее в формате ICO в корне сайта под именем favicon.ico. После этого на каждой странице в заголовке документа следует добавить строку, приведенную в листинге 10.11.

Листинг 10.11. Ссылка на персональную иконку сайта

```
<html>
<head>
 <link rel="shortcut icon" href="http://www.mysite.ru/favicon.ico">
</head>
```

```
<body>
...
</body>
</html>
```

Не все браузеры поддерживают подобную возможность. Если браузер "не признает" эту строку, он ее просто игнорирует и продолжает анализировать код дальше.

Замечание

Можно пойти на хитрость и сохранить изображение в формате BMP, который поддерживается почти всеми графическими редакторами в отличие от ICO. Несмотря на другой формат, файл все равно должен называться favicon.ico.

Создание тени

Добавление тени к объекту, будь это текст или рамка, — весьма простой и эффективный способ выделить объект на странице и привлечь к нему внимание читателя. Главное, следует побеспокоиться, чтобы все тени от разных элементов были направлены в одну сторону. Хотя это и не всегда бывает заметно на первый взгляд, подсознательно человек выделяет такое несоответствие и оценивает его негативно.

Для создания тени от объекта существует несколько методов:

- *графический*, когда текст и тень к нему создаются в графическом редакторе, а затем на веб-страницу вставляются как изображение через тег `IMG`. Это наиболее популярный способ для добавления в документ самых разнообразных эффектов, которые невозможно получить средствами HTML или CSS;
- *с помощью фильтров*. Этот путь поддерживается только одним браузером Internet Explorer, поэтому универсальным его считать нельзя;
- *через стилевое свойство* `text-shadow`. Хотя этот параметр и введен в спецификацию HTML, ни один браузер его в настоящее время не поддерживает. Про это решение можно было бы и не упоминать, но у многих разработчиков возникает вопрос, почему при наличии этого свойства никто его не использует. Вот потому и не используют, что результата никакого не видно;
- *с помощью слоев*. Для текста или блочного элемента можно создать его дубликат и подложить под источник с небольшим сдвигом. Цвет в этом случае подбирается другой, более светлый, чтобы создать эффект тени. Этот способ наиболее универсален и поддерживается всеми браузерами. Единственная его слабая сторона — недостаточная гибкость: приходится

редактировать текст в двух местах, за счет чего повышается дополнительный объем информации.

Не рассматривая здесь создание изображения в графическом редакторе, остановимся на двух способах добавления тени — через фильтр и с помощью слоев.

Использование фильтров

Фильтры представляют собой инструмент для определенных визуальных манипуляций с объектами веб-страницы. Фильтры могут применяться не ко всем элементам, в частности не даст результата добавление фильтров к следующим тегам: `APPLET`, `EM`, `H1...H6`, `OPTION`, `P`, `SELECT` и некоторым другим. Всегда можно использовать фильтры с изображениями, таблицами и тегами `DIV` и `SPAN`, но только в том случае, если у них установлена ширина или высота с помощью параметров `width` и `height`.

Добавление тени происходит с помощью фильтра `dropShadow`, который имеет следующий синтаксис:

```
filter: dropShadow(color=цвет тени, offX=смещение по вертикали,
offY=смещение по горизонтали)
```

Цвет следует указывать в шестнадцатеричном значении (например `#c0c0c0`) или по имени (`silver`), параметр `offX` определяет, на сколько пикселов следует сдвинуть тень по горизонтали относительно базового объекта. Положительное значение смещает тень вправо, а отрицательное — влево. Аналогично работает и параметр `offY`, но только в вертикальном направлении. Здесь положительное число указывает смещение вниз, а отрицательное — вверх.

Хотя фильтр нельзя напрямую применять к тексту, можно поместить его в контейнер `DIV` и установить для него стиль, как показано в листинге 10.12.

Листинг 10.12. Добавление тени к тексту с помощью фильтра

```
<html>
<head>
<style type="text/css">
 DIV.shadow {
  width: 90%;
  filter: dropShadow(color=#cccccc, offX=5, offY=5)
}
</style>
</head>
<body>
<div class=shadow>
```

```
 <h1>Чебурашка</h1>
 Имя произошло от слова "чебурахнулся", что в разговорном просторечном
выражении означает падение с небольшой высоты со стуком, обычно ногами
кверху.
</div>
</body>
</html>
```

Результат использования данного фильтра продемонстрирован на рис. 10.4, тень установлена для заголовка и текста под ним.

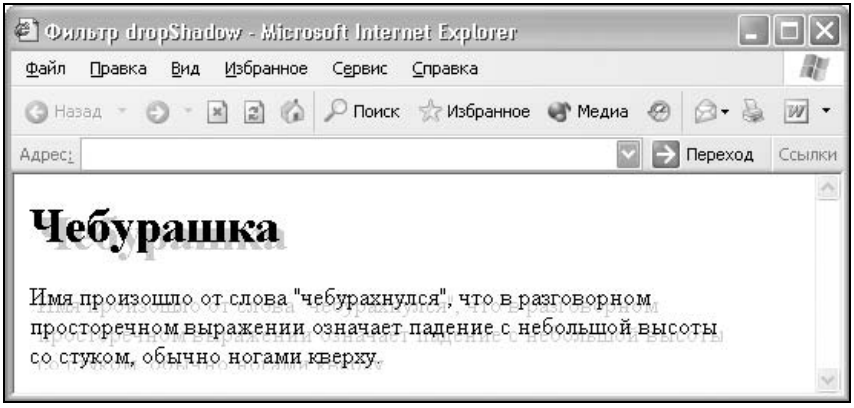


Рис. 10.4. Тень от текста, добавленная с помощью фильтра

Само название фильтра и его параметров не чувствительно к регистру, поэтому допустимо использовать как прописные, так и строчные символы или смешивать их, как показано в данном примере.

Применение слоев

Использование слоев позволяет создать универсальный код для добавления тени от текста или другого объекта, которая будет отображаться в других браузерах, помимо Internet Explorer. Нельзя сказать, что этот подход единственно правильный из-за некоторых ограничений и особенностей, о которых будет сказано далее. Тем не менее для каких-то целей указанный метод вполне пригодится.

Итак, чтобы создать тень, в код документа необходимо добавить контейнер `DIV`, который содержит копию объекта, и установить у него относительное позиционирование через свойство `position`. Параметры `left` и `top` управляют положением тени по горизонтали и по вертикали, а `color` задает ее цвет. Чтобы тень по уровню находилась ниже исходного объекта, следует вос-

пользоваться параметром z-index, который определяет порядок наложения слоев. Чем больше значение аргумента, тем выше слой находится относительно других элементов. Отрицательное значение, наоборот, указывает, что слой должен располагаться ниже всех объектов веб-страницы (листинг 10.13).

Листинг 10.13. Добавление тени к тексту с помощью слоя

```
<html>
<head>
<style type="text/css">
 H1 {
  margin: 0px                /* Убираем отступы вокруг заголовка */
}
 DIV.shadow H1 {
  position: relative;        /* Относительное позиционирование */
  left: 5px;                 /* Сдвиг тени вправо */
  top: -1em;                 /* Сдвиг тени вверх от исходного положения */
  color: #ccc;               /* Цвет тени */
  z-index: -1                /* Порядок слоя */
}
</style>
</head>
<body>
 <h1>Чебурашка</h1>
 <div class=shadow><h1>Чебурашка</h1></div>
 ...
</body>
</html>
```

В этом примере показано добавление тени для заголовка страницы. Следует отметить некоторые особенности, связанные с данным примером, и в частности слоями.

- При использовании относительного позиционирования и сдвига слоя от первоначального положения место, которое занимал слой, остается неизменным. Поэтому расстояние между заголовком и текстом под ним в этом случае больше, чем обычно (рис. 10.5).
- Первоначально слой с тенью располагается строго под исходным объектом, поэтому для его смещения и требуется относительное позиционирование.
- Для сдвига слоя вверх необходимо знать высоту исходного объекта, чтобы правильно позиционировать тень. Поскольку в примере тень добавля-

ется к тексту, то слой сдвигается вверх на `1em`, эта единица равна высоте текущего шрифта. Точное наложение заголовков друг на друга в этом случае не происходит из-за небольших отступов.

- Браузер Netscape (а также Mozilla и Firefox) не отобразит тень, добавленную в данном примере, из-за присутствия `z-index: -1`. Он интерпретирует этот параметр как указание задать положение слоя ниже основного уровня веб-страницы (у которого значение равно нулю). Любое иное положительное значение параметра `z-index` заставляет тень отображаться поверх текста, а не под ним.

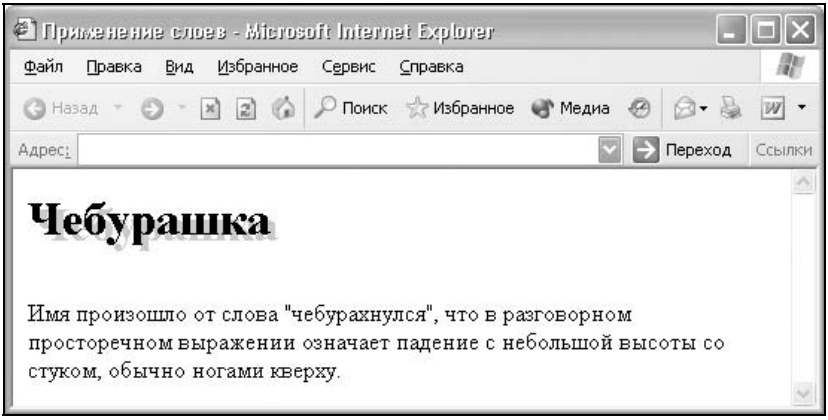


Рис. 10.5. Вид тени, созданной с помощью слоев

Чтобы обойти указанную особенность браузера Netscape, следует поменять слои местами, в таком случае `z-index` вообще не понадобится, слои примут нужный порядок автоматически (листинг 10.14). При этом тень будет отображаться несколько по-иному, и ее положение следует настроить с помощью атрибутов `left` и `top`.

Листинг 10.14. Изменение порядка слоев для отображения тени

```
<html>
<head>
<style type="text/css">
 H1 {
  margin: 0px;                /* Убираем отступы вокруг заголовка */
  color: #ccc                 /* Цвет тени */
}
 DIV H1 {
  position: relative;         /* Относительное позиционирование */
  left: 5px;                  /* Сдвиг тени влево */
```

```
  top: -1em;                  /* Сдвиг тени вверх от исходного положения */
  color: black                /* Цвет текста заголовка */
}
</style>
</head>
<body>
 <h1>Чебурашка</h1>
 <div><h1>Чебурашка</h1></div>
 ...
</body>
</html>
```

Из-за того, что свойства разных селекторов наследуются, необходимо раздельно задавать цвет заголовка и тени. Обратите также внимание, что положительное значение параметра `left` в данном случае смещает тень влево, а не вправо, как в листинге 10.13.

Использование спецсимволов

Кроме тегов в код HTML допустимо вставлять специальные символы, которые изображают буквы, отсутствующие на клавиатуре, например буквы греческого языка, а также различные знаки типа денежных, знак copyright, параграфа, угловых скобок и т. д. Чтобы браузер понимал, что имеет дело не с обычным текстом, спецсимволы всегда начинаются с амперсанда (&), после него обычно идет ключевое слово или шестнадцатеричное значение, а заканчиваются они точкой с запятой.

Полный список всех символов достаточно большой и редко используется целиком. Поэтому в табл. 10.3 приведены только основные и наиболее распространенные знаки, которые повсеместно применяются на сайтах.

Таблица 10.3. *Часто применяемые спецсимволы*

Имя	Код	Вид	Описание
			Неразрывный пробел
&sect;	&#167;	§	Символ параграфа
&copy;	&#169;	©	Знак copyright
&reg;	&#174;	®	Знак зарегистрированной торговой марки
&frac14;	&#188;	¼	Дробь — одна четверть
&frac12;	&#189;	½	Дробь — одна вторая
&frac34;	&#190;	¾	Дробь — три четверти

Таблица 10.3 *(окончание)*

Имя	Код	Вид	Описание
&quot;	&#34;	"	Двойная кавычка
&amp;	&#38;	&	Амперсанд
&lt;	&#60;	<	Знак "меньше"
&gt;	&#62;	>	Знак "больше"
&laquo;	&#171;	"	Левая двойная угловая скобка
&raquo;	&#187;	"	Правая двойная угловая скобка
&larr;	&#8592;	←	Стрелка влево
&uarr;	&#8593;	↑	Стрелка вверх
&rarr;	&#8594;	→	Стрелка вправо
&darr;	&#8595;	↓	Стрелка вниз
&harr;	&#8596;	↔	Стрелка влево–вправо
&ndash;	&#8211;	–	Тире
&mdash;	&#8212;	—	Длинное тире

Хотя угловые скобки и символ амперсанда есть на клавиатуре, в некоторых случаях напрямую их в код документа вставлять нельзя. Это, например, происходит, когда требуется отобразить на веб-странице HTML-код. Содержимое угловых скобок будет рассматриваться как теги, и не станет выводиться в окне браузера. В этом случае как раз и пригодятся знаки, которые выглядят как скобки, но сами ими не являются (листинг 10.15).

Листинг 10.15. Применение спецсимволов для отображения кода

```
<html>
<body>
<p><b>Пример. Создание таблицы с помощью скрипта</b></p>
<p>&lt;html&gt;<br>
  &lt;body&gt;</p>
<p>&lt;table width=400 border=1&gt;<br>
  &lt;script language=&quot;JavaScript&quot;&gt;<br>
  for (i=1; i&lt;6; i++) {<br>
  document.writeln(&quot;&lt;tr&gt;&quot;);<br>
  for (j=1; j&lt;6; j++) document.write(&quot;&lt;td&gt;&quot; + i + j +
&quot;&lt;/td&gt;&quot;);<br>
  document.writeln(&quot;&lt;/tr&gt;&quot;);<br>
  }<br>
```

```
  &lt;/script&gt;<br>
  &lt;/table&gt; </p>
<p>&lt;/body&gt;<br>
  &lt;/html&gt;</p>
<p>Copyright &copy; 2002 &#8212; 2005 Влад Мержевич</p>
</body>
</html>
```

На рис. 10.6 показано, что получится в итоге применения спецсимволов на веб-странице.

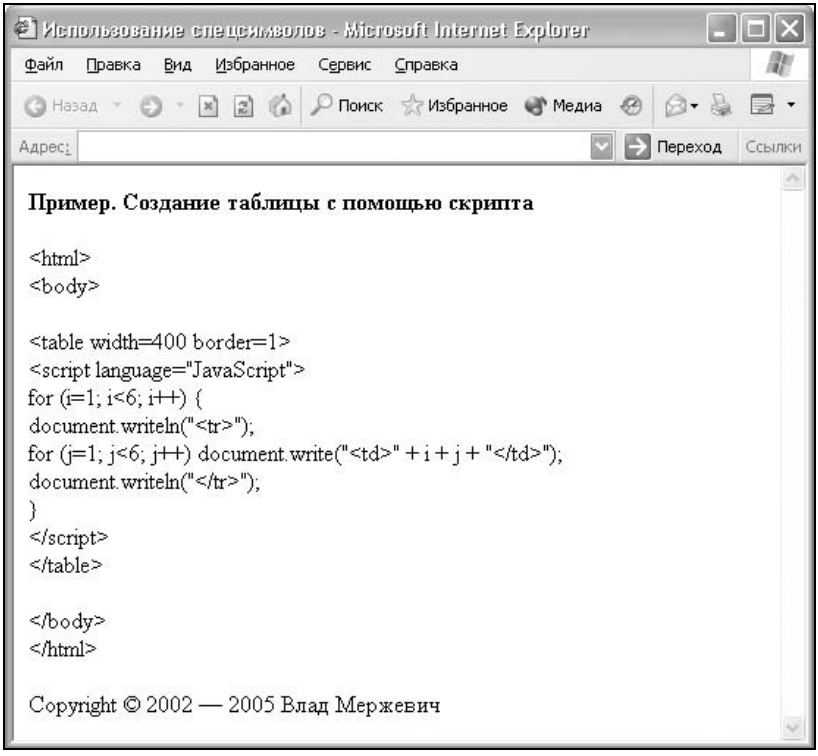


Рис. 10.6. Вид спецсимволов в окне браузера

Область применения указанных знаков не ограничивается выводом кода HTML на веб-страницу. В некоторых случаях спецсимволы могут добавляться вместо небольших рисунков, например со стрелками. Поскольку спецсимволы воспринимаются как текст, то к ним применимы те же правила, что и к тексту — их можно добавлять в любое место строки, задавать размер шрифта, изменять цвет и т. д.

ГЛАВА 11

Создание меню

Под *меню* понимают набор ссылок, позволяющих переходить к разным разделам или документам сайта. Меню непосредственно связано с *навигацией по сайту* — системой организации документов и их взаимодействия между собой. Другими словами, навигация дает пользователю представление о структуре сайта и возможность перемещаться к нужной странице. Термин *навигация* давно уже стал широким понятием и включает в себя не только способ перехода от страницы к странице, но также вид и представление ссылок. По этой причине к навигации относят элементы страницы, которые имеют к ней косвенное отношение, например меню. Тем не менее это уже связанные понятия и, говоря о навигации по сайту, обычно упоминают и меню, с помощью которого пользователь загружает в окно браузера требуемые веб-страницы.

Правильно организованное меню предоставляет пользователю возможность быстрого доступа к нужным ему разделам, показывает, где он находится в данный момент в структуре сайта и что на нем еще можно посмотреть. С этой целью придерживаются следующих рекомендаций.

- Пункт меню, совпадающий с текущей веб-страницей, не делают ссылкой, чтобы не путать посетителя. В самом деле, если имеется ссылка, то появляется желание нажать на нее, в результате чего откроется тот же документ, который мы только что видели. После этого возникает мысль, что если произошла загрузка страницы, то это новый документ, но содержимое говорит о том, что его уже читали. В общем, получается противоречие, которого легко избежать, если просто заменить ссылку обычным текстом.
- Число пунктов меню обычно делают не очень большим. В противном случае имеет смысл разбить меню на подменю или организовать объем информации на сайте по-другому.

- Ничто не мешает сочетать на сайте различные виды навигации. Например, горизонтальное меню может применяться для доступа к основным разделам сайта, а вертикальное — для его подразделов.

Условно все типы меню можно отнести к следующим категориям.

- *Вертикальное меню.* Пункты меню располагаются друг под другом, и число их может быть достаточно велико. В силу своей универсальности вертикальное меню встречается на сайтах наиболее часто.
- *Горизонтальное меню.* В этом случае пункты помещаются по горизонтали, но чтобы это не привело к появлению горизонтальной полосы прокрутки, их число ограничивают или располагают в два-три ряда.
- *Ниспадающее меню.* Обычно выглядит как горизонтальное меню, но когда курсор мыши наводится на пункты, открывается дополнительное подменю.
- *Всплывающее меню.* При наведении на ссылку курсора мыши такое меню появляется в виде панели с набором вариантов перехода. Как только курсор уводится прочь со ссылки или с меню, оно пропадает.
- *Контекстное меню.* Открывается при нажатии в окне браузера правой кнопкой мыши. Подобный тип меню уже встроен в браузер по умолчанию (мы говорим о системе Windows), но Internet Explorer позволяет переназначить его, добавив на сайт свое собственное. Из-за того, что такой тип меню поддерживается только одним браузером и его использование не является очевидным, он применяется достаточно редко.

К разновидностям меню также можно отнести различные списки, в том числе раскрывающиеся, и вкладки.

Горизонтальное меню

Горизонтальное меню является одним из распространенных и популярных элементов навигации, используемых на сайтах. Как следует из названия, пункты меню располагаются по горизонтали, как правило, в верхней части страницы. Перечислим следующие особенности, присущие горизонтальному меню:

- ширина веб-страницы ограничена разрешением монитора, его размерами, настройками браузера и операционной системы. По этой причине большое количество пунктов меню делать не рекомендуется. Иначе это может привести к появлению горизонтальной полосы прокрутки, что не позволяет удобно пользоваться сайтом, или стать причиной изменения вида и форматирования меню;
- горизонтальное меню располагают в верхней части веб-страницы, чтобы его можно было видеть без прокрутки содержимого. Иногда горизонтальное меню для удобства пользователей дублируют внизу страницы.

Технически создание горизонтального меню сводится всего к двум способам — с помощью таблиц и с применением слоев. Это не значит, что подобных меню может быть всего два, остальные варианты просто являются их модификациями.

Создание меню с помощью таблиц

Таблицы обладают некоторыми преимуществами по сравнению со слоями. В частности, таблицы удобно использовать, менять параметры их отображения и выравнивать по любому краю содержимое ячеек. Также для меню важно то, что ячейки при уменьшении ширины окна браузера не смещаются с горизонтальной линии, а лишь уменьшаются в размерах. Иными словами, заданное форматирование таблицы остается неизменным.

Самый простой вариант создания меню — когда в каждой ячейке располагается один пункт (рис. 11.1).

Рис. 11.1. Вид горизонтального меню, созданного через таблицы и стили

Используя тег `TABLE` совместно со стилями, можно получить самый разнообразный вид нашего меню. В частности, в листинге 11.1 показано, как через селектор `TABLE` добавить рамку вокруг таблицы, а с помощью контекстного селектора `.menu TD` установить белые линии между ячейками и поля вокруг их содержимого.

Листинг 11.1. Использование таблицы для создания меню

```
<html>
<head>
<style type="text/css">
 TABLE.menu {
  background: #fc0;                  /* Цвет фона меню */
  width: 80%;                        /* Ширина меню */
  border: 1px solid black;           /* Рамка вокруг таблицы */
  text-align: center                 /* Выравнивание текста по центру */
}
 .menu TD {
  border: 1px solid white;           /* Линия между ячейками */
  padding: 4px                       /* Поля вокруг текста */
}
</style>
</head>
```

```
<body>
 <table align=center cellspacing=0 class=menu>
  <tr>
   <td><a href=link1.html>Чебурашка</a></td>
   <td><a href=link2.html>Крокодил Гена</a></td>
   <td><a href=link3.html>Шапокляк</a></td>
   <td><a href=link4.html>Крыса Лариса</a></td>
  </tr>
 </table>
</body>
</html>
```

При создании горизонтального меню не обойтись без подсвечивания фона ячейки таблицы при наведении на нее курсора мыши. Подобный эффект повышает привлекательность меню и удобство работы с ним (рис. 11.2).

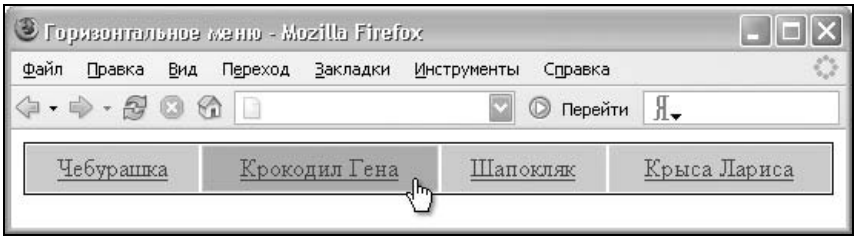


Рис. 11.2. Подсветка ячеек в таблице

Для создания подсветки текста воспользуемся псевдоклассом `hover`, который управляет стилем ссылки при наведении на нее курсора мыши. Остается только указать желаемый цвет фона с помощью свойства `background`. Также можно изменить и другие параметры, например цвет текста. Чтобы в качестве ссылки выступала вся ячейка целиком, а не только текст в ней, следует добавить для селектора `A` свойство `display: block`. В данном случае оно превращает ссылку в блочный элемент, заставляя ее занимать все свободное пространство. На виде ссылки это никак не отражается, но подсветка ячейки теперь будет происходить при наведении курсора мыши в любое ее место. Обратите внимание на то, что с этой же целью атрибут `padding` перенесен из `TD` в селектор `A` (листинг 11.2).

Листинг 11.2. Создание подсветки для ячеек таблицы

```
<html>
<head>
<style type="text/css">
```

```
 TABLE.menu {
  background: #ccc;          /* Цвет фона меню */
  width: 100%;               /* Ширина меню */
  border: 1px solid black;   /* Рамка вокруг таблицы */
  text-align: center         /* Выравнивание текста по центру */
}
 .menu TD {
  border: 1px solid white    /* Линия между ячейками */
}
 .menu A {
  display: block;            /* Ссылка на всю ячейку */
  padding: 4px;              /* Поля вокруг ссылок */
  width: 100%                /* Работает в браузере Internet Explorer,
                                вызывает ошибки в Netscape и Firefox */
}
 .menu A:hover {
  background: #fc0           /* Цвет фона при наведении курсора мыши */
}
</style>
</head>
<body>
<table align=center cellspacing=0 class=menu>
<tr>
 <td><a href=link1.html>Чебурашка</a></td>
 <td><a href=link2.html>Крокодил Гена</a></td>
 <td><a href=link3.html>Шапокляк</a></td>
 <td><a href=link4.html>Крыса Лариса</a></td>
</tr>
</table>
</body>
</html>
```

При выполнении приведенного кода браузеры неоднозначно интерпретируют некоторые атрибуты. Чтобы вся ячейка целиком превратилась в ссылку, для браузера Internet Explorer недостаточно добавить свойство `display`. Необходимо также указать `width: 100%`, что расценивается как указание установить ширину ссылки на всю доступную область ячейки. Только так Internet Explorer будет менять цвет фона ячейки при наведении на нее курсора мыши. В противном случае подсветка ячейки происходит только при наведении курсора на текст ссылки. Однако добавление `width` со значением 100 % приводит к ошибкам в отображении меню в браузерах Netscape и Firefox. Область подсветки превышает размеры ячейки и выходит по ширине за ее пределы.

