Юрий Едомский

Техника Web-дизайна

для студента

Санкт-Петербург
«БХВ-Петербург»
2005

УДК 681.3.06
ББК 32.973.26-018.2
Е32

Едомский Ю.

Е32 Техника Web-дизайна для студента. — СПб.: БХВ-Петербург, 2005. — 400 с.: ил.

ISBN 5-94157-742-7

Рассмотрен широкий круг вопросов, связанных с разработкой web-сайтов с использованием HTML и CSS, начиная от простых страниц до создания интерактивных и динамических сайтов. Даны основы компьютерной графики, описана работа с программами Photoshop и ImageReady в плане подготовки изображений к публикации на сайте. Рассмотрены вопросы редактирования сайтов в программе Dreamweaver. Для создания динамических страниц приведены необходимые сведения о программировании на JavaScript.

Книга будет полезна студентам при изучении интернет-технологий, написании курсовых и дипломных проектов, а также преподавателям и всем желающим создавать самостоятельно интерактивные и динамические страницы.

Для широкого круга пользователей

УДК 681.3.06
ББК 32.973.26-018.2

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Евгений Рыбаков*
Зав. редакцией	*Григорий Добин*
Редактор	*Елена Сухогузова*
Компьютерная верстка	*Екатерины Трубниковой*
Корректор	*Татьяна Кошелева*
Дизайн обложки	*Игоря Цырульникова*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 06.10.05.
Формат $60\times90^{1}/_{16}$. Печать офсетная. Усл. печ. л. 32,25.
Тираж 3000 экз. Заказ №
"БХВ-Петербург", 190005, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.02.953.Д.006421.11.04 от 11.11.2004 г. выдано Федеральной службой
по надзору в сфере защиты прав потребителей и благополучия человека.

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

ISBN 5-94157-742-7

© Едомский Ю. Е.
© Оформление, издательство "БХВ-Петербург", 2005

Содержание

Введение

Книга предназначена для широкого круга читателей — от начинающих создателей web-страниц до имеющих определенный опыт, но желающих расширить свои знания. Прежде всего, она адресована студентам, работающим над курсовыми и дипломными проектами. Полезную информацию найдут для себя и школьники, делающие первые шаги в освоении сети Интернет, а также специалисты, желающие заявить о себе и своих достижениях в Интернете, но не имеющие достаточных знаний в области компьютерных технологий.

Требования к новичкам не слишком велики:

- владение элементарными приемами работы с операционной системой Windows, а именно умение открыть нужную программу, создать папку, сохранить в ней файл или найти ранее сохраненный файл;
- элементарные навыки практической работы в Интернете: наличие представления о том, что для просмотра документов, расположенных в Интернете или создаваемых для этих целей, предназначена программа браузер и хотя бы небольшой опыт работы с такой программой.

Главное достоинство книги состоит в том, что в ней можно найти ответы на многие вопросы, которые обычно рассредоточены по разным источникам. Например, для подготовки изображений к публикации в Интернете необходима литература по компьютерной графике, с целью ускорения работы потребуется знакомство с редактором web-сайтов, а при создании динамических страниц понадобится литература по программированию. Обычно специальная литература по каждому из этих направлений имеет

большой объем и достаточно сложна для начинающих. Несмотря на широкий спектр излагаемого материала и небольшой объем книги, важные вопросы излагаются достаточно подробно, а сокращение достигается за счет материалов, без которых вполне можно обойтись.

Прочитав первые две главы и хотя бы часть третьей, новичок уже сможет приступить к созданию собственных web-страниц. Дочитав третью главу, можно дополнить свои страницы рисунками причудливых форм, анимированными изображениями и специально подготовленными фоновыми рисунками. В целом *главы 2* и *3* могут быть полезны разработчикам разного уровня.

В *главе 2* приводится первая часть описания программы создания сайтов Macromedia Dreamweaver MX 2004. Знакомство с этой программой продолжится и в последующих главах, когда ее использование может существенно ускорить работу над сайтом.

Материалы *главы 3* будут полезны всем, кто раньше не занимался компьютерной графикой. В ней достаточно подробно рассмотрены вопросы подготовки изображений к публикации на сайте как в программе Adobe Photoshop, так и в программе ImageReady, поставляемой фирмой Adobe вместе с программой Photoshop и существенно расширяющей ее возможности именно по созданию изображений для web-страниц.

Новичку не следует торопиться с переходом к последующим главам. Получив навыки практической работы по созданию простых страниц с иллюстрациями и поняв, что не удается расположить рисунки и фрагменты текста в произвольных местах страницы, можно переходить к созданию таблиц (*глава 4*).

Назначение таблиц состоит не только в привычном расположении структурированных данных в определенном порядке, но и в осуществлении компоновки элементов страницы. Используя для табличной компоновки страницы программу Dreamweaver, можно значительно облегчить эту работу.

В *главе 5* предлагается познакомиться с еще одним методом компоновки — фреймами. Здесь необходимо понять, в каких случаях лучше использовать таблицы, а в каких фреймы и в чем их принципиальное отличие.

Глава 6 имеет достаточно обособленный характер. Основное назначение форм, рассматриваемых в этой главе, — создание об-

ратной связи между пользователями и владельцем сайта. После прочтения материалов *главы 6* можно не только заявить о себе в сети Интернет, но и облегчить посетителю сайта отправку отзывов на размещенные материалы. Даже если в ближайшее время вы не планируете организацию такого диалога, пропускать этот материал не следует, так как формы используются в качестве объектов программирования в *главе 8*.

Глава 7 знакомит со свойствами каскадных таблиц стилей (CSS), которые позволяют существенно дополнить и расширить возможности языка HTML и в случае совместного использования с программой-сценарием на языке JavaScript создавать динамические страницы. По всей видимости, на изучение этой главы придется потратить больше времени, чем на изучение некоторых предыдущих, но это, безусловно, окупится теми практическими возможностями, которые станут доступными. В конце главы можно будет познакомиться с еще одним методом компоновки страницы с использованием слоев.

Главу 8 следует рассматривать как введение в программирование на языке JavaScript. Программирование на любом из алгоритмических языков, в том числе и на JavaScript, является достаточно сложной задачей особенно на ранней стадии изучения. Называть себя программистом после прочтения главы, и даже после практического выполнения приведенных там примеров, будет преждевременным. Тем не менее содержащиеся в *главе 8* сведения позволяют получить практические навыки по созданию простых, но достаточно полезных программ, познакомиться с основными понятиями программирования. Тот, кому изложенный в главе материал покажется слишком сложным, сможет воспользоваться приведенными примерами, не вдаваясь в тонкости их реализации. Тот же, кто почувствует интерес к программированию, прежде всего, должен попытаться решить несколько собственных задач, базирующихся на предлагаемых примерах. В случае успеха в дальнейшем можно приступить к изучению специальной литературы по программированию, а полученные начальные знания позволят более осмысленно выбрать специальную литературу и облегчат процесс ее освоения.

Глава 9 посвящена применению мультимедийных эффектов, разработанных фирмой Microsoft и поддерживаемых браузером

MS Internet Explorer. Начиная с версии 5.5 браузера, фирма Microsoft изменила синтаксис задания фильтров, принцип функционирования динамических фильтров, расширила перечень фильтров. Все эти изменения нашли отражение в *главе 9*.

Каждая глава содержит множество примеров, иллюстрирующих пройденный материал. Все примеры работоспособны в браузере Microsoft Internet Explorer 6.0. Этот браузер выбран не случайно. Дело в том, что он является не только самым популярным среди посетителей Интернета, но и, как правило, превосходит конкурентов по своим возможностям. При попытке просмотра созданных документов в браузерах более ранних версий, а в особенности других производителей, желаемого результата можно не увидеть. Это, прежде всего, относится к материалам *глав 8* и *9*.

Для успешного освоения материалов книги читателю следует установить на своем компьютере программы Macromedia Dreamweaver MX 2004 и Adobe Photoshop 8.0 CS. Это последние на момент издания книги версии программ. Конечно, отличия последних версий программ от предыдущих не столь значительны, чтобы не суметь разобраться в их работе по приведенному описанию. Тем не менее определенные отличия есть, что потребует дополнительного времени для освоения программ. Кроме того, со временем все равно придется осваивать более новые версии, а делать это, перепрыгивая несколько ступеней, сложнее.

В книге не рассматриваются вопросы применения современной и весьма перспективной технологии Flash фирмы Macromedia. Возможно, что это ее недостаток, который со временем придется исправить. По всей видимости, читатель, получивший определенный опыт и уверенность в своих силах, сам заинтересуется технологией Flash и воспользуется для этого специальной литературой.

Глава 1

Создание простых web-страниц

1.1. Основные понятия и определения

Документы, которыми осуществляется обмен информацией в Интернете, обычно представляют собой текстовые файлы, написанные с использованием специального языка HTML (HyperText Markup Language, язык разметки гипертекста). Язык HTML — это набор меток, которые расставляются по тексту и определяют внешний вид документа в окне браузера.

Браузеры — специальные программы, предназначенные для просмотра документов, размеченных HTML-метками. В настоящее время самым распространенным браузером является Microsoft Internet Explorer.

Метки — небольшие последовательности символов, ограниченные угловыми скобками и предлагающие браузеру определенные действия (команды). Например, встретив в тексте HTML-документа метку, предписывающую сделать фрагмент текста полужирным, браузер выполнит это требование. Точно также, используя соответствующие метки, можно предложить браузеру изменить размер и цвет шрифта, выровнять текст, вставить рисунки и т. д. Некоторые метки применяются попарно, одна из них называется *открывающей*, другая — *закрывающей*. Действие метки будет распространяться на фрагмент документа, заключенный между открывающей и закрывающей меткой. В англоязычной литературе метки именуются словом *tag* (ярлык, бирка, признак).

Этот термин механически перешел и в отечественную литературу, причем часть авторов пишет *тэг*, а не менее значительная часть — *тег*. Слово "метка" более точно отражает существо вопроса, поэтому мы будем использовать именно этот термин.

Открывающая метка может иметь следующую структуру:

```
<имя [параметр][="значение"] [параметр][="значение"] …>
```

Элементы, написанные в квадратных скобках, являются необязательными составляющими метки. Это означает, что метки могут не содержать *параметры* или содержать параметры без значений. Но если параметры есть, квадратные скобки писать не нужно. Имя метки, а также параметры отделяются друг от друга пробелами. После символа "<" и перед символом ">" пробелы не ставятся. Многоточие означает, что параметров может быть много. *Значения параметров* записываются в прямых кавычках, однако, если значение параметра не содержит пробелов, то кавычки можно не писать.

Закрывающая метка содержит только имя, перед которым ставится правый слеш:

```
</имя>
```

Как видим, закрывающая метка не содержит никаких параметров. Ее назначение — ограничение действия открывающей метки. Не всякая метка имеет в паре закрывающую метку.

Для написания меток используются только латинские буквы. Допускается использование как строчных, так и заглавных букв.

Структура HTML-документа:

```
<HTML>
<HEAD>
<TITLE>
Заголовок окна документа
</TITLE>
</HEAD>
<BODY>
Тело документа
</BODY>
</HTML>
```

Такую структуру должен иметь любой HTML-документ. Она включает три пары меток с именами `HTML`, `HEAD`, `BODY`, которые в данном случае не имеют параметров.

В головной части документа, то есть внутри пары меток `<HEAD>…</HEAD>`, могут размещаться другие метки. На первых порах будем использовать только единственную и обязательную пару меток `<TITLE>…</TITLE>`, предназначенную для размещения заголовка окна браузера. После открытия документа текст заголовка будет воспроизводиться в строке заголовка окна браузера (синяя верхняя полоса окна).

В теле документа размещается текст и метки, определяющие его внешний вид в окне браузера.

1.1.1. Задание цвета

В HTML для задания цвета используется 6-разрядное шестнадцатеричное число, перед которым ставится символ #. В качестве цветовой модели используется RGB (Red — красный, Green — зеленый, Blue — синий). Шестнадцатеричная система счисления имеет 16 цифр (0, 1, , 9, A, B, C, D, E, F). Для задания каждой из трех составляющих цвета отводится два разряда шестнадцатеричного числа, что позволяет задать значения от 00 до FF, соответствующие значениям 0 255 в десятичной системе счисления. Например, запись `#FF0000` означает, что красная составляющая цвета имеет максимальное значение FF (255 в десятичной системе), т. е. максимальную яркость, а остальные составляющие цвета отсутствуют. Очевидно, что таким образом задан ярко-красный цвет. Для задания темно-красного цвета используется запись `#800000`. В этом случае красная составляющая имеет значение 80 (128 в десятичной системе), остальные составляющие по-прежнему отсутствуют. Изменяя значения отдельных составляющих, можно задать 16777216 различных цветов. Например: `#0000FF` — синий, `#00FFFF` — голубой, `#FFFF00` — желтый, `#808080` — серый, `#FFFFFF` — белый, `#000000` — черный. Современные браузеры позволяют не писать перед числом символ #, однако лучше этой возможностью не пользоваться.

Для задания цвета также можно использовать английские названия цветов. Например: `red` — красный, `cyan` — голубой, `magenta` — пурпурный, `yellow` — желтый, `white` — белый, `black` — черный.

1.1.2. Задание размера шрифта

В HTML размер шрифта задается в условных единицах от 1 до 7. Разными браузерами шрифт одного и того же размера может воспроизводиться по разному. На рис. 1.1 показано воспроизведение шрифтов браузером Microsoft Internet Explorer.

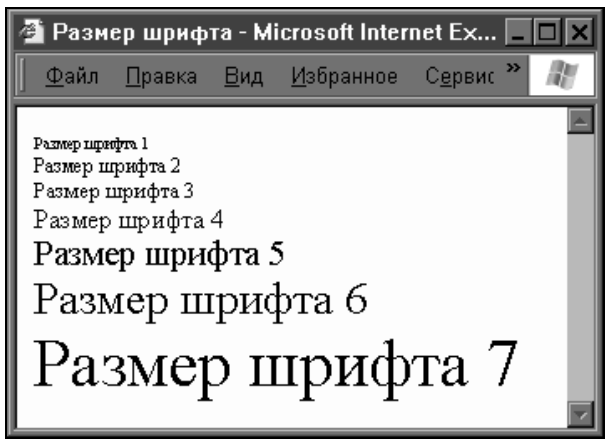


Рис. 1.1. Воспроизведение шрифтов разного размера браузером Microsoft Internet Explorer

Язык HTML не позволяет задать размеры шрифта, отличающиеся от приведенных выше. Для задания произвольных размеров необходимо использовать свойства *каскадных таблиц стилей (CSS)* (*см. главу 7*).

1.2. Создание HTML-документа в Блокноте

Блокнот (Notepad) — простой текстовый редактор, входящий в стандартный набор компонентов MS Windows. Откроем Блокнот и наберем следующий текст, представленный в листинге 1.1.

Листинг 1.1. Содержимое простого HTML-документа

```
<HTML>
<HEAD>
<TITLE>Моя первая страница</TITLE>
</HEAD>
<BODY>
Мой первый HTML-документ
Здесь помещается текст моего первого HTML-документа, который
пока не содержит форматирования и воспроизводится браузером с
параметрами шрифта, принятыми по умолчанию.
</BODY>
</HTML>
```

После завершения набора документа выполним команду **Сохранить** меню **Файл**. В открывшемся диалоговом окне **Сохранить как** выберем папку, в которую хотим сохранить документ. В строке **Имя файла** введем имя с обязательным расширением htm или html (например mydoc.htm). В раскрывающемся списке **Тип файла** выберем пункт **Все файлы** и нажмем кнопку **Сохранить**. После этого можно закрыть блокнот, отыскать созданный документ в папке и дважды щелкнуть по имени файла. Благодаря расширению htm документ откроется в браузере, и вы увидите заголовок окна, набранный после метки `<TITLE>`, и текст, набранный после метки `<BODY>`.

Дальнейшую работу с документом можно продолжать, не закрывая окно браузера. Для этого нужно выбрать команду **Просмотр HTML-кода** меню **Вид**. Откроется окно блокнота с текстом создаваемого документа. Дополним текст документа фразой: `Продолжаю редактировать мой первый документ`. Дадим команду **Сохранить** и, свернув блокнот, перейдем к окну браузера. Выполним команду **Обновить** меню **Вид** (дублируется кнопкой **Обновить** на панели инструментов) и убедимся в том, что в тексте документа произошли изменения. После этого примера дальнейшая работа с документом становится очевидной. Она будет заключаться в открытии Блокнота, внесении в текст документа нужных изменений, сохранении документа и обновлении содержимого окна браузера.

1.3. Метки форматирования шрифта

Чтобы отформатировать шрифт нашего документа, то есть изменить его внешний вид, нужно расставить по тексту *метки форматирования шрифта*, которые и будут рассмотрены в этом разделе.

Метки форматирования шрифта принято разделять на метки *физического* и *логического* форматирования. Метки физического форматирования позволяют задать конкретные параметры шрифта, например, размер, цвет, начертание и так далее, которые будут реализованы браузером. Метки логического форматирования определяют лишь характер помечаемого текста, например: заголовок, аббревиатура, цитата и так далее, а браузер отобразит текст с теми физическими характеристиками, которые предусмотрены для подобного текста. Если одного и того же результата можно достичь как с помощью меток физического форматирования, так и меток логического форматирования, следует отдать предпочтение меткам логического форматирования.

1.3.1. Метки логического форматирования шрифта

В данном разделе рассматриваются метки логического форматирования, их запись и результат отображения в браузере.

- Увеличение размера шрифта: `<BIG>...</BIG>`.

  Пример записи: `<BIG>Текст</BIG>`.

  Размер шрифта увеличивается на единицу относительно текущего уровня.

- Уменьшение размера шрифта: `<SMALL>...</SMALL>`.

  Пример записи: `<SMALL>Текст</SMALL>`.

  Размер шрифта уменьшается на единицу относительно текущего уровня.

- Разметка аббревиатур: `<ACRONYM>...</ACRONYM>`.

  Пример записи: `<ACRONYM TITLE="Hypertext Markup Language">HTML</ACRONYM>`.

  В метке можно использовать параметр TITLE, который позволяет задать полную форму записи аббревиатуры, появ-

ляющуюся в браузере в виде всплывающей подсказки, при наведении указателя мыши на текст.

- Разметка цитат `<CITE>...</CITE>`.

  Пример записи: `<CITE>Текст</CITE>`.

  Текст выводится курсивом.

- Разметка удаленного текста: `<DEL>...</DEL>`.

  Пример записи: `<DEL>Текст</DEL>`.

  Текст перечеркивается горизонтальной линией.

- Разметка терминов или определений: `<DFN>...</DFN>`.

  Пример записи: `<DFN>Текст</DFN>`.

  Текст выводится курсивом.

- Разметка текста вставляемого в документ и изменяющего его по отношению к предыдущей версии: `<INS>...</INS>`.

  Пример записи: `<INS>Текст</INS>`.

  Текст подчеркивается.

- Разметка важных фрагментов текста: `<EM>...</EM>`.

  Пример записи: `<EM>Текст</EM>`.

  Текст выводится курсивом.

- Разметка текста, вводимого пользователем с клавиатуры: `<KBD>...</KBD>`.

  Пример записи: `<KBD>Текст</KBD>`.

  Текст выводится моноширинным шрифтом (все символы одинаковой ширины).

- Разметка важных фрагментов текста: `<STRONG>...</STRONG>`.

  Пример записи: `<STRONG>Текст</STRONG>`.

  Текст выводится полужирным шрифтом.

- Разметка заголовков (6 уровней):

  `<H1>...</H1>`

  `<H2>...</H2>`

  ...

  `<H6>...</H6>`

Пример записи: `<H4>Заголовок</H4>`.

Текст выводится полужирным шрифтом. Размер шрифта зависит от выбранного уровня заголовка (`<H1>` — самый крупный, `<H6>` — самый мелкий). Метки могут иметь параметр ALIGN со значениями: `LEFT` — выравнивание заголовка по левому краю (действует по умолчанию), `RIGHT` — выравнивание заголовка по правому краю, `CENTER` — центрирование заголовка.

Пример записи: `<H1 ALIGN=CENTER>Заголовок</H1>`.

1.3.2. Метки физического форматирования шрифта

Наряду с метками логического форматирования, используются и метки физического форматирования, которые позволяют задать конкретные параметры шрифта не только для всего текста, но и для отдельных его элементов.

- Задание гарнитуры, размера и цвета шрифта всего текста: `<BASEFONT>`.

  Пример записи: `<BASEFONT FACE="Times New Roman" SIZE=5 COLOR=Green>Текст`.

  Метка не имеет закрывающей метки и действует на весь текст, расположенный ниже.

  Описание параметров:

  - `FACE="значение"` — позволяет указать гарнитуру (рисунок) шрифта;
  - `SIZE="значение"` — размер шрифта (целое число от 1 до 7);
  - `COLOR="значение"` — цвет шрифта.

  В данном примере запись значения параметра `FACE` без кавычек не допустима, так как состоит из нескольких слов, а значения параметров `SIZE` и `COLOR` можно писать без кавычек. Задавая гарнитуру шрифта в параметре `FACE`, следует помнить, что такой гарнитуры на компьютере пользователя может не оказаться. В этом случае текст будет воспроизводиться гарнитурой, предусмотренной по умолчанию, что может существенно изменить вид документа. В связи с этим иногда в

параметре FACE указывают через запятую в порядке убывания приоритета несколько гарнитур, относящихся к одному семейству, заканчивая список именем семейства, к которому они принадлежат. Например:

```
<BASEFONT FACE="Arial, Helvetica, sans-serif">Текст.
```

- Задание гарнитуры, размера и цвета шрифта фрагмента текста: <FONT>…</FONT>.

  Пример записи: <FONT FACE="Helvetica" SIZE=-1 COLOR=Red>Текст</FONT>.

  Параметры аналогичны параметрам метки <BASEFONT>, за исключением параметра SIZE. Запись значения параметра SIZE со знаком минус означает, что размер шрифта уменьшается на соответствующую величину относительно заданного по умолчанию или заданного в метке <BASEFONT>. Запись без знака означает, что размер задан абсолютно.

- Разметка полужирного шрифта: <B>…</B>.

  Пример записи: <B>Текст</B>.

  Рекомендуемый аналог <STRONG>.

- Разметка курсивного шрифта: <I>…</I>.

  Пример записи: <I>Текст</I>.

  Рекомендуемые аналоги <EM>, <DFN>.

- Разметка подчеркнутого шрифта: <U>…</U>.

  Пример записи: <U>Текст</U>.

  Рекомендуемый аналог: <INS>.

- Разметка моноширинного шрифта: <TT>…</TT>.

  Пример записи: <TT>Текст</TT>.

  Рекомендуемый аналог: <KBD>.

- Разметка шрифта, перечеркнутого горизонтальной линией: <STRIKE>…</STRIKE> или <S>…</S>.

  Пример записи: <STRIKE>Текст</STRIKE>.

  Рекомендуемый аналог: <DEL>.

- Создание подстрочного индекса: _….

Пример записи: `X<SUB>2</SUB>`.

Результат действия: X_2.

- Создание надстрочного индекса: `<SUP>…</SUP>`.

  Пример записи: `Y<SUP>3</SUP>`.

  Результат действия: Y^3.

Пример форматирования шрифта

Созданный ранее в блокноте первый HTML-документ не содержал форматирования и воспроизводился браузером шрифтом, принятым по умолчанию. Пришла пора создать второй документ, похожий по содержанию, но использующий некоторые из рассмотренных выше меток для форматирования шрифта (листинг 1.2).

Листинг 1.2. HTML-документ, содержащий метки форматирования шрифта

```
<HTML>
<HEAD>
<TITLE>Вторая страница</TITLE>
</HEAD>
<BODY>
<BASEFONT SIZE=4>
<H1 ALIGN=CENTER>Мой второй HTML-документ</H1>
Здесь помещается текст моего второго <ACRONYM TITLE="Hyper Text
Markup Language">HTML</ACRONYM>-документа, который содержит
заголовок первого уровня и аббревиатуру с текстом всплывающей
подсказки. Для всего текста установлен размер 4. <STRONG>Эти
слова воспроизводятся полужирным шрифтом.</STRONG>
</BODY>
</HTML>
```

Вид документа в окне браузера показан на рис. 1.2.

1.3.3. Вставка специальных символов

К *специальным символам* относятся служебные символы, используемые в языке HTML (< — меньше, > — больше, & — амперсанд, " — кавычки), а также некоторые специальные знаки.

В HTML для отображения специальных символов предусмотрены последовательности символов, некоторые из которых приведены в табл. 1.1.

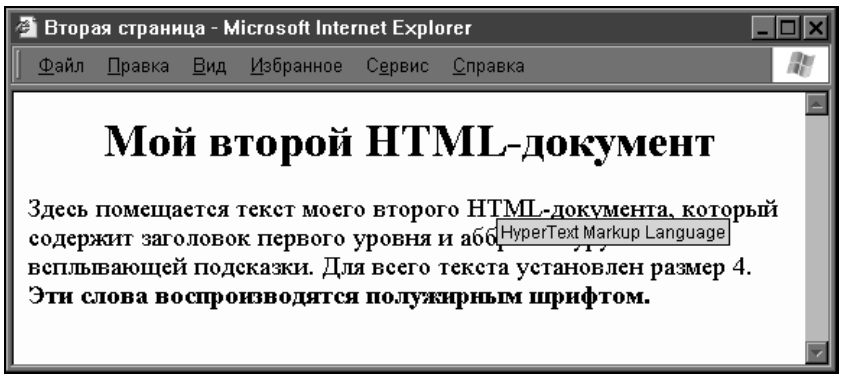


Рис. 1.2. Результат форматирования шрифта

Таблица 1.1. *Последовательность для отображения некоторых символов*

Последовательность	Символ
`<`	< знак меньше
`>`	> знак больше
` `	неразрывный пробел (non-breaking space)[1]
`&`	& амперсанд
`&quot;`	" прямые кавычки
`&copy;`	© знак авторского права (Copyright)
`&reg;`	® знак зарегистрированной торговой марки (Registered)

[1] *Неразрывный пробел называется так потому, что слова, между которыми он находится, не могут быть разнесены браузером на разные строки, и* если *не будут помещаться на одной строке, то будут перенесены вместе. Поэтому его, в частности, можно использовать для разделения величины и единицы измерения. Например:* `120 км/час`.

Таблица 1.1 *(окончание)*

Последовательность	Символ
`&laquo;`	« левая парная кавычка
`&raquo;`	» правая парная кавычка
`&mdash;`	— длинное тире

Существует и более универсальный способ отображения символов. Нужные символы вводятся в документ при помощи специального кода, который включает знак амперсанда (`&`), знак номера (`#`), десятичный код символа и точку с запятой. Этот способ применяется также для отображения символов, которые невозможно ввести с клавиатуры. Примеры специальных кодов некоторых символов приведены в табл. 1.2.

Таблица 1.2. *Специальные коды некоторых символов*

Код	Символ
` `	неразрывный пробел
`&#169;`	© — знак авторского права
`&#171;`	« — левая кавычка
`&#187;`	» — правая кавычка

Чтобы определить код символа, в **Главном меню Windows** (кнопка **Пуск**) нужно воспользоваться командой **Программы | Стандартные | Служебные | Таблица символов**. В диалоговом окне **Таблица символов** в раскрывающемся списке **Шрифт** выбирается название гарнитуры, после чего появится список содержащихся в данной гарнитуре символов. Выделив щелчком нужный символ, в нижней части диалогового окна прочтем код символа. Символы некоторых гарнитур кодируются кодом ASCII (American Standard Code for Information Interchange), для которого используются два разряда шестнадцатеричного числа (записываются после символов 0x). Символы других гарнитур кодируются Юникодом (Unicode), использующим четырехразрядные шестнадцатеричные числа (записываются после символов U+). В обоих случаях их нужно предварительно перевести в десятичный код, для чего можно использовать Калькулятор (**Программы |**

Стандартные | Калькулятор) в режиме инженерных расчетов. Для примера выберите гарнитуру Arial и убедитесь в правильности кодов, приведенных в табл. 1.2.

Если гарнитура шрифта, используемая для основного текста, не содержит нужного символа, то для использования символа в HTML-документе требуется изменить гарнитуру и после этого отобразить нужный символ при помощи кода. Например, пусть в HTML-документе необходимо использовать символ "→". По **Таблице символов** определяем, что такой символ содержится в гарнитуре **Symbol** и кодируется шестнадцатеричным кодом `AE`. Переведя `AE` в десятичный, код получим `174`. В HTML-документе необходимо написать:

```
<FONT FACE="symbol">&#174;</FONT>
```

1.4. Метки форматирования текста

Под форматированием текста подразумевается изменение внешнего вида документа путем разбиения его на абзацы, выравнивания абзацев, проведения горизонтальных разделительных линий, создания списков.

- Создание абзаца: `<P>…</P>`.

  Пример записи: `<P ALIGN=RIGHT>Текст абзаца</P>`.

  Параметр `ALIGN` позволяет задать выравнивание строк абзаца и может иметь следующие значения:

  - `LEFT` — выравнивание по левому краю (действует по умолчанию);
  - `RIGHT` — выравнивание по правому краю;
  - `CENTER` — центрирование строк;
  - `JUSTIFY` — выравнивание по ширине (по правой и левой сторонам).

Примечание

Закрывающая метка `</P>` не обязательна.

Абзацы отделяются друг от друга интервалами, равными пустой строке.

Первая строка абзаца не имеет отступа.

- Переход на новую строку в пределах абзаца: <BR>.

  Текст, располагающийся после метки <BR>, будет воспроизводиться с начала новой строки, при этом строки абзаца сохранят выравнивание, заданное для всего абзаца.

- Вставка текста, предварительно отформатированного в текстовом редакторе: <PRE>…</PRE>.

  Пример записи: <PRE>Текст</PRE>

  Использованные в предварительно отформатированном тексте переходы на новую строку, пробелы, символы табуляции сохранятся при выводе его браузером. Текст будет выводиться моноширинным шрифтом;

- Отображение фрагмента текста с отступом: <BLOCKQUOTE>… </BLOCKQUOTE>.

  Пример записи: <BLOCKQUOTE>Текст</BLOCKQUOTE>.

  Основное назначение <BLOCKQUOTE> — выделение фрагментов текста (например цитат) путем изменения их положения относительно остального текста. Текст сдвигается на несколько позиций вправо, перед и после текста устанавливаются интервалы.

- Создание маркированного списка: <UL>…</UL>.

  Список представляет собой последовательность абзацев, начала которых при выводе списка браузером будут отмечены специальным значком (маркером). До и после списка будут добавлены интервалы.

  Пример записи:

```
<UL TYPE=disc>
 <LI>Первый абзац списка
 <LI>Второй абзац списка
 …
 <LI>Последний абзац списка
</UL>
```

  Метка <LI> отмечает начало абзаца, что обеспечивает его вывод с новой строки.

Параметр TYPE позволяет задать тип маркера и может принимать значения:

- `disc` — закрашенный кружок;
- `circle` — не закрашенный кружок (действует по умолчанию);
- `square` — закрашенный квадрат.

Следует обратить внимание, что вопреки принципу нечувствительности к регистру, характерному для HTML, перечисленные выше значения параметра `TYPE`, нужно писать только строчными буквами.

Допускается использование параметра `TYPE` в метке `<LI>`.

В качестве маркеров можно использовать специальные символы. Для этого можно применить следующий способ:

```
<UL>
<FONT FACE="wingdings">&#74;</FONT>Первая строка списка<BR>
<FONT FACE="wingdings">&#76;</FONT>Вторая строка списка<BR>
<FONT FACE="wingdings">&#79;</FONT>Третья строка списка<BR>
<FONT FACE="wingdings">&#60;</FONT>Четвертая строка списка<BR>
<FONT FACE="wingdings">&#38;</FONT>Пятая строка списка
</UL>
```

Дополнив фрагмент соответствующими метками, в окне браузера увидим результат, приведенный на рис. 1.3.

Чтобы в качестве маркера использовать рисунок, нужно вместо метки `<FONT>` поставить метку рисунка `<IMG>`. Например:

```
<UL>
<IMG SRC="имя файла рисунка">Первая строка списка<BR>
<IMG SRC="имя файла рисунка">Вторая строка списка
</UL>
```

О размещении рисунков *см. в главе 3*.

- Создание нумерованного списка: `<OL>…</OL>`.

Пример записи:

```
<OL TYPE=a START=3>
 <LI>Первый элемент списка
```

```
  <LI>Второй элемент списка
  ...
  <LI>Последний элемент списка
</OL>
```

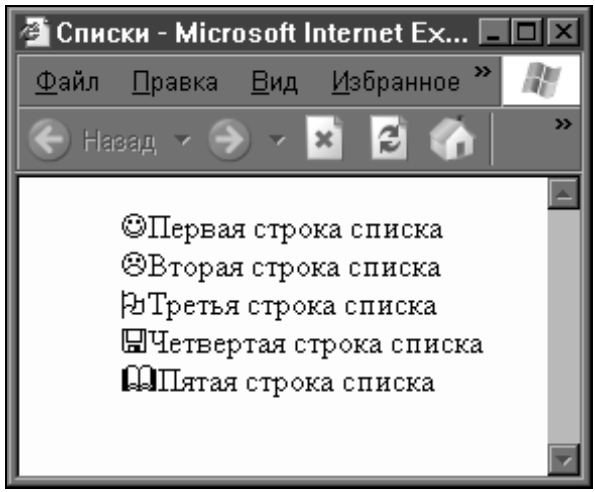


Рис. 1.3. Маркированный список с различными символами в качестве маркеров

В отличие от маркированного списка, абзацы входящие в список, нумеруются. Вид нумерации задается параметром `TYPE`, который может принимать следующие значения:

- `A` — нумерация большими латинскими буквами;
- `a` — нумерация маленькими латинскими буквами;
- `I` — нумерация большими римскими цифрами;
- `i` — нумерация маленькими римскими цифрами;
- `1` — нумерация арабскими цифрами (действует по умолчанию).

По умолчанию нумерация списка задается начиная с первого номера. В случае продолжения нумерации параметр `START` позволяет задать начальный номер списка. Для всех видов нумерации номер должен быть указан арабской цифрой.

Метка `<LI>` может иметь параметры: `TYPE` — изменяет вид номера только для данного элемента списка, `VALUE` — изменяет вид номера данного элемента списка с изменением нумерации последующих абзацев.

Пример записи:

```
<OL>
<LI> Первый элемент списка
<LI TYPE=a VALUE=1> Первый элемент второго списка
<LI TYPE=a> Второй элемент второго списка
<LI VALUE=2> Второй элемент списка
<LI> Третий элемент списка
</OL>
```

В окне браузера такой список будет выглядеть как на рис. 1.4.

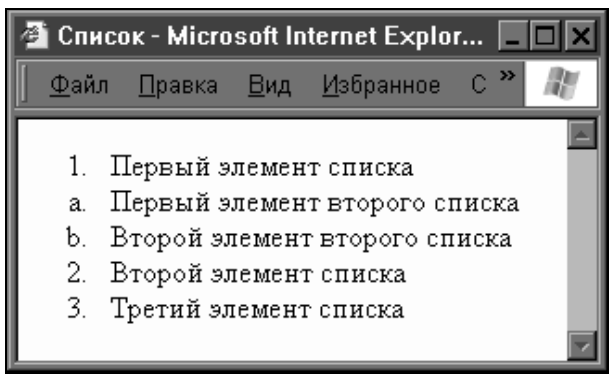


Рис. 1.4. Нумерованный список с двумя видами нумерации

Допускается создание многоуровневого списка, когда один список вкладывается в другой независимо от типа. Список более низкого уровня вложения будет изображаться в браузере смещенным вправо относительно списка предыдущего уровня вложения (листинг 1.3).

Листинг 1.3. Создание многоуровневого списка

```
<OL>
 <LI>Холодные закуски
```

```
  <UL>
   <LI>Винегреты
    <UL>
     <LI>Винегрет овощной
     <LI>Винегрет с рыбой
    </UL>
   <LI>Салаты
    <UL>
     <LI>Салат зеленый
     <LI>Салат "Весна"
     <LI>Салат витаминный
    </UL>
  </UL>
 <LI>Супы
  <UL>
   <LI>Щи
    <UL>
     <LI>Щи из свежей капусты
     <LI>Щи из квашеной капусты
     <LI>Щи боярские
    </UL>
   <LI>Борщи
    <UL>
     <LI>Борщ московский
     <LI>Борщ украинский
     <LI>Борщ флотский
    </UL>
  </UL>
</OL>
```

Внешний вид списка в окне браузера представлен на рис. 1.5.

- Создание списка определений: <DL>…</DL>.

  Список определений включает определяемый термин, перед которым ставится метка <DT> и абзац с его определением, которому предшествует метка <DD>.

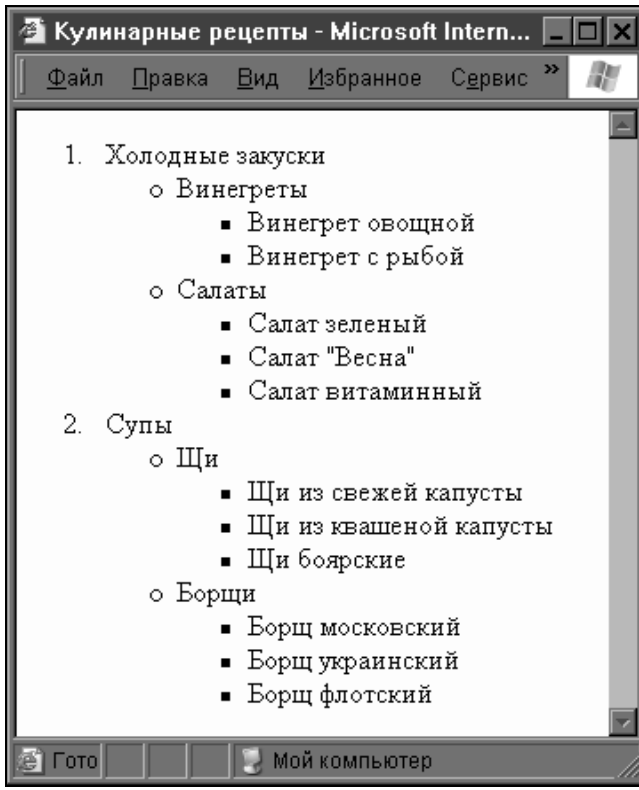


Рис. 1.5. Многоуровневый список

Пример записи:

```
<DL>
<DT>Квадрат
<DD>Прямоугольник, у которого все стороны равны
<DT>Ромб
<DD>Параллелограмм, у которого все стороны равны
</DL>
```

В окне браузера увидим результат, приведенный на рис. 1.6.

- Создание горизонтальной разделительной линии: `<HR>`.

  Создается рельефная разделительная линия. До и после линии вставляется пустая строка.

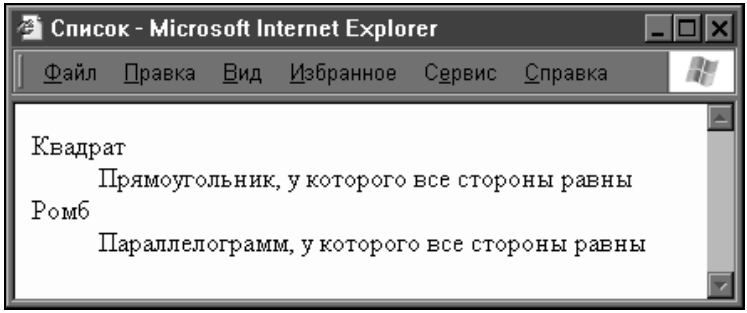


Рис. 1.6. Список определений

Пример записи: `<HR ALIGN=LEFT WIDTH=60% SIZE=7>`.

В метке могут использоваться параметры:

- `ALIGN` — выравнивает линию по левому краю, по правому краю или по центру в зависимости от значения (LEFT, RIGHT, CENTER – по умолчанию);
- `WIDTH` — позволяет задать длину линии в пикселях или в процентах от ширины окна браузера (в этом случае после значения записывается знак %);
- `SIZE` — позволяет задать толщину линии в пикселях;
- `NOSHADE` — отменяет рельефность линии;
- `COLOR` — позволяет задать цвет линии (при задании цвета линия теряет рельефность).

1.5. Создание гипертекстовых ссылок

С помощью *гипертекстовых ссылок* реализуется важнейший принцип функционирования сети Интернет — доступ к информации посредством специально выделенных фрагментов документа, обращение к которым ведет к загрузке новых документов. Такой фрагмент текста называется *гипертекстом*. Обычно гипертекст в окне браузера отличается от остального текста цветом и подчеркиванием. Курсор, наведенный на гипертекст, изменяет свою форму, принимая вид руки с вытянутым указательным

пальцем. Щелчок левой клавишей мыши по гипертексту повлечет загрузку связанного с ним документа. В качестве гипертекста можно использовать не только фрагменты текста, но и рисунки, а ссылаться можно не только на другие HTML-документы, но и на другие ресурсы Интернета. Создание гипертекстовой ссылки осуществляется с помощью метки `<A>…</A>`.

Пример записи:

```
<A HREF="адрес документа или ресурса">Текст, по которому осуществляется ссылка</A>
```

Параметр `HREF` предназначен для задания адреса документа или ресурса, на который осуществляется ссылка. В простейшем случае это может быть документ, входящий в состав того же сайта, что и документ из которого осуществляется ссылка, и располагающийся с ним в одной папке. Тогда значением параметра `HREF` будет только имя файла этого документа:

```
<A HREF="friends.htm">Мои друзья</A>
```

Адресуемые документы могут располагаться и в других папках, в этом случае необходимо указать путь к файлу документа по правилам адресации, принятым в операционной системе. Следует обратить внимание, что в качестве разделителя нужно использовать правый (/), а не левый слеш, как принято в Windows. Например:

```
<A HREF="mydir/mydoc1.htm">Мой первый документ</A>
```

Такая адресация означает, что в папке с документом, из которого осуществляется ссылка, располагается папка **mydir**, а в ней адресуемый документ **mydoc1.htm**.

В метке `<A>` можно использовать параметр `TARGET` со значениями:

- `_blank` — документ, на который осуществляется ссылка, откроется в новом окне браузера;
- `_self` — документ откроется в том же окне или фрейме, что и исходный файл (по умолчанию);
- `_parent` — документ откроется в родительском фрейме;
- `_top` — документ займет все окно браузера.

Последние два параметра используются при наличии фреймов (*см. главу 5*). Например:

```
<A HREF="doc2.htm" TARGET=_blank>Второй документ</A>
```

1.5.1. Ссылки на внешние ресурсы Интернета

В данном разделе рассмотрим гипертекстовые ссылки на внешние ресурсы Интернета.

- Ссылка на ресурс WWW (World Wide Web — Всемирная паутина).

  Ресурсами WWW являются многочисленные сайты, размещенные на web-серверах сети Интернет. Обмен информацией осуществляется в соответствии с протоколом HTTP (HyperText Transfer Protocol, протокол передачи гипертекста). Название протокола в параметре HREF является составной частью адреса ресурса. Например, ссылка на сайт поисковой системы АПОРТ имеет следующий вид:

  ```
  <A HREF="http://www.aport.ru">Поисковая система АПОРТ</A>
  ```

- Ссылка на FTP-сервер.

  FTP-серверы предназначены для хранения файлов различного назначения, которые можно загрузить на свой компьютер. Обмен информацией осуществляется в соответствии с протоколом FTP (File Transfer Protocol, протокол передачи файлов). Как и в предыдущем случае, название протокола присутствует в адресе ресурса. Например, ссылка на FTP-сервер фирмы Microsoft имеет следующий вид:

  ```
  <A HREF="ftp://ftp.microsoft.com">FTP-сервер фирмы Microsoft</A>
  ```

- Ссылка на группу новостей.

  Пусть с помощью ссылки необходимо подключиться к группе новостей my_news. Ссылка на группу новостей записывается следующим образом:

  ```
  <A HREF="news:my_news">Доступ к группе новостей my_news</A>
  ```

  В данном случае щелчок по гиперссылке приведет к загрузке почтовой программы, например Outlook Express, и активизации группы новостей my_news. Возможность практической работы с группой новостей зависит от ее доступности для данного пользователя.

- Ссылка на адрес электронной почты.

  Данная ссылка позволяет запустить почтовую программу, например Outlook Express, и открыть окно создания сообщения. В зависимости от значения параметра `HREF` в этом окне могут быть заполнены строки: **Кому**, **Копия**, **Тема** и возможно даже набран текст письма. Например, для открытия окна создания сообщения с установленным адресом отправления me@mymail.ru необходимо создать следующую ссылку:

  ```
  <A HREF="mailto: me@mymail.ru">Пишите письма</A>
  ```

  Для заполнения остальных строк после адреса нужно добавить знак "`?`", а затем после "`subject=`" написать тему послания, после "`&cc=`" — адрес копии, после "`&body=`" — текст письма. Например:

  ```
  <A HREF="mailto:me@mymail.ru?subject=Привет от посетителя сайта &cc=q@qqq.ru &body=Ваш сайт мне понравился">Пишите письма</A>
  ```

  Используя такую возможность, можно, например, организовать отправку заказов на продукцию, рекламируемую на сайте, или отзывов посетителей на качество сайта. Посетителю останется только нажать кнопку **Отправить**.

1.5.2. Адресация внутри одного документа

Для документов большого размера, включающих несколько разделов, представляет интерес создание ссылок на отдельные разделы документа. Разместив в начале документа оглавление со ссылками на разделы, можно щелчком мыши по названию раздела сделать его видимым в окне браузера, не прибегая к использованию полос прокрутки. Прежде чем это станет возможным, каждому из разделов нужно присвоить определенное имя. Для этой цели также используется метка `<A>`, но с параметром `NAME`. Метка должна располагаться в начале раздела, а параметру `NAME` присваивается имя раздела. Например:

```
<A Name="раздел1"></A>Первая строка раздела 1
Остальной текст раздела
<A Name="раздел2"></A>Первая строка раздела 2
Остальной текст раздела
<A Name="раздел3"></A>Первая строка раздела 3
Остальной текст раздела
```

В этом случае текст между метками <A>…</A> может отсутствовать.

Ссылка на раздел документа осуществляется следующим образом:

```
<A HREF="#раздел1">Раздел 1</A>
```

Допускается переход к разделам другого документа, в котором также каждому разделу должно быть присвоено имя. В этом случае при создании ссылки в значение параметра HREF добавляется имя файла документа:

```
<A HREF="doc2.htm#раздел1">Раздел 1 документа 2</A>
```

1.5.3. Ссылки на документы других типов

Ссылка на документ, с которым браузер умеет работать самостоятельно (например рисунок) приводит к загрузке документа в окно браузера. Ссылка на документ, с которым браузер не может самостоятельно работать, предполагает загрузку документа в соответствующей для его формата программе.

Адресация, например, на видеофрагмент формата MPEG приведет к его демонстрации в Проигрывателе Windows Media, а на файл, созданный в программе MS Excel, — к открытию программы MS Excel и загрузке в нее этого файла.

Примеры ссылок:

```
<A HREF="1.jpg">Смотрите рисунок</A>
<A HREF="1.mpeg">Смотрите видео</A>
<A HREF="1.xls">Работаем в программе MS Excel</A>
```

1.5.4. Изменение цвета гипертек ста

Цвет гипертекста можно изменить с помощью следующих параметров метки <BODY>:

- LINK – цвет гиперссылки, которой еще не пользовались;
- VLINK – цвет просмотренной гиперссылки;
- ALINK – цвет активной (просмотренной последней) гиперссылки.

Пример записи:

```
<BODY LINK="green" VLINK="blue" ALINK="red">
```

Более широкие возможности по изменению цвета гиперссылок и исключению подчеркивания открываются при использовании свойств каскадных таблиц стилей (*см. разделы 7.8.1* и *7.7.3*).

1.6. Включение дополнительной информации о документе

Для включения в документ дополнительной информации, которая будет использоваться браузером или поисковыми системами, предназначена метка `<META>`. Метка должна располагаться в разделе `<HEAD>...</HEAD>`. Допускается использование на одной странице нескольких меток `<META>` с различными параметрами.

1.6.1. Обновление содержимого страницы

Для страницы с часто изменяемой информацией можно предусмотреть принудительную перезагрузку страницы браузером через определенный промежуток времени. В этом случае используется метка `<META>` с параметрами `HTTP-EQUIV` и `CONTENT`:

```
<META HTTP-EQUIV="refresh" CONTENT="30">
```

Значение параметра `HTTP-EQUIV "refresh"` указывает браузеру на необходимость перезагрузить страницу, а в параметре `CONTENT` указывается время в секундах, через которое должна осуществляться перезагрузка.

1.6.2. Переадресация документа

В случае изменения адреса web-сайта многие пользователи могут пытаться получить доступ к нему, используя старый адрес. Чтобы не терять посетителей, можно предусмотреть переадресацию сайта. Для этой цели по старому адресу необходимо расположить короткую страницу с соответствующим информационным текстом, включив в нее метку `<META>` с параметрами:

```
<META HTTP-EQUIV="refresh"
CONTENT="10;URL=http://www.mynew.ru">
```

В параметре URL указывается новый адрес сайта, на который произойдет переход через 10 с.

1.6.3. Включение информации для поисковых систем

Краткое описание содержимого сайта включается следующим образом:

```
<META NAME="description" CONTENT="текст описания">
```

Перечень ключевых слов, т. е. слов, которые наиболее точно отражают содержание сайта, можно включить в метку <META>, разделяя слова пробелами или запятыми:

```
<META NAME="keywords" CONTENT="список ключевых слов">.
```

Также можно включить имя и фамилию автора сайта:

```
<META NAME="author" CONTENT="имя фамилия">.
```

1.7. Метки-контейнеры <DIV> и <SPAN>

В тех случаях, когда фрагменту документа нужно задать определенные свойства, его можно поместить в один из контейнеров <DIV>…</DIV> или <SPAN>…</SPAN>. Главным образом это касается изменения стиля форматирования объекта или совершения над ним определенных действий (*см. главы 7, 8*). Принципиальное отличие меток <DIV> и <SPAN> состоит в том, что в контейнер <SPAN> можно поместить одно слово или даже его часть и остальной текст будет воспроизводиться браузером с сохранением целостности строки или слова. При использовании контейнера <DIV> часть текста, оказавшаяся в контейнере, будет воспроизводиться с новой строки, а текст после контейнера также перейдет на новую строку.

Глава 2

Создание простых страниц в редакторе Macromedia Dreamweaver MX 2004

С появлением новых информационных технологий появляются и начинают стремительно развиваться программные продукты, облегчающие труд разработчиков web-страниц. Создание в Блокноте даже простых HTML-документов показало, что это достаточно трудоемкая работа. Использование при создании сайтов специальной программы-редактора позволит значительно ускорить работу и избежать некоторых ошибок, которые неизбежно возникают во время выполнения однообразных механических действий. В свою очередь следует помнить, что какой бы совершенной ни была программа, создаваемый ею документ, часто будет несколько хуже оптимального варианта, который может создать квалифицированный специалист. Поэтому осваивать редактор сайтов следует параллельно с изучением языка HTML. Внося необходимые коррективы в создаваемый в редакторе документ, можно не только ускорить работу, но и не снизить качество web-документа.

В настоящее время существует множество программ для создания web-документов. Мы познакомимся с одной из лучших и популярных программ — *Macromedia Dreamweaver MX 2004* (Dreamweaver).

2.1. Интерфейс программы Macromedia Dreamweaver MX 2004

При первом запуске программы Macromedia Dreamweaver MX 2004 появляется стартовая страница, позволяющая выбрать объект создания или редактирования, с которым будет осуществляться работа (рис. 2.1).

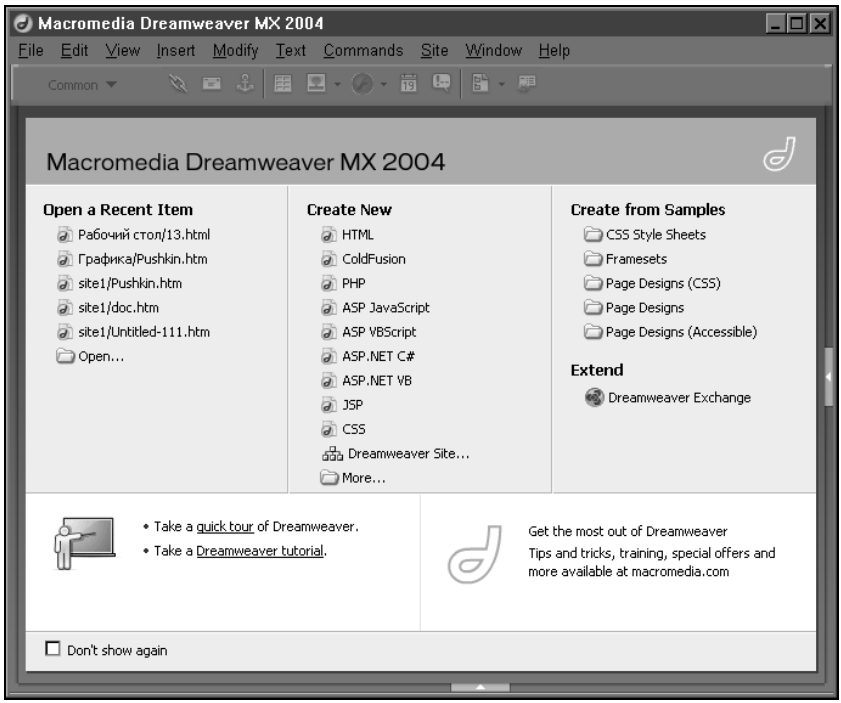


Рис. 2.1. Вид стартовой страницы программы Macromedia Dreamweaver MX 2004

Стартовая страница содержит следующие разделы:

- **Open a Recent Item** (Открыть недавно созданный документ) включает в себя список последних (не более 10) документов, с которыми велась работа. В случае отсутствия документа в списке, он может быть открыт с помощью диалогового окна

Open (Открыть), которое появится после щелчка по соответствующему значку в нижней части раздела.

- **Create New** (Создать новый документ) содержит список документов, которые можно создать. Выбор пункта **HTML** позволит приступить к созданию HTML-документа. Нажав кнопку **More** (Больше) в нижней части раздела, можно открыть диалоговое окно **New Document** (Новый документ) с расширенным списком документов.
- **Create from Samples** (Создание по образцам). Выбор любой кнопки этого раздела приведет к открытию диалогового окна **New Document** (Новый документ), в котором будет выделен соответствующий вид документа. Более подробно с содержимым окна **New Document** (Новый документ) мы будем знакомиться по мере необходимости.
- **Extend** (Расширение возможностей) — выход в Интернет на сайт фирмы Macromedia.

Установив в нижней левой части окна флажок **Don't Show Again** (Больше не показывать), можно избавиться от открытия стартовой страницы при повторных запусках программы. В этом случае открытие окна **New Document** (Новый документ) можно осуществлять командой меню **File | New** (Файл | Создать). Доступ к 10 недавно созданным документам будет возможен по команде меню **File | Open Recent** (Файл | Открыть недавно созданный документ), а открытие любого документа — **File | Open** (Файл | Открыть). В случае если вы решите вернуться к использованию стартовой страницы, то командой меню **Edit | Preferences** (Правка | Настройки) необходимо открыть диалоговое окно **Preferences** (Настройки) и в категории **General** (Общая) установить флажок **Show start page** (Показ стартовой страницы).

В ближайшее время нам предстоит работа только с HTML-документами, поэтому в разделе **Create New** (Создать новый документ) стартовой страницы следует выбрать вид документа — HTML. После выбора данного пункта загрузится заготовка для создания новой HTML-страницы. Прежде чем приступить к работе, познакомимся с элементами интерфейса программы.