Есть четыре варианта решения указанной проблемы.

1. Вообще убрать свойство `width` из стилей. Тогда все браузеры, кроме Internet Explorer, будут работать именно так, как и было задумано (см. рис. 11.2). В Internet Explorer подсветка ячеек при этом осуществляется только при наведении курсора на текст.
2. Уменьшить значение атрибута `width` до 97 %. Браузеры Netscape и Firefox подсветку ячеек отобразят должным образом, а Internet Explorer и Opera с небольшими полями справа и слева.
3. Удалить свойство `padding` у селектора `A`. Ошибка в браузере Netscape происходит из-за его наличия.
4. Несоответствие между браузерами получается только при использовании процентной записи. Стоит перейти на пикселы, как все проблемы снимаются. В листинге 11.3 приведен стиль для горизонтального меню, ширина которого задается в пикселах. Остальной код будет без изменений, как в листинге 11.2.

Листинг 11.3. Стиль для таблицы фиксированной ширины

```
<style type="text/css">
 TABLE.menu {
  background: #ccc;           /* Цвет фона меню */
  border: 1px solid black;    /* Рамка вокруг таблицы */
  text-align: center          /* Выравнивание текста по центру */
}
 .menu TD {
  width: 150px;               /* Ширина каждой ячейки */
  border: 1px solid white     /* Линия между ячейками */
}
 .menu A {
  display: block;             /* Ссылка на всю ячейку */
  padding: 4px;               /* Поля вокруг ссылок */
  width: 150px                /* Ширина ячейки */
}
 .menu A:hover {
  background: #fc0            /* Цвет фона при наведении курсора мыши */
}
</style>
```

Ширина таблицы складывается из ширины всех ячеек, поэтому ее можно не указывать. При этом способе ширина ячеек жестко задана и одинакова независимо от содержащегося в них текста.

Применение слоев

Горизонтальное меню, созданное с помощью слоев, по виду мало отличается от своего табличного собрата. И в том и другом случае можно активно применять стили для изменения оформления, а скорость загрузки фактически одинакова. Тем не менее приведем метод создания горизонтального меню с использованием слоев, чтобы было с чем сравнить и из чего выбрать понравившийся вариант.

При выборе слоев необходимо решить несколько задач — состыковка слоев друг с другом по горизонтали, выравнивание меню по центру, изменение цвета фона при наведении курсора на ссылку и т. д. Вначале рассмотрим выравнивание горизонтального меню, основанного на слоях.

В отличие от тега `TABLE`, у которого есть удобный параметр `align`, выравнивание набора слоев происходит через стили. Чтобы задать выравнивание по центру, необходимо создать контейнер `DIV` (назовем его `menu`) и указать ему желаемую ширину через свойство `width`. После этого добавить `margin-left` и `margin-right` со значением `auto`. Следует констатировать, что такая конструкция не работает в браузере Internet Explorer, поэтому для него придется пойти другим путем. А именно для селектора `BODY` установить `text-align: center`. Такая директива заставляет все элементы веб-страницы выравниваться по центру, поэтому если требуется задать выравнивание текста по левому или правому краю, то это следует указывать специально.

Совмещение слоев по горизонтали осуществляется путем использования атрибута `float: left` для класса `item`, который задает оформление пунктов меню. Вместе с тем действие `float` следует отменить для текста, расположенного ниже меню, иначе он будет располагаться справа от меню, а не внизу. Для этого после всех слоев, формирующих горизонтальное меню, нужно указать строку `<br style="clear: both">`.

Для названий пунктов, состоящих более чем из одного слова, желательно добавить свойство `white-space` со значением `nowrap`, текст тогда отображается одной строкой. Дело в том, что при автоматическом переносе текста изменяется высота слоя, поэтому пункты меню могут оказаться разновеликими. Чтобы этого не произошло, следует запретить переносы строк.

Если ширина слоя не задана явно через параметр `width`, она будет установлена автоматически на основе содержимого слоя. Изменение вида пункта меню можно осуществлять через параметр `border` и его производные — `border-left`, `border-right`, `border-top` и `border-bottom` (листинг 11.4).

Листинг 11.4. Создание меню с помощью слоев

```
<html>
<head>
```

```
<style type="text/css">
 BODY {
  text-align: center  /* Выравниваем содержимое веб-страницы по центру */
}
 .menu {
  width: 90%;               /* Ширина меню относительно окна браузера */
  margin-left: auto;  /* Отступ слева для браузера Opera и Netscape */
  margin-right: auto; /* Отступ справа для браузера Opera и Netscape */
  white-space: nowrap       /* Запрещаем переносы строк в тексте */
}
 .item {
  float: left;                      /* Состыковка с соседним слоем */
  width: 24%;                       /* Ширина каждого слоя */
  background: #ccc;                 /* Цвет фона меню */
  border-left: 1px solid black      /* Черная линия слева */
}
 .item A {
  display: block;                   /* Ссылка как блочный элемент */
  padding: 7px                      /* Поля вокруг текста ссылки */
}
 .item A:hover {
  background: #fc0                  /* Цвет фона подсветки */
}
</style>
</head>
<body>
 <div class=menu>
  <div class=item><a href=link1.htm>Чебурашка</a></div>
  <div class=item><a href=link2.htm>Крокодил Гена</a></div>
  <div class=item><a href=link3.htm>Шапокляк</a></div>
  <div class=item style="border-right: 1px solid black"><a
href=link4.htm>Крыса Лариса</a></div>
 </div>
 <br style="clear: both">
 Продолжение следует...
</body>
</html>
```

Окончательное меню, созданное с помощью слоев, показано на рис. 11.3.

Те же проблемы с браузерами и подсветкой пунктов меню, что были описаны для варианта с таблицей, присутствуют и здесь. Кроме того, обратите внимание на то, что ширина каждого слоя указана 24 %, а не 25 %, как можно было логично решить, разделив общую ширину меню (100 %) на че-

тыре пункта. Вначале разберемся, откуда взялась ширина 100 %, а не 90 %, как это явно указано в стиле. Это связано с вложенностью элементов и их отношением между собой. Установив для верхнего слоя `menu` значение `width: 90%`, мы тем самым отмеряем его ширину относительно родительского элемента, в качестве которого выступает окно браузера. Для вложенных слоев `item` теперь уже слой `menu` выступает в качестве родителя. При этом его ширина принимается за 100 % независимо от ширины относительно других родителей.

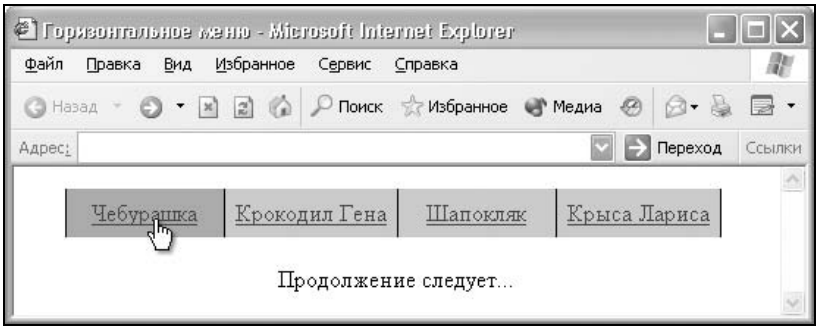


Рис. 11.3. Вид горизонтального меню, созданного с помощью слоев

Браузеры по-разному трактуют значение `width`. Например, Netscape и Firefox прибавляют к ширине элемента величину границы, заданной атрибутом `border`. Когда речь идет о ширине 100 %, это приводит к нарушению макета, поскольку слои по горизонтали в один ряд уже не помещаются, или к появлению горизонтальной полосы прокрутки. Чтобы избежать подобных помех, ширина каждого пункта меню установлена в 24 %. Как вариант — можно убрать горизонтальные линии или задавать ширину слоев в пикселах за вычетом толщины границы.

Текстовые вкладки

Вкладки — это один из элементов навигации, любимый как пользователями сайта за их наглядность и очевидность действия, так и дизайнерами за то, что вкладкам можно придавать любой подходящий вид без потери их функциональности. Вдобавок этот элемент хорошо выделяется на веб-странице, и сразу становится понятно, что вкладки нужны для перехода между разделами сайта. На рис. 11.4 показан один из возможных вариантов создания вкладок с помощью рисунков.

Создать графические вкладки можно в любом подходящем графическом редакторе. Ссылки делаются либо с помощью карт-изображений, либо разрезанием одной картинки на фрагменты. Мы, однако, сделаем вкладки ис-

ключительно простыми средствами, через стили, например, как показано на рис. 11.5.

Рис. 11.4. Вариант создания и размещения вкладок

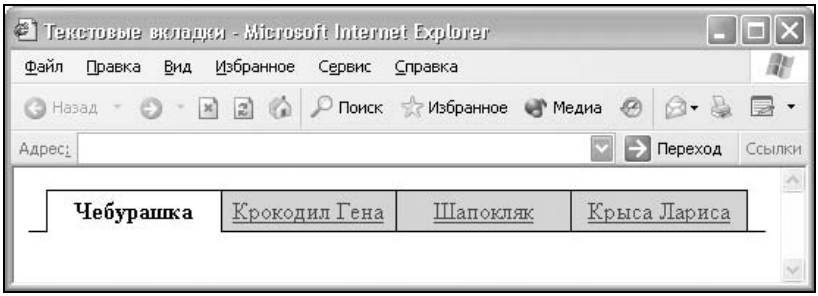


Рис. 11.5. Пример вкладок, созданных с помощью таблицы и стилей

Для создания вкладок потребуется таблица из шести ячеек. Четыре из них образуют сами вкладки, а две крайние ячейки нужны больше для красоты, они формируют горизонтальную линию и служат для отступа от краев.

Потребуется всего два селектора класса: один изменяет стиль текущей вкладки, назовем его `open`, а второй селектор, с именем `close`, будет управлять видом неактивной вкладки. Рамка создается с помощью атрибута `border`. Чтобы не образовывалась двойная рамка в местах состыковки ячеек, следует границу справа убрать. Для этого используется параметр `border-right: none`. Можно также воспользоваться атрибутом `border-collapse` со значением `collapse`, применяя его к тегу `TABLE`.

Для текущей вкладки (селектор `open`) следует спрятать также и нижнюю границу. Стиль самой правой и левой ячеек можно описать прямо в теге `TD`, но при частом использовании вкладок на сайте лучше создать отдельный класс (листинг 11.5).

Листинг 11.5. Использование таблицы для создания вкладок

```
<html>
<head>
<style type="text/css">
```

```
 .open {
  border: 1px solid black;          /* Рамка вокруг текущей вкладки */
  border-right: none;               /* Границу справа убираем */
  border-bottom: none;              /* Линию снизу тоже прячем */
  text-align: center;               /* Выравнивание текста по центру */
  font-weight: bold                 /* Жирное начертание текста */
}
 .close {
  border: 1px solid black;          /* Рамка вокруг вкладок */
  background: #cfd6d4;              /* Цвет фона вкладок */
  border-right: none;               /* Границу справа убираем */
  text-align: center                /* Выравнивание текста по центру */
}
</style>
</head>
<body>
 <table width=100% cellspacing=0 cellpadding=4>
  <tr>
   <td width=10 align=center style="border-bottom: 1px solid ↳
black"> </td>
   <td width=25% class=open>Чебурашка</td>
   <td width=25% class=close><a href=link2.hml>Крокодил Гена</a></td>
   <td width=25% class=close><a href=link3.hml>Шапокляк</a></td>
   <td width=25% class=close><a href=link4.hml>Крыса Лариса</a></td>
   <td width=10 align=center style="border-left: 1px solid black; ↳
border-bottom: 1px solid black"> </td>
  </tr>
 </table>
</body>
</html>
```

Внутри текущей вкладки ссылку не делают, чтобы не сбивать с толку пользователей. Наоборот, текст в ней выделяют другим цветом или жирным начертанием.

Через слои также можно создать вкладки, причем самого разного вида. Для этого следует указать слой с именем, например `tabs` (он будет служить контейнером), а внутрь него поместить набор отдельных слоев, образующих вкладки. Чтобы слои располагались по горизонтали, надо использовать теги `SPAN` или для селектора `DIV` добавить стилевой параметр `display: inline`. По умолчанию тег `DIV` расценивается как блочный элемент, до и после него автоматически добавляется перенос строки. Аргумент `inline` атрибута `display` отменяет эту особенность, превращая `DIV` во встроенный элемент.

Сами вкладки образуются добавлением рамки вокруг текста, причем за счет применения полей нижняя граница вкладок накладывается на линию слоя

tabs. Для текущей вкладки нижнюю линию надо спрятать, но проще это сделать, установив ее цвет таким же, как у фона веб-страницы (листинг 11.6).

Чтобы соприкасающиеся вкладки не создавали двойную линию, граница справа прячется путем присваивания значения none атрибуту border-right. Но это же приводит к тому, что самая крайняя вкладка справа окажется без линии. Чтобы этого не случилось, для последнего слоя добавляется внутренний стиль через параметр style.

Листинг 11.6. Использование слоев для создания вкладок

```
<html>
<head>
<style type="text/css">
 .tabs {
  border-bottom: 1px solid black;   /* Линия под вкладками */
  padding-bottom: 1px;            /* Отступ снизу для Internet Explorer
*/
  padding-bottom: 0px;              /* Отступ снизу для Firefox */
  padding-left: 20px                /* Отступ слева от вкладок */
}
 .tabs DIV {
  padding: 2px 15px 1px;            /* Поле сверху, по горизонтали и снизу
                                    от текста */
  border: 1px solid black;          /* Рамка для создания вкладки */
  border-right: none;               /* Границу справа убираем */
  background: #cfd6d4;              /* Цвет фона вкладок */
  display: inline                   /* Превращаем блочные элементы
                                    во встроенные */
}
 .tabs .open {
  border-right: none;               /* Убираем линию справа */
  border-bottom: 1px solid white;   /* Маскируем линию снизу */
  background: white                 /* Цвет фона текущей вкладки */
}
</style>
</head>
<body>
 <div class=tabs>
  <div><a href=link1.hml>Чебурашка</a></div> ↳
  <div class=open>Крокодил Гена</div> ↳
  <div><a href=link3.hml>Шапокляк</a></div> ↳
  <div style="border-right: 1px solid black"><a href=link4.hml>Крыса
                                                    Лариса</a></div>
```

```
 </div>
</body>
</html>
```

При использовании приведенного метода создания вкладок с помощью слоев нужно принимать во внимание следующие моменты. Теги `SPAN` или `DIV`, у которых добавлен атрибут `display: inline`, чувствительны к пробелам. Перенос строки также в некоторых случаях воспринимается как пробел, поэтому если между вкладками появились необоснованные промежутки, запишите код слоев в виде одной строки без пробелов.

Браузеры неоднозначно интерпретируют, как отображать границы у вложенных элементов. Речь идет только о том случае, когда внутри блочного элемента располагается встроенный. Для примера создадим два вложенных слоя с рамками разного цвета — красной и черной:

```
<div style="border: 2px solid red">
 <span style="border: 2px solid black; padding: 0px">Lorem ipsum dolor
                                                    sit amet...</span>
</div>
```

В браузере Internet Explorer и Opera верхняя и нижняя границы будут сливаться, и под текстом мы увидим черную линию. В Netscape и Firefox сливаются только верхние границы, а между нижними оказывается промежуток величиной один пиксел. Его легко убрать, если добавить `padding: 1px` для тега `SPAN`. Но это в свою очередь приведет к смещению границ в других браузерах.

Таким образом, вид стилевых вкладок будет немного различаться в разных браузерах, как показано на рис. 11.6.

Рис. 11.6. Вид вкладок, созданных с помощью стилей:
а — в браузерах Internet Explorer и Opera; *б* — в браузерах Netscape и Firefox

Обратите внимание на то, что на рис. 11.6, *а* горизонтальная линия располагается на один пиксел выше, чем на рис. 11.6, *б*. В остальном полученные результаты оказываются идентичными.

Поскольку несоответствие между браузерами касается только нижней границы встроенных элементов, то следует "перевернуть" вкладки, проведя го-

ризонтальную линию сверху. Заодно можно инвертировать цвета, в этом случае активная вкладка выделяется цветом, а остальные — нет (рис. 11.7).

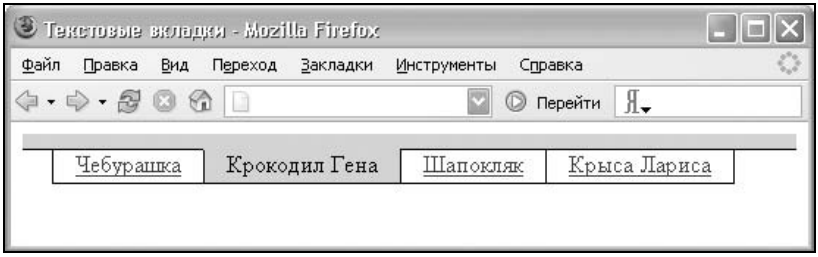


Рис. 11.7. Пример инвертированных вкладок

Код по сравнению с предыдущим листингом претерпит незначительные изменения, в частности цвет фона теперь указывается для активной вкладки (класс `open`), а линия проводится сверху. Кроме того, сверху вкладок через тег `DIV` добавлен прямоугольник, фоновый цвет которого совпадает с цветом активной вкладки. При этом ее верхняя граница маскируется линией того же цвета, что и фон (листинг 11.7).

Листинг 11.7. Изменение вида вкладок

```
<html>
<title>Текстовые вкладки</title>
<head>
<style type="text/css">
 .tabs {
  border-top: 1px solid black;       /* Линия сверху вкладок */
  padding-left: 20px                 /* Отступ слева от вкладок */
}
 .tabs DIV {
  padding: 0px 15px 3px;            /* Поле сверху, по горизонтали и снизу
                                    от текста */
  border: 1px solid black;           /* Рамка для создания вкладки */
  border-right: none;                /* Убираем линию справа */
  display: inline                   /* Превращаем блочные элементы
                                    во встроенные */
}
 .tabs .open {
  border-right: none;                /* Убираем линию справа */
  border-top: 1px solid #cfd6d4;     /* Маскируем линию сверху */
  background: #cfd6d4                /* Цвет фона текущей вкладки */
}
```

```
</style>
</head>
<body>
<div style="background: #cfd6d4; height: 10px"></div>
<div class=tabs>
  <div><a href=link1.hml>Чебурашка</a></div>
  <div class=open>Крокодил Гена</div>
  <div><a href=link3.hml>Шапокляк</a></div>
  <div style="border-right: 1px solid black"><a href=link4.hml>Крыса
Лариса</a></div>
 </div>
</body>
</html>
```

Опять же, примите во внимание то, что код во избежание появления пустых промежутков между вкладками надо набирать без пробелов.

Графические вкладки

Использование графических изображений для создания вкладок привлекает внимание посетителей, формирует эстетическую привлекательность сайта и расширяет возможности по его дизайну. Можно просто сделать подходящие рисунки с текстом для каждой вкладки, но интереснее использовать подсветку вкладок, когда они меняют свой цвет, выделяясь таким образом из группы при наведении на них курсора мыши.

Создание вкладок

Для создания вкладок вначале потребуется два рисунка, их можно подготовить в любом подходящем графическом редакторе. Первый рисунок (рис. 11.8, *а*) служит для обозначения неактивного пункта, а второй (рис. 11.8, *б*) заменяет его при наведении на пункт курсора мыши. Рисунки должны быть одинаковыми по размеру.

а *б*

Рис. 11.8. Рисунки для создания вкладок: *а* — изображение вкладки; *б* — изображение, которое будет появляться при наведении на вкладку курсора

Подобный прием, когда один рисунок меняется на другой при наведении на него курсора мыши, называется *эффект перекатывания*, или *rollover* (рис. 11.9).

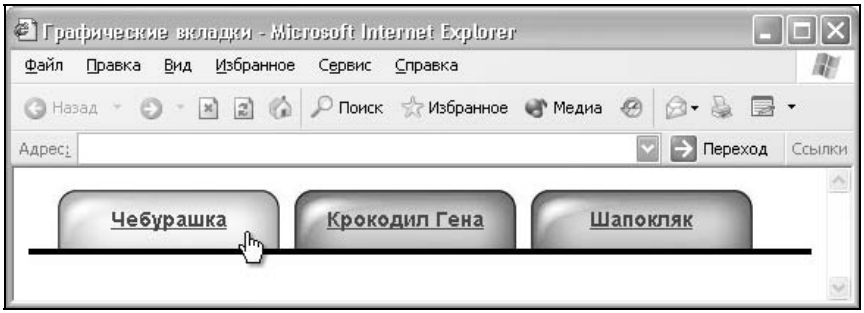


Рис. 11.9. Вид графических вкладок с эффектом перекатывания

При использовании графических изображений для создания вкладок воспользуемся слоями и стилями. Структура кода остается аналогичной текстовым вкладкам, а основные изменения коснутся таблицы стилей.

Самый верхний слой-контейнер с именем `menu` определяет высоту вкладок, параметры надписей и стиль линии внизу. Высота слоя, задаваемого параметром `height`, должна равняться высоте наших рисунков, в данном случае она составляет 40 пикселов. Отступы от начала нижней линии до границы крайних вкладок управляются аргументом `padding`, а высота и цвет линии снизу атрибутом `border-bottom`.

Поскольку ширина и высота всех рисунков задана заранее, то размер текста меняться не должен, иначе может получиться, что он выйдет за пределы рисунка. Увеличить или уменьшить шрифт можно через настройки браузеров. Чтобы этого не произошло, требуется жестко задать размер шрифта в абсолютных единицах, например пикселах, через параметр `font-size`.

Теперь перейдем к отдельным пунктам вкладок. Чтобы не загромождать стиль разными именами, воспользуемся контекстными селекторами. Конструкция `.menu DIV` означает, что стиль будет работать только для тега `DIV`, который размещается внутри слоя с именем `menu`. Осталось только сделать для него описание (листинг 11.8).

Параметр `float` предназначен для размещения слоев по горизонтали, без него по умолчанию слои располагаются друг под другом. Обязательно надо установить ширину (`width`) и высоту (`height`) слоя, равную размерам рисунка. Аргумент `margin-right` управляет расстоянием между вкладками, а `background` определяет путь к файлу с изображением, он может быть как абсолютным, так и относительным, как указано в данном примере. Исходная картинка, показанная на рис. 11.8 *а*, называется greentab.jpg и устанавливается как фоновое изображение.

Осталось только описать ссылки. Также воспользуемся контекстными селекторами и создадим стиль для элемента `.menu A`, который будет действовать только для ссылок в слое `menu`. Это позволяет использовать тег `A` в дальнейшем, уже не заботясь о том, что он где-то описан. Параметр `padding`

отвечает за сдвиг текста внутри слоя, чтобы надпись располагалась приблизительно по центру рисунка. Высота и ширина, установленные как 100 %, требуются, чтобы в качестве ссылки выступала вся область рисунка, а не только текстовая ссылка. Если убрать эти параметры, то эффект перекатывания возникнет лишь при наведении курсора на надпись.

Смена одной картинки на другую осуществляется через псевдокласс `hover`. Он отвечает за событие, возникающее при наведении курсора мыши на ссылку. Тогда происходит замена фонового изображения на другое, опять же через параметр `background`. В этом примере картинка называется orangetab.jpg и показана на рис. 11.8, *б*.

Листинг 11.8. Графические вкладки с подсветкой

```
<html>
<head>
<style type="text/css">
 .menu {
  height: 40px;                      /* Высота вкладок */
  padding: 0px 20px;                 /* Поля по вертикали и горизонтали */
  border-bottom: 4px solid black;    /* Параметры линии внизу */
  font-family: Arial, sans-serif;    /* Шрифт надписи */
  font-weight: bold;                 /* Жирное начертание */
  font-size: 14px                    /* Размер шрифта надписи */
}
 .menu DIV {
  float: left;                       /* Состыковка с соседней вкладкой */
  width: 151px;                      /* Ширина вкладки */
  height: 40px;                      /* Высота вкладки */
  margin-right: 10px;                /* Расстояние между вкладками */
  text-align: center;                /* Выровнять надпись по центру */
  background:
       url(greentab.jpg)             /* Путь к начальному рисунку */
       no-repeat                     /* Установить фон без повторений */
}
 .menu A {
  display: block;                    /* Ссылка как блочный элемент */
  width: 100%;                       /* Ссылка на всю ширину вкладки */
  height: 100%;                      /* Ссылка на всю высоту вкладки */
  padding: 12px 0px          /* Поля для размещения текста по центру */
}
 .menu A:hover {
  background:
       url(orangetab.jpg)            /* Путь к замещаемой картинке */
       no-repeat                     /* Отображать фон без повторений */
}
```

```
</style>
</head>
<body>
 <div class=menu>
  <div><a href=link1.html>Чебурашка</a></div>
  <div><a href=link2.html>Крокодил Гена</a></div>
  <div><a href=link3.html>Шапокляк</a></div>
 </div>
</body>
</html>
```

Чтобы фон отображался в браузерах без смещений относительно исходного положения, поля для управления положением текста по горизонтали следует задать равными нулю. По центру надпись выравнивается с помощью свойства `text-align`. В то же время выравнивание по вертикали происходит за счет установки полей сверху и снизу от надписи. Так, в примере они задаются равными 12 пикселам через атрибут `padding` для селектора `A`. Эта величина зависит от размера шрифта и высоты изображения и подбирается в каждом случае индивидуально.

Браузер Opera 7 черную линию внизу вкладок отображает за рисунками, а не поверх них, как показано на рис. 11.9. Это единственное отличие в отображении данного примера, которое наблюдается в разных браузерах.

Приведенный способ создания вкладок имеет как свои достоинства, так и недостатки. К преимуществам относится простота и быстрота построения вкладок. Среди недостатков — меню не работает в браузере Opera 6 и ниже, а также медленная скорость работы. Это связано с тем, что загрузка в память второго рисунка происходит не заранее, а только по мере обращения к нему, т. е. при наведении курсора мыши на вкладку. Соответственно замедляется и смена одной картинки на другую, поскольку требуется время на ее загрузку. Этого можно избежать, если уменьшить объем файлов изображений или осуществить предварительную загрузку картинок в память. Еще один способ ускорения работы меню связан с объединением рисунков в одну картинку и сдвигом фонового изображения, как показано далее.

Предварительная загрузка изображений

При создании эффекта перекатывания, как показано в листинге 11.8, загружается только первый рисунок. После наведения курсора мыши на изображение начинается загрузка второго рисунка, что происходит не мгновенно и портит впечатление от ожидания. Поэтому желательно необходимые изображения загружать заранее, еще до их демонстрации, делать для них так называемый *прелоад* (от англ. preload — предварительная загрузка). Этот процесс осуществляется через скрипты, в частности в JavaScript использует-

ся оператор `new`, с помощью которого создается новый объект типа `Image`. Задавая ему адрес требуемого изображения, в итоге получаем, что рисунок оказывается загруженным в память еще до его отображения.

В листинге 11.9 показано создание функции `preloadImage`, она проверяет, все ли указанные в качестве аргументов изображения загружены в память, и если нет, то загружает их. Вызов функции осуществляется с помощью события `onLoad`, которое указывается в теге `BODY`. Это событие наступает в момент загрузки всего документа.

Листинг 11.9. Предварительная загрузка изображения через скрипт

```
<html>
<head>
<script language="JavaScript">
function newImage(arg) {
// Проверяем, есть ли вообще изображения на странице
  if (document.images) {
   result = new Image(); // Создаем новый объект типа Image
   result.src = arg; // Присваиваем ему адрес изображения
   return result; // Возвращаем результат
  }
}

// Первоначально изображения в память не загружены
preloadFlag = false;
function preloadImages() {
  frame = new Array(); // Заводим новый массив с именем frame

// В массиве arg хранятся все аргументы функции preloadImages
  arg = preloadImages.arguments;

// Проверяем, есть ли вообще изображения на странице
  if (document.images) {
// Пробегаемся по всем аргументам функции
    for (i=0; i<arg.length; i++) {
// И для каждого изображения вызываем функцию newImage()
       frame[i] = newImage(arg[i]);
    }
    preloadFlag = true; // Да, все изображения загружены!
  }
}
</script>
</head>
```

```
<body onLoad="preloadImages('greentab.jpg', 'orangetab.jpg')">
 ...
</body>
</html>
```

Первой вызывается функция `preloadImages`, ее задача — загрузить в память браузера изображения, которые перечислены в виде аргументов. Заметим, что в данном случае требуется указывать не только имя файла, но и путь к нему. Если рисунки хранятся в папке `images`, тогда функция будет вызываться как `preloadImages('images/1.gif', 'images/2.gif')`. Вначале пробегаем по всем аргументам этой функции и каждый из указанных рисунков загружаем в память, для этого вызываем нашу функцию `newImage`. Чтобы браузер заранее, еще до появления изображения на веб-странице "знал" о существовании рисунка, необходимо создать новый объект типа `Image` через оператор `new` и связать его с путем к графическому файлу. Этим функция `newImage` и занимается.

Еще один способ загрузки изображений связан с кэшированием. Суть в следующем. Если требуемое изображение показать раньше, чем необходимо, то последующее отображение произойдет быстрее. Это связано с тем, что при первом использовании графический файл попадает в *кэш* браузера — специальное место на жестком диске для временного хранения информации. Потом изображение загружается быстрее, поскольку браузер открывает его с локального ресурса, а не из Интернета. Процесс сохранения файлов в кэше и повторное их использование принято называть *кэшированием*.

Но тут кроется противоречие. Как отобразить рисунок заранее, если его нужно показать только в определенном месте? На помощь придут слои. Рисунок надо поместить внутри контейнера `DIV` и задать ему абсолютное позиционирование совместно со свойством `display: none`. При этом содержимое слоя не будет отображаться, но и мешать выводу остальной информации также не станет. Другой вариант — выводить слой за пределами окна браузера, указав ему отрицательные координаты (листинг 11.10).

Листинг 11.10. Использование слоя для предварительной загрузки изображений

```
<html>
<body>
<div style="position: absolute; left: -200px">
 <img src=orangetab.jpg width=151 height=40>
</div>
...
<img src=orangetab.jpg width=151 height=40></body>
</html>
```

Чтобы предварительно загружаемый рисунок не влиял на остальной код, в стиле слоя необходимо добавить `position: absolute` и отрицательное значение параметра `left`. Поскольку отсчет ведется от левого верхнего угла слоя, то величина `left` должна быть больше ширины картинки.

Объединение изображений в одно

Заминка с отображением вкладок происходит только при загрузке второго рисунка. Раз так, то почему бы вообще от него не отказаться? С этой целью надо соединить две картинки в одну, как показано на рис. 11.10.

Рис. 11.10. Исходное изображение для создания вкладок

В стиле теперь нужно указать путь только к одному графическому файлу, а затем сместить фоновый рисунок по горизонтали. Все это можно сделать с помощью свойства `background`. В качестве аргументов следует указать два числа через пробел: первое определяет положение левого верхнего угла рисунка по горизонтали (оно в данном случае и будет изменяться), а второе — по вертикали. В листинге 11.8 изменения коснутся только стиля, да и то в месте, где упоминается новое изображение. В листинге 11.11 изображение называется green_orange.jpg; предполагается, что на рис. 11.10 слева находится исходная вкладка, а справа — вкладка, которая появляется при наведении на нее курсора мыши.

Листинг 11.11. Сдвиг фонового изображения

```
<style type="text/css">
 .menu DIV {
  float: left;                    /* Состыковка с соседней вкладкой */
  width: 151px;                   /* Ширина вкладки */
  height: 40px;                   /* Высота вкладки */
  margin-right: 10px;             /* Расстояние между вкладками */
  text-align: center;             /* Выровнять надпись по центру */
  background:
       url(green_orange.jpg)      /* Путь к совместному рисунку */
       no-repeat                  /* Установить фон без повторений */
}
 .menu A:hover {
  background:
       url(green_orange.jpg)      /* Путь к совместному рисунку */
       -151px                     /* Сдвигаем фон влево */
```

```
      0px                          /* По вертикали оставляем на месте */
      no-repeat                    /* Отображать фон без повторений */
}
</style>
```

Отрицательное значение для смещения фона требуется только в том случае, если в файле green_orange.jpg первоначальная вкладка находится слева. Вводимое значение при этом равняется ширине вкладки.

Вертикальное меню

Меню, в котором ссылки расположены друг под другом, называется *вертикальным*. К его достоинствам относят то, что ширину можно задавать как в пикселах, так и в процентах, подстраивая таким образом под любой макет. К тому же число пунктов вертикального меню может быть достаточно велико, при увеличении его высоты просмотр происходит посредством вертикальной полосы прокрутки веб-страницы. Меню также можно снабдить своей собственной полосой прокрутки для случая большого числа пунктов, как это, например, показано на рис. 5.27.

Простое меню с рамкой

Вначале создадим простое меню, состоящее из четырех пунктов. Ссылкой будет служить не только текст, но и пустое пространство справа от него (рис. 11.11). Для этого следует в стиле селектора `A` указать `display: block` и установить значение ширины `width` как 100 %. Но как уже упоминалось, подобная запись вызовет ошибки в браузере Netscape. Дело в том, что Internet Explorer ширину блока устанавливает такой, как она задана атрибутом `width`, а Netscape (Mozilla и Firefox в том числе) приплюсует к ней толщину границ и величину полей. Отсюда два решения — либо убрать из стилей атрибут `padding`, что не всегда возможно, либо уменьшить ширину области ссылки. На рис. 11.11 показано, как различается вид вертикального меню в разных браузерах при наведении на один из пунктов курсора мыши.

а *б*

Рис. 11.11. Вид пунктов меню в разных браузерах:
а — Internet Explorer и Opera; *б* — Netscape и Firefox

Как видно из рис. 11.11, браузер Netscape незначительно увеличивает ширину меню; это следует учитывать, если требуется точное позиционирование элементов на странице.

Ширина меню в пикселах или процентах устанавливается через свойство `width`, которое применяется к классу `menu` (листинг 11.12). Здесь же задаются параметры рамки вокруг меню и отступы от нее до ссылок. Для того чтобы вокруг ссылки при наведении на нее курсора мыши появлялась цветная граница, требуется сделать следующее. Вначале к селектору `A` добавляем невидимую рамку, цвет которой совпадает с цветом веб-страницы, так, в примере используется белый цвет. Маскирование границы необходимо, поскольку текст ссылки иначе станет смещаться от своего исходного положения. Теперь для псевдокласса `A:hover` остается только указать желаемые параметры рамки, цвет текста и фона.

Листинг 11.12. Создание простого вертикального меню

```
<html>
<head>
<style type="text/css">
 .menu {
  width: 200px;              /* Ширина меню в пикселах */
  padding: 5px;              /* Отступы от рамки до пунктов меню */
  border: 1px solid black    /* Рамка вокруг меню */
}
 .menu A {
  width: 97%;                /* Ширина пунктов меню */
  padding: 2px;              /* Отступ от рамки вокруг ссылки до текста */
  display: block;            /* Ссылка как блочный элемент */
  border: 1px solid white;   /* Маскируем рамку вокруг ссылки */
  text-decoration: none      /* Убираем подчеркивание у ссылок */
}
 .menu A:hover {
  background: #faf3d2;              /* Цвет фона под ссылкой */
  color: #800000;                   /* Новый цвет ссылки */
  border: 1px dashed #634f36        /* Рамка вокруг ссылки */
}
</style>
</head>
<body>
 <div class=menu>
  <div><a href=link1.html>Чебурашка</a></div>
  <div><a href=link2.html>Крокодил Гена</a></div>
  <div><a href=link3.html>Шапокляк</a></div>
```

```
  <div><a href=link4.html>Крыса Лариса</a></div>
 </div>
</body>
</html>
```

Чтобы ширина меню при использовании процентной записи не увеличивалась в браузере Netscape, уберите атрибут `padding` у класса `menu`.

Меню с подсветкой

Обычно выделение активного пункта меню, т. е. пункта, на котором стоит курсор мыши, происходит путем изменения цвета фона под текстом. Менее распространенным, в то же время наглядным и эффектным способом акцентирования внимания на элементе является появление цветного прямоугольника рядом с пунктом при наведении на него курсора мыши. На рис. 11.12 показано меню, пункты которого подсвечиваются подобным образом.

Рис. 11.12. Изменение оформления активного пункта меню

Легче всего добавить прямоугольник слева или справа от ссылки через свойство `border-left` или `border-right`. При этом толщина границы должна быть достаточно большой для того, чтобы из линии превратиться в прямоугольник. Поскольку требуется лишь менять ее цвет, то для селектора `A:hover` проще установить атрибут `border-left-color`, в качестве его значения выступает требуемый цвет линии слева от элемента. Аналогично цвет линии для случая, если она располагается справа, меняется через `border-right-color`.

В листинге 11.13 показано создание меню, у которого слева от каждого пункта добавляется зеленая линия толщиной 10 пикселов. Это достигается тем, что для селектора `A` используется свойство `border-left`. Чтобы ссылка была на всю ширину пункта, следует добавить атрибут `display: block` и установить `width` со значением 100 %. Поскольку границу вокруг меню рисовать не нужно, то подобная величина не повлияет на вид меню в разных браузерах.

При наведении курсора мыши на пункт меню срабатывает селектор `A:hover`, который управляет оформлением ссылок в этом случае. Для него в данном

примере цвет границы слева изменяется на оранжевый, а также становится другим цвет текста и фона.

Чтобы отделить один пункт от другого, между ними проводится линия серого цвета путем добавления параметра `border-bottom` к селектору `A`. Для самого нижнего пункта граница также будет отображаться, что не всегда желательно, поэтому она убирается за счет применения встроенного стиля и атрибута `border-bottom` со значением `none`.

Листинг 11.13. Меню с подсветкой пунктов

```
<html>
<head>
<style type="text/css">
 .menu {
  width: 200px                        /* Ширина меню */
}
 .menu A {
  display: block;                     /* Ссылка как блочный элемент */
  width: 100%;                        /* Ссылка на всю ширину меню */
  padding: 5px;                       /* Поля вокруг надписи */
  border-left: 10px solid #13694e;    /* Линия слева */
  border-bottom: 1px solid silver;    /* Линия между пунктами */
  background: #74a18e;                /* Исходный цвет фона */
  color: white;                       /* Исходный цвет текста */
  text-decoration: none               /* Убираем подчеркивание у ссылок */
}
 .menu A:hover {
  border-left-color: orange;          /* Меняем цвет линии слева */
  background: #a18e74;                /* Новый цвет фона под ссылкой */
  color: #ffffcc                      /* Новый цвет ссылки */
}
</style>
</head>
<body>
 <div class=menu>
  <div><a href=link1.html>Чебурашка</a></div>
  <div><a href=link2.html>Крокодил Гена</a></div>
  <div><a href=link3.html>Шапокляк</a></div>
  <div><a href=link4.html style="border-bottom: none">Крыса Лариса</a></div>
 </div>
</body>
</html>
```

Хотя вид данного меню в разных браузерах будет одинаковым, его ширина различается.

Меню и подменю

Меню, которое содержит большое число пунктов, имеет смысл разделить на части, тем самым создавая так называемые *подменю*. Подобная организация меню хорошо воспринимается пользователем, к тому же она достаточно часто используется в различных программах, следовательно, будет привычной и на сайте.

Вариантов создания подменю может быть множество, например, на рис. 11.13 показан способ, когда подменю отображается справа от выделенного пункта.

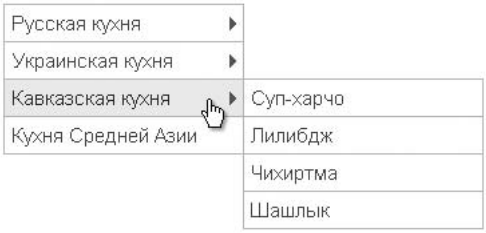


Рис. 11.13. Вид меню с выбранным подменю

Небольшая стрелка справа от текста нужна для наглядности, она показывает, что за пунктом скрывается подменю.

Создать универсальное меню, приведенное на рис. 11.13, с помощью одного CSS не удается и для этого придется подключать JavaScript. Код при этом разбивается на три части: HTML, CSS и JavaScript, которые далее и рассмотрим по отдельности.

Из-за того, что требуется применять упорядоченность элементов и прямую зависимость их друг от друга, слои и теги `DIV` и `SPAN` в частности, не очень годятся на роль создания подобного меню. Лучше использовать списки, а именно теги `UL` и `LI`. Их внутренняя организация как нельзя лучше подходит для создания меню различного уровня вложенности. Начинается меню с тега `UL`, а затем каждый пункт меню верхнего уровня добавляется через тег `LI`. Создание подменю происходит за счет конструкции `UL`, которая помещается внутрь `LI` (листинг 11.14).

Листинг 11.14. Структура меню при использовании списков

```
<ul>
 <li>Пункт 1.
```

```
  <ul>
    <li>Подпункт 1.1.</li>
    <li>Подпункт 1.1.</li>
  </ul>
 </li>
 <li>Пункт 2.
  <ul>
    <li>Подпункт 2.1.</li>
    <li>Подпункт 2.1.</li>
  </ul>
 </li>
</ul>
```

Формирование подменю третьего уровня вложенности происходит аналогично, путем добавления тега `UL` внутрь подпункта `LI`.

Стиль предназначен для оформления пунктов меню и сокрытия подменю. Поскольку тег `UL` отображает элементы как список, то вначале следует спрятать маркеры, установив для селектора `UL` атрибут `list-style: none`. Также надо убрать различные отступы и поля, которые по умолчанию добавляются для элементов списка, и задать ширину всех меню через параметр `width` (листинг 11.15).

Подменю вначале оказывается скрытым за счет использования свойства `display` со значением `none`. Координаты его появления относительно текущего пункта меню определяются значениями атрибутов `top` и `left`. Обычно по вертикали выделенный пункт меню и его подпункты находятся на одном уровне (см. рис. 11.13), а по горизонтали подменю смещается вправо на ширину меню за вычетом толщины границы. В браузере Netscape за счет того, что ширина меню складывается из значения параметра `width` и полей вокруг текста, подменю немного накладывается на основное меню. Тем не менее это никак не портит впечатления и не влияет на работу меню. Но чтобы наложение происходило корректно, следует сделать две вещи. Во-первых, установить белый или другой цвет фона у пунктов меню. Поскольку фоновый цвет исходно прозрачный, то если это не сделать, через него будут просвечивать линии границы, что довольно некрасиво. И, во-вторых, использовать свойство `z-index`, которое определяет порядок наложения элементов друг на друга. В частности, значение для подменю должно быть больше, чем для меню.

Рамка вокруг пунктов добавляется к селектору `A`. Чтобы при этом не возникало сдвоенных линий в местах стыка пунктов, граница внизу убирается. Самые нижние пункты остаются в этом случае без линии, и для них специально создается новый класс с именем `brd`, который добавляет нижнюю границу. Можно применять этот класс к тегу `UL`, но Netscape отобразит в

этом случае линию с "дыркой". Так что наилучшим вариантом будет добавление класса `brd` к тегу `A`, как показано в данном примере.

Чтобы в меню различать пункты, у которых нет подменю, и пункты с подменю, логично добавить к таким пунктам небольшой рисунок, например треугольник-стрелку, как на рис. 11.13. Изображение добавляется как фоновое с помощью свойства `background`, применяемого к селектору `A`. В качестве значений указывается путь к графическому файлу и положение картинки. Так `right center` означает, что рисунок будет одновременно выравниваться по правому краю и по центру вертикали пункта. Отменить показ рисунка для отдельных пунктов можно указав `background-image: none` в теге `A`.

Скрытие и показ подменю происходит за счет использования псевдокласса `hover`, применяемого к селектору `LI`, и включения в стиль `display: block`. Это свойство отображает подменю, как только курсор мыши наводится на пункт, содержащий подменю. При выводе курсора за пределы области значение `display` возвращается в первоначальное, заданное как `none`.

Увы, но в браузерах Internet Explorer и Opera `hover` не работает для тега `LI`, из-за этого подменю никак не будет отображаться. Для таких браузеров требуется использование скрипта. С этой целью вводится еще один класс с именем `over`, для которого указываем `display: block`. Затем программа в цикле просматривает все элементы `LI`, которые находятся внутри нашего меню, и присваивает им при наведении на них курсора мыши класс `over`.

Листинг 11.15. Создание меню с вложенными подменю

```
<html>
<head>
<script language="JavaScript">
function startMenu() {

// Проверяем, поддерживает ли текущий браузер DOM
 if (document.getElementById) {

// Получаем объект с именем идентификатора menu
   nav = document.getElementById('menu');

// Пробегаемся по всем дочерним элементам нашего меню
   for (i=0; i<nav.childNodes.length; i++) {
     node = nav.childNodes[i];

// Если дочерний элемент - LI, то идем дальше
     if (node.nodeName == 'LI') {
```

```
// При наведении курсора на пункт меню, присваиваем LI класс over
       node.onmouseover = function() {
         this.className = 'over';
       }
       node.onmouseout = function() {

// При перемещении курсора за пределы пункта убираем класс over
         this.className = '';
       }
     }
   }
 }
}
</script>
<style type="text/css">
 UL {
  width: 180px;                      /* Ширина меню */
  list-style: none;                  /* Для списка убираем маркеры */
  margin: 0px;                       /* Нет отступов вокруг */
  padding: 0px;                      /* Убираем поля вокруг текста */
  font-family: Arial, sans-serif;    /* Рубленый шрифт для текста меню */
  font-size: 11pt                    /* Размер названий в пункте меню */
}
 UL LI  {
  position: relative          /* Подпункты позиционируются относительно */
}
 LI UL {
  position: absolute;         /* Подменю позиционируются абсолютно */
  display: none;              /* Скрываем подменю */
  top: 0px;                   /* По высоте положение подменю исходное */
  left: 179px;                /* Сдвигаем подменю вправо */
  z-index: 1                  /* Основное меню находится ниже подменю */
}
 LI A {
  display: block;             /* Ссылка как блочный элемент */
  width: 100%;                /* Ссылка на всю ширину пункта */
  padding: 5px;               /* Поля вокруг надписи */
  text-decoration: none;      /* Подчеркивание у ссылок убираем */
  background:                 /* Для основных пунктов отображаем рисунок */
       url(bullet.gif)        /* Указываем путь к рисунку со стрелкой */
       right center           /* Выравниваем рисунок по центру справа */
       no-repeat;             /* Отменяем повторение фона */
  color: #666;                /* Цвет текста */
  border: 1px solid #ccc;     /* Рамка вокруг пунктов меню */
```

```
  background-color: white;   /* Белый цвет фона */
  border-bottom: none        /* Границу снизу не проводим */
}
 LI UL LI A {
  background-image: none;    /* Для подпунктов рисунок убираем */
  z-index: 2                 /* Подменю располагаются выше меню */
}
 LI A:hover {
  color: maroon;                    /* Цвет текста активного пункта */
  background-color: #f0f0f0         /* Цвет фона активного пункта */
}
 LI:hover UL, LI.over UL {
  display: block             /* При выделении пункта курсором мыши
                             отображается подменю */
}
 .brd {
  border-bottom: 1px solid #ccc     /* Линия снизу */
}
</style>
</head>
<body onLoad="startMenu()">
<!--[if IE]>
<style type="text/css">
 UL LI {
  float: left         /* В браузере Internet Explorer пункты выводятся
                      без разрывов */
}
</style>
<![endif]-->

 <ul id=menu>
   <li><a href=russian.html>Русская кухня</a>
    <ul>
     <li><a href=linkr1.html>Бефстроганов</a></li>
     <li><a href=linkr2.html>Гусь с яблоками</a></li>
     <li><a href=linkr3.html>Крупеник новгородский</a></li>
     <li><a href=linkr4.html class=brd>Раки по-русски</a></li>
    </ul>
   </li>
   <li><a href=ukrainian.html>Украинская кухня</a>
    <ul>
     <li><a href=linku1.html>Вареники</a></li>
     <li><a href=linku2.html>Жаркое по-харьковски</a></li>
     <li><a href=linku3.html>Капустняк черниговский</a></li>
```

```
    <li><a href=linku4.html class=brd>Потапцы с помидорами</a></li>
   </ul>
  </li>
  <li><a href=caucasus.html>Кавказская кухня</a>
   <ul>
    <li><a href=linkc1.html>Суп-харчо</a></li>
    <li><a href=linkc2.html>Лилибдж</a></li>
    <li><a href=linkc3.html>Чихиртма</a></li>
    <li><a href=linkc4.html class=brd>Шашлык</a></li>
   </ul>
  </li>
  <li><a href=asia.html style="background-image: none" class=brd>Кухня
                                                    Средней Азии</a></li>
 </ul>
</body>
</html>
```

Приведенное меню не работает в браузере Opera 6 и ниже. В браузере Internet Explorer возникает ошибка, когда между пунктами меню появляются вертикальные разрывы. Чтобы избавиться от этого недостатка, достаточно добавить `float: left` к селектору `UL LI`. Но это же, в свою очередь, приведет к ошибке при отображении меню в браузере Netscape. Есть несколько способов решения, позволяющих получить одинаково работающий результат.

- Код меню следует писать без пробелов и переносов строк. Этот подход работает не всегда, к тому же становится неудобно редактировать текст, поэтому можно порекомендовать обращаться к этому способу лишь при необходимости.
- Использовать в стилях конструкцию `* HTML селектор`, которая понимается только браузером Internet Explorer. Таким образом, в контейнере `STYLE` следует написать следующие строки:

```
* HTML UL LI {
  float: left
}
```

- Воспользоваться условными комментариями. Любой текст в коде HTML можно закомментировать, и он при этом не будет выводиться на вебстранице. Для этого текст следует поместить между элементами `<!--` и `-->`. Internet Explorer поддерживает специальный синтаксис `<!--[if IE]>`, в задачу которого входит интерпретировать код, если перед нами Internet Explorer; остальные браузеры видят комментарий и не отображают его. Завершить все это требуется строкой `<![endif]-->`. Поскольку указанные

элементы можно писать только внутри контейнера BODY, то следует добавить тег STYLE, как показано далее.

```
<!--[if IE]>
<style type="text/css">
 UL LI {
  float: left
}
</style>
<![endif]-->
```

Следует отметить, что для других браузеров тоже существует вариант, когда им "подкладывают" стиль, отличный от стиля Internet Explorer. С этой целью применяется тег COMMENT, который добавляет комментарий. Хитрость состоит в том, что этот тег понимает только Internet Explorer, соответственно, все, что находится внутри этого контейнера, этим браузером не отображается. Остальные браузеры этот тег не понимают, поэтому они просто его игнорируют, но при этом содержимое этого тега нормально обрабатывается. Получается, что вначале мы указываем стиль для браузера Internet Explorer, а затем внутри тега COMMENT повторяем стиль для остальных браузеров (листинг 11.16). Когда идет несколько одинаковых описаний стиля для одного и того же селектора, то применяется тот стиль, который располагается в коде ниже других.