Окно программы Macromedia Dreamweaver MX 2004 включает следующие элементы (рис. 2.2):

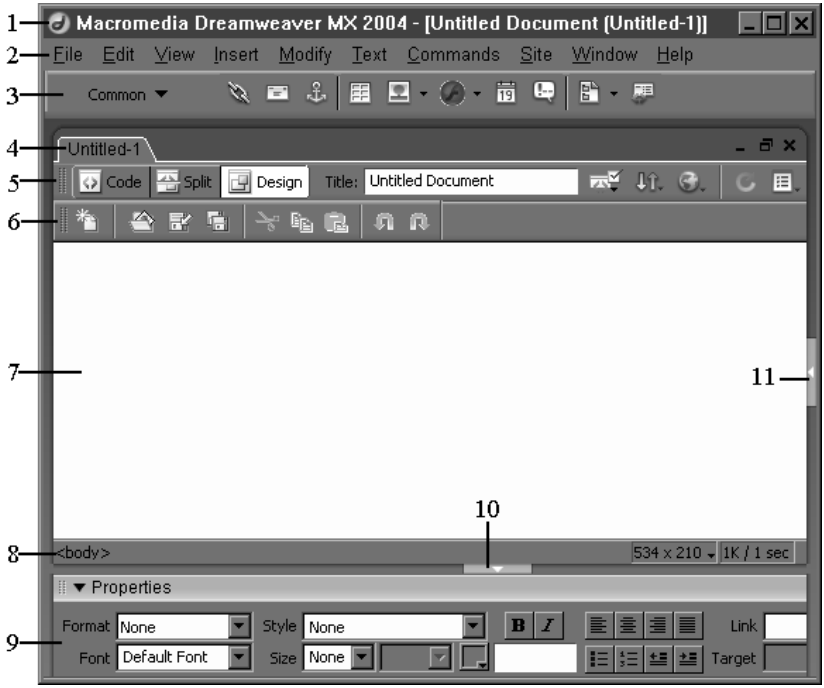


Рис. 2.2. Окно программы Macromedia Dreamweaver MX 2004

1. *Строка заголовка* — содержит название программы, заголовок создаваемого документа, записываемый в метке `<TITLE>`, и имя файла документа. Для вновь создаваемого документа в качестве имени файла будет записано [Untitled Document (Untitled-1)]. После изменения содержимого метки `<TITLE>` и сохранения документа в строке заголовка произойдут соответствующие изменения.
2. *Строка меню* — содержит команды, необходимые для работы программы и настройки внешнего вида окна.
3. *Панель инструментов* ***Insert*** (Вставка) предназначена для быстрого создания различных элементов. Если панель отсутствует, то ее можно отобразить с помощью команды меню **Window | Insert** (Окно | Вставка) или комбинацией клавиш <Ctrl>+<F2>, или меню **View | Toolbars | Insert** (Вид | Инструменты | Вставка). В левой части данной панели располага-

ется кнопка, раскрывающая меню со списком групп инструментов, входящих в ее состав. На кнопке отображается название текущей группы инструментов. Последняя команда раскрывающегося меню **Show as Tabs** (Показать в виде вкладок) позволяет изменить внешний вид панели (рис. 2.3).

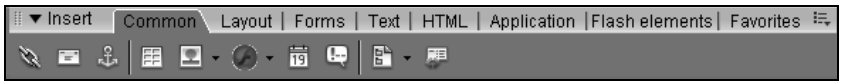


Рис. 2.3. Представление панели инструментов **Insert** в виде вкладок

Щелчок по маркеру в виде треугольной стрелки (слева от слова "Insert") раскрывает или сворачивает содержимое панели, а стрелка разворачивается на 90°. Панель содержит множество вкладок, на каждой из которых располагаются кнопки соответствующих инструментов. Такой вид панель **Insert** (Вставка) имела в предыдущих версиях программы Dreamweaver. Кому-то он может показаться более удобным. Чтобы вернуться к первоначальному виду панели, следует щелкнуть правой кнопкой мыши по слову "Insert" и выбрать команду **Show as Menu** (Показать в виде меню). Непосредственное знакомство с группами инструментов панели **Insert** (вставка) будет продолжено в дальнейшем.

4. *Окно документа* с ярлыком имени файла web-страницы, над которым ведется работа. Если в программе открыто несколько документов, то на экране будут отображаться несколько ярлыков. Щелчок по ярлыку позволяет перейти к нужному файлу. При работе с новым документом на ярлыке в качестве имени файла будет написано Untitled-1 (номер соответствует порядковому номеру документа, создаваемого в текущий сеанс работы). После сохранения документа на ярлыке появится имя файла.

5. *Панель инструментов* ***Document*** (Документ) содержит кнопки управления созданием текущего документа, в частности кнопки выбора режима работы с документом. Установить или убрать панель можно с помощью команды меню **View | Toolbars | Document** (Вид | Инструменты | Документ). Кнопка **Code** (Код) на панели инструментов **Document** (Документ) по-

зволяет выбрать режим работы с HTML-кодом, предусматривающий ручную расстановку меток в тексте документа. Работа в этом режиме практически не отличается от работы в Блокноте. Новый документ содержит первоначальный набор меток, определяющих структуру документа. Все метки нам уже известны (*см. главу 1*), за исключением двух. Обе они не являются обязательными и поэтому не были включены в структуру HTML-документа. Однако программа Dreamweaver всегда включает их в новый документ, поэтому, если мы не собираемся постоянно удалять эти метки из текста, необходимо уточнить их назначение. Метка

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

помогает браузеру определить версию языка HTML, которая использовалась при написании документа. W3C — сокращенное название организации World Wide Web Consortium, разработавшей стандарт HTML. DTD (Document Type Definition) означает определение типа документа. В нашем случае документ создается с использованием языка HTML версии 4.01, **http://www.w3.org/TR/html4/loose.dtd** – адрес web-сайта Консорциума (Consortium) W3C.

Метка

```
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
```

предназначена для указания кодировки текста на русском языке. Если после `charset=` указана кодировка не `"windows-1251"`, то, чтобы избежать проблем при работе с текстами на русском языке, необходимо изменить настройку программы Dreamweaver в категории **New Document** (Новый документ). Для этого можно воспользоваться командой меню **Edit | Preferences** (Правка | Настройки). В открывшемся диалоговом окне **Preferences** (Настройки) выбрать категорию **New Document** (Новый документ). В раскрывающемся списке **Default encoding** (Кодировка по умолчанию) необходимо выбрать **Кириллица (Windows)** и нажать кнопку **OK**. Изменения вступят в силу после повторного открытия документа.

С помощью кнопки **Design** (Конструирование) выбирается визуальный режим работы над документом. В этом режиме

работа по созданию web-страницы практически не отличается от создания текстового документа в обычном текстовом редакторе (например MS Word). Пользователю не нужно заботиться о расстановке меток, причем он их может просто не знать. Программа Dreamweaver будет сама расставлять в нужных местах необходимые метки. Периодически переключаясь в режим **Code** (код), можно контролировать этот процесс.

В ряде случаев, в частности на начальных этапах обучения, целесообразно иметь перед глазами сразу два окна, в одном из которых можно будет наблюдать текст документа с расставленными метками, а в другом осуществлять работу в визуальном режиме. Такой режим включается кнопкой **Split** (Многооконный). В этом случае рабочее поле делится на две части. В нижней части можно набирать документ в визуальном режиме, а в верхней — наблюдать процесс автоматической расстановки меток.

6. *Панель инструментов* ***Standard*** (Стандартная) содержит кнопки для выполнения стандартных операций в среде Windows. Кнопки данной панели дублируют соответствующие команды меню **File** (Файл): **New** — создать новый документ, **Open** — открыть документ, **Save** — сохранить, **Save All** — сохранить как, а также команды меню **Edit** (Правка): **Cut** — вырезать, **Copy** — копировать, **Paste** — вставить, **Undo** — отмена последнего выполненного действия, **Redo** — возврат отмененного действия. Установить или убрать панель можно с использованием команды меню **View | Toolbars | Standard** (Вид | Инструменты | Стандартная).

   Панели **Document** (Документ) и **Standard** (Стандартная) можно перемещать. Для этого указатель мыши нужно установить в любое место между кнопками; нажать левую клавишу мыши и, удерживая ее, перетащить панель. Иногда эти панели располагают на одной горизонтальной линии, что позволяет увеличить размер рабочей области.

7. *Рабочая область окна документа* предназначена непосредственно для работы с документом. Внешний вид рабочей области зависит от выбранного режима работы. Как уже отмечалось, выбор режима осуществляется кнопками на панели **Document** (Документ).

8. *Строка состояния* ***Status bar***. В левой части строки состояния находится селектор меток. В зависимости от положения курсора в тексте документа селектор отображает все метки, между которыми располагается курсор. Щелчок по имени метки в селекторе позволяет выделить весь фрагмент документа, который находится внутри выбранной метки. В правой части строки состояния расположены два информационных поля. В первом показан размер рабочей области окна документа в визуальном режиме в пикселях. Во втором поле находится дробное выражение, в числителе которого указан объем файла страницы с учетом подключенных к ней файлов (например рисунков), а в знаменателе — время загрузки страницы по каналу связи со скоростью передачи данных 28,8 Кбит/сек.
9. *Панель свойств* ***Properties***. Вид панели изменяется в зависимости от того, в каком месте документа находится курсор или какой объект выделен. Данная панель позволяет просматривать и изменять свойства объектов. Если панель отсутствует на экране, то ее можно отобразить с помощью меню **Window | Properties** (Окно | Свойства). Панель свойств можно перемещать. Для этого необходимо указатель мыши навести на корешок панели (две вертикальные линии), при этом он должен принять вид четырехнаправленной стрелки; нажать левую кнопку мыши и, удерживая ее, переместить панель в нужное место экрана. Содержимое панели может бать свернуто щелчком по треугольной стрелке слева от слова **Properties.**
10. Кнопка с треугольной стрелкой, щелчок по которой позволяет быстро установить/свернуть панель **Properties** (Свойства).
11. Кнопка с треугольной стрелкой, щелчок по которой позволяет установить/свернуть инструментальные панели, содержащие большой набор инструментов, знакомство с которыми будет осуществляться по мере необходимости. На рис. 2.2 данные кнопки находятся в режиме свернутого состояния.

2.2. Определение нового сайта

Даже самый простой *сайт* содержит не одну, а несколько HTML-страниц, связанных между собой гипертекстовыми ссыл-

ками. Каждая страница может содержать иллюстрации, следовательно, с ней будет связано несколько графических файлов. Таким образом, число файлов, входящих в состав сайта даже в простых случаях, может измеряться десятками. Создавая сайт в Блокноте, мы стремились сохранять создаваемые файлы в одной папке, что облегчало их поиск и адресацию. Программа Dreamweaver будет оказывать существенную помощь в объединении разрозненных файлов в один сайт, но для этого новый сайт должен быть уже определен. Рекомендуется сделать это до начала работы над первой страницей.

Если вы начинаете работу с программой со стартовой страницы, то для определения сайта в разделе **Create New** (Создать новый документ) необходимо выбрать **Dreamweaver Site** (Сайт), после чего откроется диалоговое окно **Site Definition for...** (Определение сайта...). Если работа над HTML-документом уже ведется, то путь к открытию окна **Site Definition for...** (Определение сайта...) становится более длинным. Для этого необходимо использовать команду меню **Site | Manage Sites** (Сайт | Управление сайтами), после чего откроется диалоговое окно **Manage Sites** (Управление сайтами) (рис. 2.4).

Рис. 2.4. Вид диалогового окна **Manage Sites**

В диалоговом окне **Manage Sites** (Управление сайтами) нужно нажать кнопку **New** (Создать) и в раскрывшемся списке выбрать

Site (Сайт), что приведет к открытию диалогового окна **Site Definition for** (Определение сайта...). Дальнейшая работа осуществляется в этом диалоговом окне на вкладке **Basic** (Основная), которая подразумевает пошаговое выполнение следующих действий.

На первом шаге необходимо задать имя сайта. В текстовое поле **What would you like to name your site?** (Какое имя вы хотите дать вашему сайту?) нужно ввести имя сайта, которое будет использоваться только в программе Dreamweaver, например MySite. Нажав кнопку **Next**, перейдем ко второму шагу.

На втором шаге предлагается сделать выбор между использованием серверной технологии и отказом от нее. Установив переключатель в положение **Yes, I want to use a server technology** (Да, я хочу использовать серверную технологию), вы соглашаетесь на использование серверной технологии. Отказ от ее использования подразумевает положение переключателя **No, I do not want to use a server technology** (Нет, я не хочу использовать серверную технологию). На первых порах создания сайтов следует ограничиваться выбором второго варианта.

Третий шаг предусматривает выбор места расположения файлов создаваемого сайта. Выбирать можно из следующих вариантов:

- **Edit local copies on my machines, then upload to server when ready (recommended)** — редактировать файлы на моем компьютере, после чего загрузить их на сервер (рекомендуется);
- **Edit directly on server using local network** — редактировать файлы на сервере, используя локальную сеть.

Очевидно, что первый вариант является наиболее простым и универсальным, именно его и следует выбрать. В текстовой строке **Where on your computer do you want to store your files?** (Где на вашем компьютере вы хотите хранить ваши файлы?) указать папку, в которой будут располагаться все файлы сайта. Щелкнув по значку с изображением папки (справа от текстовой строки), откроем диалоговое окно **Choose local root folder for site...** (Выбор локальной корневой папки для сайта...). В этом окне, используя стандартные методы Windows, можно создать папку, если она не была создана ранее, и выбрать место ее расположения (например, папка называется MyFiles и располагается в папке

"Мои документы"). После этого, нажав кнопку **Next**, можно перейти на четвертый шаг.

На четвертом шаге в списке **How do you connect to your remote server?** (Как вы соединяетесь с вашим удаленным сервером?) выбрать **None**, так как пока мы не будем загружать файлы создаваемого сайта на удаленный сервер.

На пятом шаге выводится справочная информация, в частности напоминание о месте нахождения папки сайта. После нажатия кнопки **Done** (Закрыть) процесс определения сайта завершается.

Что касается вкладки **Advanced** (Продвинутый) диалогового окна **Site Definition for...** (Определение сайта...), то она содержит больше возможностей по определению параметров сайта, но требует больших знаний и опыта. В простейшем случае ее можно использовать для изменения уже установленных параметров.

Вернемся к диалоговому окну **Manage Sites** (Управление сайтами) (рис. 2.4). В окне перечислены все созданные сайты. Изменить установленные параметры любого из них можно, выделив название сайта и нажав кнопку **Edit** (Правка). После этого откроется диалоговое окно **Site Definition for...** (Определение сайта...). Пройдя, все перечисленные выше пять шагов, можно осуществить необходимые изменения. Нажатие кнопки **Duplicate** (Копия), приведет к созданию копии выделенного сайта, а кнопки **Remove** (Удалить) — к его удалению.

Примечание

Следует иметь в виду, что при копировании сайта, копирование входящих в него файлов не производится. Несмотря на появление двух сайтов с разными именами, в них могут быть использованы одни и те же файлы. Удаление сайта ведет лишь к удалению его имени из списка, но использовавшиеся в нем файлы сохраняются.

Сохранить сведения о сайте можно в специальном файле с расширением ste. Для этого нужно выделить имя сайта и нажать на кнопку **Export** (Экспорт). В открывшемся диалоговом окне **Export Site** (Экспорт сайта) указывается имя файла и выбирается папка, в которой его нужно сохранить. Наличие файла, хранящего сведения о сайте, позволяет быстро восстановить запись о нем, не повторяя процедуру определения. Для этой цели ис-

пользуется кнопка **Import** (Импорт), открывающая диалоговое окно **Import Site** (Импорт сайта). В диалоговом окне требуется отыскать нужный файл, выделить его имя и нажать **OK**. Работу в диалоговом окне **Manage Sites** (Управление сайтами) следует завершать нажатием кнопки **Done** (Закрыть).

2.3. Создание отдельных страниц

Как уже отмечалось, созданию нового HTML-документа предшествует выбор в разделе **Create New** (Создать новый документ) стартовой страницы вида документа **HTML**, в результате чего появится заготовка для создания новой HTML-страницы. При отсутствии стартовой страницы можно использовать команду меню **File | New** (Файл | Создать), которая дублируется кнопкой **New** (Создать) на панели инструментов **Standard** (Стандартная). Для этих же целей можно использовать сочетание клавиш <Ctrl>+<N>. В результате выполнения команды открывается диалоговое окно **New Document** (Новый документ), приведенное на рис. 2.5.

В диалоговом окне на вкладке **General** (Общая) содержится два списка: перечень категорий создаваемых документов (**Category**) и список шаблонов документов, входящих в данную категорию. После выбора категории и шаблона необходимо нажать кнопку **Create** (Создать). Самой распространенной категорией является категория **Basic Page** (Базовая страница) и шаблон HTML, предназначенный для создания обычного HTML-документа. Очень часто приходится выбирать именно эту категорию и этот шаблон. В этом случае открытие диалогового окна **New Document** (Новый документ) становится ненужной промежуточной операцией. Если программа настроена соответствующим образом, то избежать открытия этого окна можно только при использовании сочетания клавиш <Ctrl>+<N>. Настройка осуществляется в уже знакомом диалоговом окне **Preferences** (Настройки), которое открывается по команде меню **Edit | Preferences** (Правка | Настройки). В данном окне выбирается категория **New Document** (Новый документ) и снимается флажок **Show New Document dialog box on Control+N** (Показывать Новый документ при нажатии клавиш Control+N). Кроме того, необходимо убедиться, что в списке **Default document type** (Тип документа по умолчанию) вы-

брано **HTML**. После этого можно приступать к работе по созданию HTML-документов. Что касается других категорий и шаблонов окна **New Document** (Новый документ), то с некоторыми из них мы познакомимся в дальнейшем, при изучении соответствующих разделов.

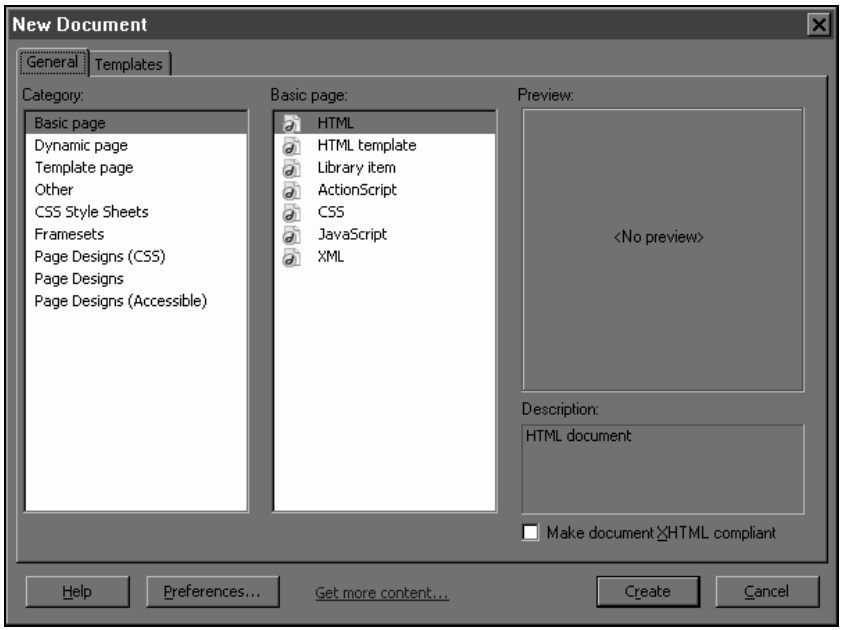


Рис. 2.5. Диалоговое окно **New Document**

Примечание

Перейти к созданию нового документа без открытия окна **New Document** (Новый документ) можно также с помощью сочетания клавиш <Ctrl>+<Shift>+<N>, но в этом случае документ, с которым ведется работа, будет закрыт, а на его месте появится чистый шаблон.

Приступая к работе над новым документом, следует определиться, какому сайту он будет принадлежать. Необходимо открыть диалоговое окно **Manage Sites** (Управление сайтами), используя команду меню **Site | Manage Sites** (Сайт | Управление сайтами), в списке

определенных ранее сайтов выделить нужный и нажать кнопку **Done** (Закрыть). Если сайт ранее не был определен, то это можно сделать, не выходя из окна, нажав кнопку **New** (Создать). Дальнейшие действия описаны в разделе *"Определение нового сайта"*.

При первом сохранении документа используется команда меню **File | Save** (Файл | Сохранить) или кнопка **Save** (Сохранить) на панели инструментов **Standard** (Стандартная). В появившемся диалоговом окне **Save As** (Сохранить как) будет предложено сохранить файл в папке активного сайта. Если активизация сайта через диалоговое окно **Manage Sites** (Управление сайтами) не производилась, то для сохранения документа будет предложена папка сайта, который был активизирован последним.

Активизация того или иного сайта ведет к открытию панели инструментов **Files** (Файлы). Впрочем, данная панель может быть открыта так же, как и любая другая. Для этого достаточно в меню **Window** (Окно) щелкнуть по имени панели (рис. 2.6).

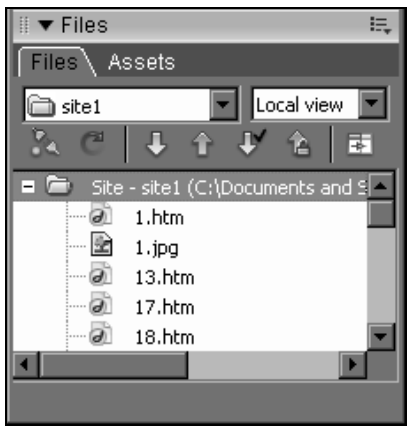


Рис. 2.6. Инструментальная панель **Files**

Панель **Files** (Файлы) содержит список всех файлов, входящих в состав сайта. Двукратный щелчок по имени файла, открывает его в рабочей области окна Dreamweaver. Раскрывающийся список сайтов располагается в левом верхнем углу панели. Выбор сайта из списка позволяет не только просмотреть его содержимое, но и делает его активным.

Как уже отмечалось, программа Dreamweaver позволяет одновременно работать с несколькими документами, принадлежащими разным сайтам. После сохранения документа имя папки сайта воспроизводится в заголовке программы Dreamweaver вместе с именем файла. Переход от одного документа к другому осуществляется щелчком по ярлыку с именем файла документа.

Чтобы завершить работу над документом, можно воспользоваться одним из стандартных способов закрытия окна для операционной системы Windows, например командой меню **File | Close** (Файл | Закрыть).

2.4. Работа с текстом

2.4.1. Форматирование шрифта и текста

Для форматирования будем использовать группу инструментов **Text** (Текст) панели **Insert** (Вставка) (рис. 2.7) и инструменты панели **Properties** (Свойства) (рис. 2.8).

Рис. 2.7. Группа инструментов **Text** панели **Insert**

Рис. 2.8. Панель **Properties**

Инструменты этих панелей частично дублируют, а частично дополняют друг друга. Установим смешанный режим работы над документом (кнопка **Split** (Многооконный) на панели **Document** (Документ)). В нижнюю часть рабочего поля, предназначенную для работы в визуальном режиме, введем заголовок:

```
Мой первый документ, созданный в программе Dreamweaver
```

Нажмем клавишу **Enter** и продолжим набирать текст:

```
Здесь находится текст моего первого HTML-документа, который
отформатирован с использованием инструментов группы Text па-
нели Insert и инструментов панели Properties. Текст содержит
заголовок и абзац. Внутри абзаца имеется аббревиатура и не-
сколько слов, набранных жирным шрифтом.
```

Теперь можно приступить к форматированию, продолжая работать в нижней части рабочей области окна, т. е. в визуальном режиме. Поместим курсор в любое место заголовка и воспользуемся одним из двух способов: либо нажмем кнопку **h2** (Heading 2) в группе **Text** (Текст) панели **Insert** (Вставка), либо на панели **Properties** (Свойства) раскроем список **Format** и выберем **Heading 2**.

В отличие от заголовков, форматируемый элемент текста должен быть предварительно выделен. Выделим в тексте аббревиатуру `HTML` и нажмем кнопку **W3C** (Acronym) в группе инструментов **Text** (Текст). В открывшемся диалоговом окне **Acronym** в строке **Full Text** (Полный текст) наберем полную форму записи аббревиатуры "`Hyper Text Markup Language`". Поле **Language** (Язык) можно не заполнять. После нажатия кнопки **OK** увидим появление в верхней части рабочей области окна метки `<ACRONYM>` с параметром `TITLE`.

Выделим слово "`Текст`", которое должно воспроизводится полужирным шрифтом, и нажмем кнопку **S** (Strong) в группе **Text** (Текст).

Продолжим форматирование шрифта. Последовательно выделяя в тексте слова "`Insert`" и "`Properties`" и используя кнопки **B** (Bold) или **S** (Strong), сделаем шрифт полужирным.

Перейдем к форматированию текста. Для центрирования заголовка курсор поместим в любое место заголовка и нажмем кнопку  — **Align Center** (Выравнивание по центру) на панели **Properties** (Свойства). Для абзаца зададим отступ от края страницы слева, воспользовавшись кнопкой  — **Text Indent** (Отступ текста) на панели **Properties** (Свойства).

Форматирование документа завершено. Остается в строке **Title** (Название) панели **Document** (Документ) написать название документа (например "`Первый документ`") и убедиться, что оно появилось в метке `<TITLE>`.

Перед тем как сохранить документ сделаем активным сайт MySite, который был создан в *разделе 2.2*, и, используя команду меню **File | Save** (Файл | Сохранить), сохраним документ под именем First в папке MyFiles.

Рабочая область окна программы Dreamweaver после завершения форматирования и сохранения документа показана на рис. 2.9.

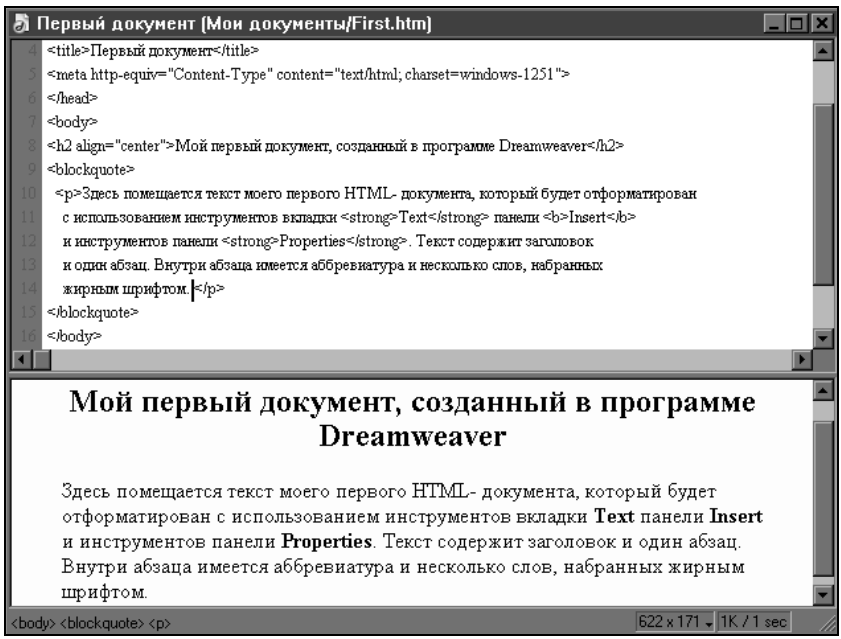


Рис. 2.9. Рабочая область окна программы Dreamweaver после форматирования и сохранения документа

В процессе создания документа может возникнуть желание посмотреть, как он будет выглядеть непосредственно в окне браузера. Для этого на панели инструментов **Document** (Документ) следует нажать кнопку — **Preview/Debug in Browser** (Предварительный просмотр/отладка в браузере) и в раскрывшемся списке выбрать **Preview in iexplore** (Предварительный просмотр в браузере Internet Explorer) либо нажать клавишу <F12>.

Теперь, когда основные принципы форматирования рассмотрены на простом примере, можно более детально познакомиться с

инструментами группы **Text** (Текст) и содержимым панели инструментов **Properties** (Свойства).

Форматирование заголовков

Группа инструментов **Text** (Текст) панели инструментов **Insert** (Вставка) для форматирования заголовков содержит кнопки **h1**, **h2** и **h3**, позволяющие задать заголовки соответствующего уровня от 1 до 3. Список **Format** (Формат) панели инструментов **Properties** (Свойства) позволяет выбрать любой из 6 возможных уровней форматирования заголовков от **Heading 1** до **Heading 6**, а значение **None** позволяет отказаться от установленного ранее форматирования. Курсор должен находиться в любом месте форматируемого заголовка.

Изменение размера шрифта

Выбирается из списка **Size** (размер) на панели **Properties** (свойства). Прежде чем приступить к изменению размера шрифта, следует обратить внимание на настройку программы. Воспользуемся командой меню **Edit | Preferences** (Правка | Настройки). В категории **General** (Общая) открывшегося диалогового окна **Preferences** (Настройки) должен быть снят флажок **Use CSS instead of HTML tags** (Использовать CSS вместо HTML-меток). В противном случае программа Dreamweaver будет задавать размеры шрифта, используя каскадную таблицу стилей (CSS), а не относительные единицы, применяемые в HTML (об использовании CSS *см. главу 7*). После изучения CSS флажок можно вернуть на место. В списке **Size** (Размер) на панели **Properties** (Свойства) можно выбрать числа от 1 до 7 со знаками "+", "−" или без знака. Выбор значения со знаком означает, что размер увеличивается или уменьшается на соответствующую величину относительно заданного размера по умолчанию или в метке `<BASEFONT>`. Значение без знака означает абсолютное задание размера. Форматируемый текст должен быть предварительно выделен.

Изменение гарнитуры шрифта

Выбор гарнитуры шрифта осуществляется из списка **Font** (Шрифт) на панели **Properties** (Свойства). Раскрыв список, можно выбрать нужное семейство гарнитур или **Default Font** (Шрифт по

умолчанию). Выбор **Default Font** (Шрифт по умолчанию) позволяет удалить выбранную ранее гарнитуру. В конце списка гарнитур находится строка **Edit Font List** (Редактирование списка шрифта), выбор которой открывает одноименное диалоговое окно для редактирования списка. С помощью этого окна можно не только дополнить список гарнитур из числа имеющихся на компьютере, но и удалить из списка все семейство или отдельную гарнитуру.