Листинг 11.16. Разный стиль для браузеров

```
<html>
<body>
<style type="text/css">
 DIV { background: green; color: white; padding: 4px }
</style>
<comment>
 <style type="text/css">
  DIV { background: silver; color: green }
 </style>
</comment>
 <div>Lorem ipsum dolor sit amet...</div>
</body>
</html>
```

Internet Explorer отобразит белые буквы на зеленом фоне, а остальные браузеры — зеленые символы на сером фоне.

Ниспадающее меню

Ниспадающее меню представляет собой разновидность горизонтального меню, но при наведении курсора мыши на пункт происходит открытие дополнительного списка, из которого выбираются подпункты (рис. 11.14). Ниспадающее меню давно уже применяется в различных операционных системах, поэтому принцип его действия понятен пользователю. Однако на сайтах подобное меню встречается не повсеместно, поэтому, глядя на список ссылок, сложно сразу сказать, какой перед нами тип меню — обычное или ниспадающее. Кроме того, к недочетам любого меню, которое первоначально скрывает часть данных, относится то, что выбор вариантов сразу не виден, и чтобы он стал доступен, требуется навести курсор на текст.

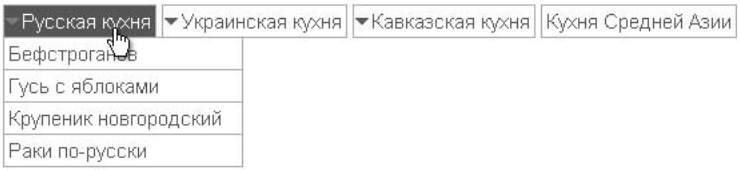


Рис. 11.14. Вариант представления ниспадающего меню

Как вариант, к ниспадающему меню можно отнести тег `SELECT`, когда в списке отображается только одна строка. Возможностей по оформлению такого меню практически нет, поэтому списки применяют для навигации не так часто.

Одним из вариантов создания ниспадающего меню является модификация листинга 11.15, предназначенного для получения всплывающих подменю. В самом деле, если пункты расположить по горизонтали, то мы получим желаемый результат. С этой целью требуется для селектора `LI` добавить свойство `float: left` (листинг 11.17).

Текст скрипта по сравнению с листингом 11.15 не поменяется, а в коде стиля будут небольшие изменения. В частности, чтобы выделить пункты меню, содержащие подменю, слева от текста добавляется небольшая стрелка вниз, как показано на рис. 11.14. С этой целью используется свойство `background` у селектора `A` со значением `left center`, оно помещает рисунок по левому краю пункта и по центру вертикали. Чтобы картинка не накладывалась на текст, необходимо добавить отступ слева через параметр `padding-left`. При этом для пункта без подменю надо не забыть убрать рисунок и отступ.

Для пунктов подменю (селектор `LI UL`) обязательно следует задать их ширину через атрибут `width`. При этом пункты становятся одной ширины и выстраиваются друг под другом. Иначе аккуратный строй ссылок может сломаться, а подменю рассыплется.

Листинг 11.17. Ниспадающее меню, созданное с помощью стилей

```
<html>
<head>
<script language="JavaScript">
См. листинг 11.15
</script>
<style type="text/css">
UL {
list-style: none;                    /* Для списка убираем маркеры */
margin: 0px;                         /* Нет отступов вокруг */
padding: 0px;                /* Убираем поля вокруг текста */
font-family: Arial, sans-serif;      /* Рубленый шрифт для текста меню */
font-size: 11pt                      /* Размер текста в пункте меню */
}
UL LI {
position: relative;          /* Подпункты позиционируются относительно */
background: white;           /* Цвет фона пунктов меню */
float: left;                 /* Пункты первого уровня
располагаются горизонтально */
padding: 4px 0px;            /* Поля вокруг пунктов первого уровня */
margin-right: 4px     /* Расстояние между пунктами по горизонтали */
}
UL LI UL LI {
padding: 0px                 /* Отменяем поля вокруг рамки
пунктов меню второго уровня */
}
LI UL {
position: absolute;          /* Подменю позиционируются абсолютно */
display: none;        /* Скрываем подменю */
width: 180px;         /* Ширина пунктов подменю */
margin-top: 3px              /* Сдвигаем подменю немного вниз */
}
* HTML LI UL {               /* Только для браузера Internet Explorer */
left: 0px;                   /* Положение подменю по горизонтали */
top: 21px                    /* Положение подменю по вертикали */
}
LI A {
padding: 3px;         /* Поля вокруг надписей основного меню */
padding-left: 12px;          /* Добавляем отступ слева для рисунка */
text-decoration: none;       /* Подчеркивание у ссылок убираем */
color: #666;                 /* Цвет текста */
border: 1px solid #ccc;      /* Рамка вокруг пунктов меню */
```

```
background:                    /* Для основных пунктов отображаем рисунок */
       url(bullet.gif)         /* Указываем путь к рисунку со стрелкой */
       left center             /* Выравниваем рисунок по центру слева */
       no-repeat               /* Отменяем повторение фона */
}
LI UL LI A {
display: block;                /* Отображаем ссылку как блочный элемент */
padding: 4px;                  /* Поля вокруг надписей подменю */
width: 180px;                  /* Ширина пунктов подменю */
border-bottom: none;           /* Границу снизу не проводим */
background-image: none         /* Для подпунктов рисунок убираем */
}
LI A:hover {
color: white;         /        * Цвет текста активного пункта */
background-color: #666         /* Цвет фона активного пункта */
}
LI:hover UL, LI.over UL {
display: block         /* При выделении пункта курсором мыши
                          открывается подменю */
}
.brd {
border-bottom: 1px solid #ccc        /* Добавляем линию снизу */
}
</style>
</head>
<body onLoad="startMenu()">
<ul id=menu>
 <li><a href=russian.html>Русская кухня</a><ul>
  <li><a href=linkr1.html>Бефстроганов</a></li>
  <li><a href=linkr2.html>Гусь с яблоками</a></li>
  <li><a href=linkr3.html>Крупеник новгородский</a></li>
  <li><a href=linkr4.html class=brd>Раки по-русски</a></li>
 </ul>
 </li>
 <li><a href=ukrainian.html>Украинская кухня</a><ul>
  <li><a href=linku1.html>Вареники</a></li>
  <li><a href=linku2.html>Жаркое по-харьковски</a></li>
  <li><a href=linku3.html>Капустняк черниговский</a></li>
  <li><a href=linku4.html class=brd>Потапцы с помидорами</a></li>
 </ul>
 </li>
 <li><a href=caucasus.html>Кавказская кухня</a><ul>
  <li><a href=linkc1.html>Суп-харчо</a></li>
```

```
  <li><a href=linkc2.html>Лилибдж</a></li>
  <li><a href=linkc3.html>Чихиртма</a></li>
  <li><a href=linkc4.html class=brd>Шашлык</a></li>
 </ul>
 </li>
 <li><a href=asia.html style="background-image: none; padding-left:
3px">Кухня Средней Азии</a></li>
</ul>
<br style="clear: left">
Продолжение следует...
</body>
</html>
```

При создании меню следует принимать во внимание наследование стиля элементов. Поскольку тег `UL` является родительским элементом по отношению к `LI`, то и стиль, который описан для `UL`, передается его потомкам, в частности тегу `LI`. С одной стороны, это упрощает написание стиля, ведь часть параметров можно не указывать, раз они применяются автоматически. С другой стороны, не все атрибуты нужны, поэтому их приходится повторять для дочерних элементов, но уже с другим значением. Так, в данном примере для добавления рисунка стрелки к пунктам меню первого уровня (селектор `LI A`) текст смещается вправо через атрибут `padding-left`. В то же время для пунктов второго уровня (селектор `LI UL LI A`) сдвиг текста не нужен, ведь у них изображение не используется. Поэтому требуется задать новое значение параметру `padding-left` или `padding`, как более универсальному.

В браузере Netscape замечен следующий недостаток. При наведении на текст меню первого уровня курсора мыши пункты, стоящие от него справа, самопроизвольно смещаются по горизонтали. Сдвиг происходит незначительный, но все равно воспринимается как небрежность. Чтобы ликвидировать этот недочет, следует убрать перенос строк и пробелы перед тегом `UL`. Обратите внимание, что в примере `UL` начинается сразу после закрывающего тега `A` (...`</a><ul>`).

Использование абсолютного позиционирования для подменю заставляет его располагаться ниже пункта основного меню, т. е. в том месте, где нам и нужно. Исключением является браузер Internet Explorer, который помещает подменю совсем не так. Надо присоединить атрибуты `left` и `top` со значениями `0` и `21px`. Последняя величина равна высоте пункта, зависит от текущих параметров меню и подбирается экспериментально. Чтобы параметры `left` и `top` не испортили картину в других браузерах, следует воспользоваться конструкцией `* HTML LI UL`, активной только в браузере Internet Explorer.

Плавающее меню

Большинство различных вариантов меню на веб-странице добавляются и управляются при помощи JavaScript. Для того чтобы иметь представление о технологии создания меню через скрипты, рассмотрим один из типов — *плавающее меню*. Такое меню всегда располагается поверх основного контента веб-страницы и при прокрутке не перемещается вместе с содержимым, а "скользит" относительно него, оставаясь при этом на исходном месте. Положение плавающего меню остается заданным относительно окна браузера.

Чтобы получать значения координат меню и менять их динамически, вначале следует создать слой и указать для него идентификатор с именем `menu`. Вид меню может быть произвольным и легко настраивается через стили. Как правило, в него добавляют ссылки на популярные и часто используемые разделы сайта, чтобы они всегда были под рукой.

Меню в окне браузера может занимать четыре положения: по горизонтали (слева или справа) и по вертикали (вверху или внизу). Главное, чтобы оно не мешало воспринимать информацию на странице, а помогало ориентироваться на сайте. Для этого вводятся две переменные: `verticalPos` и `horizontalPos`, которые задают исходную позицию меню. Когда меню располагается слева, то координаты задаются от левого верхнего угла окна браузера до левого верхнего угла слоя `menu` через стилевые свойства `left` и `top`. Для положения справа лучше воспользоваться атрибутом `right`, он определяет расстояние от правого края окна браузера до правого края слоя (листинг 11.18). По вертикали отсчет ведется от верхнего края браузера, и для размещения слоя внизу окна требуется вначале вычислить текущую высоту окна и отнять от нее заданную координату. Поскольку отсчет идет от верхнего угла слоя, то в этом случае необходимо задавать большее значение переменной `startY`, чем при расположении слоя вверху.

Скрипт работает следующим образом. Встроенная функция `setTimeout` вызывает пользовательскую функцию `slideMenu` через определенный промежуток времени, равный 10 миллисекундам. Сама функция `slideMenu` задает новые координаты слоя с учетом высоты рабочей области окна (метод `clientHeight`) и смещения документа при его прокрутке (метод `scrollTop`). Использование свойства `scrollTop` необходимо, чтобы отслеживать прокручивание содержимого и добиться плавности движения меню.

Листинг 11.18. Создание плавающего меню

```
<html>
<head>
<script language="JavaScript">
```

```
function floatMenu() {
// Указывает, где находится меню по вертикали
// Возможные варианты - top и bottom
 verticalPos = "top"; // Меню находится вверху

// Указывает, где находится меню по горизонтали
// Возможные варианты - left и right
 horizontalPos = "right"; // Меню располагается справа
 startX = 10;  // Начальная координата по горизонтали
 startY = 50;   // Начальная координата по вертикали

 function coordMenu(id) {
// Обращаемся к нашему меню по его id через getElementById
  el = document.getElementById(id);

// Получаем координаты меню
  el.currentPosition = function(x, y) {
    if (horizontalPos == "left") this.style.left = x;
    else  this.style.right = x;
    this.style.top = y;
  }

// Положение по горизонтали остается неизменным
 el.x = startX;

// Если меню расположено вверху окна, то вертикальная
// текущая координата не меняется
  if (verticalPos == "top") el.y = startY;

// Если меню расположено внизу окна, то из высоты окна вычитаем
// исходную вертикальную координату меню
  else el.y = document.body.clientHeight - startY;
  return el;
 }
 slideMenu = function()      {
// Число 20 в знаменателе определяет плавность хода,
// чем оно больше, тем медленнее движется меню
  if (verticalPos == "top")
    obj.y += (document.body.scrollTop + startY - obj.y)/20;
  else
    obj.y += (document.body.clientHeight + document.body.scrollTop - ↳
startY - obj.y)/20;

// Сохраняем текущие координаты меню
  obj.currentPosition(obj.x, obj.y);
```

```
// Вызываем функцию slideMenu каждые 10 миллисекунд
  setTimeout("slideMenu()", 10);
 }
 obj = coordMenu("menu");
 slideMenu();
}
</script>
<style type="text/css">
 #menu {
  width: 150px;                    /* Ширина меню */
  border: 1px solid navy;          /* Рамка вокруг меню */
  background: #ffffee;             /* Цвет фона */
  position: absolute               /* Абсолютное позиционирование */
}
 .title {
  background: navy;                /* Цвет фона под надписью */
  color: white;                    /* Цвет заголовка */
  font-family: Arial, sans-serif;  /* Рубленый шрифт */
  font-weight: bold;               /* Жирное начертание */
  font-size: 80%                   /* Размер текста заголовка */
}
 .content A {
  border-bottom: 1px solid silver;  /* Линии между ссылками */
  padding-bottom: 4px;       /* Расстояние от ссылки до линии под ней */
  display: block                   /* Ссылка на всю ширину меню */
}
 .title, .content {
  padding: 4px               /* Поля вокруг ссылок и заголовка */
}
</style>
</head>
<body onLoad="floatMenu()">
 <div id=menu>
  <div class=title>Навигация по сайту</div>
  <div class=content>
    <a href=link1.html>Домой</a>
    <a href=link2.html>Статьи</a>
    <a href=link3.html>Форум</a>
    <a href=link4.html>Помощь</a>
  </div>
 </div>
<!-- Таблица для создания вертикальной полосы прокрутки -->
 <table height=1000>
```

```
  <tr><td> </td></tr>
 </table>
</body>
</html>
```

Для наглядности работы меню добавлена невидимая таблица с большой высотой, она необходима для появления вертикальной полосы прокрутки. В обычных случаях при наличии содержания необходимости в таблице нет.

ПРИЛОЖЕНИЕ

Свойства CSS

Любую задачу по верстке веб-страниц можно решить двумя путями: посмотреть, как это сделано на другом сайте, модернизировав затем код HTML под свои нужды, и воспользоваться исходными возможностями CSS. И в том и в другом случае базовыми кирпичиками выступают свойства CSS, с помощью которых выстраивается требуемый вид сайта. Чтобы изначально понимать, что именно можно создать на веб-странице, нужно знать стилевые атрибуты, их значения, а также особенности их поведения в разных браузерах. В данном *приложении* приводится описание основных стилевых свойств с примерами их использования.

Пояснения

Описание каждого атрибута происходит по одному шаблону: содержит таблицу браузеров с номерами версий, которые поддерживают указанное свойство, синтаксис написания, допустимые значения параметра, наследование и к каким элементам он применяется. Каждый параметр иллюстрируется примером, который показывает область его использования. В большинстве случаев приводится рисунок, который демонстрирует результат примера, что позволяет наглядно понять, как действует тот или иной параметр.

Браузеры

Хотя большинство свойств CSS описаны достаточно давно, разработчики браузеров не всегда включали их поддержку в браузер или делали это не в полном объеме. По этой причине часто возникала ситуация, когда стандарты нельзя было применять, потому что они просто не работали. К счастью, эта ситуация исправляется, и современные браузеры поддерживают спецификацию CSS практически в полном объеме, хотя еще и не до конца. Тем не менее при создании универсальных веб-страниц, которые корректно отображаются в разных браузерах, необходимо знать, какие параметры и где будут работать, а какие нет.

Для этого в таблице к каждому параметру приведены три популярных браузера — Internet Explorer, Netscape (синонимы: Netscape Navigator и Netscape Communicator) и Opera. Поклонникам браузеров Mozilla и Firefox следует знать, что они, как и Netscape 6, основаны на одном ядре Gecko, а следовательно, свойства, которые описаны для Netscape версии 6 и 7, работают и в этих браузерах. В частности Firefox 1.0 поддерживает те же параметры CSS, что и Netscape 7, а Mozilla в зависимости от версии — те же, что и Netscape 6 или Netscape 7.

В таблице браузеров используются следующие обозначения:

- *да* — параметр полностью поддерживается браузером со всеми допустимыми аргументами;
- *нет* — атрибут браузером не воспринимается и игнорируется;
- *частично* — параметр поддерживается лишь частично, например, не все допустимые аргументы действуют или атрибут применяется не ко всем элементам, которые указаны в спецификации;
- *ошибки* — свойство понимается браузером, но при его работе возможно появление различных ошибок. В примечании к описанию свойства обычно указывается, какого рода ошибки обнаруживаются в браузере.

Синтаксис

Каждое свойство CSS записывается в следующем обобщенном виде:

```
Селектор { свойство1: аргументы; свойство1: аргументы }
```

Селектором называют имя стиля, в котором указаны параметры форматирования; делятся они на несколько типов: селекторы тегов, классы и идентификаторы (ID).

Селекторы тегов используются для определения стилей встроенных тегов HTML. Классы применяются для создания стилей, которые можно применять к любому тегу HTML, для создания выделений или изменения стиля блока текста. Селекторы ID обычно используются совместно со скриптами, чтобы можно было управлять параметрами стиля динамически.

Далее в фигурных скобках указывают имя параметра CSS и через двоеточие его желаемое значение. Несколько параметров можно перечислить через двоеточие или задать отдельно каждый параметр, как показано далее:

```
P { color: green; background: #f0f0f0 }
P { color: green } P { background: #f0f0f0 }
```

В первой строке для селектора `P` одновременно устанавливается цвет текста и фона, а во второй — вначале задается цвет текста, а затем уже цвет фона. Поскольку селектор указан один, а свойства разные, то они будут приме-

няться одновременно. Поэтому приведенные формы записи параметров дают один результат.

Если для одного селектора определяются одинаковые атрибуты с разными значениями, то использоваться будет тот, который указан в коде последним.

```
P { color: green } P { color: red }
```

В строке показано изменение цвета фона у параграфа `P`. Вначале задается зеленый цвет, а затем красный, который и будет применен к тексту.

Любые свойства CSS, а также их значения не чувствительны к регистру, поэтому их можно набирать как прописными, так и строчными символами. Но при этом традиционно их всегда пишут маленькими буквами.

При описании синтаксиса применяются следующие обозначения:

- вертикальная черта между значениями (например `fixed | scroll`) указывает на логическое исключающее ИЛИ. Это означает, что надо выбрать только один аргумент из предложенных;
- двойная вертикальная черта (`border-style || color`) обозначает объединяющее ИЛИ (ИЛИ/И). Каждый аргумент может использоваться самостоятельно или совместно с другими через пробел;
- квадратные скобки (`[left | center | right]`) помечают группу, из которой, как правило, выбирается одно значение;
- два числа в фигурных скобках (`{a,b}`) говорят о том, что предшествующее им значение следует повторять не менее `a`, но не более `b` раз.

Значение по умолчанию

Если какой-то параметр в стиле не приводится явно, то браузер, тем не менее, самостоятельно применяет его со значением, которое установлено по умолчанию. Подобные аргументы не всегда являются оптимальными, поэтому их можно переназначить, если напрямую указать свойство с желаемым значением.

Наследование

Наследование — это перенос правил форматирования для элементов, находящихся внутри других. Например, если для параграфа `P` задан красный цвет текста, а для курсива `I`, который находится внутри параграфа, нет, то в этом случае вложенный элемент наследует свойства своего родителя и курсивный текст также будет красным.

Наследование полезно для задания свойств, применяемых к элементу по умолчанию. Так, достаточно задать параметры форматирования тега `TABLE`, и к ячейкам таблицы `TD` они будут применены автоматически. Точно так же можно определить свойства тега `BODY`, который порождает стиль всех остальных элементов веб-страницы.

Существует два варианта применения наследования. Если свойства элемента наследуются, то для его дочернего элемента их можно не указывать, кроме случая, когда использование параметра нежелательно. Наоборот, если свойства элемента не наследуются, то для дочернего элемента их требуется указывать снова или пропустить, когда они не нужны.

Применение

Свойства CSS можно применять далеко не ко всем элементам веб-страницы подряд, а только к тем, с которыми они "дружат". Большинство атрибутов работают со всеми элементами, а некоторые только с блочными или позиционированными. Хотя и указано, что определенные свойства могут применяться ко всем элементам, это не всегда приводит к какому-либо результату. Например, для изображений совершенно бессмысленно устанавливать атрибуты, которые манипулируют с текстом. Так что в каждом случае следует решать самостоятельно, когда применять параметр, а когда нет.

Параметр *background*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Параметр позволяет установить одновременно до пяти атрибутов стиля фона. Значения могут идти в любом порядке, браузер сам определит, какое из них соответствует нужному атрибуту. Некоторые параметры не поддерживаются браузером Netscape 4.x. Для более подробного ознакомления с аргументами смотрите свойства каждого параметра отдельно.

Синтаксис

```
background: background-attachment || background-color || background-image
|| background-position || background-repeat
```

Аргументы

Любые комбинации пяти значений, определяющих стиль фона, в произвольном порядке. Значения разделяются между собой пробелом. Ни один аргумент не является обязательным, поэтому неиспользуемые значения можно опустить.

Значение по умолчанию

Нет.

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П1

```
<html>
<head>
<style type="text/css">
 BODY { background: url(bg.gif) repeat fixed }
 P { background: #c0c0c0; padding: 4px }
</style>
</head>
<body>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.</p>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Браузер Netscape 4.x работает некорректно с отображением фона. Если вокруг элемента нет никакой границы, то фон будет только под текстом, а не у всей области. Даже если граница добавлена, между ней и областью будет небольшой промежуток, избавиться от которого нет возможности.

Параметр *background-attachment*

Браузер	**Internet Explorer**						**Netscape Navigator**			**Opera**		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Нет	Да	Да	Да	Да	Да

Описание

Параметр `background-attachment` устанавливает, будет ли фоновое изображение прокручиваться вместе с содержимым элемента. Изображение может быть зафиксировано на одном месте и оставаться неподвижным или перемещаться совместно с документом.

Синтаксис

```
background-attachment: fixed | scroll
```

Аргументы

Значение `fixed` делает фоновое изображение элемента неподвижным, `scroll` — позволяет перемещаться фону вместе с содержимым.

Значение по умолчанию

```
scroll
```

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П2

```
<html>
<head>
<style type="text/css">
 BODY {
  background-image: url(bg.gif);
  background-attachment: fixed
}
</style>
</head>
<body>
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Хотя это свойство применяется ко всем элементам, в браузере Internet Explorer реальный результат будет заметен только для селектора `BODY`.

Параметр *background-color*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Устанавливает фоновый цвет элемента. Хотя этот параметр не наследует свойства своего родителя из-за того, что начальное значение цвета фона устанавливается прозрачным, он совпадает с фоном текущего элемента.

Синтаксис

```
background-color: цвет
```

Аргументы

Значение цвета может задаваться по его названию, шестнадцатеричному значению или с помощью RGB (см. параметр `color`). Кроме значения цвета, есть еще один допустимый аргумент — `transparent`, устанавливающий прозрачный фон.

Значение по умолчанию

```
transparent
```

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П3

```
<html>
<head>
<style type="text/css">
 BODY { background-color: #3366cc }
 P { background-color: yellow }
```

```
 SPAN { background-color: #98560f; color: white }
 .select { background-color: rgb(249, 231, 16) }
</style>
</head>
<body>
<p><span>Lorem ipsum dolor sit amet</span>, consectetuer adipiscing elit,
sed diem nonummy nibh euismod tincidunt ut lacreet dolore <span
class=select>magna aliguam erat volutpat.</span></p>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Браузер Netscape 4.x не применяет цвет фона к блоку текста и отступам вокруг него, а только к тексту в элементе. Чтобы избавиться от этого недостатка, можно сделать вокруг области границу нулевой толщины. Начиная с версии 5.5 в браузере Internet Explorer значение `transparent` можно применять к фреймам и плавающим фреймам.

Параметр *background-image*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Частично	Да	Да	Да	Да	Да

Описание

Устанавливает фоновое изображение для элемента. Если одновременно для элемента задан цвет фона, он будет показан, пока фоновая картинка не загрузится полностью. То же произойдет, если изображение не доступно или отключен показ рисунков в браузере. В случае наличия в рисунке прозрачных областей через них будет проглядывать фоновый цвет.

Синтаксис

```
background-image: url(путь к файлу) | none
```

Аргументы

В качестве значения используется путь к графическому файлу, который указывается внутри конструкции `url()`. Либо аргумент может принимать значение `none`, когда фоновое изображение не требуется.

Значение по умолчанию

none

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П4

```
<html>
<head>
<style type="text/css">
  BODY { background-image: url(images/bg.gif) }
</style>
</head>
<body>
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

В браузере Netscape 4.x свойство не применяется к элементам форм, а также к тегам IMG, HR и TABLE.