Изменение начертания шрифта

Для изменения начертания шрифта предназначены следующие кнопки группы инструментов **Text** (текст): **B** (Bold) — полужирный, ***I*** (Italic) — курсивный, **S** (Strong) — важный (отображается полужирным шрифтом), **em** (Emphasis) — важный (отображается курсивом). Кнопки **B** (Bold), **I** (Italic) имеются также на панели **Properties** (Свойства) и полностью дублируют аналогичные кнопки группы **Text** (Текст). Можно настроить программу таким образом, чтобы при нажатии кнопок **B** (Bold) появлялась метка <STRONG>, а при нажатии кнопки **I** (Italic) — метка <EM>. Для настройки нужно использовать команду меню **Edit | Preferences** (Правка | Настройки). В категории **General** (Общая) открывшегося диалогового окна **Preferences** (Настройки) требуется установить флажок **Use and in place of <b> and <i>** (использовать и вместо <b> и <i>).

Изменение цвета шрифта

Для изменения цвета шрифта необходимо выделить нужный элемент текста, нажать кнопку **Text Color** (Цвет текста) на панели **Properties** (Свойства) и в открывшейся палитре выбрать нужный цвет. Впрочем, если цвет уже известен, то его можно просто вписать в поле справа от кнопки. В верхней части палитры имеется кнопка **Default Color** (Цвет по умолчанию), позволяющая очистить поле, и кнопка **System Color Picker** (системная цветовая палитра), открывающая окно с большими возможностями по выбору цвета.

Создание нового абзаца

В визуальном режиме (активная кнопка **Design**) новый абзац создается нажатием клавиши <Enter>. Чтобы перейти на новую строку, не создавая нового абзаца, нужно использовать сочета-

ние клавиш <Shift>+<Enter>, тогда в текст документа вместо метки <P> будет вставлена метка <BR>.

Другой способ перехода на новую строку в пределах абзаца — нажатие крайней правой кнопки из группы **Text** (Текст) —  . В случае, если кнопка имеет другой вид, то щелчком по стрелке нужно раскрыть меню и выбрать **Line Break** (Разрыв строки). На кнопке сохраняется условное обозначение последней использованной команды. Полное содержание меню будет рассмотрено в *разделе 2.4.2.*

В режиме ручной расстановки меток (активная кнопка **Code**) нажатие кнопки  — **Paragraph** (Абзац) в группе **Text** (Текст) приведет к появлению в тексте документа метки <P>.

Вставка предварительно отформатированного текста

Текст необходимо набрать в Блокноте и осуществить форматирование, задавая пробелы (несколько подряд), символы табуляции, переходы на новые строки и т. д. Чтобы поместить созданный текст в HTML-документ, сохранив форматирование без изменений, нужно проделать следующее:

- выделить в Блокноте весь текст и скопировать его в буфер обмена (команда **Копировать** меню **Правка**);
- в программе Dreamweaver перейти в режим ручной расстановки меток, установить курсор в то место документа, куда нужно вставить текст и нажать кнопку  — **Preformatted Text** (предварительно отформатированный текст) из группы **Text** (Текст);
- вставить текст из буфера обмена после метки <PRE>, дав команду меню **Edit | Paste** (Правка | Вставить).

Перейдя в режим визуального форматирования, можно предварительно оценить полученный результат, но лучше всего это сделать, открыв документ в окне браузера.

Создание абзацных отступов

Каждый абзацный отступ смещает левую и правую границы абзаца на несколько позиций, а в текст документа вставляется метка <BLOCKQUOTE>...</BLOCKQUOTE>.

Для создания отступа абзаца курсор необходимо поместить в любое место абзаца и нажать кнопку  — **Block Quote** (Блок выделения) из группы инструментов **Text** (Текст) или кнопку  — **Text Indent** (Увеличить отступ текста) на панели **Properties** (Свойства). Уменьшить абзацный отступ позволяет кнопка  — **Text Outdent** (Уменьшить отступ текста) на панели **Properties** (Свойства).

Выравнивание текста

На панели **Properties** (Свойства) расположены четыре кнопки —  , предназначенные для выравнивания текста. Выравнивание распространяется на абзац, в котором находился курсор в момент нажатия кнопки. Способ выравнивания очевиден не только из его изображения на кнопке, но и из названия: **Align Left** (выравнивание по левому краю), **Align Center** (выравнивание по центру), **Align Right** (выравнивание по правому краю), **Justify** (выравнивание по ширине).

Создание маркированных и нумерованных списков

При создании списков следует помнить, что список — это последовательность абзацев. В группе инструментов **Text** (Текст) панели **Insert** (Вставка) для создания списка имеются следующие кнопки:

- **ul** (Unordered List) — маркированный список;
- **ol** (Ordered List) — нумерованный список;
- **li** (List Item) — элемент списка.

Списки можно создавать двумя способами:

1. Создать последовательность абзацев, нажимая на клавишу <Enter> в конце каждого абзаца. Выделить все абзацы и нажать кнопку **ul** или **ol**.
2. Нажав на кнопку **ul** или **ol**, набрать первый элемент списка. Каждый последующий элемент списка создается автоматически при переходе на следующий абзац с помощью клавиши <Enter>. После набора последнего элемента необходимо перейти на следующий абзац и произвести повторный щелчок по соответствующей кнопке **ul** или **ol**. Создание списка прекратится.

Кнопки **ul** (Unordered List) и **ol** (Ordered List) полностью дублируются одноименными кнопками  на панели **Properties** (Свойства). Кнопка **li** предназначена для создания элементов списка в режиме ручной расстановки меток в тексте документа.

Создание списков определений

Для создания списков определений в группе инструментов **Text** (текст) панели **Insert** (Вставка) имеется три кнопки:

- **dl** (Definition List) — список определений;
- **dt** (Definition Term) — определяемый термин;
- **dd** (Definition Description) — описание определения.

Списки определений создаются двумя способами:

1. Последовательность выделенных абзацев преобразуется в список определений после нажатия кнопки **dl**. Особенностью преобразования является то, что нечетные абзацы становятся определяемыми терминами, а четные — описаниями определений.
2. Активизировать кнопку **dl.** После набора определяемого термина следует нажать клавишу <Enter>, набрать его описание и снова нажать <Enter>. По окончании ввода описания последнего термина необходимо нажать клавишу <Enter> и произвести повторный щелчок по кнопке **dl.** Создание списка прекратится.

Кнопки **dt** и **dd** предназначены для создания соответственно определяемых терминов и описаний определений в режиме ручной расстановки меток в тексте документа.

2.4.2. Вставка специальных символов

Меню, содержащее список специальных символов вместе с командой **Line Break** (Разрыв строки), открывается щелчком по кнопке с треугольником справа от кнопки  из группы инструментов **Text** (Текст) (рис. 2.10). Как уже отмечалось, вид кнопки зависит от того, какая команда списка использовалась последней.

Рис. 2.10. Меню, содержащее список специальных символов

Рассмотрим назначение команд списка:

- **Line Break** — переход на новую строку в пределах абзаца. Выбор данной команды позволяет вставить в текст документа метку `<BR>`. Переход на новую строку с помощью клавиш <Shift>+<Enter> также приведет к вставке метки `<BR>`;
- **Non-Breaking Space** — вставляет в текст символ неразрывного пробела;
- **Left Quote** — левые кавычки;
- **Right Quote** — правые кавычки;
- **Em Dash** — длинное тире;
- **Pound** — фунт стерлингов;
- **Euro** — евро;
- **Yen** — иена;
- **Copyright** — знак авторского права;
- **Registered Trademark** — знак регистрации;
- **Trademark** — торговая марка;

- **Other Characters** — другие символы. Данная команда открывает диалоговое окно **Insert Other Character** (Вставка другого символа), которое содержит большой набор дополнительных символов.

Вставка каждого символа может сопровождаться появлением диалогового окна с текстом, предупреждающим, что не все браузеры могут воспроизводить специальные символы. Установив в этом окне флажок **Don't show me again** (Далее не показывать), можно избавиться от его появления в дальнейшем.

2.5. Создание гипертекстовых ссылок

2.5.1. Создание гиперссылки по фрагменту уже имеющегося текста

Фрагмент текста, по которому будет осуществляться ссылка, предварительно выделяется. Раскрывающийся список **Link** (Связь) на панели **Properties** (Свойства) содержит список файлов, на которые ранее уже осуществлялись ссылки. Если нужный файл находится в этом списке, то достаточно щелкнуть по нему мышкой, и гиперссылка будет создана.

Так как в нашем документе создается первая ссылка, то список **Link** (Связь) будет пустым. В этом случае необходимо щелкнуть по значку **Browse for File** (Поиск файла) с изображением папки, правее списка. Раскроется диалоговое окно **Select File** (Выбор файла) и в нем будет показано содержимое папки сайта, которому принадлежит создаваемый документ. Необходимо выделить файл, на который осуществляется ссылка, и щелкнуть по кнопке **OK**. Ссылка готова.

Представляет интерес случай, когда файл, на который осуществляется ссылка, находится в другой папке. Ссылка на него создается аналогично, за исключением перехода в диалоговом окне **Select File** (Выбор файла) в нужную папку. В данном случае после нажатия **OK** откроется диалоговое окно, в котором будет предложено сохранить копию файла в папке сайта (рис. 2.11).

Рис. 2.11. Диалоговое окно, предлагающее поместить копию файла в папку сайта

Текст диалогового окна предупреждает: "Этот файл находится за пределами корневой папки сайта "MySite" и может стать недоступным после публикации сайта. Ваша корневая папка: C:\Documents and Settings\Юрий\Мои документы\MySite\. Вы хотите скопировать файл сейчас?". Увидев такое диалоговое окно и убедившись, что именно в предлагаемом вам сайте вы хотели бы видеть адресуемый документ, нужно нажать кнопку **Да**. При нажатии **Нет**, в текст документа будет вставлен путь к файлу за пределами папки сайта. В дальнейшем, при перемещении папки на другой компьютер или публикации сайта на сервере, адресуемый файл, скорее всего, будет недоступным.

После того как гиперссылка создана становится активным список **Target** (Цель) на панели **Properties** (Свойства). При желании открыть адресуемый документ в новом окне необходимо выбрать **_blank**. В остальных случаях документ будет открываться по умолчанию в текущем окне. Выбор других вариантов из списка **Target** (Цель) имеет смысл только при работе с фреймами (*см. главу 5*).

2.5.2. Создание ссылки вместе с гипертекстом

В случае, когда текст, по которому будет осуществляться ссылка, пока отсутствует и должен быть набран в процессе создания

ссылки, необходимо открыть группу инструментов **Common** (Простые) панели **Insert** (Вставка). Щелкнув по первой кнопке этой группы — **Hyperlink** (Гиперссылка), откроем одноименное диалоговое окно (рис. 2.12).

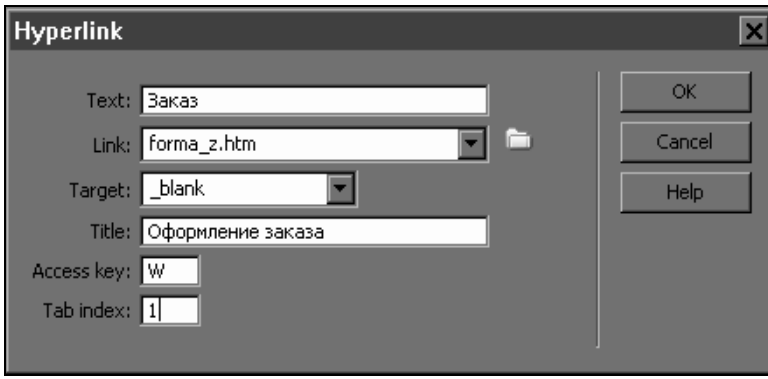


Рис. 2.12. Диалоговое окно создания гиперссылки

В строке **Text** (Текст) нужно написать текст, по которому будет осуществляться ссылка. Назначение списка **Link** (Связь), значка с изображением папки справа от него и списка **Target** (Цель) аналогично назначению этих элементов на панели **Properties** (Свойства). В строку **Title** (Название) можно ввести текст всплывающей подсказки. В поле **Access key** (Клавиша доступа) можно указать клавишу, которая в сочетании с клавишей <Alt> (например <Alt>+<W>) позволит в браузере осуществить переход по ссылке без помощи мыши. Эта возможность реализована не во всех браузерах. В строку **Tab index** можно ввести порядковый номер ссылки. В этом случае в окне браузера переход от одной ссылки к другой с помощью клавиши <Tab> сначала будет осуществляться по нумерованным ссылкам от меньшего номера к большему, а затем по ненумерованным ссылкам в порядке их расположения.

Для открытия диалогового окна **Hyperlink** (Гиперссылка) можно также воспользоваться командой меню **Insert** | **Hyperlink** (Вставка | Гиперссылка).

2.5.3. Создание гиперссылки внутри одного документа

Гиперссылки внутри одного документа осуществляются с целью быстрой загрузки в окно браузера нужного раздела длинного документа. Поиск раздела с помощью полосы прокрутки может занять много времени, тогда как, имея перед глазами оглавление документа со ссылками на отдельные разделы, щелчком по названию нужного раздела можно загрузить его в окно браузера. Чтобы это стало возможным, в начале каждого раздела требуется помещать его персональное имя. Для этого необходимо перед началом раздела установить курсор и в группе инструментов **Common** (Общие) панели **Insert** (Вставка) нажать кнопку  — **Named Anchor** (Именной якорь). Откроется одноименное диалоговое окно, в котором в строке **Anchor name** (Имя якоря), не используя пробелы, нужно набрать условное имя раздела (например "paragraph1"). После нажатия **OK** перед разделом появится значок с изображением якоря. Если значок не появился, то следует дать команду меню **View | Visual Aids | Invisible Elements** (Вид | Визуальные средства | Невидимые элементы). То же самое следует проделать и с другими разделами. После этого можно приступить к созданию оглавления документа, которое целесообразно разместить в его начале.

В отличие от условных имен разделов, которые предназначены для внутреннего использования и могут не нести никакой смысловой нагрузки, строки оглавления предназначены для посетителей сайта и должны достаточно точно характеризовать раздел документа, к которому осуществляется переход. Создав оглавление, нужно установить связь между его строками и именами разделов, которые помечены значками с изображением якоря. Для этого необходимо:

1. Выделить первую строчку оглавления и с помощью полосы прокрутки расположить текст документа таким образом, чтобы изображение первого якоря оказалось в поле зрения.
2. Установить указатель мыши на значок **Point to File** (Направление к файлу) на панели **Properties** (Свойства).
3. Нажать левую кнопку мыши и, не отпуская ее, переместить указатель, который должен иметь вид значка **Point to File** (Направление к файлу), на изображение якоря.

4. Отпустить кнопку мыши и обратить внимание на то, что ссылка готова, а в список **Link** (Связь) попало имя раздела.

Аналогичные действия нужно проделать с другими строками оглавления и соответствующими якорями. Результаты работы можно посмотреть в браузере, нажав клавишу <F12>.

2.5.4. Создание гиперссылки на почтовый адрес

Иногда возникает необходимость создания ссылки на почтовый адрес. Особенностью такой гиперссылки является то, что вместо имени файла, на который осуществляется ссылка, указывается адрес электронной почты. Щелчок по гиперссылке в окне браузера приведет к открытию почтовой программы (например MS Outlook Express) с заготовкой нового письма и с заполненной адресной строкой. Пользователю остается только написать письмо, указать его тему и нажать кнопку **Отправить**.

Установим курсор в то место, где должна разместиться гиперссылка и нажмем кнопку — **Email Link** (Связь по электронной почте) в группе **Common** (Общие) панели **Insert** (Вставка). В открывшемся диалоговом окне **Email Link** (Связь по электронной почте) в строке **Text** (Текст) написать текст гиперссылки, а в строке **E-Mail** — адрес электронной почты. Пример заполнения строк данного диалогового окна показан на рис. 2.13.

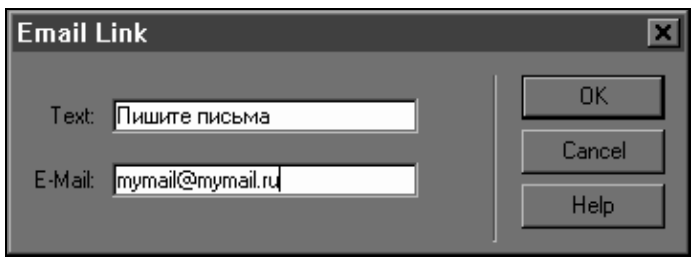


Рис. 2.13. Диалоговое окно создания ссылки на адрес электронной почты

После нажатия кнопки **OK** в документе появится готовая к работе гиперссылка.

Глава 3

Подготовка и размещение графических изображений

3.1. Форматы графических файлов, используемых в web-сайтах

Прежде чем приступить к непосредственному изложению вопросов по подготовке и размещению иллюстраций на web-сайте, следует рассмотреть используемые с этой целью форматы графических файлов, а также их особенности.

- *Формат GIF* (Graphics Interchange Format — формат обмена графической информацией). Файлы в этом формате имеют расширение gif.

  Достоинства формата GIF:

  - в одном файле могут храниться несколько изображений (кадров), которые демонстрируются через заданные промежутки времени (GIF-анимация);
  - изображения могут содержать прозрачные цвета;
  - возможность сохранения и последующего вывода изображения с чередованием строк (interlacing), т. е. изображение появляется целиком после передачи только части строк, но отдельные детали уточняются по мере загрузки, в отличие от обычного построчного вывода сверху вниз (разница проявляется при небольшой скорости загрузки документа);

- сравнительно высокая степень сжатия без потерь информации и, следовательно, относительно небольшие размеры файлов.

Недостаток:

- Изображение может содержать только 256 цветов.

Рекомендации по использованию:

- изображения с ограниченным количеством цветов (рисованные изображения, графики, диаграммы);
- простые изображения и изображения, содержащие текст (кнопки, меню, списки);
- анимационные изображения.

❐ *Формат JPEG* (Joint Photographic Experts Group — объединенная группа экспертов в области фотографии). Файлы в формате JPEG имеют расширение jpg.

Достоинства формата JPEG:

- изображение может содержать 16 777 216 цветов (24 бита на пиксель);
- очень высокая степень сжатия информации с сохранением высокого качества изображения, несмотря на потерю информации при сжатии (степень сжатия и качество изображения выбираются при сохранении);
- позволяет задать прогрессивный (progressive) режим вывода изображения на экран, при котором изображение появляется полностью после передачи лишь части информации, в отличие от обычного вывода с построчной разверткой сверху вниз. Разницу можно заметить при небольшой скорости загрузки документа.

Недостаток:

- Отсутствуют возможности формата GIF по созданию прозрачного цвета и анимации.

Рекомендации по использованию:

- Многоцветные фотографические изображения;
- Изображения с большим количеством цветов и оттенков.

- *Формат PNG* (Portable Network Graphics – переносимая сетевая графика). Расширение файлов в этом формате — png. Формат задуман в качестве альтернативы форматам GIF и JPEG. Значительно уступает формату JPEG в степени сжатия информации, а формату GIF — в возможности создания анимации. Несмотря на многолетнее существование, широкого распространения не получил.

3.2. Размещение графики в HTML-документах

Графические изображения размещаются в HTML-документах с помощью метки `<IMG>`. Пример записи:

```
<IMG SRC="имя файла рисунка">
```

Параметр `SRC` предназначен для задания имени файла рисунка, который должен располагаться в той же папке, что и HTML-документ. Если рисунки располагаются в других папках, то в этом случае необходимо указать путь к файлу рисунка. Например:

```
<IMG SRC="pictures/pic1.jpg">
```

Рисунок pic1.jpg находится в папке pictures, которая расположена в той же папке, что и HTML-документ.

Рисунки могут располагаться и на других ресурсах сети Интернет. В этом случае необходимо указать адрес ресурса, путь к файлу и имя файла рисунка. Например:

```
<IMG SRC=http://www.mysite1.ru/pictures/pic1.jpg>
```

Параметр `SRC` является обязательным. Кроме него могут использоваться следующие параметры метки `<IMG>`:

- `ALT="текст"` — позволяет задать альтернативный текст, который будет воспроизводиться браузером, если загрузка рисунка невозможна. Кроме того, этот текст будет отображаться в виде всплывающей подсказки при наведении курсора на рисунок;
- `ALIGN="значение"` — задает выравнивание текста, идущего за меткой `<IMG>`, относительно рисунка. Параметр `ALIGN` может принимать следующие значения:

- TOP — самый высокий элемент строки (не обязательно текстовый) выравнивается по верхней границе изображения;
- TEXTTOP — самый высокий текстовый элемент строки выравнивается по верхней границе изображения;
- MIDDLE — базовая линия строки выравнивается посередине изображения;
- ABSMIDDLE — середина строки выравнивается посередине изображения;
- BOTTOM — базовая линия строки выравнивается по нижней границе изображения (действует по умолчанию);
- ABSBOTTOM — нижняя граница строки выравнивается по нижней границе изображения.

В качестве примера рассмотрим следующий документ (листинг 3.1).

Листинг 3.1. Размещение изображения в HTML-документе. Строка, идущая за изображением, выравнивается по его верхней границе

```
<HTML>
<HEAD>
<TITLE>
Графика
</TITLE>
</HEAD>
<BODY>
<IMG SRC=1.jpg ALT="Петродворец" ALIGN=top>
Эта строка <IMG SRC=1.jpg ALT="Петродворец" WIDTH=50> располагается за рисунком и выравнивается параметром ALIGN со значением TOP.
</BODY>
</HTML>
```

В состав строки, идущей за рисунком, входит уменьшенная копия того же рисунка (параметром WIDTH уменьшена ширина до 50 px). Строка выравнивается по верхнему краю уменьшенного рисунка. Результат в окне браузера показан на рис. 3.1.

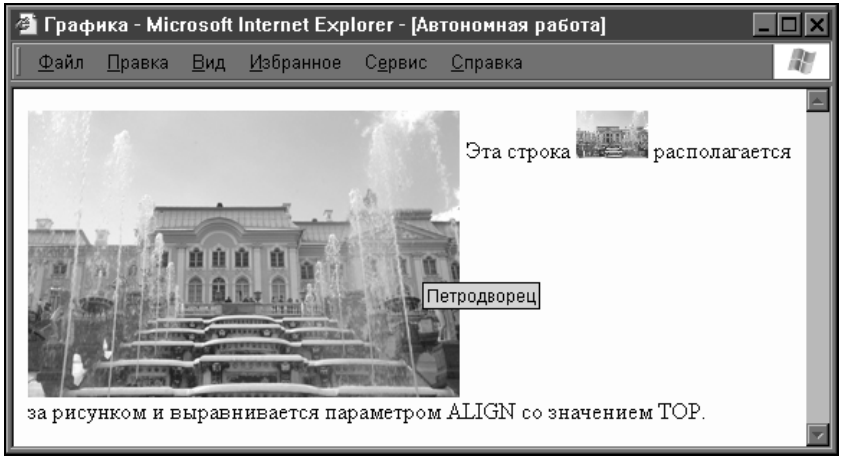


Рис. 3.1. Пример выравнивания самого высокого элемента строки по верхней границе изображения

При использовании перечисленных значений параметра `ALIGN`, справа от рисунка выводится только одна строка. Чтобы текст обтекал рисунок, можно использовать следующие значения:

- `LEFT` — изображение располагается слева, текст обтекает его справа;
- `RIGHT` — изображение располагается справа, текст обтекает его слева.

Прервать обтекание рисунка текстом можно меткой `<BR>` с параметром `CLEAR`, который принимает значения `LEFT` и `RIGHT` в зависимости от значения параметра `ALIGN` (листинг 3.2).

Листинг 3.2. Размещение изображения в HTML-документе, текст обтекает рисунок справа

```
<HTML>
<HEAD>
<TITLE>
Графика
</TITLE>
</HEAD>
```

```
<BODY>
<IMG SRC=1.jpg ALT="Петродворец" ALIGN=left>
Этот текст располагается за рисунком и выравнивается параметром ALIGN со значением LEFT. Текст обтекает рисунок справа.
<BR CLEAR=LEFT> В этом месте обтекание прерывается.
</BODY>
</HTML>
```

В браузере это будет выглядеть так, как показано на рис. 3.2.

Рис. 3.2. Пример обтекания рисунка текстом и прерывания обтекания

Другие параметры метки `<IMG>`:

- `WIDTH="значение"` — ширина изображения в пикселях или в процентах от ширины элемента, в котором находится изображение (полное окно, ячейка таблицы);
- `HEIGHT="значение"` — высота изображения в пикселях или в процентах от высоты элемента, в котором находится изображение, если высота элемента определена;
- `HSPACE="значение"` — зазор между текстом и изображением в пикселях по горизонтали;
- `VSPACE="значение"` — зазор между текстом и изображением в пикселях по вертикали;
- `BORDER="значение"` — толщина рамки вокруг изображения.

Изменяя размеры изображения, следует использовать только один из параметров `WIDTH` или `HEIGHT`. Другой параметр изменится автоматически пропорционально первому. Одновременное изменение двух параметров в случае нарушения пропорций приведет к искажению изображения.

Примечание

Если предполагается всегда выводить изображение с уменьшенными размерами, то лучше предварительно уменьшить размеры исходного рисунка, а не прибегать к использованию параметров `WIDTH` или `HEIGHT`. Это позволит уменьшить размер передаваемого по сети файла.

Параметры `HSPACE` и `VSPACE` необходимо использовать для увеличения зазоров между рисунком и текстом, который по умолчанию слишком мал. Так в последнем примере следовало бы задать `HSPACE=10` и `VSPACE=7`.

Рамки вокруг рисунков устанавливаются редко, но в одном случае параметр `BORDER` может пригодиться. Если по рисунку осуществляется ссылка, то вокруг него появится рамка. Чтобы избавиться от нее можно использовать параметр `BORDER=0`.

По рисункам можно осуществлять ссылки не только на другие HTML-документы, но и на другие ресурсы сети Интернет. Для создания ссылок по рисункам применяются те же способы, что и для гипертекста. Разница заключается лишь в том, что вместо гипертекста должна быть вставлена метка рисунка. Например:

```
<A HREF="doc.htm"><IMG SRC=1.gif BORDER=0></A>
```

Ссылка осуществляется по рисунку 1.gif на документ doc.htm.

3.3. Работа с графикой в программе Dreamweaver

3.3.1. Размещение графики

Разместить графические изображения не web-странице можно следующими способами:

- Воспользоваться командой **Image** (Изображение) меню **Insert** (Вставка).

- Активизировать кнопку — **Image** (Изображение) в группе инструментов **Common** (Общие). Треугольная стрелка справа от кнопки позволяет раскрыть список **Images** (Изображения), в который входит и кнопка **Image** (Изображение). Видимой является пиктограмма той кнопки, которая была выбрана последней. В семействе **Images** (Изображения) также находятся кнопки:
  - **Image Placeholder** (Временный заместитель изображения) — вставка временного объекта, который должен обозначить место будущего изображения на странице (в дальнейшем должен быть заменен реальным изображением);
  - **Rollover Image** (Ролловер) — графическая кнопка, изменяющая свой вид в зависимости от действий пользователя, например при наведении на нее курсора. Программа на JavaScript, управляющая сменой изображений будет вставлена автоматически. Пример создания ролловера будет рассмотрен в конце главы;
  - **Fireworks HTML** — вставка HTML-документа, созданного в графическом редакторе Fireworks;
  - **Navigation Bar** (Навигационная панель) — вставка связанных между собой графических кнопок для навигации по страницам сайта;
  - группа из трех кнопок, предназначенных для создания на изображении чувствительных областей, по которым можно осуществлять ссылки на другие документы. В эту группу входят кнопки: **Draw Rectangle Hotspot** (Создание чувствительной области прямоугольной формы), **Draw Oval Hotspot** (Создание чувствительной области овальной формы), **Draw Polygon Hotspot** (Создание чувствительной области в форме многоугольника). В дальнейшем будет рассмотрена возможность создания чувствительных областей (изображений-карт) средствами программы Photoshop (ImageReady).

Независимо от выбранного способа вставки изображения откроется диалоговое окно **Select Image Source** (Выбор источника изображения) (рис. 3.3).