Параметр *background-position*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Нет	Да	Да	Да	Да	Да

Описание

Задает положение левого верхнего угла фонового изображения, установленного с помощью параметра background-image.

Синтаксис

```
background-position: [проценты | значение] | [left | center | right] ||
[top | center | bottom]
```

Аргументы

У этого параметра два значения: положение по горизонтали (`left`, `center`, `right`) и по вертикали (`top`, `center`, `bottom`). Положение можно также задавать в процентах, пикселах или других единицах.

Значение по умолчанию

`0%`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П5

```
<html>
<head>
<style type="text/css">
 BODY { background-image: url(mybg.gif); background-position: right top
background-repeat: no-repeat }
</style>
</head>
<body>
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</body>
</html>
```

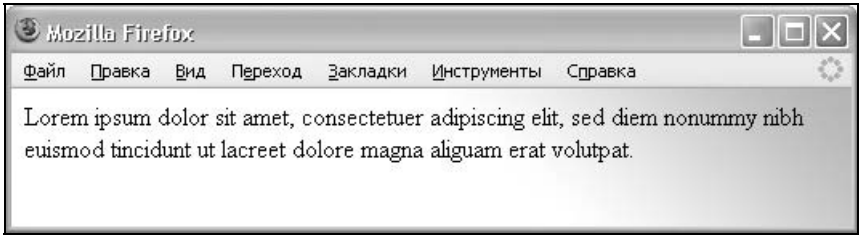


Рис. П1. Положение фонового рисунка на веб-странице

Результат этого примера показан на рис. П1. Обратите внимание на правый верхний угол окна браузера, где видно затенение фона. Это используется рисунок `mybg.gif`, упомянутый в примере.

Применяется

Ко всем элементам.

Параметр *background-repeat*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Ошибки	Да	Да	Да	Ошибки	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Определяет, как будет повторяться фоновое изображение, установленное с помощью параметра `background-image`, и по какой оси. Можно установить повторение рисунка только по горизонтали, по вертикали или в обе стороны.

Синтаксис

```
background-repeat: no-repeat | repeat | repeat-x | repeat-y
```

Аргументы

Значение `no-repeat` устанавливает одно фоновое изображение в элементе без его повторений, положение которого определяется атрибутом `background-position` (по умолчанию в левом верхнем углу). Другими допустимыми значениями являются `repeat` (повторяемость по вертикали и горизонтали), `repeat-x` (по горизонтали), `repeat-y` (по вертикали).

Значение по умолчанию

`repeat`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П6

```
<html>
<head>
<style type="text/css">
 BODY { background-image: url(mybg.gif); background-repeat: repeat-x }
</style>
</head>
<body>
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Браузер Internet Explorer 4 повторяет фоновый рисунок вниз и вправо. Более правильное поведение для изображения фона — повторение в обоих направлениях по горизонтали (для параметра `repeat-x`) или по вертикали (параметр `repeat-y`). Поскольку браузер Netscape 4.x не поддерживает свойство `background-position`, то вывод фонового рисунка всегда начинается с левого верхнего угла элемента. Кроме того, в этом браузере имеются проблемы с отображением фона при использовании атрибута `background-repeat`.

Параметр *border*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Частично	Частично	Да	Да	Частично	Да	Да	Да	Да	Да	Да	Да

Описание

Параметр позволяет одновременно установить толщину, стиль и цвет рамки вокруг элемента. Значения могут идти в любом порядке, разделяясь пробелом; браузер сам определит, какое из них соответствует нужному атрибуту.

Для установки рамки только на определенных сторонах элемента воспользуйтесь параметрами `border-top`, `border-bottom`, `border-left`, `border-right`.

Синтаксис

```
border: border-width || border-style || color
```

Аргументы

Значение `border-width` определяет толщину рамки. Для управления видом рамки предоставляется восемь значений параметра `border-style`. Их названия и результат действия приведены в табл. П2.

Первые два стиля — `dotted` и `dashed` — поддерживаются браузером Netscape начиная с шестой версии, а браузером Internet Explorer с версии 5.5.

Аргумент `color` устанавливает цвет рамки, значение может быть в любом допустимом для CSS формате.

Значение по умолчанию

Нет.

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П7

```
<html>
<body>
<div style="border: 2px solid black; background: #fc3; padding: 10px">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</div>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Internet Explorer 4/5 не применяет свойство `border` к встроенным элементам.

Параметр *border-bottom*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Частично	Частично	Да	Да	Частично	Да	Нет	Да	Да	Да	Да	Да

Описание

Параметр позволяет одновременно установить толщину, стиль и цвет границы внизу элемента. Значения могут идти в любом порядке, разделяясь пробелом; браузер сам определит, какое из них соответствует нужному атрибуту. Аналогично этому атрибуту характеристики границы устанавливаются на верхней, левой и правой стороне с помощью параметров `border-top`, `border-left` и `border-right` соответственно.

Синтаксис

```
border-bottom: border-width || border-style || color
```

Аргументы

Значение `border-width` определяет толщину рамки. Для управления видом рамки предоставляется восемь значений параметра `border-style`. Их названия и результат действия приведены в табл. П2. Аргумент `color` устанавливает цвет рамки, значение может быть в любом допустимом для CSS формате.

Значение по умолчанию

Нет.

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П8

```
<html>
<body>
<div style="border-bottom: 2px solid black; border-right: 2px solid
black; background: #fc3; padding: 10px">
```

```
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</div>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

В браузере Internet Explorer 4/5 для встроенных элементов атрибут не работает.

Параметр *border-bottom-color*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Частично	Частично	Да	Да	Частично	Да	Нет	Да	Да	Частично	Да	Да

Описание

Устанавливает цвет границы внизу элемента. Изменить цвет на других сторонах можно через атрибуты `border-top-color`, `border-left-color` и `border-right-color`, которые определяют цвет верхней, левой и правой границы.

Синтаксис

```
border-bottom-color: цвет
```

Аргументы

Значение цвета может задаваться по его названию, шестнадцатеричному значению, либо с помощью RGB (см. параметр `color`).

Значение по умолчанию

Нет.

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П9

```
<html>
<body>
<p style="border-bottom-color: #ccc; border-right-color: #ccc; border-
style: solid; padding: 4px">Lorem ipsum dolor sit amet, consectetuer
adipiscing elit. Sed diem nonummy nibh euismod tincidunt ut lacreet
dolore magna aliguam erat volutpat.</p>
</body>
</html>
```

Результат этого примера показан на рис. П2. С помощью стилей устанавливается серая граница слева и снизу блока.

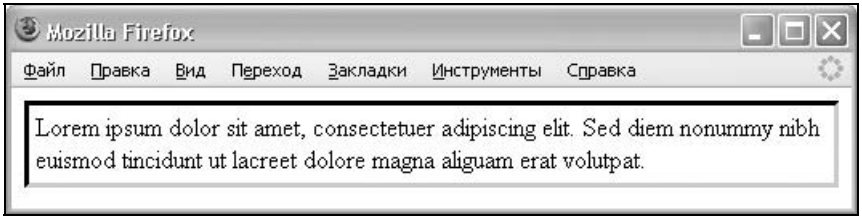


Рис. П2. Изменение цвета сторон границы

Применяется

Ко всем элементам.

Примечание

В Internet Explorer 4/5 цвет рамки не устанавливается для встроенных элементов. В браузере Opera 3.5 атрибут не дает эффекта для изображений, таблиц и их ячеек, а также для элементов форм.

Параметр *border-bottom-style*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Част.*	Част.	Да	Да	Част.	Да	Част.	Да	Да	Част.	Да	Да

* Част. — поддерживается частично.

Описание

Устанавливает стиль границы внизу элемента. Стиль других сторон определяется параметрами `border-top-style`, `border-left-style` и `border-right-style`.

Синтаксис

```
border-bottom-style: стиль
```

Аргументы

Для управления видом рамки предоставляется восемь значений параметра `border-style`. Их названия и результат действия приведены в табл. П2.

Значение по умолчанию

Нет.

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П10

```
<html>
<head>
<style type="text/css">
 A { border-style: outset; background: #fc0;
  border-top-color: #ffea95; border-left-color: #ffea95 }
 A:hover { border-style: inset; border-bottom-color: #ffea95;
  border-right-color: #ffea95 }
</style>
</head>
<body>
<a href=link.html>Lorem ipsum dolor sit amet</a>
</body>
</html>
```

Результат данного примера показан на рис. П3. Вокруг ссылки отображается рамка, в которой используется стиль `outset`. За счет добавления псевдокласса `hover` при наведении на ссылку курсора мыши стиль рамки меняется на `inset`. Также модифицируются и некоторые цвета границ.

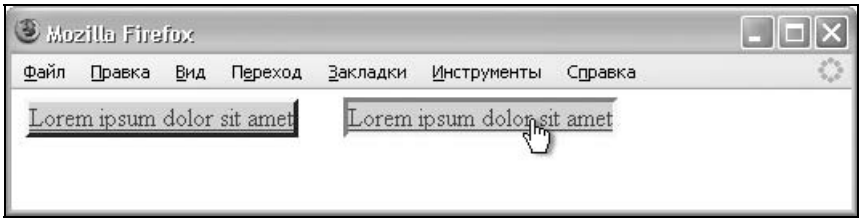


Рис. П3. Изменение стиля границы при наведении курсора мыши

Применяется

Ко всем элементам.

Параметр *border-bottom-width*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Частично	Частично	Да	Да	Частично	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Устанавливает толщину границы внизу элемента. Для изменения толщины границы на других сторонах используйте свойства `border-top-width`, `border-left-width` и `border-right-width` или универсальный атрибут `border-width`.

Синтаксис

```
border-bottom-width: thin | medium | thick | значение
```

Аргументы

Три переменные — `thin` (2 пиксела), `medium` (4 пиксела) и `thick` (6 пикселов) задают толщину границы внизу. Для более точного значения толщину можно указывать в пикселах или других единицах.

Значение по умолчанию

`medium`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П11

```
<html>
<body>
<h1 style="border-color: silver; border-style: double; border-bottom-width:
7px; border-right-width: 7px">Lorem ipsum dolor sit amet</h1>
...
</body>
</html>
```

Результат данного примера показан на рис. П4. Для стиля рамки `double` рекомендуется указывать толщину не меньше 2 пикселов, иначе двойные линии станут сливаться друг с другом.

Рис. П4. Применение границ разной толщины к заголовку страницы

Применяется

Ко всем элементам.

Примечание

Если стиль границы не установлен с помощью параметра `border-style`, Netscape 4.x все равно создает видимые границы. При установке стиля границы они отрисовываются не на всех сторонах. Internet Explorer 4/5 правильно работает для блочных элементов, но встроенные элементы полностью игнорирует.

Параметр *border-collapse*

Браузер	**Internet Explorer**						**Netscape Navigator**			**Opera**		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Нет	Да	Да	Да	Нет	Нет	Нет	Нет	Да	Нет	Ошибки	Ошибки

Описание

Устанавливает, как отображать границы вокруг ячеек таблицы. Этот параметр играет роль, когда для ячеек установлена рамка и в месте стыка ячеек получается линия двойной толщины. Добавление значения `collapse` заставляет браузер анализировать подобные места в таблице и убирать двойные линии. При этом между ячейками остается только одна граница, одновременно принадлежащая обеим ячейкам. То же правило соблюдается и для внешних границ, когда вокруг самой таблицы добавляется рамка.

Синтаксис

```
border-collapse: collapse | separate
```

Аргументы

- `collapse` — линия между ячейками отображается как одна;
- `separate` — вокруг каждой ячейки отображается своя собственная рамка, в местах соприкосновения ячеек отображаются сразу две линии.

Значение по умолчанию

```
separate
```

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П12

```
<html>
<head>
<style type="text/css">
 TABLE { width: 300px; border: 4px double black;
  border-collapse: collapse }
 TH { text-align: left; background: #ccc; padding: 5px;
  border: 1px solid black }
 TD { padding: 5px; border: 1px solid black }
</style>
</head>
<body>
 <table cellspacing=0>
  <tr><th>Струна</th><th>Нота</th></tr>
  <tr><td>1</td><td>D</td></tr>
  <tr><td>2</td><td>G</td></tr>
```

```
  <tr><td>3</td><td>C</td></tr>
 </table>
</body>
</html>
```

Результат данного примера представлен на рис. П5. Заметьте, что все линии вокруг ячеек отображаются толщиной один пиксел.

Рис. П5. Вид таблицы при использовании свойства `border-collapse`

Применяется

К тегу `TABLE`.

Примечание

Браузер Opera иногда проявляет некоторые причуды в отображении границ. Например, линии могут появляться в том месте, где их по соображениям разработчиков быть не должно.

Параметр *border-color*

Браузер	**Internet Explorer**						**Netscape Navigator**			**Opera**		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Част.*	Част.	Да	Да	Част.	Да	Част.	Да	Да	Част.	Да	Да

* Част. — поддерживается частично.

Описание

Устанавливает цвет границы на разных сторонах элемента. Параметр позволяет задать цвет границы сразу на всех сторонах элемента или определить его только для указанных сторон.

Синтаксис

```
border-color: color {1,4}
```

Аргументы

Разрешается использовать одно, два, три или четыре значения, разделяя их пробелом. Эффект зависит от количества аргументов и приведен в табл. П1.

Таблица П1. *Зависимость параметра* `border-color` *от числа аргументов*

Число аргументов	Результат
1	Цвет рамки будет установлен для всех сторон элемента
2	Первый аргумент устанавливает цвет верхней и нижней границы, второй аргумент — левой и правой
3	Первый аргумент задает цвет верхней границы, второй — одновременно левой и правой границы, а третий — нижней границы
4	Поочередно устанавливается цвет верхней, правой, нижней и левой границы

Значение по умолчанию

Нет.

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П13

```
<html>
<head>
<style type="text/css">
 P { border-color: red rgb(0,0,255) red; border-style: dashed;
border-width: thin; padding: 4px }
 DIV { border-color: #008a77 #008a77 #00b9a0 #00b9a0; border-style:
solid; padding: 3px }
</style>
</head>
<body>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit.</p>
```

```
<div>Sed diem nonummy nibh euismod tincidunt ut lacreet dolore magna
aliguam erat volutpat.</div>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Netscape 4.x и Opera 3.6 не могут устанавливать разные цвета границ для индивидуальных сторон элемента, как например `border-color: red green blue orange`. Internet Explorer 4/5 не может применять цвета границ к встроенным элементам.

Параметр *border-style*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Част.	Част.	Да	Да	Част.	Да	Част.	Да	Да	Да	Да	Да

* Част. — поддерживается частично.

Описание

Устанавливает стиль рамки вокруг элемента. Браузер Netscape 4 понимает только одно значение, которое определяет стиль сразу для всех границ. Для остальных браузеров допустимо устанавливать индивидуальные стили для разных сторон элемента.

Синтаксис

```
border-style: стиль
```

Аргументы

Для управления видом рамки предоставляется восемь значений параметра `border-style`. Их названия, вид и поддержка разными браузерами приведены в табл. П2.

Разрешается использовать одно, два, три или четыре значения, разделяя их пробелом. Результат зависит от количества аргументов и приведен в табл. П3.

Таблица П2. *Тип стиля границы элемента*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
dashed	–	–	+	+	–	+	–	+	+	+	+	+
dotted	–	–	+	+	–	+	–	+	+	+	+	+
double	+	+	+	+	+	+	+	+	+	+	+	+
groove	+	+	+	+	+	+	+	+	+	+	+	+
inset	+	+	+	+	+	+	+	+	+	+	+	+
outset	+	+	+	+	+	+	+	+	+	+	+	+
ridge	+	+	+	+	+	+	+	+	+	+	+	+
solid	+	+	+	+	+	+	+	+	+	+	+	+

Плюс означает, что этот вид рамки корректно отображается браузером, минус — аргумент не поддерживается и рамка отображается сплошной.

Таблица П3. *Зависимость параметра* `border-style` *от числа аргументов*

Число аргументов	Результат
1	Стиль рамки будет задан для всех сторон элемента
2	Первый аргумент устанавливает стиль верхней и нижней границы, второй аргумент — левой и правой
3	Первый аргумент задает стиль верхней границы, второй — одновременно левой и правой границы, а третий — нижней границы
4	Поочередно устанавливается стиль верхней, правой, нижней и левой границы

Значение по умолчанию

Нет.

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П14

```
<html>
<head>
<style type="text/css">
 P { border-style: solid double double solid; border-width: 2px 5px 5px
2px; padding: 5px }
</style>
</head>
<body>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.</p>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Стиль границы в браузере Internet Explorer не применяется к встроенным элементам. Использование параметра `border-style: double` совместно с `border-width: thin` устанавливает сплошную границу, а не двойную. В браузере Opera 3.5 нельзя установить стиль рамки для изображений, таблиц, ячеек и элементов форм.

Параметр *border-width*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Задает толщину границы одновременно на всех сторонах элемента или индивидуально для каждой стороны. Способ изменения толщины зависит от числа аргументов.

Синтаксис

```
border-width: thin | medium | thick | значение
```

Аргументы

Три переменные — `thin` (2 пиксела), `medium` (4 пиксела) и `thick` (6 пикселов) — задают толщину границы. Для более точного значения толщину можно указывать в пикселах или других единицах. Разрешается использовать одно, два, три или четыре значения, разделяя их пробелом. Эффект зависит от количества аргументов и приведен в табл. П4.

Таблица П4. *Зависимость результата использования толщины границы от числа аргументов параметра* `border-width`

Число аргументов	Результат
1	Толщина рамки будет установлена для всех сторон элемента
2	Первый аргумент устанавливает толщину верхней и нижней границы, второй аргумент — левой и правой
3	Первый аргумент задает толщину верхней границы, второй — одновременно левой и правой границы, а третий — нижней границы
4	Поочередно устанавливает толщину верхней, правой, нижней и левой границы

Значение по умолчанию

`medium`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П15

```
<html>
<head>
<style type="text/css">
 P { border-style: double; border-width: 3px 7px 7px 4px; padding: 7px }
</style>
</head>
<body>
```

```
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.</p>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Если стиль границы не установлен с помощью параметра `border-style`, Netscape 4.x все равно создает видимые границы. При установке стиля границы они отрисовываются не на всех сторонах. В Internet Explorer 4/5 этот параметр правильно работает для блочных элементов, но встроенные элементы игнорирует полностью.

Параметр *bottom*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Нет	Да	Да	Да	Нет	Да	Нет	Да	Да	Нет	Да	Да

Описание

Устанавливает положение нижнего края содержимого элемента без учета толщины рамок и отступов. Отсчет координат зависит от параметра `position`, он обычно принимает значение `relative` (относительное положение) или `absolute` (абсолютное положение).

При относительном позиционировании элемента отсчет ведется от нижнего края родительского элемента (рис. П6), а при абсолютном — относительно нижнего края окна документа (рис. П7).

Синтаксис

```
bottom: значение | проценты | auto
```

Аргументы

В качестве значений принимаются любые единицы длины, принятые в CSS, — например пикселы (px), дюймы (in), пункты (pt) и др. Значение параметра `bottom` может быть и отрицательным, в этом случае вероятны

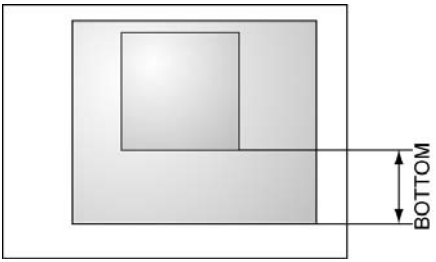


Рис. П6. Значение параметра `bottom` при относительном позиционировании элемента

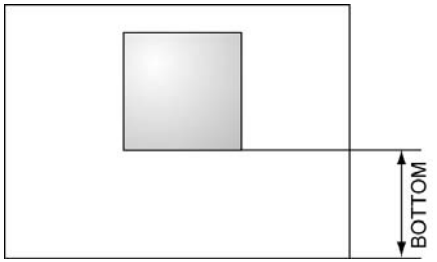


Рис. П7. Значение параметра `bottom` при абсолютном позиционировании элемента

наложения разных элементов друг на друга. При задании значения в процентах положение элемента вычисляется в зависимости от высоты родительского элемента. Аргумент `auto` не изменяет положение элемента.

Значение по умолчанию

`auto`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П16

```
<html>
<body>
<p>Duis te feugifacilisi.</p>
<div style="bottom: 20px; position: absolute; width: 90%; background:
#f0f0f0; padding: 10px; border: solid 1px black">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</div>
</body>
</html>
```

Результат данного примера представлен на рис. П8. Поскольку слой позиционирован абсолютно, то он не влияет на вывод остальных элементов вебстраницы и всегда отображается внизу окна браузера.

Применяется

Ко всем элементам.

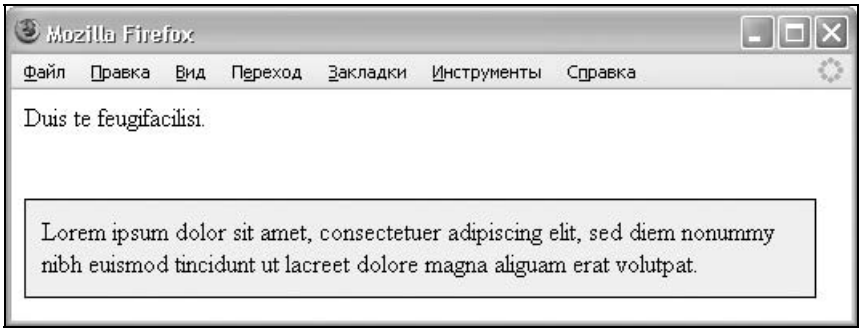


Рис. П8. Размещение слоя внизу веб-страницы

Параметр *clear*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Ошибки	Да	Да	Да	Ошибки	Да	Ошибки	Да	Да	Ошибки	Да	Да

Описание

Параметр устанавливает, с какой стороны элемента запрещено его обтекание другими элементами. Если установлено обтекание элемента с помощью параметра `float`, свойство `clear` отменяет его действие для указанных сторон.

Синтаксис

```
clear: both | left | none | right
```

Аргументы

- `both` — отменяет обтекание элемента одновременно с правого и левого края. Этот аргумент рекомендуется установить, когда требуется снять обтекание элемента, но точно неизвестно с какой стороны;
- `left` — отменяет обтекание с левого края элемента. При этом все другие элементы на этой стороне будут опущены вниз и будут располагаться под текущим элементом;
- `right` — отменяет обтекание с правой стороны элемента;

- none — отменяет действие данного свойства, и обтекание элемента происходит так, как задано с помощью параметра float или других настроек.

Значение по умолчанию

none

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П17

```
<html>
<body>
<div style="float: left; background: #fd0; border: solid 1px black;
padding: 10px; width: 40%">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</div>
<div style="clear: left">
Duis autem dolor in hendrerit in vulputate velit esse molestie consequat,
vel illum dolore eu feugiat nulla facilisis.
</div>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

При работе со свойством clear в некоторых браузерах возникают ошибки, когда обтекание элемента при установке значения float не отменяется.

Параметр *clip*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Частично	Нет	Да	Да	Нет	Частично	Да

Описание

Параметр `clip` определяет область позиционированного элемента, в которой будет показано его содержимое. Все, что не помещается в эту область, будет обрезано и становится невидимым. На данный момент единственно доступная форма области — прямоугольник. Все остальное остается лишь в мечтах. Параметр `clip` работает только для абсолютно позиционированных элементов.

Синтаксис

```
clip: rect(сверху справа снизу слева) | auto
```

Аргументы

В качестве аргументов используется расстояние от края элемента до области вырезки. Если край области нужно оставить без изменений, следует поставить параметр `auto`.

Значение по умолчанию

`auto`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П18

```
<html>
<body>
<div style="position: absolute; clip: rect(20px auto auto 20px); width:
300px; color: white; background: #7f4c3e">
<b>Р.Л. Стивенсон</b><br>
Лето в стране настало,<br>
Вереск опять цветет.<br>
Но некому готовить<br>
Вересковый мед.
</div>
</body>
</html>
```

Результат данного примера показан на рис. П9. Область отображения информации устанавливается на 20 пикселов ниже и правее левого верхнего угла слоя.

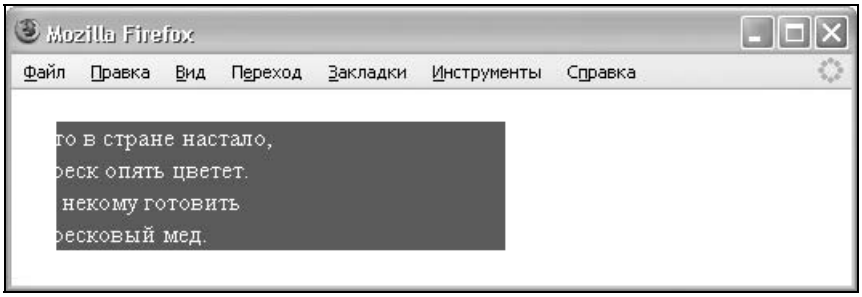


Рис. П9. Обрезание текста через свойство `clip`

Применяется

Ко всем элементам.

Примечание

Браузер Opera оставляет цвет фона элемента неизменным, а обрезает только его содержимое. Другие браузеры маскируют и содержимое, и фон.

Параметр *color*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Частично	Да	Да	Да	Да	Да

Описание

Определяет цвет текста элемента.

Синтаксис

```
color: значение
```

Аргументы

Значение цвета можно задавать тремя способами.

- По его названию.

  Браузеры поддерживают некоторые цвета по их названию.

- По шестнадцатеричному значению.

  Для задания цветов используются числа в шестнадцатеричном коде. Шестнадцатеричная система в отличие от десятичной системы базирует-

ся, как следует из ее названия, на числе 16. Цифры будут следующие: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Цифры от 10 до 15 заменены латинскими буквами. Числа больше 15 в шестнадцатеричной системе образуются объединением двух чисел в одно. Например, числу 255 в десятичной системе соответствует число FF в шестнадцатеричной системе. Чтобы не возникало путаницы в определении системы счисления, перед шестнадцатеричным числом ставят символ решетки #, например #666999. Каждый из трех цветов — красный, зеленый и синий — может принимать значения от 00 до FF. Таким образом, обозначение цвета разбивается на три составляющие #rrggbb, где первые два символа отмечают красную составляющую цвета, два средних — зеленую, а два последних — синюю. Допускается использовать сокращенную форму вида #rgb, где каждый символ следует удваивать. Так, запись #fe0 следует расценивать как #ffee00.

- С помощью RGB.

  Можно определить цвет, используя значения красной, зеленой и синей составляющей в десятичном исчислении. Значение каждого из трех цветов может быть в диапазоне от 0 до 255. Также можно задавать цвет в процентном отношении. При этом способе записи применяется ключевое слово `rgb()`, внутри скобок через запятую перечисляются желаемые значения.

Значение по умолчанию

`black`

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П19

```
<html>
<body>
<p><span style="color: red">L</span>orem ipsum dolor sit amet,
consectetuer adipiscing elit, sed diem nonummy nibh euismod tincidunt ut
lacreet dolore magna aliguam erat volutpat.</p>
<p style="color: rgb(49, 151, 116)"><span style="color: #f00">Ut</span>
wisis enim ad minim veniam, quis nostrud exerci tution ullamcorper
suscipit lobortis nisl ut aliquip ex ea commodo consequat.</p>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

В браузере Netscape 4.x свойство не применяется к элементам форм, таблицам и тегу `HR`.