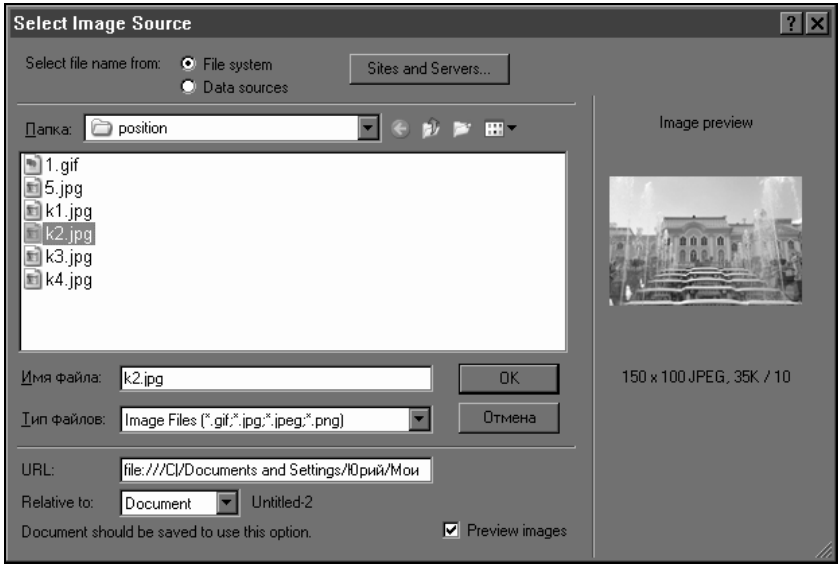


Рис. 3.3. Диалоговое окно **Select Image Source**

Для вставки рисунков, находящихся на компьютере, следует переключатель **Select file name from** (Выбор имени файла из) установить в положение **File system** (Обычный файл). Указание абсолютного пути к файлу рисунка в строке **URL** и предупреждающая запись в нижней части окна **Document should be saved to use this option** (Документ должен быть сохранен, чтобы использовать этот выбор) свидетельствуют о том, что документ еще не был сохранен в папке сайта. После сохранения документа абсолютный адрес будет автоматически заменен именем файла рисунка. В списке **Relative to** (Относительно) можно выбрать категорию, относительно чего задается путь к файлу: рисунка, текущего документа (Document) или корневой папки сайта (Site Root). Установив флажок **Preview images** (Предварительный просмотр изображений), можно осуществлять предварительный просмотр выбранного изображения. В остальном диалоговое окно **Select Image Source** (выбор источника изображения) аналогично диалоговым окнам открытия документа.

При выборе файла изображения, находящегося за пределами папки сайта, откроется диалоговое окно, предупреждающее об этом (*см. раздел 2.5.1*). Следует нажать кнопку **OK** и тогда будет предложено сохранить копию изображения в папке сайта, что избавит от дальнейших проблем, связанных с его адресацией.

3.3.2. Изменение свойств изображения

После того как изображение выделено, т. е. вокруг него появилась рамка и три маркера, на панели свойств **Properties** (Свойства) можно изменять его параметры (рис. 3.4).

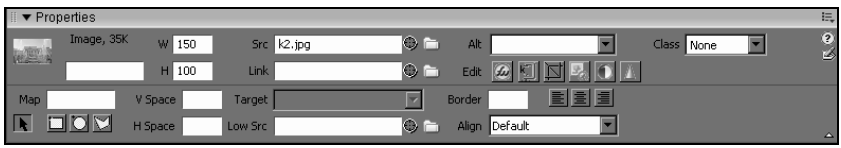


Рис. 3.4. Внешний вид панели **Properties**, отображающей свойства выделенного изображения

Поле **Image** (Изображение) предназначено для задания индивидуального имени изображения, которое может потребоваться, например, если оно станет объектом программирования.

В полях **W** и **H** можно изменить ширину и высоту изображения, а в **V Space** и **H Space** задать зазоры между текстом и изображением по вертикали и горизонтали.

Поле **Map** (Карта) предназначено для ввода имени карты чувствительных элементов на изображении, а кнопки под ним — для создания самих элементов карты различной формы. В дальнейшем будет рассмотрена возможность создания чувствительных областей (изображений-карт) средствами программы Photoshop (ImageReady).

В поле **Src** должно находиться имя файла изображения, которое можно изменить, используя кнопку **Browse for File** (Поиск файла).

Поле **Link** (Ссылка) предназначено для ввода имени файла, на который осуществляется ссылка по рисунку. После ввода имени файла для ссылки станет активным список **Target** (Цель), в котором можно осуществить выбор окна или фрейма для загрузки

адресуемого документа. В данном списке возможен выбор следующих значений:

- **blank** — документ откроется в новом окне браузера;
- **_self** — документ откроется в том же окне или фрейме (по умолчанию);
- **_parent** — документ откроется в родительском фрейме;
- **_top** — документ займет все окно браузера.

Последние два значения используются только при наличии фреймов (*см. главу 5*).

Поле **Low Src** предназначено для ввода имени файла рисунка, загрузка которого в документ предшествует загрузке основного изображения. Для этого целесообразно использовать аналогичный рисунок, но более низкого качества и со значительно меньшим размером файла. При медленной загрузке информации из Интернета и большом объеме файла основного изображения ввод имени предшествующего рисунка в данное поле позволит пользователю получить представление об изображении до загрузки основного файла.

В поле **Alt** можно внести текст, который будет появляться в окне браузера при наведении указателя мыши на рисунок, в качестве всплывающей подсказки.

Список **Class** предназначен для выбора стиля форматирования. Список будет заполнен, если к документу подключены каскадные таблицы стилей (CSS) (*см. главу 7*).

В поле **Border** (Рамка) вводится толщина рамки изображения. Введя значение “0” можно избавится от рамки, которая устанавливается, если по рисунку осуществляется ссылка.

Три кнопки справа от поля **Border** (рамка) предназначены для выравнивания рисунка по горизонтали:

- **Align Left** — выравнивание по левому краю;
- **Align Center** — выравнивание по центру;
- **Align Right** — выравнивание по правому краю.

Раскрывающийся список **Align** (Выравнивание) позволяет выбрать способ выравнивания текста относительно изображения.

Все значения уже комментировались в начале главы в пояснениях к параметрам метки <IMG>.

Серия кнопок **Edit** (Правка) предназначена для редактирования изображения. Нажав первую из них, **Edit** (Правка), можно открыть изображение в графическом редакторе. В соответствии с настройками по умолчанию для этих целей используется графический редактор MS Photo Editor. Возможно, что после знакомства с графическим редактором Adobe Photoshop в следующей части этой главы, вы предпочтете открывать изображение для редактирования именно в этом редакторе. Чтобы это стало возможным, командой **Edit | Preferences** (Правка | Настройки) необходимо открыть диалоговое окно **Preferences** (Настройки) и выбрать категорию File Types/Editors (Файл Типы/Редакторы). В столбце **Extensions** (Расширения) выделить строку **.jpg .jpe .jpeg** и добавить в столбец **Editors** (Редакторы) новый графический редактор. Для этого нужно над столбцом нажать кнопку со значком "+" и в открывшемся диалоговом окне **Select External Editor** (Выбор внешнего редактора) выбрать файл Adobe\Photoshop\Photoshop.exe. После появления записи о новом редакторе в столбце **Editors** (Редакторы), его нужно выделить и нажать кнопку **Make Primary** (Сделать основным). Те же действия можно проделать для изображений с расширением gif.

Кнопка **Optimize Fireworks** (Оптимизация файлов редактора Fireworks) предназначена для редактирования файлов, созданных в этом редакторе.

Кнопка **Crop** (Обрезка) позволяет осуществить кадрирование (обрезку) изображения, не выходя из программы Dreamweaver.

Кнопка **Brightness and Contrast** (Яркость и контраст) открывает диалоговое окно **Brightness/Contrast** (Яркость/Контраст) (рис. 3.5), позволяющее с помощью ползунка Brightness изменить яркость изображения, а ползунка Contrast — контраст изображения. Просмотр результата данных изменений будет возможен при установленном флажке **Preview** (Предварительный просмотр).

Кнопка **Sharpen** (Резкость) открывает диалоговое окно, позволяющее регулировать резкость изображения (рис. 3.6).

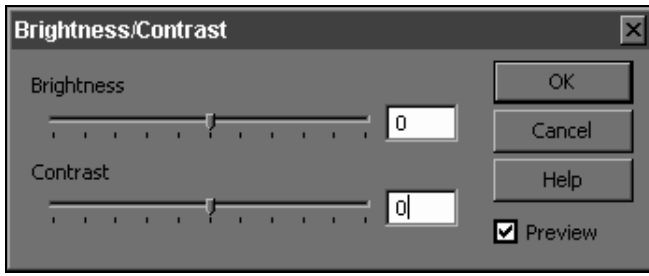


Рис. 3.5. Диалоговое окно **Brightness/Contrast**

Рис. 3.6. Диалоговое окно **Sharpen**

3.4. Подготовка изображений к публикации на сайте

Практически каждый рисунок, предназначенный для публикации на сайте, должен быть предварительно подготовлен. Подготовка может включать широкий диапазон операций — от изменения размеров и обрезки до применения эффектов, фильтров и наложения текста. Для решения этих и многих других задач лучше всего подходит программа Adobe Photoshop. Основное назначение программы Photoshop — обработка существующих изображений. Кроме того, программа располагает необходимыми инструментами и для создания несложных изображений, например кнопок и разделов меню. Владельцы программы Photoshop одновременно являются обладателями еще одной программы — ImageReady. Программа ImageReady не требует отдельной установки и предна-

значена специально для подготовки web-графики. С ее помощью можно создавать GIF-анимацию, фрагментированные изображения, изображения-карты и т. д. С основными возможностями программ Photoshop и ImageReady по подготовке графики для публикации на Web мы и познакомимся в этом разделе.

3.4.1. Интерфейс программы Adobe Photoshop

При загрузке программы Adobe Photoshop, можно заметить, что окно не содержит никакого документа и даже чистой заготовки для его создания. Это подчеркивает, что основное назначение программы — обработка изображений, полученных другими средствами, например, путем сканирования фотографий. Интерфейс программы Photoshop показан на рис. 3.7.

К основным элементам интерфейса относятся:

1. *Строка заголовка программы.* Это стандартный элемент практически любой программы, работающей под управлением операционной системы Windows. В нем располагаются известные элементы управления окном и название программы, в данном случае Adobe Photoshop.
2. *Строка меню.* Меню содержит все команды, которые можно выполнить в данной программе: **File** (Файл), **Edit** (Правка), **Image** (Изображение), **Layer** (Слой), **Select** (Выделение), **Filter** (Фильтр), **View** (Вид), **Window** (Окно), **Help** (Помощь). С отдельными командами этих групп мы будем знакомиться по мере необходимости.
3. *Панель параметров.* Панель содержит необходимые средства для изменения параметров выбранного инструмента, причем содержимое панели изменяется в зависимости от выбранного инструмента.
4. *Панель инструментов.* Каждый инструмент представлен кнопкой. Выбор инструмента производится щелчком по соответствующей кнопке. Наличие маркера в виде черного треугольника в правой нижней части кнопки свидетельствует о том, что под этой кнопкой скрыта группа инструментов. Чтобы увидеть и выбрать инструмент, достаточно нажать кнопку мыши и удерживать ее до появления списка группы. На па-

нели инструментов отображена кнопка того инструмента, который выбирался последним. Кроме инструментов, на этой панели располагаются и другие элементы управления, с которыми познакомимся позже.

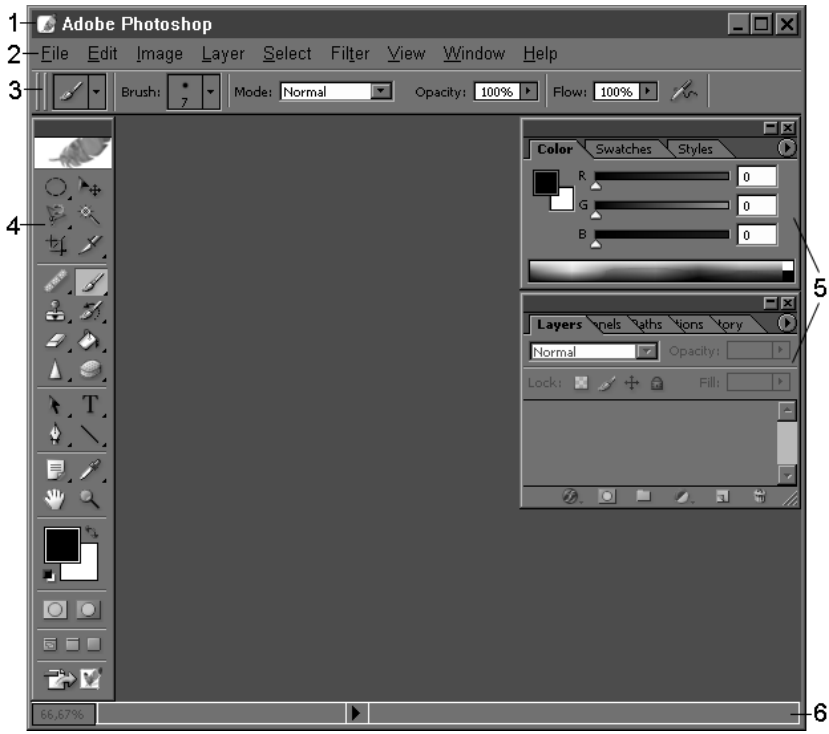


Рис. 3.7. Интерфейс программы Adobe Photoshop

5. *Инструментальные палитры.* Несколько палитр могут группироваться в одном окне. Каждая палитра находится на определенной вкладке. Переход из одной палитры в другую осуществляется щелчком по ярлыку с ее названием. Палитры существенно расширяют возможности программы по обработке изображений.
6. *Строка состояния.* Она отображает полезную информацию о документе и активном инструменте.

3.4.2. Настройка интерфейса программы

Строка заголовка, как и строка меню, всегда отображается на экране в строго определенном месте. Остальные элементы окна можно убирать с экрана и возвращать на прежнее место. Для этой цели предназначено меню **Window** (Окно). Наличие флажка рядом с пунктом меню говорит о том, что данный элемент уже установлен. Щелчок по имени элемента, помеченного флажком, удаляет его с экрана. Повторный щелчок возвращает элемент на прежнее место.

Основными элементами меню **Window** (Окно) являются:

- **Tools** — панель инструментов;
- **Options** — панель параметров;
- **Actions**, **Brushes**, **Channels**, **Character**, **Color**, **Histogram**, **History**, **Info**, **Layer Comps**, **Layers**, **Navigator**, **Paragraph**, **Paths**, **Styles**, **Swatches**, **Tool Presets** — инструментальные палитры;
- **Status Bar** — строка состояния.

Удаление видимой палитры ведет к удалению окна, в котором она находится. Перемещение окон с палитрами осуществляется аналогично перемещению окон в операционной системе Windows. Установив указатель на ярлыке с именем палитры и нажав левую клавишу мыши, палитру можно перемещать из одного окна в другое и даже в отдельное окно.

Аналогично можно перемещать и панель инструментов.

Для перемещения панели параметров необходимо поместить указатель мыши в левую часть панели над двумя рельефными вертикальными линиями, нажать левую клавишу и переместить панель.

3.4.3. Отмена и возврат выполненных действий

Операции, совершаемые над изображением, отображаются отдельной строкой в палитре **History** (История). Палитра позволяет вернуться к любому шагу изменения изображения, для чего необходимо щелкнуть по названию соответствующей операции в палитре. Все действия, выполненные после выделенной опера-

ции, будут отменены. До тех пор, пока не будет выполнена новая операция, отмененные ранее операции можно восстанавливать.

3.4.4. Сохранение изображений

Сохранять изображения можно с помощью традиционных команд меню **File | Save** (Файл | Сохранить) или **File | Save As** (Файл | Сохранить как). Однако в программе Photoshop имеется специальная команда **File | Save for Web** (Сохранить для Web), открывающая одноименное диалоговое окно с большими возможностями для изменения параметров изображений и демонстрации результатов. Диалоговое окно **Save For Web** (Сохранить для Web) показано на рис. 3.8.

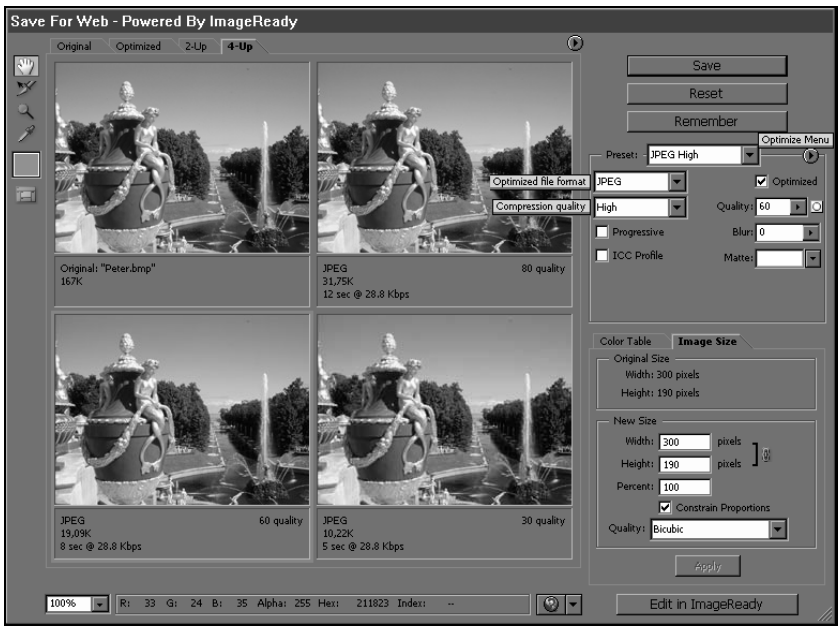


Рис. 3.8. Диалоговое окно **Save For Web**

В левой части окна **Save For Web** (Сохранить для Web) располагается панель инструментов. Остановимся на назначении трех инструментов. Инструмент — **Hand** (Рука) предназначен для

перемещения изображения в окнах просмотра, если изображение не помещается в окне полностью.

Инструмент  — **Zoom** (Масштаб) позволяет изменять масштаб изображения в окнах просмотра. Чтобы увеличить изображение, нужно щелкнуть по нему данным инструментом. Для уменьшения рисунка, в момент щелчка необходимо удерживать клавишу **Alt**.

Инструмент  — **Eyedropper** (Пипетка) позволяет выбрать на изображении определенный цвет, который появится в поле **Eyedropper Color** (Цвет пипетки), расположенном ниже кнопки.

Правее панели инструментов располагается поле просмотра. В зависимости от выбранной вкладки, поле просмотра может содержать:

- **Original** (Оригинал) — окно просмотра с оригиналом, т. е. изображением до изменения параметров;
- **Optimized** (Оптимизированное изображение) — окно просмотра с изображением после изменения параметров;
- **2-Up** (Два изображения) — два окна просмотра с вариантами изображения;
- **4-Up** (четыре изображения) — четыре окна просмотра с вариантами изображения.

В нижней части каждого окна просмотра располагаются основные параметры изображения: формат и размер файла, время передачи файла при определенной скорости и др.

На вкладке **Original** (Оригинал) всегда демонстрируется вид изображения до изменения параметров. Параметры вариантов изображения в остальных окнах просмотра могут изменяться пользователем. Для этого необходимо выделить окно, щелкнув по нему мышкой, и приступить к изменению параметров изображения в правой части диалогового окна. Можно либо выбрать имя набора параметров из списка **Preset** (Установки), подходящего в наибольшей степени, либо выбрать нужный формат из списка **Optimized file format** (Формат оптимизируемого файла) и начать коррекцию параметров. После коррекции набор параметров можно сохранить под определенным именем и использовать в дальнейшем при работе с другими изображениями. Для этого нужно раскрыть меню **Optimize Menu** (Круглая кнопка) и вы-

брать команду **Save Settings** (Сохранить установки). Откроется диалоговое окно **Save Optimization Settings** (Сохранить оптимизированные установки), в котором набору параметров нужно дать имя. Он будет сохранен в папке **Optimized Settings** (Оптимизированные установки). При повторных обращениях к списку **Preset** (Установки) созданный набор параметров будет уже присутствовать. Параметры для форматов JPEG и GIF отличаются друг от друга.

Для формата JPEG можно (рис. 3.8):

- изменить степень сжатия изображения, следовательно, его качество, используя раскрывающийся список **Compression Quality** (Качество сжатия) или поле **Quality** (Качество), расположенные в одном ряду. Чем сильнее сжатие, тем ниже качество изображения. Список **Compression Quality** (Качество сжатия) содержит следующие значения качества:
  - **Low** — низкое;
  - **Medium** — среднее;
  - **High** — высокое;
  - **Very High** — очень высокое;
  - **Maximum** — максимальное.

  Поле **Quality** (Качество) позволяет плавно изменять качество изображения (степень сжатия), устанавливая значения от 0 до 100;
- установить флажок **Progressive** (Прогрессивный). В этом случае изображение будет отображаться в окне браузера поэтапно. На первом этапе оно будет иметь низкое разрешение, которое будет увеличиваться с каждым шагом. Увеличение разрешения означает проявление все более мелких деталей изображения. Это будет заметно при невысокой скорости загрузки изображения по сети и позволит оценить содержимое изображения до его полной загрузки. Если флажок не установлен, то изображение будет разворачиваться на экране построчно сверху вниз и оценить его содержимое возможно только после загрузки значительной части изображения;
- установить флажок **ICC Profile** (ICC профиль), в результате чего будут сохранены данные управления цветом, которые

позволят браузеру улучшить качество изображения. Размер файла при этом незначительно возрастает;

- установить флажок **Optimized** (Оптимизированный), предварительно сняв флажок **Progressive** (Прогрессивный), что позволит не только оптимизировать передачу цвета, но и несколько уменьшить размер файла;
- изменить значение поля **Blur** (Размытие) от 0 до 2. С увеличением значения увеличивается размытие, что позволяет сгладить резкие границы деталей изображения;

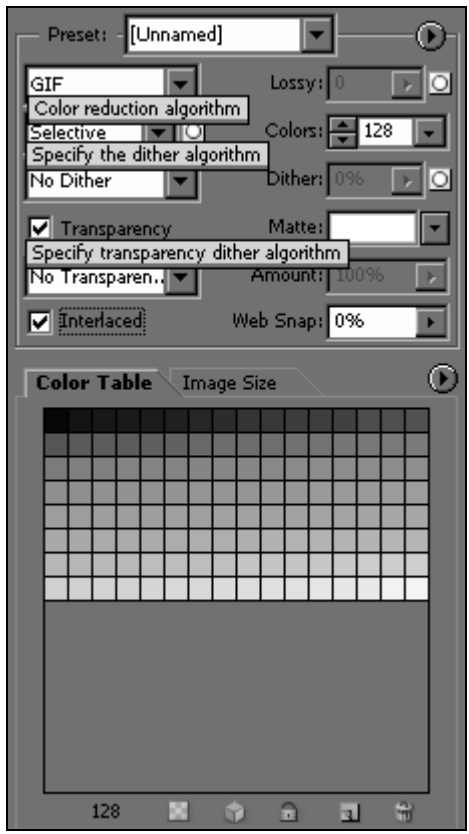


Рис. 3.9. Панель установки параметров формата GIF в диалоговом окне **Save For Web** и палитра цветов **Color Table**

- задать в поле **Matte** (Фон), то есть цвет, который будет использоваться вместо прозрачного фона, не поддерживающегося форматом JPEG. Цвет можно выбрать из списка:
  - **None** — по умолчанию;
  - **Eyedropper Color** — выбранный пипеткой;
  - **White** — белый;
  - **Black** — черный;
  - **Other** — другой (открывается диалоговое окно **Color Picker** (Выбор цвета)).

Вкладка **Image Size** (Размер изображения) является общей для всех форматов. Она предназначена для изменения размера изображения. При установленном флажке **Constrain Proportions** (Сохранять пропорции) достаточно поменять ширину (**Width**) или высоту (**Height**) либо в поле **Percent** (Процент) задать изменение размеров в процентах.

В раскрывающемся списке **Quality** (Качество) устанавливается способ интерполяции (улучшения качества изображения после изменения размеров). Среди предлагаемых вариантов можно рекомендовать **Bicubic** (Бикубическое сглаживание). Размеры изменятся после нажатия на кнопку **Apply** (Применить).

При выборе формата GIF изменится содержимое панели установки параметров (рис. 3.9)

На панели установки параметров формата GIF можно выполнить следующие действия:

- в раскрывающемся списке **Color reduction algorithm** (Алгоритм сокращения цветов) выбрать один из алгоритмов перехода от полноцветного изображения к изображению с меньшим числом цветов, так как формат GIF позволяет использовать не более 256 цветов. Палитра используемых цветов показана на вкладке **Color Table** (Таблица цветов), расположенной в нижней части панели. Такое преобразование сопровождается искажением цвета некоторых фрагментов изображения и ухудшением его качества. Чтобы потери, связанные с уменьшением количества цветов, свести к минимуму, разработаны специальные алгоритмы такого перехода. Возможен выбор одного из следующих алгоритмов:

- **Selective** (Селективный) — учитываются особенности восприятия цвета глазом и требования web-безопасности (по умолчанию);
- **Perceptual** (Перцептуальный) — учитываются только особенности восприятия цвета человеческим глазом;
- **Adaptive** (Адаптивный) — используются цвета, встречающиеся в изображении наиболее часто;
- **Restrictive (Web)** (Web-безопасный) — используются только web-безопасные цвета (к web-безопасным относятся 216 цветов);
- **Custom** (Пользовательский) — пользователь может сам расширить палитру используемых цветов, выбирая их пипеткой на оригинале изображения и нажимая кнопку **Adds eyedropper color to palette** (Добавить цвет выбранный пипеткой в палитру), которая располагается в нижней части вкладки **Color Table** (вторая справа), предварительно выделив окно просмотра с недостающим цветом.

Рекомендуется использовать алгоритм, дающий максимальное качество изображения. В том же ряду кнопок первой слева идет еще одна полезная кнопка — **Maps selected colors to transparent** (Сделать выделенные цвета прозрачными). После выделения одного или нескольких цветов в палитре (удерживая <Shift> или <Ctrl>), нажав эту кнопку, можно выделенные цвета сделать прозрачными. В списке **Color reduction algorithm** (Алгоритм сокращения цветов) можно также выбрать одну из палитр с фиксированными цветами:

- **Black&White** — изображение преобразуется в черно-белое двуцветное изображение;
- **Grayscale** (оттенки серого) — изображение преобразуется в черно-белое с 256 оттенками серого цвета;
- **Windows** — сохраняются 256 цветов из стандартной палитры Windows;

- в списке **Specify the dither algorithm** (Определить алгоритм имитации) можно выбрать один из алгоритмов имитации цветов, отсутствующих на экране монитора: **Pattern** (Узор), **Diffusion** (Диффузия), **Noise** (Шум). В случае выбора одного из алгоритмов в поле **Dither** (Степень имитации) с помощью ползунка можно установить степень имитации;

- установить флажок **Transparency** (Прозрачность), обеспечив сохранение информации о прозрачных областях изображения. В случае снятия флажка, прозрачные области будут окрашены цветом, заданным в поле **Matte** (Фон);
- выбрать из списка **Specify transparency dither algorithm** (Определить алгоритм имитации прозрачности) один из алгоритмов имитации полупрозрачных тонов. Список содержит те же алгоритмы, что используются для имитации цвета. Необходимость такой имитации возникает в связи с тем, что сохранить плавный переход от пикселей изображения к прозрачному фону, например полученный в результате растушевки, невозможно. Можно лишь сохранить плавный переход от пикселей изображения к фону, заданному в поле **Matte** (Фон). Если же в поле **Matte** (фон) установлено значение **None** (Нет), то переход станет резким. Использование одного из алгоритмов списка **Specify transparency dither algorithm** (Определить алгоритм имитации прозрачности) позволит имитировать плавный переход. Поле **Amount** (Количество) позволяет регулировать глубину имитации перехода (от 0 до 100 %) для алгоритма **Diffusion Transparency Dither** (Диффузионная имитация прозрачности);
- установить флажок **Interlaced** (Чересстрочная развертка), что обеспечит пошаговую загрузку изображения по сети, с постепенным повышением четкости изображения, аналогично режиму **Progressive** (Прогрессивный) для формата JPEG;
- в поле **Lossy** (Потери) можно установить величину допустимых потерь качества изображения (от 0 до 100), наблюдая результат на образце, получив взамен уменьшение размера файла;
- в поле **Web Snap** (Приближение к Web) устанавливается степень приближения цветов данного изображения к web-безопасной палитре. Рекомендуемое значение "0".

3.4.5. Изменение размеров изображения

Изменение размеров графического объекта может потребоваться уже на ранних стадиях работы, например, при использовании в одном рисунке фрагментов изображений разного масштаба. Из-

меняя размеры изображения, следует помнить, что во всех случаях это ведет к ухудшению его качества. Чем в большей степени изменяется размер графического объекта относительно исходного, тем сильнее это становится заметным. Поэтому целесообразно задавать нужный размер рисунка на этапе его создания, например при сканировании.