Параметр *cursor*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Нет	Нет	Да	Да	Нет	Да	Нет	Да	Да	Нет	Нет	Да

Описание

Устанавливает форму курсора, когда он находится в пределах элемента. Вид курсора зависит от операционной системы и заданных параметров.

Прежде чем воспользоваться возможностью переделать вид курсора, решите, будет ли он использоваться к месту. Многих пользователей подобные изменения могут ввести в заблуждение, когда, например, вместо традиционной руки, появляющейся при наведении на ссылку, возникает нечто другое. В большинстве случаев, лучше оставить все как есть.

Синтаксис

```
cursor: auto | crosshair | default | e-resize | help | move | n-resize |
ne-resize | nw-resize | pointer | s-resize | se-resize | sw-resize | text
| w-resize | wait || url('путь к курсору')
```

Аргументы

- `auto` — вид курсора по умолчанию для текущего элемента;
- `url` — позволяет установить свой собственный курсор, для этого нужно указать путь к файлу, в котором содержится форма курсора в формате CUR или ANI. Данный параметр работает только в браузере Internet Explorer 6.

Остальные допустимые аргументы приведены в табл. П5.

Таблица П5. *Вид и значения возможных курсоров мыши*

Вид	Значение	Пример
	default	`cursor:default`
	crosshair	`cursor:crosshair`
	pointer	`cursor:pointer`
	move	`cursor:move`
	text	`cursor:text`
	wait	`cursor:wait`
	help	`cursor:help`
	n-resize	`cursor:n-resize`
	ne-resize	`cursor:ne-resize`
	e-resize	`cursor:e-resize`
	se-resize	`cursor:se-resize`
	s-resize	`cursor:s-resize`
	sw-resize	`cursor:sw-resize`
	w-resize	`cursor:w-resize`
	nw-resize	`cursor:nw-resize`

Значение по умолчанию

`auto`

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П20

```
<html>
<head>
<style type="text/css">
 .cross { cursor: crosshair }
 .help { cursor: help }
```

```
</style>
</head>
<body>
<span class="cross">На этом тексте курсор мыши примет вид
перекрестья.</span>
<br>
<a href="help1.htm" class=help>СПРАВКА 1 </a><br>
<a href="help2.htm" class=help>СПРАВКА 2</a>
</body>
</html>
```

Применяется

Ко всем элементам.

Параметр *display*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Част.*	Част.	Част.	Част.	Част.	Част.	Част.	Част.	Част.	Част.	Част.	Част.

* Част. — поддерживается частично.

Описание

Многоцелевой атрибут, который определяет, должен ли элемент быть показан в документе. Отличается от похожего на него по действию параметра `visible` тем, что `display` не резервирует пустое пространство под элементом, а временно удаляет его из документа. Благодаря этому свойству параметр `display` часто применяют для создания раскрывающегося списка, когда требуется, чтобы один текст заменял собой другой при нажатии на ссылку.

Синтаксис

CSS1:

```
display: block | inline | list-item | none
```

CSS2:

```
display: block | compact | inline | inline-table | list-item | none |
run-in | table | table-caption | table-cell | table-column-group | table-
footer-group | table-header-group | table-row | table-row-group
```

Аргументы

Если элемент должен быть показан в документе, устанавливается значение `block`, `inline` или `list-item`. Значение `none` временно удаляет элемент из документа.

- `block` — элемент отображается как блочный. Применение этого значения для встроенных элементов, например тега `SPAN`, заставляет его вести себя подобно блокам — происходит перенос строк в начале и в конце содержимого;
- `inline` — элемент отображается как встроенный. Использование блочных тегов, таких как `DIV` и `P`, автоматически создает перенос и показывает содержимое этих тегов с новой строки. Аргумент `inline` отменяет эту особенность, поэтому содержимое блочных элементов начинается с того места, где окончился предыдущий элемент;
- `list-item` — элемент выводится как блочный и добавляется маркер списка;
- `none` — временно удаляет элемент из документа. Занимаемое им место не резервируется, и веб-страница формируется так, словно элемента и не было. Изменить значение параметра и сделать его вновь видимым можно с помощью скриптов, обращаясь к свойствам через объектную модель. В этом случае происходит переформатирование данных на странице с учетом вновь добавленного элемента.

Значение по умолчанию

`inline`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П21

```
<html>
<head>
<style type="text/css">
 #menu {
  background: #fc0; border: 1px solid black; padding: 2px; width: 100px }
 #submenu {
  background: #ccc; position: relative; padding: 2px;
  width: 100px; display: none }
</style>
```

```
<script language="JavaScript">
function hideLayer() {
  document.getElementById("submenu").style.display = "none";
}
function showLayer() {
  document.getElementById("submenu").style.display = "block";
}
</script>
</head>
<body>
<div id=menu><a href=list.html onMouseOver="showLayer()"
onMouseOut="hideLayer()">Lorem</a></div>
<div id=submenu>Lorem ipsum dolor sit amet...</div>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Список возможных значений этого атрибута, понимаемый разными браузерами, очень короткий — `block`, `inline`, `list-item` и `none`. Все остальные допустимые аргументы поддерживаются браузерами выборочно.

Параметр *float*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Ошибки	Ошибки	Да	Да	Ошибки	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Определяет, по какой стороне будет выравниваться элемент, при этом остальные элементы будут обтекать его с других сторон. Когда значение параметра `float` равно `none`, элемент выводится на странице как обычно, самое большое — одна строка обтекающего текста может быть на той же линии, что и элемент.

Синтаксис

```
float: left | right | none
```

Аргументы

- `left` — выравнивает элемент по левому краю, а все остальные элементы вроде текста огибают его по правой стороне;
- `right` — выравнивает элемент по правому краю, а все остальные элементы огибают его по левой стороне;
- `none` — обтекание элемента не задается.

Значение по умолчанию

`none`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П22

```
<html>
<body>
<div style="float: left; background: #fc0; border: solid 1px black;
padding: 10px; width: 40%">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</div>
<div style="margin-left: 20px; float: left; width: 50%">
Duis autem dolor in hendrerit in vulputate velit esse molestie consequat,
vel illum dolore eu feugiat nulla facilisis.
</div>
</body>
</html>
```

Результат этого примера показан на рис. П10. В данном случае обязательно следует указывать ширину всех слоев через атрибут `width`. Иначе в некоторых браузерах слои будут располагаться друг под другом. Расстояние между слоями регулируется значением `margin-left`.

Применяется

Ко всем элементам.

Примечание

Браузеры Internet Explorer 4/5 и Netscape 4.x по-разному работают с атрибутом `float`, особенно это касается изображений. Если Internet Explorer для тега `IMG` создает его обтекание, то Netscape требует, чтобы параметр `float` применялся к тегу `DIV`, внутрь которого помещен рисунок. Кроме того, Internet Explorer 5 не поддерживает свойство `float` для встроенных элементов и списков, а Netscape 4.x не применяет `float` для списков, таблиц и их ячеек.

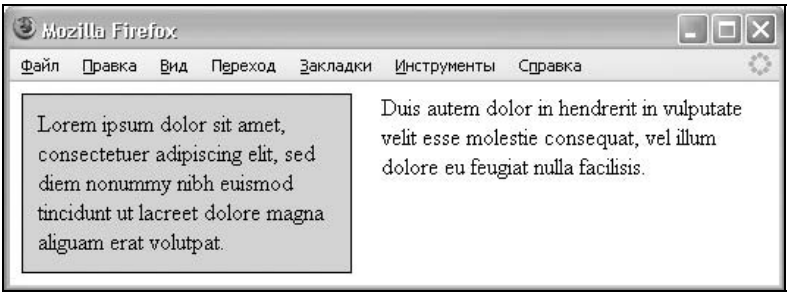


Рис. П10. Размещение слоев по горизонтали с помощью свойства `float`

Параметр *font*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да

Описание

Параметр `font` позволяет установить одновременно несколько атрибутов стиля шрифта. Значения могут идти в любом порядке, браузер сам определит, какое из них соответствует нужному атрибуту.

Синтаксис

CSS1:

```
font: font-style || font-variant || font-weight || font-size ||
line-height || font-family
```

CSS2:

```
font: caption || icon || menu || messagebox || smallcaption || statusbar
```

Аргументы

Любые комбинации значений, определяющих стиль фона, в произвольном порядке. Значения разделяются пробелом. Ни один аргумент не является обязательным, поэтому неиспользуемые значения можно опустить. Для более подробного ознакомления с аргументами смотрите свойства каждого параметра отдельно.

Значение по умолчанию

Нет.

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П23

```
<html>
<body>
<div style="font: 12pt sans-serif">
<h1 style="font: 200% serif">Duis te feugifacilisi</h1>
Duis autem dolor in hendrerit in vulputate velit esse molestie consequat,
vel illum dolore eu feugiat nulla facilisis.
</div>
</body>
</html>
```

Результат примера показан на рис. П1.11. Поскольку заголовок `H1` располагается внутри слоя, для которого задан размер шрифта 12 пунктов, то за 100 % будет приниматься именно этот размер.

Рис. П11. Отображение текста разными шрифтами

Применяется

Ко всем элементам.

Примечание

Константы CSS2 обращаются к параметрам браузера и операционной системы пользователя, поэтому результат действия этих аргументов различается в зависимости от настроек. Применение этих атрибутов связано с необходимостью использования системных настроек шрифтов на своей странице. Не все браузеры поддерживают эти новшества.

Параметр *font-family*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да

Описание

Устанавливает семейство шрифта, которое будет использоваться для оформления текста содержимого. Список шрифтов может включать одно или несколько названий, разделенных запятой. Если в имени шрифта содержатся пробелы, например Trebuchet MS, оно должно заключаться в кавычки.

Когда браузер встречает первый шрифт в списке, он проверяет его наличие на компьютере пользователя. Если такого шрифта нет, берется следующее имя из списка и также анализируется на присутствие. Поэтому несколько шрифтов увеличивают вероятность того, что хотя бы один из них будет обнаружен на клиентском компьютере. Заканчивают список обычно ключевым словом, которое описывает тип шрифта, — `serif`, `sans-serif`, `cursive`, `fantasy` или `monospace`. Таким образом, последовательность шрифтов лучше начинать с экзотических типов и заканчивать обобщенным именем, которое задает вид начертания.

Синтаксис

```
font-family: имя шрифта [, имя шрифта[, ...]]
```

Аргументы

Любое количество имен шрифтов, разделенных запятыми. Универсальные семейства шрифтов:

- `serif` — шрифты с засечками (антиквенные), типа Times;
- `sans-serif` — рубленые шрифты (шрифты без засечек или гротески), типичный представитель — Arial;
- `cursive` — курсивные шрифты;
- `fantasy` — декоративные шрифты;
- `monospace` — моноширинные шрифты, ширина каждого символа в таком семействе одинакова.

Значение по умолчанию

Шрифт, установленный в браузере по умолчанию. Обычно это Times New Roman.

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П24

```
<html>
<body>
<h1 style="font-family: Verdana, Arial, Helvetica, sans-serif">Duis te
feugifacilisi</h1>
<p style="font-family: 'Times New Roman', Times, serif">Lorem ipsum dolor
sit amet, consectetuer adipiscing elit, sed diem nonummy nibh euismod
tincidunt ut lacreet dolore magna aliguam erat volutpat.</p>
</body>
</html>
```

Применяется

Ко всем элементам.

Параметр *font-size*

Браузер	**Internet Explorer**						**Netscape Navigator**			**Opera**		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Частично	Частично	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да

Описание

Определяет размер шрифта элемента. Размер может быть установлен несколькими способами. Набор констант (xx-small, x-small, small, medium, large, x-large, xx-large) задает размер, который называется абсолютным. По правде говоря, он не совсем абсолютный, поскольку зависит от настроек браузера и операционной системы.

Другой набор констант (larger, smaller) устанавливает относительные размеры шрифта. Поскольку размер унаследован от родительского элемента, эти относительные размеры применяются к родительскому элементу, чтобы определить размер шрифта текущего элемента.

В конечном итоге размер шрифта сильно зависит от значения этого параметра у родительского элемента. Допустимо смешивать относительные и абсолютные размеры.

Синтаксис

```
font-size: абсолютный размер | относительный размер | значение | проценты
```

Аргументы

Для задания абсолютного размера используются следующие значения: xx-small | x-small | small | medium | large | x-large | xx-large. Относительный размер шрифта задается аргументами larger и smaller.

Также разрешается использовать любые допустимые единицы CSS: em (высота шрифта элемента), ex (высота символа x), пункты (pt), пикселы (px), проценты (%) и др. За 100 % берется размер шрифта родительского элемента.

Значение по умолчанию

medium — для элемента BODY; размер шрифта родительского элемента во всех остальных случаях.

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П25

```
<html>
<body>
<h1 style="font-size: 200%">Duis te feugifacilisi</h1>
<p style="font-size: 12pt">Lorem ipsum dolor sit amet, consectetuer
adipiscing elit, sed diem nonummy nibh euismod tincidunt ut lacreet
dolore magna aliguam erat volutpat.</p>
```

```
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

При задании абсолютного размера шрифта параметром `small` Internet Explorer 4/5 будет показывать шрифт таким же размером, что и шрифт без стиля, который имеет размер `medium`. Задание размера шрифта `font-size: medium` приводит к различным размерам шрифта в браузерах Internet Explorer и Netscape, что противоречит спецификации CSS и вводит в заблуждение многих разработчиков.

Параметр *font-style*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Частично	Да	Да	Да	Да	Да

Описание

Определяет начертание шрифта — обычное, курсивное или наклонное. Когда для текста установлено курсивное или наклонное начертание, браузер обращается к системе для поиска подходящего шрифта. Если заданный шрифт не найден, браузер использует специальный алгоритм для имитации нужного вида текста. Результат и качество при этом могут получиться неудовлетворительными, особенно при печати документа.

Курсив и наклонный шрифт при всей их похожести — это не одно и то же. Курсив — это специальный шрифт, имитирующий рукописный, наклонный же образуется путем наклона обычных знаков.

Синтаксис

```
font-style: normal | italic | oblique
```

Аргументы

- `normal` — обычное начертание текста;
- `italic` — курсив;
- `oblique` — наклонный шрифт.

Значение по умолчанию

Нет.

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П26

```
<html>
<body>
<h1>Duis te feugifacilisi</h1>
<p style="font-style: italic">Lorem ipsum dolor sit amet, consectetuer
adipiscing elit, sed diem nonummy nibh euismod tincidunt ut lacreet
dolore magna aliguam erat volutpat. Ut wisis enim ad minim veniam, quis
nostrud exerci tution ullamcorper suscipit lobortis nisl ut aliquip ex ea
commodo consequat.</p>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Аргумент `oblique` не поддерживается в браузере Netscape 4. В Internet Explorer шрифт со значением `italic` и `oblique` всегда отображается как курсив.

Параметр *font-variant*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Ошибки	Да	Да	Да	Ошибки	Да	Нет	Да	Да	Да	Да	Да

Описание

Определяет, как нужно представлять строчные буквы, — делать их все прописными или оставить без изменений.

Синтаксис

```
font-variant: normal | small-caps
```

Аргументы

Значение `normal` не изменяет регистр символов, оставляя его по умолчанию. Аргумент `small-caps` модифицирует все строчные символы как прописные уменьшенного размера (капитель).

Значение по умолчанию

`normal`

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П27

```
<html>
<body>
<h1 style="font-variant: small-caps">Duis Te Feugifacilisi</h1>
<p>Lorem ipsum dolor sit amet, Consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.</p>
</body>
</html>
```

Результат этого примера показан на рис. П12. Заметьте, что при использовании капители сохраняются заглавные символы.

Рис. П12. Изменение вида текста с помощью атрибута `font-variant`

Применяется

Ко всем элементам.

Примечание

Браузер Internet Explorer до версии 6.0 при установке значения `small-caps` сделает все буквы одного размера, в то время как прописные буквы должны в любом случае выделяться размером.

Параметр *font-weight*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да

Описание

Устанавливает насыщенность шрифта. Значение задается в интервале от 100 до 900 с шагом 100. Сверхсветлое начертание, которое может отобразить браузер, имеет значение 100, а сверхжирное — 900. Нормальное начертание шрифта (которое установлено по умолчанию) эквивалентно 400, стандартный полужирный текст — значению 700. Другие аргументы (`bolder` и `lighter`) позволяют менять насыщенность текста относительно насыщенности родительского элемента.

Синтаксис

```
font-weight: bold | bolder | lighter | normal | 100 | 200 | 300 | 400 |
500 | 600 | 700 | 800 | 900
```

Аргументы

Насыщенность шрифта задается с помощью ключевых слов: `bold` — полужирное, `bolder` — жирное; `lighter` — светлое, `normal` — нормальное начертание. Также допустимо использовать условные единицы от 100 до 900.

Значение по умолчанию

`normal`

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П28

```
<html>
<body>
<h1 style="font-weight: normal">Duis te feugifacilisi.</h1>
<span style="font-weight: 600">Lorem ipsum dolor sit amet</span>,
consectetuer adipiscing elit, sed diem nonummy nibh euismod tincidunt ut
lacreet dolore magna aliguam erat volutpat. Ut wisis enim ad minim
veniam, quis nostrud exerci tution ullamcorper suscipit lobortis nisl ut
aliquip ex ea commodo consequat.
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Браузер не всегда может адекватно показать требуемую насыщенность шрифта, это зависит от размеров текста и вида шрифта.

Параметр *height*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Част.*	Да	Да	Да	Част.	Да	Да	Да	Да	Част.	Да	Да

* Част. — поддерживается частично.

Описание

Устанавливает высоту блочных или заменяемых элементов (к ним, например, относится тег `IMG`). Высота не включает толщину границ вокруг элемента, значение отступов и полей.

Браузеры по-разному реагируют на настройки высоты элемента. Если его содержимое превышает указанную высоту, то Internet Explorer и Opera проигнорируют значение параметра `height` и автоматически подстроят высоту под контент. Браузер Netscape (а также Mozilla и Firefox) оставит высоту элемента неизменной, а содержимое будет отображаться поверх него. Из-за этой особенности может получиться наложение содержимого элементов

друг на друга, когда в коде HTML они идут последовательно. Чтобы этого не произошло, можно добавить свойство `overflow: auto` к стилю элемента.

Синтаксис

```
height: значение | проценты | auto
```

Аргументы

В качестве значений принимаются любые единицы длины, принятые в CSS, — например пикселы (px), дюймы (in), пункты (pt) и др. При использовании процентной записи высота элемента вычисляется в зависимости от высоты родительского элемента. Если родитель явно не указан, то его роль играет окно браузера. Аргумент `auto` устанавливает высоту исходя из содержимого элемента.

Значение по умолчанию

`auto`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П29

```
<html>
<body>
<div style="height: 50px; background: #fc0; padding: 7px; border: 1px
solid #ccc">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</div>
</body>
</html>
```

Применяется

Ко всем элементам, за исключением встроенных незаменяемых элементов, а также к колонкам и группам таблицы.

Примечание

Браузер Internet Explorer 4 не применяет свойство `height` к встроенным элементам, спискам и некоторым элементам форм вроде флажков, переключателей и текстовых полей. В браузере Opera 3.5 этот параметр не дает никакого эффекта для встроенных элементов, полей формы и таблиц.

Параметр *left*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Частично	Да	Да	Нет	Да	Да

Описание

Для позиционированного элемента определяет расстояние от левого края родительского элемента не включая отступ, поле и ширину рамки, до левого края дочернего элемента (рис. П13). Отсчет координат зависит от значения параметра `position`. Если его аргумент равен `absolute`, в качестве родителя выступает окно браузера и положение элемента определяется от его левого края. В случае значения `relative`, значение `left` отсчитывается от левого края родителя.

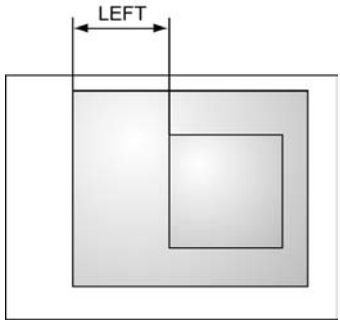


Рис. П13. Значение параметра `left` относительно родительского элемента

Синтаксис

```
left: значение | проценты | auto
```

Аргументы

В качестве значений принимаются любые единицы длины, принятые в CSS, — например пикселы (px), дюймы (in), пункты (pt) и др. Значение параметра `left` может быть и отрицательным, в этом случае возможны наложения разных элементов друг на друга. При задании значения в процентах положение элемента вычисляется в зависимости от ширины родительского элемента. Аргумент `auto` не изменяет положение элемента.

Значение по умолчанию

auto

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П30

```
<html>
<body>
<div style="position: absolute; left: 20px; background: #fc0; margin:
7px">
<div style="position: relative; left: -10px; border: 1px solid black;
padding: 10px; margin: 7px">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat. Ut wisis enim ad minim veniam, quis nostrud exerci tution
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.
</div>
</div>
</body>
</html>
```

Результат данного примера показан на рис. П14. Первый слой задан с абсолютным позиционированием, и его левая граница сдвинута на 20 пикселов вправо от левого края окна браузера. Второй слой смещается на 10 пикселов влево относительно первого слоя, который выступает в качестве родителя. За счет этих смещений черная рамка выходит за пределы оранжевого прямоугольника.

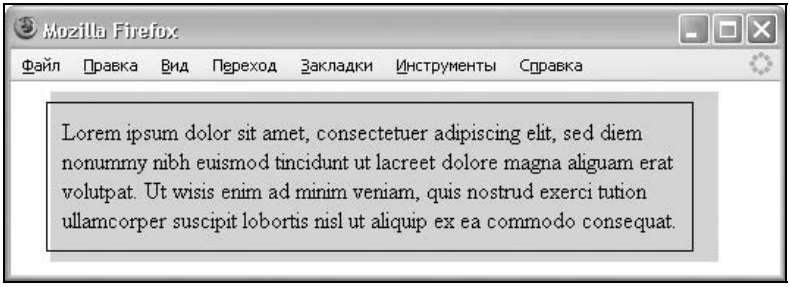


Рис. П14. Изменение положения элементов относительно друг друга

Применяется

Ко всем элементам.

Примечание

Браузер Netscape 4.x не поддерживает значение `auto`.

Параметр *letter-spacing*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Нет	Да	Да	Да	Да	Да

Описание

Определяет интервал между символами в пределах элемента. Браузеры обычно устанавливают расстояние между символами исходя из типа и вида шрифта, его размеров и настроек операционной системы. Чтобы изменить это значение и применяется данный атрибут. Допустимо использовать отрицательное значение, но в этом случае надо убедиться, что сохраняется читабельность текста.

Синтаксис

```
letter-spacing: значение | normal
```

Аргументы

В качестве значений принимаются любые единицы длины, принятые в CSS, — например пикселы (px), дюймы (in), пункты (pt) и др. Наилучший результат дает использование относительных единиц, основанных на размере шрифта (`em` и `ex`). Аргумент `normal` задает интервал между символами как обычно.

Значение по умолчанию

`normal`

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П31

```
<html>
<body>
Duis te feugifacilisi.
<div style="letter-spacing: 3px">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</div>
</body>
</html>
```

Результат этого примера показан на рис. П15. В первой строке показан обычный текст, а во второй он изображен "вразбежку" с межсимвольным интервалом в 3 пиксела.

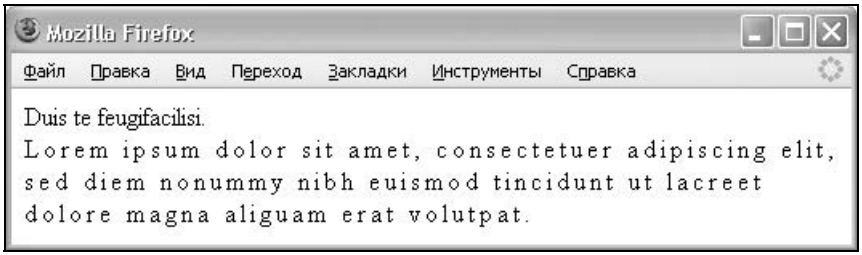


Рис. П15. Изменение межбуквенного интервала в строке текста

Применяется

Ко всем элементам.

Параметр *line-height*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да

Описание

Устанавливает интерлиньяж текста; отсчет ведется от базовой линии шрифта. При обычных обстоятельствах расстояние между строками зависит от вида и размера шрифта и определяется браузером автоматически. Отрицательное значение межстрочного расстояния не допускается.

Синтаксис

```
line-height: normal | множитель | значение | проценты
```

Аргументы

Значение `normal` заставляет браузер вычислять расстояние между строками автоматически. Любое число больше нуля воспринимается как множитель от размера шрифта текущего текста. Например, значение 1,5 устанавливает полуторный межстрочный интервал. В качестве значений принимаются также любые единицы длины, принятые в CSS, — пикселы (px), дюймы (in), пункты (pt) и др. Разрешается использовать процентную запись, в этом случае за 100 % берется высота шрифта.

Значение по умолчанию

`normal`

Наследование

Значения, присвоенные данному параметру, наследуются.

Листинг П32

```
<html>
<body>
<div style="line-height: 1.5">
<h1 style="line-height: 50%">Duis te<br><span style="color:
olive">feugifacilisi</span></h1>
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</div>
</body>
</html>
```

На рис. П16 показан результат данного примера. Для заголовка `H1` величина межстрочного расстояния установлена 50 %, из-за чего строки немного накладываются друг на друга. На читабельность текста это влияет незначительно, поскольку для каждой строки применяются разные цвета, зато это

привлекает внимание читателя. Для остального текста установлен полуторный межстрочный интервал.

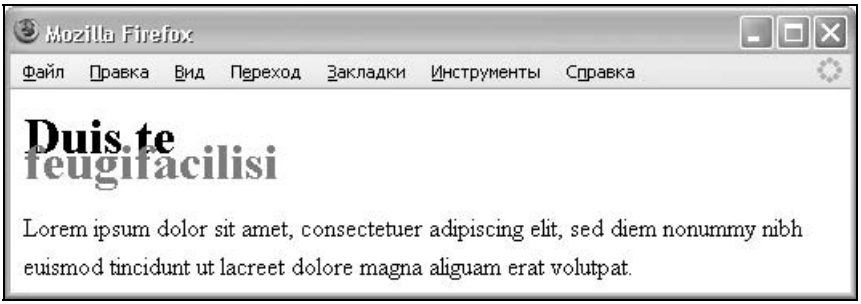


Рис. П.16. Использование интерлиньяжа в заголовках и основном тексте

Применяется

Ко всем элементам.

Параметр *list-style*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да	Да

Описание

Атрибут, позволяющий одновременно задать стиль маркера, его положение, а также изображение, которое будет использоваться в качестве маркера. Для более подробного ознакомления с аргументами смотрите описание свойств каждого параметра `list-style-type`, `list-style-position` и `list-style-image` отдельно.

Синтаксис

```
list-style: list-style-type || list-style-position || list-style-image
```

Аргументы

Любые комбинации трех значений, определяющих стиль маркеров, в произвольном порядке. Значения разделяются пробелом. Ни один аргумент не является обязательным, поэтому неиспользуемые значения можно опустить.

Значение по умолчанию

Нет.

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П33

```
<html>
<body>
<ul style="list-style: square outside">
<li>Lorem ipsum dolor sit amet</li>
<li>Consectetuer adipiscing elit</li>
<li>Sed diem nonummy nibh euismod</li>
<li>Tincidunt ut lacreet dolore magna aliguam erat volutpat. Ut wisis
enim ad minim veniam, quis nostrud exerci tution ullamcorper suscipit
lobortis nisl ut aliquip ex ea commodo consequat.</li>
</ul>
</body>
</html>
```

Применяется

К тегам `DD`, `DT`, `LI`, `OL` и `UL`, а также ко всем элементам, у которых указано свойство стиля `display: list-item`.

Параметр *list-style-image*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Нет	Да	Да	Да	Да	Да

Описание

Устанавливает адрес изображения, которое служит в качестве маркера списка. Этот атрибут наследуется, поэтому для восстановления маркера отдельных элементов списка используется значение `none`. В качестве рисунка можно использовать анимированное изображение в формате GIF.

Синтаксис

```
list-style-image: none | url('путь к файлу')
```

Аргументы

В качестве значения используется относительный или абсолютный путь к графическому файлу. Аргумент `none` отменяет изображение в качестве маркера для родительского элемента.

Значение по умолчанию

`none`

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П34

```
<html>
<body>
<ul style="list-style-image: url(/images/check.gif)">
<li>Lorem ipsum dolor sit amet</li>
<li>Consectetuer adipiscing elit</li>
<li style="list-style-image: none">Sed diem nonummy nibh euismod</li>
<li>Tincidunt ut lacreet dolore magna aliguam erat volutpat.</li>
</ul>
</body>
</html>
```

Результат данного примера показан на рис. П17. Для всех элементов списка, кроме предпоследнего, маркеры сделаны с помощью рисунка.

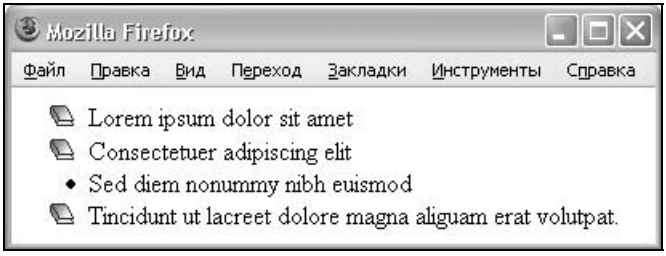


Рис. П17. Использование рисунка в качестве маркеров

Применяется

К тегам DD, DT, LI, OL и UL, а также ко всем элементам, у которых указано свойство стиля display: list-item.

Параметр *list-style-position*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Ошибки	Да	Да	Нет	Да	Да	Да	Да	Да

Описание

Определяет, как будет размещаться маркер относительно текста. Имеется два значения: outside — маркер вынесен за границу элемента списка (рис. П18) и inside — маркер обтекается текстом (рис. П19).

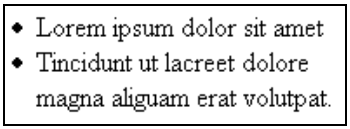


Рис. П18. Использование значения outside

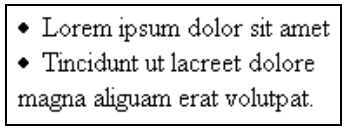


Рис. П19. Результат применения значения inside

Синтаксис

```
list-style-position: inside | outside
```

Аргументы

Когда значение параметра list-style-position равно inside, маркер является частью текстового блока и отображается в элементе списка. Для аргумента outside текст выравнивается по левому краю, а маркеры размещаются за пределами текстового блока.

Значение по умолчанию

outside

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П35

```
<html>
<body>
<ul style="list-style-position: inside">
<li>Lorem ipsum dolor sit amet</li>
<li>Consectetuer adipiscing elit</li>
<li>Sed diem nonummy nibh euismod</li>
<li>Tincidunt ut lacreet dolore magna aliguam erat volutpat. Ut wisis
enim ad minim veniam, quis nostrud exerci tution ullamcorper suscipit
lobortis nisl ut aliquip ex ea commodo consequat.</li>
</ul>
</body>
</html>
```

Применяется

К тегам DD, DT, LI, OL и UL, а также ко всем элементам, у которых указано свойство стиля display: list-item.

Примечание

В браузере Internet Explorer 6 при использовании нумерованного списка OL и значения inside параметра list-style-position числа, идущие после 10, начинают накладываться на текст списка.

Параметр *list-style-type*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Изменяет вид маркера для каждого элемента списка. Этот атрибут используется только в том случае, когда значение свойства list-style-image установлено как none. Маркеры различаются для маркированного списка (тег UL) и нумерованного (тег OL).

Синтаксис

```
list-style-type: disc | circle | square | decimal | lower-roman |
upper-roman | lower-alpha | upper-alpha | none
```

Аргументы

Для маркированного списка используются аргументы `circle`, `disc`, `square`. Для нумерованного списка: `decimal`, `lower-alpha`, `lower-roman`, `upper-alpha`, `upper-roman`. Аргумент `none` устанавливает тип маркера как у родительского элемента. Вид маркеров приведен в табл. П6.

Таблица П6. *Возможные значения маркера списка и их вид*

Тип	Пример
`disc`	●
`circle`	○
`square`	■
`decimal`	1, 2, 3, ...
`lower-roman`	i, ii, iii, ...
`upper-roman`	I, II, III, ...
`lower-alpha`	a, b, c, ...
`upper-alpha`	A, B, C, ...

Значение по умолчанию

`disc` (для `UL`); `decimal` (для `OL`).

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П36

```
<html>
<body>
<ul style="list-style-type: square">
<li>Lorem ipsum dolor sit amet</li>
<li>Consectetuer adipiscing elit</li>
<li>Sed diem nonummy nibh euismod</li>
```

```
<li>Tincidunt ut lacreet dolore magna aliguam erat volutpat. Ut wisis
enim ad minim veniam, quis nostrud exerci tution ullamcorper suscipit
lobortis nisl ut aliquip ex ea commodo consequat.</li>
</ul>
</body>
</html>
```

Применяется

К тегам `DD`, `DT`, `LI`, `OL` и `UL`, а также ко всем элементам, у которых указано свойство стиля `display: list-item`.

Примечание

Если заданы отступы с помощью параметров `margin` или `padding` и установлено значение `list-style-type: none`, браузер Netscape 4.x несмотря на это показывает маркеры.

Параметр *margin*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Ошибки	Ошибки	Да	Да	Ошибки	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Устанавливает величину отступа от каждого края элемента. Отступом является пространство от границы текущего элемента до внутренней границы его родительского элемента (рис. П20).

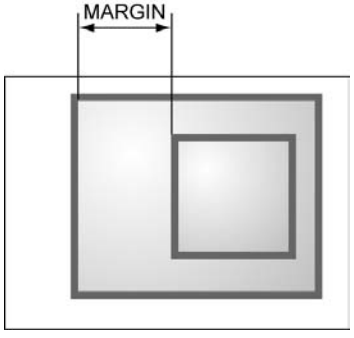


Рис. П20. Отступ от левого края элемента

Если у элемента нет родителя, отступом будет расстояние от границы элемента до края окна браузера с учетом того, что у самого окна по умолчанию тоже установлены отступы. Чтобы от них избавиться, следует устанавливать значение `margin` для селектора `BODY`, равное нулю *(см. главу 9)*.

Параметр `margin` позволяет задать величину отступа сразу для всех сторон элемента или определить ее только для указанных сторон.

Синтаксис

```
margin: значение| auto {1,4}
```

Аргументы

Разрешается использовать одно, два, три или четыре значения, разделяя их пробелом. Эффект зависит от количества аргументов и приведен в табл. П7.

Таблица П7. *Применение отступов в зависимости от числа значений*

Число аргументов	Результат
1	Отступы будут установлены для всех сторон элемента
2	Первый аргумент устанавливает отступ от верхнего и нижнего края, второй аргумент — от левого и правого
3	Первый аргумент задает отступ от верхнего края, второй — одновременно от левого и правого края, а третий — от нижнего края
4	Поочередно устанавливается отступ от верхнего, правого, нижнего и левого края

Величину отступов можно указывать в пикселах (px), процентах (%) или других допустимых для CSS единицах. Значение может быть как положительным, так и отрицательным числом. Аргумент `auto` указывает, что размер отступов будет автоматически рассчитан браузером.

Значение по умолчанию

`0`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П37

```
<html>
<head>
<style type="text/css">
 BODY { margin: 0px; padding: 0px }
 DIV.parent { margin: 20%; background: #fd0; padding: 10px }
 DIV.child { border: 3px solid #666; padding: 10px; margin: 10px }
</style>
</head>
<body>
<div class=parent>
<div class=child>
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</div>
</div>
</body>
</html>
```

Результат данного примера показан на рис. П21. Для первого слоя с именем `parent` устанавливаются отступы со значением 20 %, тогда вокруг цветного прямоугольника образуется пустое пространство. Использование параметра `margin` для второго слоя с именем `child` отображает рамку вокруг текста, отстоящую от края на указанное значение.

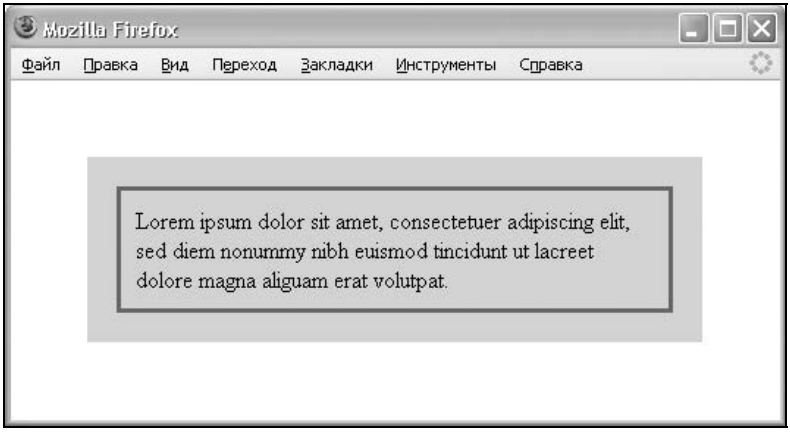


Рис. П21. Использование свойства `margin` для создания отступов вокруг элемента

Применяется

Ко всем элементам.

Примечание

Все отступы имеют проблемы или не поддерживаются полностью для встроенных элементов. Параметр `margin` хорошо поддерживается для блочных элементов в Internet Explorer 4/5, в то же время для встроенных элементов, а также ячеек таблиц он игнорируется полностью. Netscape 4.x работает корректно, пока параметр `margin` не применяется к плавающим или встроенным элементам. Opera 4 имеет проблемы с правыми и левыми отступами для встроенных элементов.

Параметр *margin-bottom*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Ошибки	Ошибки	Да	Да	Ошибки	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Устанавливает величину отступа от нижнего края элемента до его содержимого. Для управления значением отступа на других сторонах применяются свойства `margin-left`, `margin-right` и `margin-top`, которые определяют отступ слева, справа и сверху соответственно.

Синтаксис

```
margin-bottom: значение | auto
```

Аргументы

Величину нижнего отступа можно указывать в пикселах (px), процентах (%) или других допустимых для CSS единицах. Значение может быть как положительным, так и отрицательным числом. Аргумент `auto` указывает на то, что размер отступов будет автоматически рассчитан браузером.

Значение по умолчанию

`0`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П38

```
<html>
<body>
<div style="background: #008B66; color: white; padding: 10px;
margin-bottom: -7px">
<big>Lorem ipsum dolor sit amet</big>
</div>
<div style="margin-left: 40px; background: #ccc; padding: 10px">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</div>
</body>
</html>
```

Результат этого примера показан на рис. П22. За счет использования отрицательного значения параметра `margin-bottom` происходит перекрытие слоев, а добавление `margin-left` позволяет укоротить ширину слоя слева.

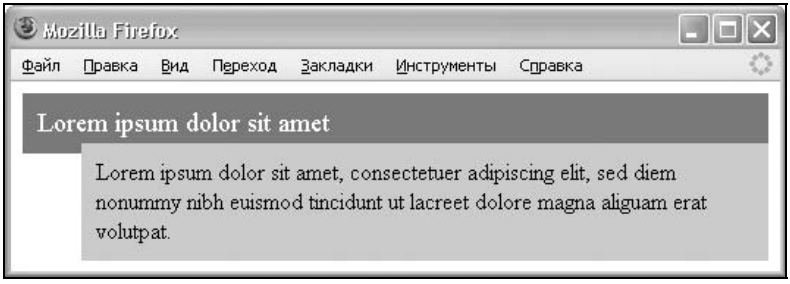


Рис. П22. Применение разных отступов для наложения элементов

Применяется

Ко всем элементам.

Параметр *overflow*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	6.0	7.0
Поддерживается	Да	Да	Да	Да	Нет	Да	Нет	Да	Да	Нет	Частично	Да

Описание

Свойство `overflow` управляет содержимым блочного элемента, если оно целиком не помещается и выходит за область заданных размеров.

Синтаксис

```
overflow: auto | hidden | scroll | visible
```

Аргументы

- `visible` — отображается все содержимое элемента, даже за пределами установленной высоты и ширины;
- `hidden` — отображается только область внутри элемента, остальное будет обрезано;
- `scroll` — всегда добавляются полосы прокрутки;
- `auto` — полосы прокрутки добавляются только при необходимости.

Значение по умолчанию

`visible`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П39

```
<html>
<body>
<div style="overflow: scroll; width: 300px; height: 150px; border:
solid 1px black">
<h2>Duis te feugifacilisi</h2>
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.
</div>
</body>
</html>
```

Результат этого примера показан на рис. П23. Добавление полос прокрутки позволяет просмотреть содержание элемента без изменения его размеров.

Рис. П23. Вид блочного элемента с полосами прокрутки

Применяется

К блочным или заменяемым элементам.

Примечание

Браузер Opera 6 не поддерживает свойство `overflow: scroll`. В браузерах Internet Explorer, Netscape и Mozilla применение `overflow` к тегу `TEXTAREA` устраняет полосы прокрутки.

Параметр *padding*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Ошибки	Ошибки	Да	Да	Ошибки	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Устанавливает значение полей вокруг содержимого элемента. Полем называется расстояние от внутреннего края рамки элемента до воображаемого прямоугольника, ограничивающего его содержимое (рис. П24).

Параметр `padding` позволяет задать величину полей сразу для всех сторон элемента или определить ее только для указанных сторон.

Синтаксис

```
padding: значение | auto {1, 4}
```

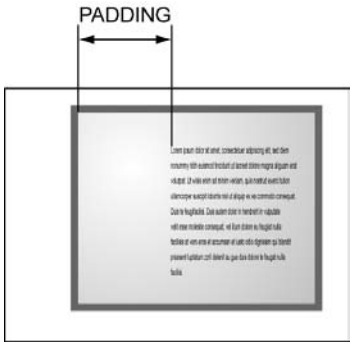


Рис. П24. Поле слева от текста

Аргументы

Разрешается использовать одно, два, три или четыре значения, разделяя их между собой пробелом. Эффект зависит от количества аргументов и приведен в табл. П8.

Таблица П8. *Применение полей к элементу в зависимости от числа значений*

Число аргументов	Результат
1	Поля будут установлены для всех сторон содержимого элемента
2	Первый аргумент устанавливает поле от верхнего и нижнего края содержимого, второй аргумент — от левого и правого
3	Первый аргумент задает поле от верхнего края содержимого, второй — одновременно от левого и правого края, а третий — от нижнего края
4	Поочередно устанавливаются поля от верхнего, правого, нижнего и левого края содержимого

Величину полей можно указывать в пикселах (px), процентах (%) или других допустимых для CSS единицах. Аргумент `auto` указывает на то, что размер полей будет автоматически рассчитан браузером.

Значение по умолчанию

0

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П40

```
<html>
<body>
<div style="background-color: #fc3; border: 2px solid black; padding:
20px">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat. Ut wisis enim ad minim veniam, quis nostrud exerci tution
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.
</div>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Все поля имеют проблемы или не поддерживаются полностью для встроенных элементов. Опера 3.6 игнорирует отрицательные значения параметра `padding`, но изменяет высоту строки встроенного элемента, основываясь на некорректных значениях полей. Internet Explorer 4/5 правильно работает для блочных элементов, но встроенные элементы игнорирует полностью. Netscape 4.x работает корректно, пока параметр `padding` не применяется к плавающим или встроенным элементам.

Параметр *padding-bottom*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Ошибки	Ошибки	Да	Да	Ошибки	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Устанавливает значение поля от нижнего края содержимого элемента. Величина поля от других границ указывается с помощью свойств `padding-top`, `padding-left` и `padding-right`, они определяют поля сверху, слева и справа.

Синтаксис

```
padding-bottom: значение | auto
```

Аргументы

Величину нижнего поля можно указывать в пикселах (px), процентах (%) или других допустимых для CSS единицах. Аргумент `auto` указывает на то, что размер поля будет автоматически рассчитан браузером.

Значение по умолчанию

```
0
```

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П41

```
<html>
<body>
<div style="background-color: #fc3; border: 2px solid black; padding:
10px; padding-bottom: 40px">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat. Ut wisis enim ad minim veniam, quis nostrud exerci tution
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.
</div>
</body>
</html>
```

Применяется

Ко всем элементам.

Параметр *position*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Ошибки	Част.*	Част.	Част.	Част.	Част.	Ошибки	Част.	Да	Нет	Да	Да

* Част. — поддерживается частично.

Описание

Устанавливает способ позиционирования элемента относительно окна браузера или других объектов на веб-странице.

Синтаксис

```
position: absolute | fixed | relative | static
```

Аргументы

Допустимые аргументы параметра `position`, их описания и поддержка браузерами приведены в табл. П9.

Таблица П9. *Значения свойства* `position`

Значение	Описание	Браузеры
`absolute`	Указывает, что элемент абсолютно позиционирован. В этом случае он не существует в обычном потоке документа подобно другим элементам, которые отображаются на веб-странице, как будто абсолютно позиционированного объекта и нет. Положение элемента задается атрибутами `left`, `top`, `right` и `bottom` относительно края окна браузера	IE4 NC4 O4
`fixed`	Устанавливает фиксированное позиционирование. По своим свойствам это значение аналогично аргументу `absolute`, но в отличие от него привязывается к указанной параметрами `left`, `top`, `right` и `bottom` точке на экране и не меняет своего положения даже при пролистывании веб-страницы. Браузеры Netscape, Mozilla и Firefox вообще не отображают полосы прокрутки, если положение элемента задано фиксированным и оно не помещается целиком в окно браузера. Хотя в браузере Opera и отображаются полосы прокрутки, но они никак не влияют на позицию элемента	NC6.1 O4
`relative`	Положение элемента устанавливается относительно родителя, иными словами, контейнера, в который вложен элемент. Если родительский элемент явно не задан, то в качестве него выступает окно браузера. Добавление атрибутов `left`, `top`, `right` и `bottom` изменяет позицию элемента и сдвигает его в ту или иную сторону от родителя в зависимости от применяемого параметра	IE4 NC4 O4
`static`	Элементы отображаются как обычно. Использование параметров `left`, `top`, `right` и `bottom` не приводит ни к каким результатам	IE4 NC4 O4

Обозначения, используемые в таблице: IE — Internet Explorer; NC — Netscape; O — Opera. Число после имени браузера указывает, начиная с какой версии поддерживается значение.

Значение по умолчанию

`static`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П42

```
<html>
<body>
<div style="font-family: Times, serif; font-size: 200%">
T<span style="position: relative; top: 10px">E</span>X и L<span
style="position: relative; top: -5px; font-size: 80%">A</span>T<span
style="position: relative; top: 10px">E</span>X
</div>
</body>
</html>
```

Результат данного примера представлен на рис. П25. Для смещения отдельных букв вверх или вниз используются относительное позиционирование и атрибут `top`. Его отрицательное значение сдвигает символ вверх на указанное число пикселов, а положительное — вниз.

Рис. П25. Изменение символов относительно своего положения

Применяется

Ко всем элементам.

Примечание

В браузере Internet Explorer 4 добавление параметра `position: absolute` к ссылкам не позиционирует их, вдобавок ссылки теряют свою функциональность. Кроме того, в этом браузере абсолютное позиционирование не работает для встроенных элементов и списков, а относительное позиционирование не применяется к ячейкам таблицы (теги `TD` и `TH`). Браузер Netscape 4.x не приме-

няет значение `relative` к полям формы, спискам, изображениям, таблицам или их ячейкам, абсолютное позиционирование не действует для списков или элементов форм.

Параметр *right*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Нет	Нет	Да	Да	Нет	Да	Нет	Да	Да	Нет	Да	Да

Описание

Для позиционированного элемента определяет расстояние от правого края родительского элемента, не включая отступ, поле и ширину рамки, до правого края дочернего элемента (рис. П26). Отсчет координат зависит от значения параметра `position`. Если его аргумент равен `absolute`, то в качестве родителя выступает окно браузера и положение элемента определяется от его правого края. В случае значения `relative` значение `right` отсчитывается от правого края родителя.

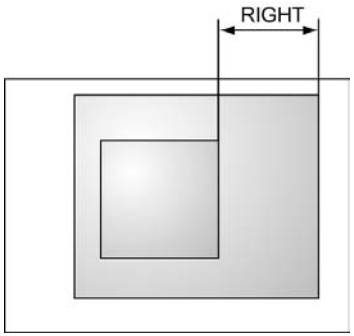


Рис. П26. Значение параметра `right`

Синтаксис

```
right: значение | проценты | auto
```

Аргументы

В качестве значений принимаются любые единицы длины, принятые в CSS, — например пикселы (px), дюймы (in), пункты (pt) и др. Значение па-

раметра `right` может быть и отрицательным, в этом случае возможны наложения разных элементов друг на друга. При задании значения в процентах положение элемента вычисляется в зависимости от ширины родительского элемента. Аргумент `auto` не изменяет положение элемента.

Значение по умолчанию

`auto`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П43

```
<html>
<head>
<style type="text/css">
 #left {
  position: absolute; top: 20px; left: 20px; width:100px;
  background: #fc3; border: 1px solid black; padding: 10px
}
 #center {
  position: relative; width: auto;
  margin: 30px 230px 0px 140px
}
 #right {
  position: absolute; top: 20px; right: 20px; width:200px;
  background: #fc3; border: 1px solid black; padding: 10px
}
</style>
</head>
<body>
  <div id=left>...</div>
  <div id=center>...</div>
  <div id=right>...</div>
</body>
</html>
```

Применяется

Ко всем элементам.

Параметр *text-align*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Частично	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Определяет горизонтальное выравнивание текста в пределах элемента. Этот атрибут наследуется, поэтому может быть установлен для целого блока для воздействия на все вложенные в него элементы.

Синтаксис

```
text-align: center | justify | left | right
```

Аргументы

- `center` — выравнивание по центру;
- `justify` — одновременное выравнивание по левому и правому краю, в этом случае браузер добавляет пробелы между словами;
- `left` — выравнивание по левому краю;
- `right` — выравнивание по правому краю.

Значение по умолчанию

`left`

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П44

```
<html>
<body>
<p style="text-align: justify"> Duis autem dolor in hendrerit in
vulputate velit esse molestie consequat, vel illum dolore eu feugiat
nulla facilisis at vero eros et accumsan et iusto odio dignissim qui
blandit praesent luptatum zzril delenit au gue duis dolore te feugat
nulla facilisi.</p>
```

```
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

В Netscape 4 при размещении текста в таблице со значением `justify` текст может и не выравниваться по ширине, но в остальных случаях все работает корректно.