Изменение размеров изображения осуществляется в диалоговом окне **Image Size** (Размер изображения) (рис. 3.10), которое вызывается с помощью команды **Image Size** (Размер изображения) меню **Image** (Изображение).

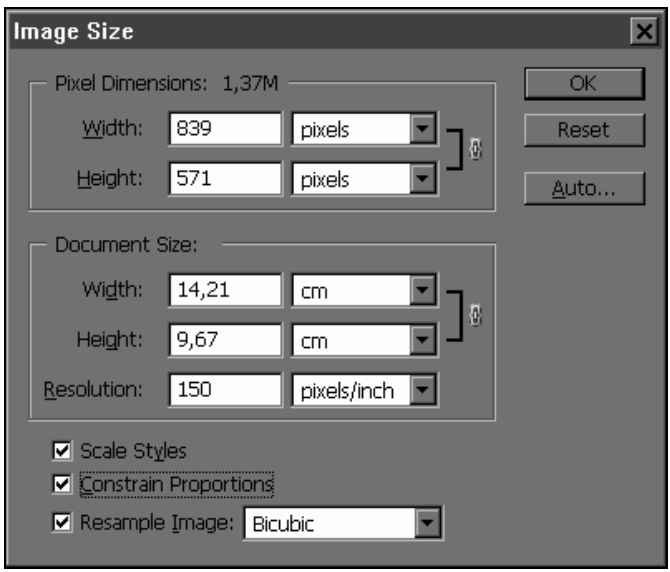


Рис. 3.10. Диалоговое окно **Image Size**

В данном окне следует, прежде всего, обратить внимание на состояние флажков в нижней части окна. Должен быть установлен флажок **Resample Image** (Интерполировать изображение), что позволит из раскрывающегося списка выбрать один из способов интерполяции изображения, т. е. способов улучшения качества после изменения размеров. Рекомендуется использовать бикубическую интерполяцию (Bicubic). Флажок **Scale Styles** (Масштабировать стили) также рекомендуется установить. Установка

флажка **Constrain Proportions** (Сохранять пропорции) позволит изменить только один размер изображения — второй изменится пропорционально. При снятом флажке **Constrain Proportions** (Сохранять пропорции) каждый размер устанавливается самостоятельно, а непропорциональное их изменение приведет к искажению изображения. Так как подготавливаемые нами изображения предназначены для демонстрации на экране монитора, то его размер должен задаваться в пикселях и устанавливаться в разделе **Pixel Dimensions:** (Величины в пикселях:). Впрочем, пиксели (**pixels**) можно заменить процентами (**percent**), выбрав **percent** из раскрывающегося списка. В этом случае можно задать изменение размеров изображения в процентах от первоначальных величин. В строке **Pixel Dimensions:** приводится размер файла изображения без сжатия. Такой размер он будет иметь в формате BMP. Эта величина в байтах определяется перемножением числа пикселей по ширине и высоте изображения и умножением результата на 3 (3 байта требуется для кодирования цвета каждого пикселя, что позволяет воспроизводить свыше 16 миллионов цветов – 2^{24}). Не следует бояться этой величины, поскольку в форматах, рекомендуемых для Web, размер файла будет значительно меньше. Размеры, приведенные в разделе **Document Size** (Размер документа), важны при печати. Они связаны с размерами в пикселях через разрешение (**Resolution**).

3.4.6. Кадрирование (обрезка) изображений

Кадрирование (обрезка) осуществляется с целью выделения из изображения фрагмента прямоугольной формы, который можно сделать самостоятельным изображением и сохранить в виде отдельного файла. Для реализации кадрирования на панели инструментов нужно выбрать инструмент  — **Crop Tool** (Обрезка), затем нажать левую клавишу мыши и, удерживая ее, растянуть рамку выделения вокруг нужного фрагмента. После выполнения данных действий по периметру выделения появится бегущая рамка с маркерами в виде квадратов. С помощью маркеров можно изменить размеры фрагмента. Кроме того, поместив указатель внутрь фрагмента, нажав и удерживая клавишу мыши, его можно переместить в любое место рисунка. Прерывание кадри-

рования производится клавишей <Esc>, а завершается нажатием клавиши <Enter> или двукратным щелчком внутри фрагмента.

Возможны три варианта кадрирования.

- *Кадрирование без масштабирования фрагмента.* Полученное изображение имеет такие же размеры, какие имел выделенный фрагмент в исходном изображении. Для этого необходимо очистить поля **Width** (Ширина), **Height** (Высота) и **Resolution** (Разрешение) на панели параметров, нажав кнопку **Clear** (Очистить) на той же панели. Далее нужно растянуть рамку выделения вокруг фрагмента и завершить кадрирование.
- *Кадрирование с увеличением выделенного фрагмента до размеров исходного изображения.* Выделенный фрагмент сохраняет пропорции исходного изображения. Такое кадрирование осуществляется после заполнения полей **Width** (Ширина), **Height** (Высота) и **Resolution** (Разрешение) соответствующими параметрами исходного изображения, для чего необходимо нажать кнопку **Front Image** (До размеров изображения) на панели параметров. Дальнейшие действия аналогичны предыдущему варианту кадрирования.
- *Кадрирование с масштабированием выделяемого фрагмента до заданных размеров.* Размеры нового изображения необходимо вписать в поля **Width** (Ширина) и **Height** (Высота) на панели параметров. Если требуется изменить единицы измерения размеров, то необходимо щелкнуть по полю правой кнопкой мыши, выбрать другую единицу измерения, после чего изменить размер и нажать <Enter>. Поле **Resolution** (Разрешение) можно оставить незаполненным, в этом случае сохранится разрешение исходного изображения. Выделяемый фрагмент будет сохранять пропорции, заданные в полях **Width** (Ширина) и **Height** (Высота). В зависимости от соотношения истинных и заданных размеров выделяемого фрагмента, он будет либо увеличен, либо уменьшен до заданных размеров.

3.4.7. Выбор цвета

Работа многих инструментов в программе Photoshop заключается в изменении цвета пикселей. Для этого используются два цвета: **Foreground** (Основной) и **Background** (Фоновый). Изменение ос-

новного и фонового цветов осуществляется в блоке установки цвета —  на панели инструментов. Цвет верхнего квадрата соответствует цвету **Foreground** (Основной), цвет нижнего квадрата — цвету **Background** (Фон). Поменять местами основной и фоновый цвет можно, щелкнув по двунаправленной стрелке. Щелчок по уменьшенным квадратикам приведет к установке цветов по умолчанию (черный и белый). Изменение цвета осуществляется в диалоговом окне **Color Picker** (Выбор цвета), которое открывается после щечка по соответствующему квадрату (рис. 3.11).

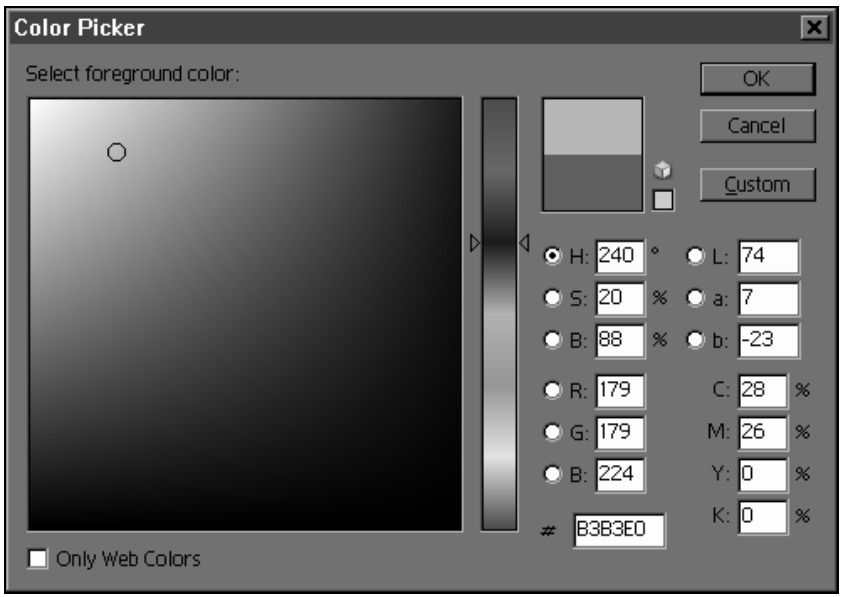
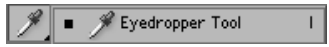


Рис. 3.11. Диалоговое окно **Color Picker**

Перемещая ползунок вдоль вертикальной полосы, можно выбрать нужный цветовой тон, а, щелкая указателем мыши по квадратному полю в левой части окна, — выбрать желаемый оттенок. Цвет можно также задать и в цифровом виде, установив значения в поля **R**, **G**, **B** (десятичные числа) или в поле # (шестнадцатеричное число). После нажатия **OK** цвет будет изменен.

Иногда требуется установить цвет по уже имеющемуся образцу. Для этой цели служит инструмент **Eyedropper Tool** (Пипетка), который располагается в группе из трех инструментов.



Выбрав инструмент, необходимо щелкнуть по пикселю нужного цвета и основной цвет будет изменен.

Примечание

Цветовой образец можно использовать и из другого изображения, которое даже не является активным.

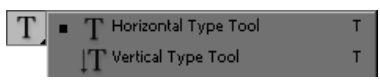
Для изменения фонового цвета нужно удерживать клавишу <Alt>. Для быстрого доступа к пипетке при работе с рисующими инструментами нужно также нажать клавишу <Alt>.

В списке **Sample Size** (Размер образца) на панели параметров инструмента **Eyedropper** (Пипетка) можно выбрать размер пипетки 1 px, 3 × 3 px и 5 × 5 px.

3.4.8. Работа с текстом

Программа Photoshop позволяет размещать на изображении тексты двух видов — *заголовочный* и *абзацный*. Рассмотрим работу только с заголовочным текстом, так как он в наибольшей степени подходит для создания web-публикаций.

Заголовочный текст можно использовать для создания надписей на имеющихся изображениях, а также в качестве самостоятельного изображения, например для создания разделов меню. Для создания текста необходимо нажать кнопку **Horizontal Type** (Горизонтальный текст) или **Vertical Type** (Вертикальный текст) из группы **Type** (Текст) на панели инструментов.



Указатель установить в точку начала набора текста и произвести однократный щелчок левой кнопкой мыши. После появления

мерцающего курсора можно начинать работу по созданию заголовочного текста.

Нажатие клавиши <Enter> на основной клавиатуре ведет к переходу на новую строку. Завершается ввод текста щелчком по крайней правой кнопке **Commit any current edits** (Завершить текущее редактирование) на панели параметров или нажатием клавиши <Enter> в цифровом блоке клавиатуры, или одновременным нажатием клавиш <Ctrl> и <Enter> на основной клавиатуре.

Параметры текста устанавливаются с помощью панели параметров инструментов **Type** (Текст) (рис. 3.12).

Рис. 3.12. Панель параметров инструмента **Type**

Необходимые установки параметров текста можно производить как до набора текста, так и после его завершения. В последнем случае текст должен быть предварительно выделен.

Особое внимание следует обратить на выбор гарнитуры шрифта в раскрывающемся списке **Set the font family** (Установка гарнитуры шрифта) панели параметров. Дело в том, что не все гарнитуры позволяют работать с кириллицей, поэтому перед набором русского текста нужно выбрать гарнитуру, дающую такую возможность. Например: **Arial**, **Comic Sans MS**, **Courier New** и др.

Кроме списка гарнитур, на панели параметров располагаются:

- кнопка переключения горизонтальный/вертикальный текст **Change the text orientation** (Изменить ориентацию текста);
- раскрывающийся список **Set the font style** (Установка начертания шрифта). Для гарнитуры **Arial** этот список содержит:
  - **Regular** — обычный шрифт;
  - **Bold** — полужирный;
  - **Italic** — курсив;
  - **Bold Italic** — полужирный и курсив.

  Для других гарнитур список может содержать не все перечисленные виды начертаний;

- раскрывающийся список **Set the font size** (Установка размера шрифта). Список содержит фиксированный набор размеров шрифта в пунктах (pt), однако при желании можно установить размер, отсутствующий в списке, введя его с клавиатуры;
- раскрывающийся список **Set the anti-aliasing method** (Установка метода сглаживания). Сглаживание осуществляется путем создания плавного перехода от граничных пикселей текста к пикселям фона. Список включает:
  - **None** (Нет) – сглаживание отсутствует;
  - **Sharp** (Резкое);
  - **Crisp** (Четкое);
  - **Strong** (Сильное);
  - **Smooth** (Гладкое).

  На рисунке показано увеличенное изображение двух вариантов написания буквы "ж". В первом случае сглаживание отсутствует (**None**), во втором — сглаживание **Crisp** (Четкое);

- три кнопки выравнивания текста: по левому краю (**Left align text**), по центру (**Center text**) и по правому краю (**Right align text**). Выравнивание производится относительно точки начала ввода текста;
- поле изменения цвета шрифта **Set the text color** (Установка цвета). Щелчок по полю раскрывает диалоговое окно **Color Picker** (Выбор цвета), в котором выбирается нужный цвет;
- кнопка выбора вида искривления текста **Create warped text** (Создание искривленного текста) позволяет раскрыть диалоговое окно **Warp text** (Искривление текста), содержащее раскрывающийся список видов искривлений **Style** (Стиль) и поля параметров искривлений. Значение **None** (Нет) в поле **Style** (Стиль) означает отсутствие искривлений;
- кнопка быстрого вывода на экран палитры **Character** (Шрифт) (рис. 3.13)

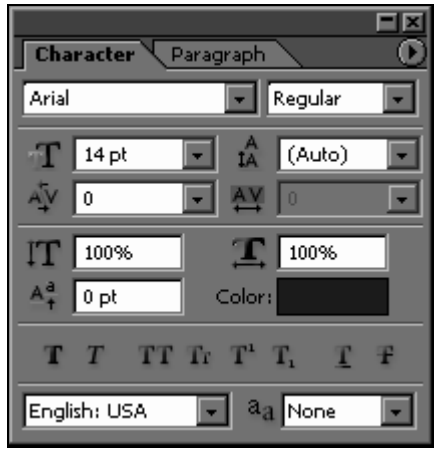


Рис. 3.13. Палитра **Character**

Палитра содержит элементы, дублирующие соответствующие элементы на панели параметров: **Set the font family** — установка гарнитуры шрифта; **Set the font style** — установка начертания шрифта; **Set the font size** — установка размера шрифта; **Set the text color** — установка цвета шрифта; **Set the anti-aliasing method** — установка метода сглаживания), а также:

- раскрывающийся список выбора расстояния между базовыми линиями соседних строк текста **Set the leading** (Установка расстояния между строками). Расстояние измеряется в пунктах (pt) и должно быть установлено до перехода на новую строку. По умолчанию устанавливается **Auto** (Авто), что означает 120 % от размера шрифта;
- два раскрывающихся списка **Set the kerning between two characters** (Установка кернинга между двумя символами) и **Set the tracking for the selected characters** (Установка трекинга для выделенных символов) предназначены для изменения расстояния между символами. *Кернинг* — изменение расстояния между символами, с учетом их начертания (например для сочетания букв АТ можно использовать меньшее расстояние, чем для сочетания букв МН). *Трекинг* — изменение расстояния между символами в зависимости от размера шрифта. Первый список действует на пару символов, между которыми установлен курсор, а второй — на выделенный текст;

- поля **Vertically scale** (Вертикальное масштабирование) и **Horizontally scale** (Горизонтальное масштабирование) предназначены для изменения размеров выделенных символов по вертикали и горизонтали соответственно. Значение нового масштаба вводится в соответствующее поле. Это дает возможность непропорционального изменения размеров текста;
- поле **Set the baseline shift** (Установка отклонения от базовой линии) позволяет задать для горизонтального текста сдвиг базовой линии выделенных символов вверх (положительные значения сдвига) или вниз (отрицательные значения сдвига). Для вертикального текста сдвиг будет осуществляться вправо (положительные значения) или влево (отрицательные значения).

Значения полей можно изменять несколькими способами: либо поместить курсор в соответствующее поле и использовать клавиши управления курсором <↑> и <↓>, либо установить указатель мыши на значок слева от соответствующего поля (при этом он изменит свой вид) и, нажав клавишу мыши, перемещать курсор влево или вправо.

В нижней части палитры располагаются кнопки, предназначенные для быстрого форматирования шрифта. Назначение кнопок следующее (слева направо): полужирный; курсивный; все заглавные буквы; начертание заглавных букв, размер строчных; верхний индекс; нижний индекс; подчеркнутый; зачеркнутый.

Создаваемый текст располагается в текстовом слое палитры **Layers** (Слои). Набор нового текста после завершения набора предыдущего приведет к созданию нового текстового слоя. При необходимости редактирования или форматирования уже созданного текста нужно по нему произвести щелчок инструментом **Type** (Текст). Слой с этим текстом автоматически становится активным.

В качестве примера использования текста рассмотрим создание надписи на рисунке:

1. Откроем файл с рисунком (размеры 300 × 200 px), на котором предстоит создать надпись, используя команду **Open** (Открыть) меню **File** (Файл).
2. Выберем инструмент **Horizontal Type** (Горизонтальный текст) и на панели параметров установим гарнитуру шрифта **Gara-**

mond, начертание курсивное (**Italic**), размер шрифта 30 pt, сглаживание четкое (**Crisp**), выравнивание текста по центру (**Center text**), цвет шрифта черный.

3. Установив курсор в площади изображения, напишем первое слово, а после нажатия клавиши **Enter,** — второе слово.
4. Выбрав инструмент — **Move Tool** (Перемещение) можно не только поместить текст в нужное место рисунка с помощью клавиш управления курсором, но и, установив на панели параметров инструмента **Move Tool** (Перемещение) флажок **Show Bounding Box** (Показать маркеры), подкорректировать его размеры.

Результаты произведенных действий показаны на рис. 3.14.

Рис. 3.14. Пример создания текста на изображении

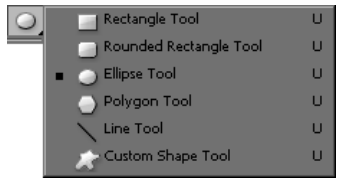
Еще несколько примеров использования текста будут рассмотрены в *разделе 3.4.14.*

3.4.9. Создание простых фигур

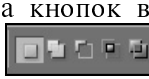
В web-документах иногда требуется расположить простые фигуры (прямоугольники, эллипсы, многоугольники и др.), а также несложные изображения, (например кнопки разделов меню). Для этих целей программа Photoshop располагает инструментами *векторной графики.*

Векторная графика, по сравнению с пиксельной, обладает рядом достоинств. Векторное изображение представляет собой совокупность линий, каждая из которых описывается математической формулой. Оно не искажается при масштабировании. Ин-

струменты векторной графики располагаются на панели инструментов и объединены в одну группу.



На панели параметров инструментов этой группы располагаются три кнопки выбора режима рисования —  . Первая кнопка **Shape Layers** (Слои формы) включает режим создания векторных форм, т. е. фигур векторной графики, заполненных цветом, установленным в поле **Color** (Цвет) на панели параметров. Фигуры будут располагаться в палитре **Layers** (Слои) на отдельном слое, поэтому векторная фигура будет иметь дополнительное преимущество изображения, располагающегося на отдельном слое. Вторая кнопка **Paths** (Контуры) включает режим создания векторных контуров, назначение которых мы рассматривать не будем. Третья кнопка **Fill pixels** (Пиксельное заполнение) позволяет создавать пиксельные изображения фигур, заполненные основным цветом. Фигуры будут располагаться в активном слое. В дальнейшем для создания простых фигур рекомендуется использовать режим **Shape Layers** (Слои формы). Именно для этого режима и будет рассмотрено содержимое панели параметров.

- Кнопки выбора инструмента, дублирующие соответствующие кнопки на панели инструментов. К ним добавлены два инструмента из другой группы: **Pen** (Перо) — для создания прямых и гладких кривых линий и **Freeform Pen** (Свободное перо) — для свободного рисования;
- Кнопка  — **Geometry options** (Геометрические параметры) раскрывает диалоговое окно настройки параметров выбранного инструмента;
- Группа кнопок взаимодействия создаваемых векторных фигур —  :
  - **Create new shape layer** — создать новый слой формы. Вновь создаваемая фигура рисуется на новом слое;

- **Add to shape area** — добавить к площади формы. Вновь создаваемая фигура добавляется к существующей фигуре на том же слое;
- **Subtract from shape area** — вычесть из площади формы. Вновь создаваемая фигура вычитается из существующей фигуры на том же слое;
- **Intersect shape area** — пересечение с площадью формы. Результатом будет область пересечения создаваемой фигуры с уже существующей фигурой на том же слое;
- **Exclude overlapping shape** — исключение накладывающейся формы. Из суммарного изображения исключаются площади пересечения четного числа фигур;

- При нажатой кнопке —  цвет в поле **Color** (Цвет) устанавливается независимо от основного цвета;
- В списке **Style** (Стиль) можно выбрать стиль заливки фигуры либо отказаться от его применения, выбрав **Default Style** (None) — стиль по умолчанию (отсутствие стиля).

На панели инструментов для построения векторных объектов расположены следующие инструменты:

- **Rectangle** (Прямоугольник);
- **Rounded Rectangle** (Прямоугольник со скругленными углами). На панели параметров появляется дополнительное поле для ввода радиуса округления (**Radius**);
- **Ellipse** (Эллипс);
- **Polygon** (Многоугольник). Дополнительное поле на панели параметров для ввода числа сторон (**Sides**);
- **Line** (Линия). Дополнительное поле **Weight** (Толщина) для задания толщины линии;
- **Custom Shape** (Готовая форма). В списке **Shape** (Форма) можно выбрать одну из готовых форм.

Выбрав кнопку нужного инструмента и убедившись, что на панели параметров включен режим **Shape Layers** (Слои формы), можно осуществить построение фигуры. Для этого нужно нажать левую кнопку мыши в начальной точке построения фигуры и, удерживая ее, нарисовать объект.

Изменение расположения фигуры на странице позволяет выполнить инструмент **Move Tool** (Перемещение). Указатель мыши необходимо поместить над объектом, нажать левую кнопку мыши и, не отпуская ее, переместить объект в необходимое место страницы.

После установления флажка **Show Bounding Box** (Показать маркеры) на панели параметров инструмента **Move Tool** (Перемещение) вокруг фигуры появляется рамка выделения объекта. С помощью угловых маркеров рамки выделения можно изменять размеры фигуры (при этом указатель мыши должен иметь вид двунаправленной стрелки).

При выполнении действий по изменению размеров графического объекта станут доступными поля панели параметров, позволяющие устанавливать точные размеры фигуры.

3.4.10. Инструменты рисования

Инструменты рисования путем воздействия на пиксели изображения позволяют изменять их цвет, воплощая определенные художественные замыслы. Они вполне пригодны, чтобы нарисовать несложное изображение или скорректировать существующее.

Рисование производится с помощью инструментов **Brush Tool** (Кисть) и **Pencil Tool** (Карандаш), составляющих одну группу инструментов.

Панель параметров инструмента **Brush Tool** (Кисть) содержит:

- список **Brush Tool** (Кисть), в котором можно выбрать кисть нужного размера и формы;
- список режимов наложения пикселей **Mode** (Режим), в котором нужно выбрать значение **Normal** (Обычный). Это наиболее распространенный режим, при котором цвет пикселя заменяется на новый без учета его предыдущего цвета;
- поле **Opacity** (Непрозрачность), позволяющее задать степень прозрачности создаваемого изображения (100 % — полная непрозрачность);

- поле **Flow** (Струя), позволяющее регулировать плотность однократного мазка (от 1 до 100 %). При небольших значениях требуется многократное воздействие на закрашиваемые пиксели, чтобы действие кисти стало заметным. Такую возможность можно использовать для создания легких теней, румянца и т. д.;
- кнопка **Set to enable airbrush capabilities** (Установка режима распылителя) переводит инструмент в режим распылителя. Действие инструмента в этом режиме будет заметно при значениях поля **Flow** меньше 100 %. В отличие от обычного режима, плотность мазка увеличивается не количеством щелчков мышью, а временем удержания инструмента над закрашиваемым объектом.

Панель параметров инструмента **Pencil Tool** (Карандаш) отличается наличием флажка **Auto Erase** (Автоматическое стирание). Установив флажок, цвет пикселей, окрашенных основным цветом, можно заменять фоновым цветом и наоборот. В этом случае инструмент используется как ластик. При снятом флажке инструмент **Pencil** (Карандаш) аналогичен инструменту **Brush** (Кисть), но с меньшими возможностями.

3.4.11. Клонирование изображений

Клонирование — это создание копии фрагмента изображения (клона) как в пределах одного изображения, так и из одного в другое. В отличие от обычного копирования, пользователь принимает решение о размерах и форме копии непосредственно в процессе построения. Клонирование осуществляется инструментом **Clone Stamp Tool** (Штамп клонирования), размещенном в группе из двух инструментов.

Инструмент **Clone Stamp** (Штамп клонирования) — это кисть, которая используется только для клонирования. Панель параметров данного инструмента полностью аналогична панели параметров **Brush** (Кисть), но дополнительно имеет два флажка. Флажок **Aligned** (Выравнивание) позволяет выбрать один из двух

режимов работы инструмента. Если флажок установлен, то процесс клонирования можно прерывать, отпустив клавишу мыши. После ее повторного нажатия продолжится построение той же копии. Если флажок снят, то в случае повторного нажатия клавиши мыши начнется построение новой копии. При установленном флажке **Use All Layers** (Использовать все слои) в клонировании участвуют все слои изображения.

Инструмент работает следующим образом. Подвести указатель мыши к точке изображения, вокруг которой располагается копируемая область, и, удерживая клавишу <Alt> (указатель должен иметь вид мишени), щелкнуть мышью. Переместить указатель в то место изображения, где предполагается разместить клон (возможно на другом рисунке, предварительно сделав его активным), и, нажав кнопку мыши, перемещать ее, наблюдая процесс появления клона. Завершив построение, отпустить клавишу мыши.

Клонирование может быть использовано для устранения на изображении дефектов большого размера, удаления нежелательных объектов. Для этого используются похожие по структуре и цвету, фрагменты того же изображения, которые клонируются на удаляемые объекты. На рис. 3.15, *a* и *b* показано изображение до и после клонирования. Клонирование прибрежной растительности позволило закрыть фрагмент рисунка, изображающего речку.

В некоторых случаях можно помещать на изображение объекты с другого рисунка.

3.4.12. Выделение фрагментов изображения

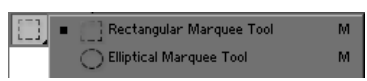
Отдельные объекты, располагающиеся на изображении (например предметы, фигуры людей, животных и т. д.), являются таковыми только для человека, смотрящего на изображение. Что касается программы Photoshop, то для нее все точки на изображении имеют одинаковую ценность и отнести их к тому или иному объекту без помощи человека программа не может. Объединить точки изображения в один объект можно с помощью операции выделения. Программа Photoshop располагает несколькими инструментами выделения объектов.

a)

b)

Рис. 3.15. Результат клонирования фрагментов изображения:
a — изображение до клонирования,
b — изображение после клонирования прибрежной растительности

В группе инструментов **Marquee** (Область) располагаются инструменты **Rectangular Marquee Tool** (Прямоугольна область) и **Elliptical Marquee Tool** (Эллиптическая область).

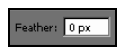


Для выделения фрагмента изображения необходимо выбрать соответствующий инструмент и при нажатой левой клавиши мыши растянуть пунктирный контур вокруг нужной области. До того как клавиша будет отпущена, можно изменять размеры об-

ласти выделения, а, удерживая еще и клавишу пробел, можно менять ее положение. Если во время построения удерживать клавишу <Shift>, то область выделения будет правильной формы (квадрат или окружность), а при нажатой клавише <Alt> — центр области выделения будет совпадать с точкой начала построения.

Рассмотрим содержимое панели параметров инструментов **Marquee** (Область). В левой части панели параметров располагаются четыре кнопки —  , позволяющие выбрать режим взаимодействия создаваемой выделенной области с областью выделения, созданной ранее.

- **New selection** (Новое выделение). В этом режиме создание новой выделенной области ведет к снятию предыдущего выделения.
- **Add to selection** (Добавить к выделенному). Создаваемая область выделения добавляется к уже имеющейся выделенной области. При наличии выделенной области переключиться в режим **Add to selection** (Добавить к выделенному) можно, нажав клавишу <Shift>.
- **Subtract from selection** (Вычесть из выделенного). Создаваемая область выделения вычитается из уже имеющейся области. При наличии выделенной области включается при нажатии клавиши <Alt>;
- **Intersect with selection** (Пересечение с выделенным). Результатом будет область выделения, принадлежащая как области созданной ранее, так и вновь создаваемой области.

Далее располагается поле 

— **Feather** (Растушевка).

Если величина, введенная в поле **Feather** (Растушевка) больше нуля, то граница выделенной области будет размыта в обе стороны от пунктирной линии на заданное число пикселей. В поле можно вводить значения от 1 до 250 пикселей. Однако следует соизмерять вводимую величину с размером выделяемой области и помнить, что поле **Feather** (растушевка) должно быть заполнено до начала выделения. Если этого не было сделано, то установить размытие на уже созданную выделенную область можно командой меню **Select | Feather** (Выделение | Растушевка), задав в диалоговом окне **Feather Selection** (Растушевка выделенного) радиус раз-

мытия. При сохранении изображения с растушевкой на прозрачном фоне плавность перехода будет утрачена. Сохранить растушевку можно, помещая фрагмент на любой фон, кроме прозрачного, или размещая изображение с прозрачным фоном и растушевкой на отдельном слое на фоне другого изображения *(см. раздел 3.4.13)*. Возможна также имитация плавного перехода, о которой говорилось в *разделе 3.4.3*.

Установка флажка **Anti-aliased** (Сглаживание) создает плавный переход от пикселей в выделенной области к соседним пикселям.

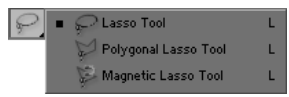
Список **Style** (Стиль) позволяет выбрать один из трех способов построения выделенной области:

- **Normal** (Обычный) — любое соотношение длины и ширины выделяемой области;
- **Fixed Aspect Ratio** (Фиксированное соотношение) — задается фиксированное соотношение между длиной и шириной;
- **Fixed Size** (Фиксированный размер) — задаются точные размеры длины и ширины.

При выборе двух последних вариантов в поля **Width** (Ширина) и **Height** (Высота) вводятся соответствующие значения.

Снятие выделения осуществляется сочетанием клавиш <Ctrl>+<D> или однократным щелчком левой кнопки мыши в любом месте документа.

Для выделения областей сложной формы предназначены несколько инструментов группы **Lasso** (Лассо).



Инструмент **Lasso Tool** (Лассо) позволяет путем перемещения указателя мыши (при нажатой левой кнопке) создать контур выделенной области произвольной формы. При нажатой клавише <Alt>, используя однократные щелчки левой кнопки мыши, также можно построить контур произвольной формы, но при этом после каждого щелчка будет создаваться прямолинейный отрезок контура.

Инструмент **Polygonal Lasso** (Многоугольное лассо) позволяет создавать контур выделения, состоящий из прямолинейных отрезков, для чего необходимо отмечать щелчком начало и конец каждого отрезка. Для построения отрезков произвольной формы нужно удерживать не только кнопку мыши, но и клавишу <Alt>. Построение завершается после замыкания контура или двукратного щелчка мышью.

Панель параметров этих двух инструментов содержит уже известные нам элементы.

Инструмент **Magnetic Lasso Tool** (Магнитное лассо) целесообразно использовать тогда, когда выделяемая область имеет достаточно контрастную границу, которая отделяет ее от остального изображения. Выбрав инструмент, нужно щелкнуть в любой точке контура и перемещать указатель мыши вдоль его линии, по возможности, точнее повторяя конфигурацию (клавишу мыши держать не обязательно). Движение будет сопровождаться появлением узловых точек, между которыми располагаются отрезки контура выделения. В случае отклонения от желаемой линии с помощью клавиши <Backspace> можно удалить одну за другой узловые точки, возвращая указатель мыши обратно, и повторить построение. Щелчком мыши можно принудительно расставлять узловые точки, заставляя контур выделения перемещаться в нужном направлении. Построение завершается однократным щелчком после замыкания контура. При неоконченном построении двойной щелчок левой кнопки мыши приводит к соединению конечной точки с началом контура выделения прямолинейным отрезком. Отказаться от начатого построения можно нажатием клавиши <Esc>.

На панели параметров инструмента **Magnetic Lasso** (Магнитное лассо), кроме уже известных элементов, находятся еще три поля:

- **Width** (Ширина) — ширина коридора, в котором осуществляется поиск контура (от 1 до 40 px). Чем меньше заданное значение, тем точнее нужно перемещать курсор вдоль контура, чтобы не потерять его;
- **Frequency** (Частота) — частота следования узловых точек (от 0 до 100). Чем больше установленное значение, тем чаще будут располагаться узловые точки;

- **Edge Contrast** (Контраст контура) — чувствительность инструмента (от 1 до 100 %). Чем меньше заданное значение, тем более чувствительным будет инструмент, т. е. появится возможность обнаруживать контуры с меньшей контрастностью.

Инструмент выделения **Magic Wand** (Волшебная палочка)  — предназначен для выделения достаточно однородных по цвету областей.

При использовании данного инструмента на панели параметров в поле **Tolerance** (Допуск) вводится числовое значение (от 0 до 255), задающее степень близости по цвету пикселей, которые попадут в область выделения. Чем меньше это значение, тем ближе по цвету должны быть пиксели к тому, по которому щелкнули инструментом **Magic Wand** (Волшебная палочка), чтобы попасть в область выделения. Установив значение 255 одним щелчком можно выделить все изображение. Установка флажка **Contiguous** (Смежный) означает, что в область выделения попадут только смежные пиксели. Флажок **Use All Layers** (Использовать все слои) устанавливается тогда, когда необходимо поместить в выделенную область пиксели, принадлежащие различным видимым слоям изображения.

Меню **Select** (Выделение), располагает дополнительными возможностями по созданию и изменению выделенных областей. Познакомимся с некоторыми из них.

- **All** — выделить все изображение.
- **Deselect** — снять выделение. Для этой цели удобнее использовать сочетание клавиш <Ctrl>+<D>.
- **Reselect** — вернуть выделение.
- **Inverse** — инвертирование выделенной области. После этой операции выделенной будет область изображения, которая до этого располагалась за пределами выделенной области.
- **Feather** (Растушевка) — открывает диалоговое окно **Feather Selection** (Растушевка выделенного) для ввода величины размытия границы выделенной области в обе стороны от пунктирной линии.
- **Modify** (Модификация) — располагает целой группой команд:
  - **Border** (Рамка). Вокруг выделенной области создается рамка выделения, толщина которой (от 1 до 64 пикселей) за-

дается в диалоговом окне **Border Selection** (Рамка выделения), после чего выделенной областью будет именно рамка, а не прежняя область;

- **Smooth** (Сглаживать). Несколько сглаживает границу выделенной области, добавляя или исключая отдельные пиксели. Радиус анализируемой области вокруг граничных пикселей (от 1 до 16) задается в диалоговом окне **Smooth Selection** (Сглаживание выделенного);
- **Expand** (Расширять). Расширяет выделенную область на величину (от 1 до 16 пикселей), указанную в диалоговом окне **Expand Selection** (Расширение выделенного);
- **Contract** (Сжимать). Сжимает выделенную область.

❐ **Transform Selection** (Изменение выделенной области) — создает маркеры вокруг выделенной области, позволяющие изменять ее размеры и вращать. На панели параметров можно задавать точные значения преобразований. Выполнение завершается нажатием клавиши **Enter** или двукратным щелчком внутри выделенной области.

Как видим, в программе Photoshop вопросам выделения отдельных областей изображения уделяется большое внимание. Это связано с тем, что операция выделения обычно предшествует многим другим операциям. В частности, выделенный фрагмент можно вырезать из изображения или копировать, вставив затем в новое или уже существующее изображение, можно реализовать различные варианты его заливки, подвергнуть воздействию фильтра и т. д. С некоторыми из этих возможностей мы сейчас и познакомимся.

3.4.13. Копирование и перемещение выделенного фрагмента

Операции копирование и перемещение производятся также, как и в других приложениях, с помощью команд меню: **Copy** (Копировать) и **Cut** (Вырезать). После выполнения этих команд, выделенный фрагмент помещается в буфер обмена и может быть вставлен как в то же самое, так и в другое изображение командой **Edit | Past** (Правка | Вставить).

Для этих же целей можно использовать инструмент **Move Tool** (Перемещение). Установив указатель мыши над выделенной областью (он приобретет вид стрелки с ножницами) и нажав кнопку мыши, можно переместить фрагмент не только в пределах текущего изображения, но и в другое изображение, предварительно открытое в другом окне. Для осуществления копирования перед началом перемещения нужно нажать и удерживать клавишу <Alt> (указатель мыши должен иметь вид двух стрелок). В этом случае перемещается не сам фрагмент, а его копия. Рассмотрим несколько примеров копирования выделенных фрагментов.

Копирование в новое изображение

Рассмотрим порядок действий копирования изображения в новый документ Photoshop.

- Выберем изображение 150 × 225 пикселей (рис. 3.16, *a*). Создадим на изображении выделенную область овальной формы, предварительно установив на панели параметров инструмента **Elliptical Marquee** (Эллиптическая область) следующие значения: **Feather** (Растушевка) — 0 px, флажок **Anti-aliased** (Сглаживание) — установлен, **Style** (Стиль) — Fixed Size, **Width** — 100 px, **Height** — 150 px. Щелкнув по изображению, получим выделенную область. Поместив указатель внутри области, и нажав кнопку мыши, переместим выделенную область в нужное место (для перемещения можно использовать и клавиши управления курсором).
- Дадим команду меню **Edit | Copy** (Правка | Копировать).
- Командой меню **File | New** (Файл | Создать) создадим новое изображение. В диалоговом окне **New** (Создать) обратим внимание, что нам предлагаются размеры изображения, соответствующие размерам копируемой области. Выберем прозрачный фон изображения (**Transparent**) и нажмем **OK**.
- Используя команду меню **Edit | Paste** (Правка | Вставить), вставим копируемый фрагмент в новое изображение. Фрагмент точно впишется в рамку изображения. Результат этих действий показан на рис. 3.16, *b*.

- Сохраним изображение в формате GIF, используя команду меню **File | Save for Web** (Файл | Сохранить для Web).

a)

b)

c)

Рис. 3.16. Копирование выделенных областей в новое изображение:
a — исходное изображение с выделенной областью,
b — результат копирования выделенной области на новое изображение с прозрачным фоном без растушевки,
c — результат копирования с растушевкой 10 пикселей

Если требуется получить изображение с размытыми краями, то до создания выделенной области необходимо в поле **Feather** (Растушевка) задать величину размытия, например 10 пикселей. Обратим внимание, что предлагаемые в диалоговом окне **New** (Создать) размеры изображения будут несколько большими, чем в предыдущем случае (рис. 3.16, *c*).

Копирование в существующее изображение

Для создания оригинальных изображений с помощью операции копирования очень часто применяют фотомонтаж. Приведем пример фотомонтажа.

a)

b)

c)

Рис. 3.17. Копирование выделенных областей в существующее изображение:
a — существующее изображение,
b — изображение с первым фрагментом для копирования,
c — изображение со вторым копируемым фрагментом

- Откроем три изображения (рис. 3.17). На втором изображении, используя инструмент **Lasso** (Лассо), выделим фрагмент для копирования (рис. 3.17, *b*). Перед копированием целесообразно осуществить растушевку выделенной области примерно на 3 пикселя, что позволит избежать резкой границы между изображениями после копирования.
- Осуществим копирование выделенного фрагмента в первое изображение (рис. 3.17, *a*), используя инструмент **Move Tool** (Перемещение). Установим указатель мыши над фрагментом. Удерживая клавишу <Alt>, нажмем левую кнопку мыши и перетащим фрагмент в нужное место.
- Возможно, возникнет необходимость изменить размеры вставленного фрагмента, чтобы сделать его соизмеримым с другими объектами изображения. Для этого достаточно на панели параметров инструмента **Move Tool** (Перемещение) установить флажок **Show Bounding Box** (Показать маркеры), вокруг фрагмента появятся маркеры, позволяющие изменить его размеры. Изменение размера завершается нажатием клавиши <Enter>.
- На третьем изображении выделим следующий копируемый объект (рис. 3.17, *c*) и аналогичным образом поместим его в первое изображение (рис. 3.17, *a*). В случае необходимости изменим его размеры. Результат показан на рис. 3.18.

Рис. 3.18. Результат копирования двух фрагментов изображений

Обратим внимание на положение флажка **Auto Select Layer** (Автоматический выбор слоя) на панели параметров инструмента **Move Tool** (Перемещение). Если флажок установлен, то можно поочередно перемещать оба вставленных фрагмента. Если же флажок снят, то будет перемещаться только один из фрагментов. Это означает, что фрагменты оказались в разных слоях. Обратимся к палитре **Layers** (Слои) и заметим, что основное изображение находится в слое **Background** (Фон), первый фрагмент в слое **Layer 1**, второй — **Layer 2** (рис. 3.19).

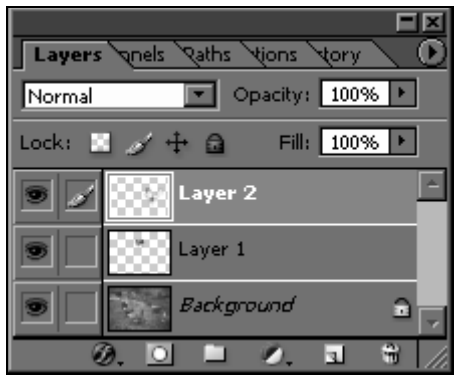


Рис. 3.19. Палитра **Layers** после копирования на изображение двух фрагментов

Чтобы заставить перемещаться другой фрагмент нужно щелкнуть мышью по имени слоя, на котором он находится. Слои играют исключительно важную роль, поэтому эта тема требует специального обсуждения.

3.4.14. Использование слоев

Нам уже известны трудности, связанные с выделением отдельных фрагментов изображений. И если такая работа проведена, то хотелось бы иметь возможность осуществлять различные манипуляции с фрагментом, не прибегая к его повторному выделению. Кроме того, если фрагмент уже занял определенное место на изображении, то повторное его перемещение не должно приводить к искажению фрагмента, находящегося под ним. Эти и многие другие задачи могут быть решены с использованием слоев.

Для получения практических навыков работы со слоями рекомендуется использовать многослойное изображение, подобное тому, которое было создано в предыдущем примере. Действия со слоями осуществляются с помощью палитры **Layers** (Слои).

Каждая строка в палитре **Layers** (Слои) содержит информацию о конкретном слое изображения. Левое поле в каждой строке позволяет управлять видимостью данного слоя. Наличие пиктограммы с изображением глаза говорит о видимости слоя, отсутствие пиктограммы — о его невидимости. Щелкнув мышью по полю, можно изменить видимость слоя.

Правое поле является индикатором активности слоя и служит для связывания слоев. Если в нем расположена пиктограмма с изображением кисти, то слой активен. Чтобы слой сделать активным, нужно щелкнуть мышкой по его имени. На активность слоя также указывает изменение цвета имени слоя и фона, на котором оно располагается. Для связывания слоев один из них нужно сделать активным и щелкнуть мышкой по соответствующему полю другого слоя, после чего в нем появится пиктограмма с изображением цепи. Таким образом, можно связать сразу несколько слоев. Повторным щелчком цепь удаляется, и слой выпадает из связки. Перемещая один из связанных слоев, мы будем наблюдать перемещение и остальных слоев из связки.

Далее в строке слоя располагается уменьшенная копия изображения слоя (миниатюра) и название слоя. Обращает на себя внимание, что один из слоев называется Background, тогда как остальные слои по умолчанию именуются Layer с указанием номера. Дело в том, что слой **Background** (Фон) по своим свойствам отличается от остальных слоев. Он всегда располагается ниже других слоев и не может иметь прозрачных фрагментов. Кроме того, его невозможно перемещать в окне рисунка. Чтобы снять с него все ограничения, его достаточно переименовать. Любое другое имя, присвоенное этому слою, делает его обычным.

Для переименования слоя **Background** (Фон) нужно дважды щелкнуть по его имени и открыть диалоговое окно **New Layer** (Создать слой). В строке **Name** (Имя) следует указать новое имя слоя. В списке **Color** (Цвет) выбрать цвет, которым будут окрашены поля этого слоя в палитре (**None** — цвет по умолчанию).

В списке **Mode** (Режим) выбирается режим взаимодействия пикселей слоя с пикселями нижележащих слоев (рекомендуется **Normal**). В поле **Opacity** (Непрозрачность) можно установить степень непрозрачности слоя (100 % — полная непрозрачность).

Возможно переименование и обычного слоя, для этого достаточно дважды щелкнуть по его имени и ввести новое.

Обычный слой можно преобразовать в слой **Background** (Фон). Для этого слой нужно выделить и воспользоваться командой меню **Layer | New | Background from Layer** (Слой | Создать | Фон из слоя).

Порядок следования слоев можно менять. Таким образом, можно менять положение находящихся в слоях объектов относительно друг друга, располагая одни поверх других или наоборот. Проще всего это делается перетаскиванием слоя. Для этого следует указатель мыши установить над именем слоя, нажать левую кнопку мыши и, не отпуская ее, перетащить слой вверх или вниз.

Перетащить слой можно и на другое изображение. После нажатия левой кнопки над именем слоя указатель мыши переместить над другим изображением и отпустить кнопку. В палитре **Layers** (Слои) другого изображения появится копия перемещаемого слоя. Это не единственный способ создания копии слоя в другом изображении. Сделав слой активным и включив инструмент **Move Tool** (Перемещение), можно нажать левую кнопку мыши непосредственно над копируемым изображением и после перемещения указателя на другое изображение отпустить кнопку. Для этой же цели можно использовать команду **Duplicate Layer** (Копия слоя) меню палитры **Layers** (Слои), которое открывается щелчком по круглой кнопке с изображением треугольной стрелки в правом верхнем углу палитры. Откроется одноименное диалоговое окно, в котором можно задать имя слоя. Раскрыв список **Document** (Документ) можно выбрать имя документа, для которого предназначена копия. В списке содержатся имена открытых файлов и команда **New** (Создать) для расположения копии слоя в новом документе.

Создать новый слой можно, воспользовавшись кнопкой **Create a new layer** (Создание нового слоя) в правой нижней части палитры или используя команду **New Layer** (Создать слой) меню палитры. В первом случае будет создан новый слой с прозрачным

фоном, во втором — откроется диалоговое окно **New Layer** (Создать слой), содержимое которого было описано выше.

Удалить слой позволяет кнопка **Delete Layer** (Удалить слой) (правая в нижней части палитры) или команда **Delete Layer** (Удалить слой) меню палитры.

В верхней части палитры **Layers** (Слои), кроме списка режимов взаимодействия пикселей слоев, в котором рекомендуется выбрать **Normal**, располагается поле **Opacity** (Непрозрачность). Изменяя значение данного поля (вводя новые значения или перемещая ползунок после щелчка по треугольной стрелке), можно регулировать степень прозрачности изображения выделенного слоя (100 % — полная непрозрачность). Там же располагаются 4 кнопки, объединенные общим названием **Lock** (Запрет). Установив соответствующий флажок, можно запретить (слева направо): изменение прозрачных областей, изменение рисунка рисующими инструментами, перемещение слоя, все перечисленные изменения.

3.4.15. Эффекты слоев

К слоям могут быть применены различные эффекты, которые позволяют видоизменить изображение. Эффекты применяются ко всем объектам, находящимся на слое. К каждому слою, кроме Background (Фон), можно применить несколько эффектов. Чтобы использовать эффект, нужно выделить слой в палитре **Layers** (Слои) и нажать кнопку **Add a layer style** (Добавить эффект слоя) в нижней части палитры (Крайняя левая кнопка). Щелчок по кнопке раскрывает меню, выбор любой команды которого приведет к открытию диалогового окна **Layer Style** (Эффект слоя). В столбце **Styles** (Эффекты) диалогового окна перечислены следующие эффекты слоев:

- **Drop Shadow** (Внешняя тень), **Inner Shadow** (Внутренняя тень);
- **Outer Glow** (Внешнее свечение), **Inner Glow** (Внутреннее свечение);
- **Bevel and Emboss** (Фаска и рельеф). Дополнительные разделы:
  - **Contour** (Профиль) — способ создания рельефа;
  - **Texture** (Текстура) — декоративная заливка;

- **Satin** (Атласное покрытие);
- **Color Overlay** (Однородная заливка);
- **Gradient Overlay** (Градиентная заливка);
- **Pattern Overlay** (Узорная заливка);
- **Stroke** (Обводка).

Для выбора соответствующего эффекта нужно установить флажок рядом с его названием, а для настройки параметров — выделить название эффекта. Более детально с некоторыми эффектами познакомимся в процессе рассмотрения примеров.

Примеры использования эффектов слоев

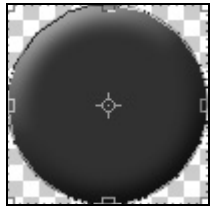
Создание кнопки

При разработке HTML-документов часто возникает необходимость в использовании графических объектов в качестве кнопок. Рассмотрим последовательность операций по созданию изображений простых кнопок.

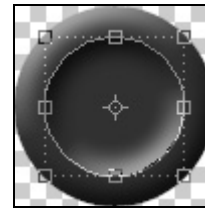
- Воспользовавшись командой меню **File | New** (Файл | Создать), откроем диалоговое окно **New** (Создать) и установим размеры окна 100 × 100 пикселей. В поле **Resolution** (Разрешение) можно оставить значение, предлагаемое по умолчанию, так как для экранного воспроизведения изображения разрешение роли не играет. В списке **Color Mode** (Режим) выбирается цветовая модель изображения RGB Color. В соседнем списке определяется глубина цвета изображения. Ее значение рекомендуется установить 8 bit. Из списка **Background Contents** (Содержимое фона) нужно выбрать **Transparent** (Прозрачный фон).
- В блоке выбора цвета на панели инструментов в качестве основного цвета (**Foreground**) выбираем желаемый цвет кнопки (например синий), а в качестве фонового цвета (**Background**) — белый.
- Из группы векторных инструментов выбираем инструмент **Ellipse Tool** (Эллипс) и рисуем круг диаметром 100 пикселей. Так как с помощью мыши это сделать затруднительно, то, построив произвольный круг, нужно выбрать инструмент **Move Tool** (Перемещение), установить флажок **Show Bounding**

Box (Показать маркеры) на панели параметров и щелкнуть мышью по одному из появившихся маркеров. На панели параметров появятся поля **X** и **Y**, в которые нужно ввести по 50 пикселей (координаты центра круга) и поля **W** и **H**, в которые нужно ввести по 100 пикселей (ширина и высота фигуры). Если в полях установлены другие единицы измерения, то перед установкой размеров нужно перейти к пикселям, щелкнув правой кнопкой по соответствующему полю. Завершим изменение размеров нажатием клавиши **Enter**.

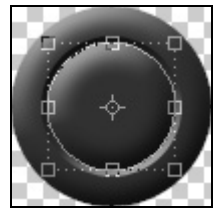
- В палитре **Layers** (Слои) нажмем кнопку **Add a layer style** (Добавить эффект слоя) в нижней части палитры и в списке эффектов выберем **Bevel and Emboss** (Фаска и рельеф). В диалоговом окне **Layer Style** (Эффект слоя) установим следующие значения параметров для этого эффекта: **Style** (Стиль) — **Inner Bevel** (Внутренняя фаска), **Technique** (Техника) — **Smooth** (Сглаженная), **Depth** (Глубина фаски) — 100 % (от размера **Size**), **Direction** (Направление) — **Up** (Выпуклость), **Size** (Размер теневой области) — 10 px, **Soften** (Смягчение границ) — 10 px. Остальные параметры оставим без изменений. Результат показан на рис. 3.20, *a*. Обратим внимание, что в палитре **Layers** (Слои) появилась запись с названием примененного эффекта. При необходимости изменить установленные параметры эффекта достаточно сделать двойной щелчок по этой записи, что приведет к открытию диалогового окна **Layer Style** (Эффект слоя).
- Создадим копию слоя, выбрав команду **Duplicate Layer** (Копия слоя) в меню палитры **Layers** (Слои).
- При включенном инструменте **Move Tool** (Перемещение) вновь щелкнем по одному из маркеров и в поля **W** и **H** панели параметров введем новое значение — 70 px. Завершим изменение размеров нажатием клавиши <Enter>. Двойным щелчком по названию эффекта в палитре **Layers** (Слои) откроем диалоговое окно **Layer Style** (Эффект слоя) и переключатель **Direction** (Направление) переведем в положение **Down** (Вогнутость), оставив остальные параметры без изменения. Получим законченное изображение нажатой кнопки (рис. 3.20, *b*), которое можно сохранить в формате GIF (команда **Save for Web** (Сохранить для Web) меню **File** (Файл)).



a)



b)



c)

Рис. 3.20. Создание изображения кнопки с использованием эффектов слоев:
a — применение эффекта выпуклости,
b — применение эффекта вогнутости к фрагменту изображения (нажатая кнопка),
c — применение эффектов выпуклости и вогнутости к разным фрагментам (не нажатая кнопка)

- Создадим копию последнего слоя (Shape 1 copy), выбрав команду **Duplicate Layer** (Копия слоя) в меню палитры **Layers** (Слои).
- Включим инструмент **Move Tool** (Перемещение), щелкнем по одному из маркеров и в поля **W** и **H** панели параметров введем 64 px. Завершим изменение размеров нажатием клавиши <Enter>.
- Двойным щелчком по названию эффекта в слое Shape 2 copy палитры **Layers** (Слои) откроем диалоговое окно **Layer Style** (Эффект слоя) и переключатель **Direction** (Направление) переведем в положение **Up** (Выпуклость), оставив остальные параметры без изменения. В результате получим изображение неактивной кнопки, которое также можно сохранить в формате GIF (рис. 3.20, *c*).

В некоторых случаях могут понадобиться кнопки разных цветов, например, когда при наведении на кнопку курсора она изменяет свой цвет. Такая задача легко решается с помощью программи-

рования на JavaScript после изучения *главы 8*. Чтобы быстро изменить цвет рисунка нужно воспользоваться командой меню **Image | Adjustments | Hue/Saturation** (Изображение | Настройки | Оттенок/Насыщенность). В диалоговом окне **Hue/Saturation** (Оттенок/Насыщенность) необходимо в списке **Edit** (Правка) установить режим **Master** (Мастер) и перемещать ползунок **Hue** (Оттенок) влево или вправо, наблюдая изменение цвета кнопки при установленном флажке **Preview** (Предварительный просмотр). Ползунок **Saturation** (Насыщенность) позволяет менять насыщенность оттенка, а **Lightness** (Яркость) — яркость.

Создание пунктов и разделов меню

Рассмотрим пример создания пунктов или разделов меню.

- В блоке выбора цвета панели инструментов установим основной цвет (**Foreground**) — темно-синий, фоновый цвет (**Background**) — белый.
- Создадим новое изображение размером 70 × 15 px, воспользовавшись командой меню **File | New** (Файл | Создать). В диалоговом окне **New** (Создать) выберем: цветовую модель изображения (**Color Mode**) — RGB Color, глубину цвета изображения — 8 bit, содержимое фона (**Background Contents**) — Background Color (Фоновый цвет).
- Чтобы стало возможным применение эффектов, переименуем слой **Background** (Фон), дважды щелкнув по имени слоя в палитре **Layers** (Слои). Согласимся с предлагаемым именем **Layer 0**.
- Применим к слою эффект **Bevel and Emboss** (Фаска и Рельеф), щелкнув по кнопке **Add a layer style** (Добавить эффект слоя) в нижней части палитры **Layers** (Слои) и выбрав в открывшемся меню название эффекта. В диалоговом окне **Layer Style** (Эффект слоя) установим следующие значения параметров для этого эффекта: **Style** (Стиль) — **Inner Bevel** (Внутренняя фаска), **Technique** (Техника) — **Smooth** (Сглаженная), **Depth** (Глубина фаски) — 100 %, **Direction** (Направление) — **Up** (Выпуклость), **Size** (Размер теневой области) — 5 px, **Soften** (Смягчение границ) — 5 px.
- Выберем инструмент **Horizontal Type** (Горизонтальный текст), на панели параметров инструмента установим гарнитуру

шрифта Times New Roman, начертание — bold (полужирный), размер — 12 pt, метод сглаживания шрифта — **None** (Сглаживание отсутствует). Поместим текст пункта меню, например Пункт 1, в площадь изображения.

- Осуществим выравнивание текста относительно краев изображения инструментом **Move Tool** (Перемещение) при установленном флажке **Show Bounding Box** (Показать маркеры). Для перемещения текста удобно использовать клавиши управления курсором.
- Сохраним рисунок, не содержащий прозрачного фона, в формате JPEG.