Параметр *table-layout*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Нет	Нет	Да	Да	Нет	Нет	Нет	Да	Да	Нет	Ошибки	Да

Описание

Определяет, как браузер должен вычислять высоту и ширину ячеек таблицы, основываясь на ее содержимом.

Синтаксис

```
table-layout: auto | fixed
```

Аргументы

- `auto` — браузер загружает всю таблицу, анализирует ее для определения размеров ячеек и только после этого отображает;
- `fixed` — это значение повышает производительность построения таблиц. Ширина колонок в этом случае определяется либо с помощью тега `COL`, либо вычисляется на основе первой строки. Если данные о форматировании первой строки таблицы по каким-либо причинам получить невозможно, то в этом случае таблица делится на колонки равной ширины.

Значение по умолчанию

`auto`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П45

```
<html>
<body>
<table width=700 cellspacing=0 border=1 bordercolor=black style=
"table-layout: fixed">
<col width=150>
<colgroup span=5 width=50 align=center>
<colgroup span=4 width=75 align=center>
  <tr>
    <td> </td><td>1995</td><td>1996</td><td>1997</td>
    <td>1998</td><td>1999</td><td>2000</td><td>2001</td>
    <td>2002</td><td>2003</td>
  </tr>
  <tr>
    <td>Нефть</td><td>5</td><td>7</td><td>2</td><td>8</td>
    <td>3</td><td>34</td><td>62</td><td>74</td><td>57</td>
  </tr>
  <tr>
    <td>Золото</td><td>3</td> <td>6</td><td>4</td><td>6</td>
    <td>4</td><td>69</td><td>72</td><td>56</td><td>47</td>
  </tr>
  <tr>
    <td>Дерево</td><td>5</td><td>8</td><td>3</td><td>4</td>
    <td>7</td><td>73</td><td>79</td><td>34</td><td>86</td>
  </tr>
</table>
</body>
</html>
```

Применяется

К элементу TABLE.

Параметр *text-indent*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Ошибки	Ошибки	Да	Да	Ошибки	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Устанавливает величину отступа первой строки блока текста (например для параграфа `P`) и на остальные строки воздействия не оказывает. Допускается отрицательное значение для создания выступа первой строки, но следует проверить, чтобы текст не выходил за пределы окна браузера.

Синтаксис

```
text-indent: значение | проценты
```

Аргументы

В качестве значений принимаются любые единицы длины, принятые в CSS, — например пикселы (px), дюймы (in), пункты (pt) и др. При задании значения в процентах отступ первой строки вычисляется в зависимости от ширины блока. Допустимо использовать отрицательные значения, но при этом в разных браузерах возможно появление ошибок.

Значение по умолчанию

```
0
```

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П46

```
<html>
<body>
<p style="text-indent: 2em">Lorem ipsum dolor sit amet, consectetuer
adipiscing elit, sed diem nonummy nibh euismod tincidunt ut lacreet
dolore magna aliguam erat volutpat. Ut wisis enim ad minim veniam, quis
nostrud exerci tution ullamcorper suscipit lobortis nisl ut aliquip ex ea
commodo consequat.</p>
```

```
</body>
</html>
```

Применяется

К блочным элементам.

Примечание

Значение `text-indent`, заданное в процентах, в некоторых браузерах может вычисляться некорректно.

Параметр *text-decoration*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Част.	Да	Да	Да	Част.	Да	Част.	Да	Да	Част.	Да	Да

* Част. — поддерживается частично.

Описание

Добавляет оформление текста в виде его подчеркивания, перечеркивания, линии над текстом и мигания. Одновременно можно применить более одного стиля, перечисляя значения через пробел.

Синтаксис

```
text-decoration: blink | line-through | overline | underline | none
```

Аргументы

- `blink` — мигающий текст;
- `line-through` — перечеркнутый текст;
- `overline` — линия проходит над текстом.
- `underline` — подчеркнутый текст;
- `none` — отменяет все эффекты.

Значение по умолчанию

`none`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П47

```
<html>
<head>
<style type="text/css">
 A { text-decoration: none }
 A:hover { text-decoration: underline }
</style>
</head>
<body>
 <a href=link1.html>Lorem ipsum dolor sit amet</a>
</body>
</html>
```

Применяется

Ко всем элементам.

Примечание

Браузер Netscape 4.x не поддерживает аргумент `overline`, Internet Explorer хотя и позволяет использовать значение `blink`, текст показывает как обычно, без всякого мерцания.

Согласно спецификации CSS, если для элемента задан вид форматирования, а для его наследника нет, то все равно свойства родителя будут передаваться его вложенным элементам. Так, если используется подчеркнутый параграф, а внутри него расположен неподчеркнутый элемент `color`, выделяющий слово другим цветом, все слова в параграфе будут подчеркнуты, включая и выделенные другим цветом слова. На практике, однако, установка `text-decoration: none` уберет все эффекты независимо от оформления родительского элемента. Единственные исключения — браузеры Opera и Internet Explorer 5 для Macintosh, которые выполняют эту часть спецификации правильно. Opera 4/5 и Netscape 6 не изменяют параметры изображений внутри тега `SPAN` при задании параметров оформления родительского элемента. Вдобавок, Netscape 6 не распространяет оформление родительского элемента на дочерние, а переносит только подчеркивание.

Несмотря на кажущуюся простоту проблемы, особенность параметра `text-decoration` в некоторых случаях служит головной болью для разработчиков сайтов.

Параметр *text-transform*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Ошибки	Да	Да	Да	Да	Да

Описание

Управляет преобразованием текста элемента в прописные или строчные символы. Когда значение отлично от `none`, регистр исходного текста будет изменен.

Синтаксис

```
text-transform: capitalize | lowercase | uppercase | none
```

Аргументы

- `none` — текст пишется как есть, без каких-либо изменений;
- `capitalize` — каждое слово будет начинаться с заглавного символа;
- `lowercase` — все символы становятся строчными (нижний регистр);
- `uppercase` — все символы становятся прописными (верхний регистр).

Значение по умолчанию

`none`

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П48

```
<html>
<head>
<style type="text/css">
 H1 { text-transform: lowercase }
 P { text-transform: capitalize }
</style>
</head>
```

```
<body>
<h1>Lorem ipsum dolor sit amet</h1>
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem
nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat
volutpat.</p>
</body>
</html>
```

Результат данного примера показан на рис. П27. Для заголовка `H1` символы устанавливаются в нижнем регистре, а для основного текста каждое слово начинается с прописной буквы.

Рис. П27. Изменение вида текста

Применяется

Ко всем элементам.

Примечание

В браузере Netscape 4.x некоторые символы, например символы русского языка, не подвергаются трансформации с помощью этого атрибута.

Параметр *top*

Браузер	**Internet Explorer**						**Netscape Navigator**			**Opera**		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Част.*	Част.	Да	Да	Част.	Да	Ошибки	Да	Да	Нет	Да	Да

* Част. — поддерживается частично.

Описание

Для позиционированного элемента определяет расстояние от верхнего края родительского элемента, не включая отступ, поле и ширину рамки, до верхнего края дочернего элемента (рис. П28). Отсчет координат зависит от значения параметра `position`. Если его аргумент равен `absolute`, в качестве родителя выступает окно браузера и положение элемента определяется от его левого края. В случае значения `relative` значение `left` отсчитывается от левого края родителя.

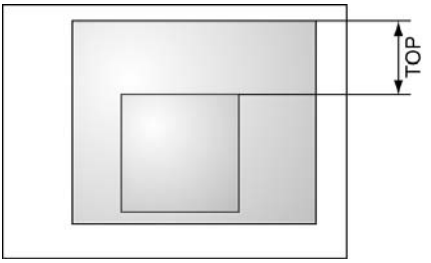


Рис. П28. Значение параметра `top`

Синтаксис

```
top: значение | проценты | auto
```

Аргументы

В качестве значений принимаются любые единицы длины, принятые в CSS, — например пикселы (px), дюймы (in), пункты (pt) и др. Значение параметра `top` может быть и отрицательным, в этом случае вероятно наложение разных элементов друг на друга. При задании значения в процентах положение элемента вычисляется в зависимости от высоты родительского элемента. Аргумент `auto` не изменяет положение элемента.

Значение по умолчанию

`auto`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П49

```
<html>
<head>
```

```
<style type="text/css">
 #menu { position: absolute; left: 350px; top: 50px; width: 120px;
  background: #e0e0e0; border: solid 1px black; text-align: left
}
 #content { position: absolute; left: 0px; top: 0px; width: 400px;
  background: #800000; color: white; text-align: left
}
</style>
</head>
<body>
<div id=content>...</div>
<div id=menu>...</div>
</body>
</html>
```

Применяется

Ко всем элементам.

Параметр *vertical-align*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Част.*	Част.	Част.	Част.	Част.	Част.	Част.	Да	Да	Да	Да	Да

* Част. — поддерживается частично.

Описание

Выравнивает элемент по вертикали относительно своего родителя.

Синтаксис

```
vertical-align: baseline | bottom | middle | sub | super | text-bottom |
text-top | top | значение | проценты
```

Аргументы

Допустимые значения атрибута `vertical-align`, а также браузеры, в которых эти значения понимаются, перечислены в табл. П10.

В качестве значения можно использовать проценты, пикселы или другие доступные в CSS единицы. Положительный аргумент смещает элемент вверх относительно базовой линии, в то время как отрицательное значение

опускает его вниз. Не все браузеры поддерживают такой способ записи, в частности Internet Explorer и Netscape только с шестой версии.

Таблица П10. *Значения выравнивания по вертикали*

Значение	Описание	Браузеры
baseline	Выравнивает базовую линию текущего элемента по базовой линии родителя. Если родительский элемент не имеет базовой линии, то за нее принимается нижняя граница элемента	IE4; NC4; O3.5
bottom	Выравнивает основание текущего элемента по нижней части элемента строки, расположенного ниже всех	IE4; NC4; O3.5
middle	Выравнивает среднюю точку элемента по базовой линии родителя плюс половина высоты родительского элемента	IE4; NC4; O3.5
sub	Элемент изображается как подстрочный, в виде нижнего индекса. Размер шрифта при этом не меняется	IE4; NC6; O3.5
super	Элемент изображается как надстрочный, в виде верхнего индекса. Размер шрифта остается прежним	IE4; NC6; O3.5
text-bottom	Нижняя граница элемента выравнивается по самому нижнему краю текущей строки	IE4; NC4; O3.5
text-top	Верхняя граница элемента выравнивается по самому высокому текстовому элементу текущей строки	IE4; NC4; O3.5
top	Выравнивание верхнего края элемента по верху самого высокого элемента строки	IE4; NC4; O3.5

Обозначения, используемые в таблице: IE — Internet Explorer; NC — Netscape; O — Opera. Число после имени браузера указывает, начиная с какой версии поддерживается значение.

Значение по умолчанию

baseline

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П50

```
<html>
<body>
<div style="font-family: Times, serif; font-size: 200%">
T<span style="vertical-align: sub">E</span>X и L<span style=
"vertical-align: 5px; font-size: 80%">A</span>T<span style=
"vertical-align: sub">E</span>X
```

```
</div>
</body>
</html>
```

Применяется

Только к встроенным элементам.

Параметр *visibility*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Частично	Да	Да	Нет	Да	Да

Описание

Предназначен для отображения или скрытия элемента, включая рамку вокруг него и фон. При скрытии элемента, хотя он и становится не виден, место, которое элемент занимает, остается за ним. Если предполагается вывод разных элементов в одно и то же место экрана, для обхода этой особенности следует использовать абсолютное позиционирование или применить свойство `display`.

Синтаксис

```
visibility: inherit | visible | hidden | collapse
```

Аргументы

- `inherit` — устанавливает значение этого свойства как у родительского элемента;
- `visible` — отображение элемента;
- `hidden` — элемент становится невидимым или, правильнее сказать, полностью прозрачным, поскольку он продолжает участвовать в форматировании страницы;
- `collapse` — если это значение применяется не к строкам или колонкам таблицы, то результат его использования будет таким же, как `hidden`. В случае использования `collapse` для содержимого ячеек таблиц они реагируют, как будто к ним было добавлено стилевое свойство `display: none`. Иными словами, заданные строки и колонки убираются, а

таблица перестраивается заново. Этот аргумент не поддерживается браузером Internet Explorer.

Значение по умолчанию

`visible`

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П51

```
<html>
<head>
<script language="JavaScript">
 function hideLayer()
  { document.getElementById("descr").style.visibility = "hidden" }
 function showLayer()
  { document.getElementById("descr").style.visibility = "visible" }
</script>
</head>
<body>
<a href=# onMouseOver="showLayer()" onMouseOut="hideLayer()"><img
src=button.gif width=98 height=33 border=0></a>
<div id=descr style="visibility: hidden">Данная эксклюзия является
подмножеством астрациональных супрематичных монотенных федоний
кадонарного экстрафазория.</div>
</body>
</html>
```

В этом примере показано динамическое отображение и скрытие слоя с именем `descr` в зависимости от того, наведен курсор мыши на рисунок или нет.

Применяется

К любым элементам.

Примечание

В браузере Netscape 4.x свойство `visibility` не применяется к элементам списка, полям форм, изображениям и таблицам.

Параметр *white-space*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Нет	Нет	Част.*	Част.	Нет	Нет	Част.	Да	Да	Нет	Да	Да

* Част. — поддерживается частично.

Описание

Параметр `white-space` устанавливает, как отображать пробелы между словами. В обычных условиях любое количество пробелов в коде HTML отображается на веб-странице как один. Исключением является тег `PRE`; помещенный в этот контейнер текст выводится со всеми пробелами, как он был отформатирован пользователем. Таким образом, `white-space` имитирует работу тега `PRE`, но в отличие от него не меняет шрифт на моноширинный.

Синтаксис

```
white-space : normal | nowrap | pre
```

Аргументы

- `normal` — текст в окне браузера выводится как обычно, переносы строк устанавливаются автоматически;
- `nowrap` — переносы строк в коде HTML игнорируются, весь текст отображается одной строкой, вместе с тем, добавление тега `BR` переносит текст на новую строку;
- `pre` — текст показывается с учетом всех пробелов и переносов, как они были добавлены разработчиком в коде HTML.

Значение по умолчанию

`normal`

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П52

```
<html>
<body>
```

```
<p style="white-space: pre">&lt;html&gt;
 &lt;body&gt;
   &lt;b&gt;Великая теорема Ферма&lt;/b&gt;&lt;br&gt;
   &lt;i&gt;X &lt;sup&gt;&lt;small&gt;n&lt;/small&gt;&lt;/sup&gt;
   + Y &lt;sup&gt;&lt;small&gt;n&lt;/small&gt;&lt;/sup&gt; =
     Z&lt;sup&gt;&lt;small&gt;n&lt;/small&gt;
     &lt;/sup&gt;&lt;/i&gt;&lt;br&gt;
   где n - целое число &gt; 2
 &lt;/body&gt;
&lt;/html&gt;
</p>
</body>
</html>
```

Результат этого примера показан на рис. П1.29. Символы `>` и `<` устанавливают открывающую и закрывающую угловую скобку, они необходимы, чтобы отобразить на веб-странице код HTML. Учтите, что любые переводы строк тоже учитываются, поэтому текст в примере начинается сразу после объявления стиля.

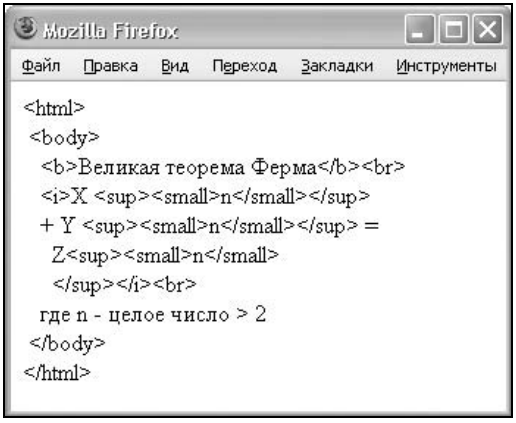


Рис. П1.29. Вид текста при использовании значения `pre` атрибута `white-space`

Применяется

К блочным элементам.

Примечание

Значение `nowrap` недоступно в браузере Netscape 4. Значение `pre` не работает в браузере Internet Explorer.

Параметр *width*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Част.*	Да	Да	Да	Част.	Да	Част.	Да	Да	Част.	Да	Да

* Част. — поддерживается частично.

Описание

Устанавливает ширину блочных или заменяемых элементов (к ним, например, относится тег `IMG`). Ширина не включает толщину границ вокруг элемента, значение отступов и полей.

Разные браузеры неодинаково работают с шириной элемента. Так, если содержимое элемента превышает его заданную ширину, то Internet Explorer и Opera используют для атрибута `width` значение `auto`. Браузеры Netscape, Mozilla, Firefox оставляют указанную ширину неизменной, а содержимое при этом выходит за границы элемента и отображается поверх. Для встроенных элементов получается наоборот. Если ширина задана заведомо больше содержимого, то Internet Explorer и Opera отобразят область согласно присвоенному значению, а остальные браузеры атрибут `width` проигнорируют.

Синтаксис

```
width: значение | проценты | auto
```

Аргументы

В качестве значений принимаются любые единицы длины, принятые в CSS, — например пикселы (px), дюймы (in), пункты (pt) и др. При использовании процентной записи ширина элемента вычисляется в зависимости от ширины родительского элемента. Если родитель явно не указан, то его роль играет окно браузера. Аргумент `auto` устанавливает ширину исходя из типа и содержимого элемента. Например, для тега `DIV` ширина по умолчанию задается как 100 %.

Значение по умолчанию

`auto`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П53

```
<html>
<body>
<div style="width: 80%; background: #fc0; padding: 7px; border: 1px solid
#ccc">
<p style="width: 300px">Lorem ipsum dolor sit amet, consectetuer
adipiscing elit, sed diem nonummy nibh euismod tincidunt ut lacreet
dolore magna aliguam erat volutpat.</p>
</div>
</body>
</html>
```

Применяется

Ко всем элементам, за исключением встроенных незаменяемых элементов, а также к колонкам и группам таблицы.

Примечание

Браузер Internet Explorer 4 не применяет свойство `width` к встроенным элементам, спискам и некоторым элементам форм вроде флажков, переключателей и текстовых полей. Netscape 4 не применяет указанное свойство к полям форм и заменяемым элементам. В браузере Opera 3.5 этот параметр не дает никакого эффекта для встроенных элементов, полей форм и таблиц.

Параметр *word-spacing*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Нет	Нет	Нет	Да	Да	Да	Нет	Да	Да	Да	Да	Да

Описание

Устанавливает интервал между словами. Если установлен параметр выравнивания `justify`, то атрибут `word-spacing` не действует, поскольку интервал между словами будет установлен принудительно, чтобы строка текста была выровнена по правому и левому краю.

Синтаксис

```
word-spacing: значение | normal
```

Аргументы

В качестве значений используются любые единицы длины, принятые в CSS, — например пикселы (px), дюймы (in), пункты (pt) и др. Значение параметра может быть и отрицательным, но следует проверять его работоспособность в разных браузерах. Процентная запись не применяется.

Значение по умолчанию

normal

Наследование

Значения, присвоенные данному параметру, наследуются.

Пример

Листинг П54

```
<html>
<body>
<p style="word-spacing: 20px">Lorem ipsum dolor sit amet, consectetuer
adipiscing elit, sed diem nonummy nibh euismod tincidunt ut lacreet
dolore magna aliguam erat volutpat. Ut wisis enim ad minim veniam, quis
nostrud exerci tution ullamcorper suscipit lobortis nisl ut aliquip ex ea
commodo consequat.</p>
</body>
</html>
```

Применяется

Ко всем элементам.

Параметр *z-index*

Браузер	Internet Explorer						Netscape Navigator			Opera		
Платформа	Windows				Macintosh		Все платформы			Все платформы		
Версия	4.0	5.0	5.5	6.0	4.0	5.0	4.x	6.0	7.0	3.5	5.0	7.0
Поддерживается	Да	Да	Да	Да	Да	Да	Да	Да	Да	Нет	Да	Да

Описание

Любые позиционированные элементы на веб-странице могут накладываться друг на друга в определенном порядке, имитируя тем самым третье измерение, перпендикулярное экрану. Каждый элемент может находиться как ни-

же, так и выше других объектов веб-страницы, их размещением по z-оси и управляет атрибут

`z-index`.

Синтаксис

```
z-index: число | auto
```

Аргументы

В качестве аргумента допустимо использовать положительное или отрицательное целое число, а также ноль. Чем больше значение, тем выше находится элемент по сравнению с теми элементами, у которых оно меньше. При равном значении `z-index` на переднем плане находится тот элемент, который в коде HTML описан ниже. Хотя спецификация и разрешает использовать отрицательные значения `z-index`, но такие элементы не отображаются в браузерах Netscape, Mozilla и Firefox.

Кроме числовых значений, применяется `auto` — порядок элементов в этом случае строится автоматически исходя из их положения в коде HTML и принадлежности к родителю, поскольку дочерние элементы имеют тот же номер, что их родительский элемент.

Значение по умолчанию

`auto`

Наследование

Значения, присвоенные данному параметру, не наследуются.

Пример

Листинг П55

```
<html>
<body>
Слой 1 наверху
<div style="position: relative; font-size: 50px; z-index: 2;
color: navy">Слой 1</div>
<div style="position: relative; top: -55px; left: 5px; color: orange;
font-size: 70px; z-index: 1">Слой 2</div>
Слой 2 наверху
<div style="position: relative; font-size: 50px; z-index: 3;
color: navy">Слой 1</div>
<div style="position: relative; top: -55px; left: 5px; color: orange;
font-size: 70px; z-index: 4">Слой 2</div>
```

```
</body>
</html>
```

На рис. П30 представлен результат данного примера. Обратите внимание на большой вертикальный промежуток между первой и второй надписью. Дело в том, что при смещении элемента со своей исходной точки место, которое он занимал, остается. Изменение местоположения текста происходит за счет использования относительного позиционирования, а также атрибутов `top` и `left`.

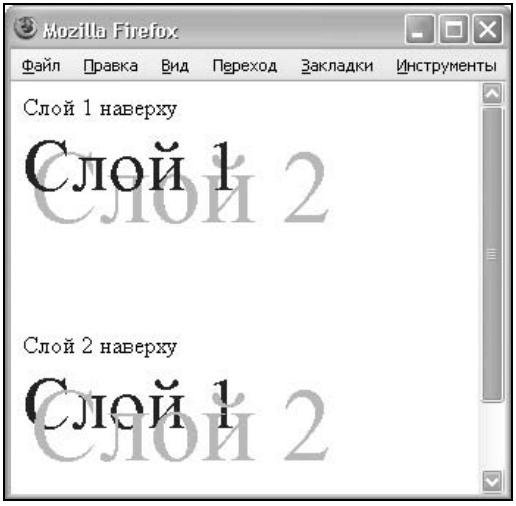


Рис. П30. Последовательность наложения элементов друг на друга

Применяется

К любым позиционированным элементам.

Примечание

Список, созданный с помощью тега `SELECT`, в браузере Internet Exolorer всегда отображается поверх других элементов, несмотря на значение `z-index`. Только начиная с версии 5.5 Internet Explorer допускает применение атрибута `z-index` к фреймам (тег `FRAME`) и плавающим фреймам (`IFRAME`). Браузер Netscape 4.x поля форм всегда отображает поверх всех других элементов веб-страницы независимо от заданного `z-index`.

Предметный указатель

Е

И

К

Л

М

Н

О

П

P

C

Продолжение рубрики см. на с. 446

T

Ф

Ц

Ш

Э

Я