Готовый рисунок выглядит следующим образом

Проделаем аналогичную работу, построив еще три рисунка для других пунктов меню. Кроме того, необходимо создать еще один комплект рисунков с аналогичными надписями, но имеющими другой цвет шрифта и фона. В случае использования программы-сценария данный комплект позволит изменять цвет пункта меню при наведении на него указателя мыши. Такая программа будет описана в *главе 8*. Для создания нового комплекта рисунков можно поменять местами основной (**Foreground**) и фоновый (**Background**) цвета в блоке установки цвета. После этого необходимо повторить все действия по созданию комплекта рисунков для пунктов меню. Теперь белый шрифт будет располагаться на темно-синем фоне.

При создании разделов меню проделаем следующие действия.

- Вновь установим в блоке установки цвета панели инструментов основной цвет (**Foreground**) — темно-синий, фоновый цвет (**Background**) — белый.
- Создадим новое изображение размером 85 × 20 px, воспользовавшись командой меню **File | New** (Файл | Создать). Остальные установки диалогового окна **New** (Создать) сделаем такими же, как при создании пунктов меню.

- Используя инструмент **Horizontal Type** (Горизонтальный текст), введем название раздела, например РАЗДЕЛ 1, предварительно установив на панели параметров гарнитуру шрифта Times New Roman, начертание — **bold** (Полужирный), размер — 12 pt, метод сглаживания шрифта — **None** (Сглаживание отсутствует). Открыв палитру **Character** (Шрифт) и выделив введенный текст, в списке **Set the tracking for the selected characters** (Установка расстояния между выделенными символами) установим расстояние между символами 75.
- Выбрав инструмент **Move Tool** (Перемещение) и установив флажок **Show Bounding Box** (Показать маркеры), осуществим выравнивание текста относительно краев изображения.
- Применим к текстовому слою эффект **Drop Shadow** (Внешняя тень). В диалоговом окне **Layer Style** (Эффект слоя) установим следующие значения параметров для этого эффекта: **Opacity** (Непрозрачность) — 100 %, **Distance** (Смещение) — 5 px, **Spread** (Размытие) — 3 %, **Size** (Размер) — 5 px. Остальные параметры оставим без изменения.
- Сохраним рисунок в формате JPEG.

В результате получим следующее изображение

Создадим еще 2 рисунка с названиями других разделов и сохраним вместе с рисунками пунктов. Все они будут использованы в *главе 8* для создания раскрывающихся меню.

Создание логотипа

Рассмотрим пример создания логотипов.

- Используя команду меню **File | New** (Файл | Создать) создадим новое изображение размером 150 × 100 px. В диалоговом окне **New** (Создать) выберем: цветовую модель изображения **Color Mode** — RGB Color, глубину цвета изображения — 8 bit, содержимое фона (**Background Contents**) — Transparent (Прозрачный фон).
- Выберем инструмент **Horizontal Type** (Горизонтальный текст) и на панели параметров установим гарнитуру шрифта Mono-

type Corsiva, размером 72 pt, метод сглаживания — Crisp (Четкое).

- Установим курсор в площади изображения, введем букву "А" и завершим ввод. Применим к букве эффект **Bevel and Emboss** (Фаска и рельеф).
- Введем букву "П", сохранив те же параметры шрифта. Буква разместится на новом слое, к которому применим эффект **Bevel and Emboss** (Фаска и рельеф).
- Инструментом **Move Tool** (Перемещение) переместим буквы относительно друг друга, чтобы получился следующий результат:

- Сохраним рисунок в формате GIF.

3.4.16. Объединение слоев

Объединение слоев может применяться к различным слоям, но мы рассмотрим только объединение изображения с текстом. В результате объединения изображение сохранится только в пределах символов текста. Текстовый слой необходимо располагать под тем графическим слоем, с которым предстоит объединение. Так как слой Background (Фон) всегда располагается ниже других слоев, то осуществлять объединение можно только с копией слоя Background (Фон) или другим обычным слоем.

Пример объединения слоев

Рассмотрим технику объединения слоев.

- Откроем изображение размером 300 × 200 px, используя команду **Open** (Открыть) меню **File** (Файл).
- Выберем инструмент **Horizontal Type** (Горизонтальный текст) и установим параметры: гарнитура шрифта Monotype Corsiva, размер шрифта 72 pt, сглаживание Crisp (Четкое), цвет шрифта черный.

- На рисунке поместим букву "П".
- Выбрав инструмент **Move Tool** (Перемещение) и установив на панели параметров флажок **Show Bounding Box** (Показать маркеры) увеличим размер буквы до размера изображения.
- Применим к тесту эффект **Bevel and Emboss** (Фаска и рельеф).
- Создадим копию слоя Background (Фон) и в палитре **Layers** (Слои) и мышью поместим ее над текстовым слоем.
- Установим указатель мыши между текстовым слоем и слоем Background copy и нажмем клавишу <Alt>, при этом указатель изменит свой вид. После щелчка мышью произойдет объединение слоев. Внешний вид палитры **Layers** (Слои) после объединения показан на рис. 3.21.

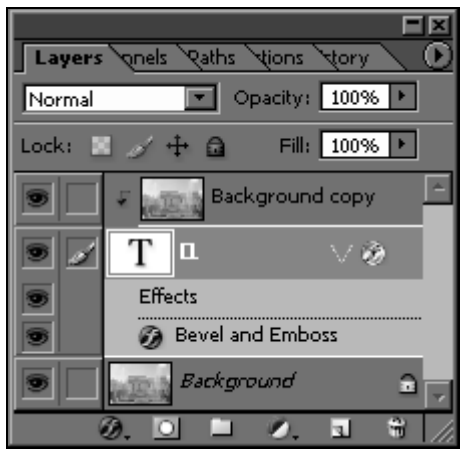


Рис. 3.21. Палитра **Layers** после объединения слоев

На рис. 3.22 представлен результат объединения на фоне слоя Background (Фон). Возможно, что для практических целей такой вариант представляет даже больший интерес, чем только результат объединения, который можно увидеть, сделав слой Background (Фон) невидимым.

Таким образом, объединение текстового слоя с изображением можно широко использовать при создании кнопок, разделов меню и логотипов для HTML-документов.

Рис. 3.22. Результат объединения слоев на фоне слоя Background

3.4.17. Заливка изображений

Заливка может использоваться при создании простых рисунков, например: кнопок, разделов меню, GIF-анимации. Возможно применение заливки и при обработке фотоизображений.

Для заливки предназначены два инструмента, находящиеся на панели инструментов: **Paint Bucket Tool** (Заливка) — однотонная или узорная заливка и **Gradient Tool** (Градиент) — градиентная заливка.

Панель параметров инструмента **Paint Bucket Tool** (Заливка) содержит следующие элементы. В списке **Fill** (Заливка) выбирается тип заливки: **Foreground** — заливка основным цветом или **Pattern** — узорная заливка. При выборе второго варианта становится доступным список образцов узорных заливок **Pattern**. В раскрывающемся списке режимов наложения пикселей **Mode** (Режим) нужно выбрать значение Normal (Обычный). В поле **Opacity** (Непрозрачность) задается степень непрозрачности заливки (100 % — полная непрозрачность). Что касается поля **Tolerance** (Допуск), а также флажков **Contiguous** (Смежный) и **Use All Layers** (Использовать все слои), то информация о них, приведенная для инструмента **Magic Wand** (Волшебная палочка), справедлива и в этом случае, если слово "выделение" заменить словом "заливка". Назначение флажка **Anti-aliased** (Сглаживание) уже объяснялось

применительно к выделению. Установка данного флажка при заливке способствует созданию плавного перехода от пикселей в области заливки к соседним пикселям.

Градиентная заливка предусматривает плавный переход от одного цвета к другому. Первым элементом панели параметров инструмента **Gradient Tool** (Градиент) является палитра градиентных переходов. Цвета некоторых переходов зависят от установленных на панели инструментов основного и фонового цветов. Далее располагаются 5 кнопок выбора типа градиентной заливки: **Linear Gradient** — линейная, **Radial Gradient** — радиальная, **Angle Gradient** — коническая, **Reflected Gradient** — зеркальная, **Diamond Gradient** — ромбовидная. Справа от кнопок располагается раскрывающийся список **Mode** (Режим), из которого нужно выбрать значение Normal. Установка флажка **Reverse** (Перемена) позволяет поменять местами начальный и конечный цвет градиентного перехода. Не все возможности градиентных переходов будут нами использоваться, поэтому устанавливать флажки **Dither** (Степень имитации) и **Transparency** (Прозрачность) и объяснять их назначение не будем.

Создание рамки вокруг рисунка

Рассмотрим пример использования заливки.

- Откроем рисунок, вокруг которого предстоит создать рамку, используя команду **Open** (Открыть) меню **File** (Файл).
- Выделим весь рисунок, используя команду меню **Select | All** (Выделение | Все).
- Выполним команду меню **Edit | Copy** (Правка | Копировать), поместив копию рисунка в буфер обмена.
- Создадим новое изображение (команда меню **File | New** (Файл | Создать)). В диалоговом окне **New** (Создать) увеличим предлагаемую ширину и высоту рисунка на 20 px. Положение переключателя **Contents** (Содержание) значения не имеет.
- Воспользуемся инструментом **Paint Bucket** (Заливка). На панели параметров выберем тип заливки **Pattern** (Узорная) и желаемый вид узора. В поле **Opacity** (Непрозрачность) установим 100 %. Так как фон нового рисунка абсолютно одно-

роден, то значение поля **Tolerance** (Допуск) может быть любым от 0 до 255. Состояние флажков в данном случае также не имеет значения. Щелкнув по полю нового изображения, осуществим его заливку.

- Используя команду меню **Edit | Paste** (Правка | Вставить), вставим рисунок из буфера обмена в поле нового рисунка (рис. 3.23).
- Сохраним рисунок.

Рис. 3.23. Создание рамки вокруг рисунка с использованием заливки

Таким образом, мы создали вокруг рисунка внешнюю рамку толщиной 10 px. Если требуется создать внутреннюю рамку, т.е. использовать для этого пиксели самого рисунка, то тогда можно поступить следующим образом:

- выделить весь рисунок;
- создать рамку выделения по периметру рисунка, для чего нужно воспользоваться командой меню **Select | Modify | Border** (Выделение | Модификация | Рамка), введя в диалоговом окне **Border Selection** (Рамка выделения) ширину рамки, например 10 px;
- выбрав инструмент **Paint Bucket** (Заливка) и выполнив необходимые установки на панели параметров, поместить указатель мыши над рамкой выделения и произвести щелчок;
- сохранить рисунок.

3.4.18. Создание прозрачного фона изображения

Форматы графических файлов предусматривают сохранение изображений прямоугольной формы. Однако желание разнообразить свою web-страницу ставит задачу размещения рисунков разнообразных форм. Этого можно достичь, сделав прозрачным фон рисунка за пределами контура объекта, изображение которого необходимо разместить на странице. Пример показан на рис. 3.24.

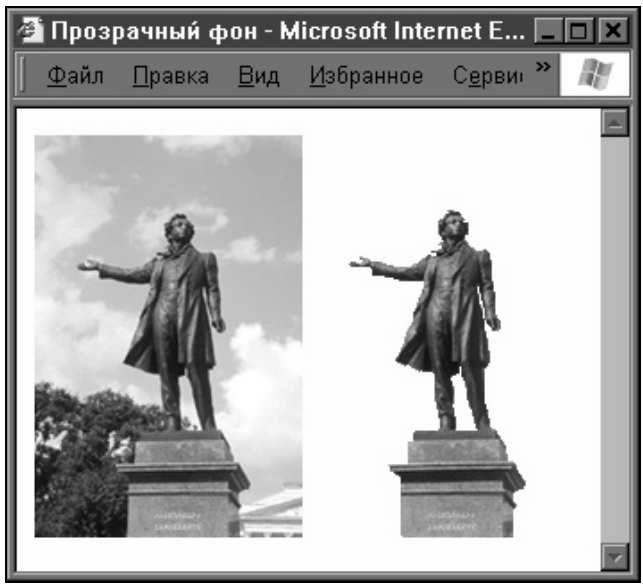
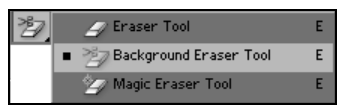


Рис. 3.24. Создание прозрачного фона изображения за пределами контура объекта

Инструменты, позволяющие удалять пиксели изображения, входят в группу **Eraser** (Ластик).



Инструмент **Background Eraser Tool** (Ластик фона) — основной инструмент для удаления фона. Изменяя установки панели параметров, можно сделать удобным применение инструмента в различных условиях. Инструмент **Background Eraser** (Ластик фона) фактически является кистью, которой и удаляется фон, поэтому, прежде всего, следует задать ее параметры. Для этого нужно щелкнуть по кнопке **Brush** (Кисть) на панели параметров, после чего откроется диалоговое окно (рис. 3.25).

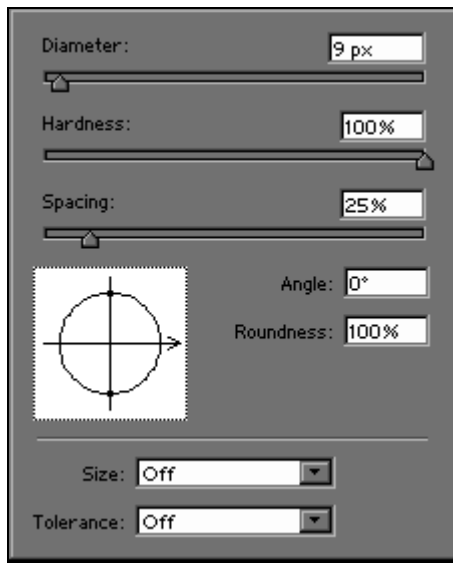


Рис. 3.25. Диалоговое окно настройки параметров кисти инструмента **Background Eraser**

В поле **Diameter** (Диаметр) устанавливается диаметр кисти. В поле **Hardness** (Жесткость) — степень размытости краев кисти (100 % — максимальная резкость краев, 0 % — максимальная размытость краев). Поле **Spacing** (Промежуток) предназначено для задания расстояния между соседними штрихами при перемещении кисти в процентах от ее диаметра. Поля **Angle** (Угол) и **Roundness** (Форма) позволяют сделать кисть овальной и изменить угол наклона овала. Результаты можно наблюдать на образце слева от полей. То же самое можно проделать непосредствен-

но на образце с помощью мыши, перемещая одну из жирных точек на окружности и стрелку.

Принцип работы инструмента заключается в том, что при нажатии клавиши мыши удаляются все пиксели, попавшие в площадь кисти и имеющие такой же или близкий (в зависимости от допуска **Tolerance** на панели параметров) цвет с пикселем, оказавшимся под крестиком. Кроме того, если в списке **Limits** (Границы) выбран режим работы инструмента **Contiguous** (Смежные), то удаляются только смежные пиксели (рис. 3.26, *b*), а если режим **Discontiguous** (Несмежные) — удаляются все пиксели в площади кисти (рис. 3.26, *c*).

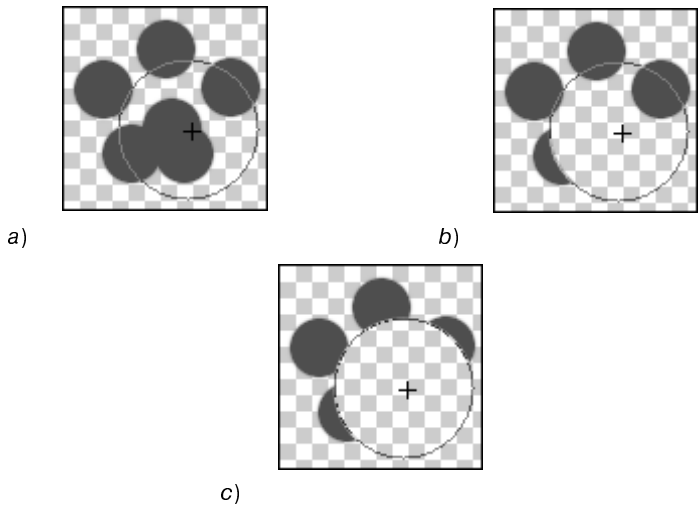


Рис. 3.26. Пример удаления пикселей:
a — исходное изображение,
b — удаляются только смежные пиксели,
c — удаляются все пиксели

Режим **Find Edges** (Поиск краев) в целом аналогичен режиму **Contiguous**, но обеспечивает большую резкость обработки краев.

Если начать перемещение инструмента, не отпуская клавишу мыши, то продолжится удаление пикселей, попадающих в площадь кисти, в соответствии с только что изложенными принципами.

Пиксель, оказавшийся под крестиком в момент нажатия клавиши мыши служит образцом, близость к которому по цвету дру-

гих пикселей приводит к их удалению. Если в списке **Sampling** (Выбор образца) установить режим **Once** (Однажды), то однажды выбранный образец в процессе движения инструмента по изображению меняться не будет. Установка режима **Continuous** (Непрерывно) означает, что каждый пиксель, оказавшийся под крестиком, становится образцом. В режиме **Background Swatch** (Фоновый цвет в качестве образца) будут удаляться только пиксели текущего фонового цвета и близкие по цвету в соответствии с допуском (**Tolerance**).

В поле **Tolerance** (Допуск), устанавливается величина допуска (от 1 до 100 %), определяющая степень близости по цвету удаляемых пикселей к образцу. Чем меньше эта величина, тем ближе к образцу должны быть по цвету пиксели, чтобы они были удалены.

Установка флажка **Protect Foreground Color** (Защита основного цвета) защищает пиксели, окрашенные основным цветом от удаления.

Инструмент **Magic Eraser** (Волшебный ластик) — удобен для удаления пикселей достаточно однородных по цвету фрагментов или выделенных фрагментов. После выбора инструмента нужно щелкнуть по одному из пикселей удаляемого фрагмента, который станет образцом.

В зависимости от значения параметра **Tolerance** (Допуск) (от 0 до 255) будут удаляться пиксели близкие по цвету к образцу. Чем меньше значение параметра **Tolerance** (Допуск), тем ближе по цвету к образцу должны быть удаляемые пиксели. Если установить 255, то будут удалены все пиксели изображения. Это удобно использовать для того, чтобы сделать прозрачным выделенный фрагмент изображения. Если удаление производится при снятом флажке **Contiguous** (Смежные), то пиксели будут удаляться независимо от их расположения на изображении. При установленном флажке удаляются только смежные пиксели.

Поле **Opacity** (Непрозрачность), позволяет регулировать степень воздействия инструмента на пиксели. Если установлено 100 %, то пиксели удаляются полностью, а фрагменты, где они находились, становятся прозрачными. При меньших значениях наступает частичная прозрачность, которая при значении 1 % практически не видна. Установка флажка **Anti-aliased** (Сглаживание) ведет к сглаживанию контура удаляемых фрагментов, а флажка

Use All Layers (Использовать все слои) — к удалению пикселей во всех видимых слоях изображения.

3.4.19. Использование фильтров

Программа Photoshop располагает большим количеством фильтров, применение которых осуществляется через меню **Filter** (Фильтр). Некоторые из них предназначены для устранения недостатков изображения (коррекции резкости, устранения мелких дефектов и т. д.). Большая часть фильтров, объединенных в несколько групп, предназначена для создания различных эффектов. Применение таких фильтров ведет к преднамеренному искажению изображения с целью имитации определенных художественных приемов. Обработанные такими фильтрами изображения можно использовать для декоративного оформления web-страниц, создания фоновых изображений.

Выбрать фильтр можно с помощью галереи фильтров, которая открывается командой **Filter Gallery** (Галерея фильтров) меню **Filter** (Фильтр). Галерея содержит не все имеющиеся в программе фильтры. Полный список групп и входящих в них фильтров находится в меню **Filter** (Фильтр). Применение некоторых фильтров может сопровождаться изменением цвета изображения, для чего используются основной (**Foreground**) и фоновый (**Background**) цвет, установленные в блоке выбора цвета панели инструментов. Изменяя цвета, и повторно применяя фильтр, можно добиваться различных результатов. Галерея содержит следующие группы фильтров:

- **Artistic** (Художественные) — преобразуют реальное изображение таким образом, как будто оно создавалось с использованием художественной техники (живописи, акварели, карандашного рисунка и др.);
- **Brush Strokes** (Мазки кисти) — аналогично фильтрам группы **Artistic** (художественные) имитируют создание изображения с использованием различных художественных манер (штриховка, обводка, разбрызгивание и др.);
- **Distort** (Деформация) — вносят в изображение геометрические искажения, имитируя просмотр через искажающие стекла;

- **Sketch** (Эскиз) — имитируют технику создания эскизных изображений (карандаш, уголь, мел и др.);
- **Styles** (Стилизация) — преобразуют плоское изображение в барельеф, создают оконтуривание, выделение краев и др.
- **Texture** (Текстура) — к изображению добавляют текстуру в виде мозаики, витража, крупных зерен и др.

Галерея открывается в отдельном окне, вид которого показан на рис. 3.27.

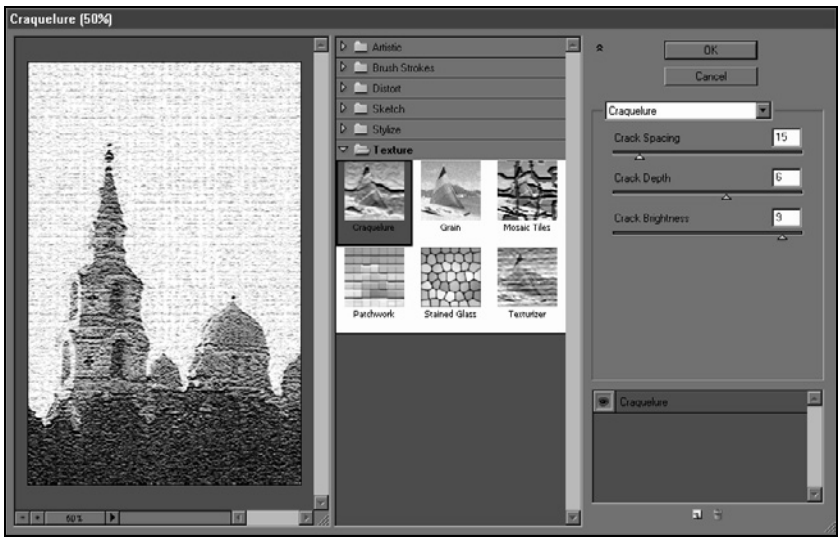


Рис. 3.27. Диалоговое окно галереи фильтров

В заголовке окна указывается название выбранного фильтра и масштаб изображения в окне просмотра. В левой части располагается окно просмотра, в котором демонстрируется изображение после обработки фильтром. Масштаб изображения можно менять, щелкая по кнопкам со значками "+" или "–" в правой нижней части окна. В центре окна перечислены названия групп фильтров. Чтобы сделать доступными фильтры группы, нужно щелкнуть по треугольной стрелке слева от названия группы. Далее можно поочередно щелкать по рисункам с названиями фильтров и наблюдать в окне просмотра результаты обработки изображения. В правой части располагается раскрывающийся

список фильтров галереи в алфавитном порядке и ползунки настроек параметров выбранного фильтра.

Для применения к изображению сразу нескольких фильтров необходимо после выбора фильтра щелкнуть по кнопке **New effect layer** (Новый слой эффекта) в нижней части окна, после чего в списке примененных фильтров над этой кнопкой появится новая запись. Следует иметь в виду, что верхняя запись списка всегда содержит название текущего фильтра, поэтому после нажатия кнопки **New effect layer** (Новый слой эффекта) в списке появятся две записи с одним названием и действие фильтра в этот момент также будет двойным.

Временно отказаться от действия фильтра, помещенного в список, позволяет щелчок мыши по кнопке с изображением глаза, слева от названия фильтра. Полностью удалить фильтр из списка можно, щелкнув по кнопке **Delete effect layer** (Удалить слой эффекта) в нижней части окна.

Результат действия нескольких фильтров зависит от порядка их расположения в списке. Чтобы изменить положение фильтра в списке нужно установить курсор над его названием, нажать клавишу мыши и, не отпуская ее, переместить название фильтра выше или ниже по списку. Фильтры будут применены к изображению после нажатия кнопки **OK** в правом верхнем углу окна.

3.4.20. Подготовка фоновых изображений

Фоновое изображение призвано сделать web-страницу более оригинальной, привлекательной и запоминающейся. При создании фоновых изображений следует придерживаться следующих рекомендаций.

- Тема фонового рисунка должна соответствовать теме страницы.
- Размеры рисунка должны быть в несколько раз меньше размеров страницы, что, прежде всего, обусловлено требованием быстрой загрузки файла рисунка по сети. В этом случае рисунок будет многократно повторяться, заполняя пространство страницы, поэтому следует обратить внимание на то, чтобы границы стыковки не портили общего впечатлении от фона.

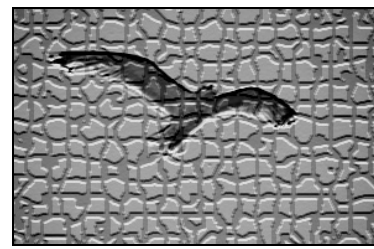
- Фоновый рисунок должен быть достаточно бледным, чтобы не мешать восприятию расположенной на странице информации и, прежде всего, текста. Это достигается одновременным увеличением яркости и уменьшением контраста.

Осуществление стилизации фонового рисунка к теме web-страницы возможно с помощью фильтров. Изменить яркость и контраст изображения можно в диалоговом окне **Brightness/Contrast** (Яркость/Контраст), которое открывается командой меню **Image | Adjustments | Brightness/Contrast** (Изображение | Настройки | Яркость/Контраст). Ползунок **Brightness** (Яркость) предназначен для увеличения (уменьшения) яркости изображения, а ползунок **Contrast** (Контраст) — контраста изображения. Установив флажок **Preview** (Предварительный просмотр), можно наблюдать результат непосредственно на изображении.

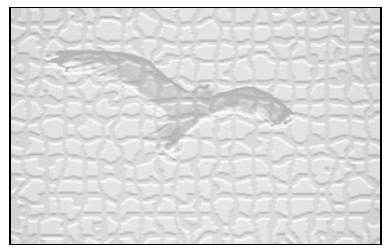
Пример создания фонового изображения:

В качестве исходного рисунка для создания фонового изображения будем использовать рис. 3.28, *a*.

a)



b)



c)

Рис. 3.28. Создание фонового изображения:
a — исходное изображение,
b — изображение после применения фильтра,
c — изображение после увеличения яркости и уменьшения контраста

- Применим к рисунку фильтр **Mosaic Tiles** (Мозаичные плитки) из группы **Texture** (Текстура). Результат показан на рис. 3.28, *b*.
- Сделаем рисунок более ярким (**Brightness = +90**) и уменьшим его контраст (**Contrast = −70**) (рис. 3.28, *c*).
- Вставим полученное изображение в HTML-документ в качестве фонового.
- В окне браузера фоновое изображение будет выглядеть следующим образом (рис. 3.29).

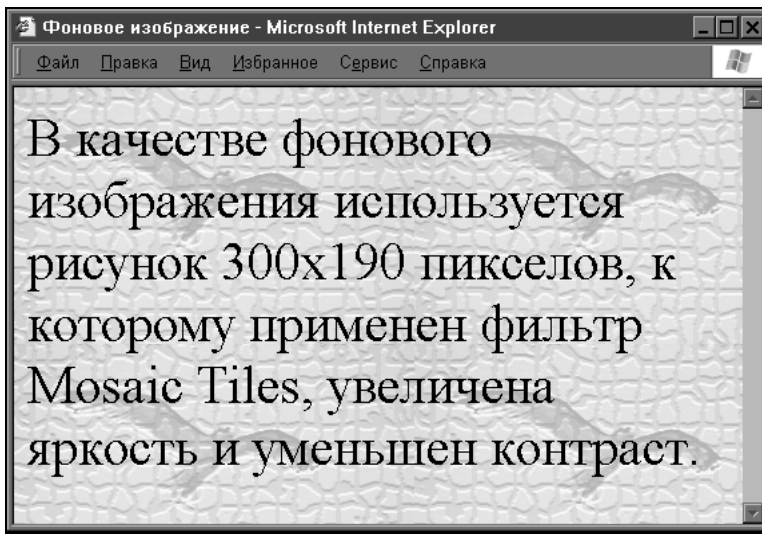


Рис. 3.29. Фоновое изображение в окне браузера

3.4.21. Создание GIF-анимации

На основе конкретных примеров в разделе будут рассмотрены общие приемы и способы создания GIF-анимации для HTTP-документов.

Имитация вращения объекта вокруг вертикальной оси

- Для создания нового изображения воспользуемся командой меню **File | New** (Файл | Создать). В открывшемся диалоговом

окне **New** (Создать) введем размеры изображения 100 × 100 px. Выберем цветовую модель изображения (**Color Mode**) — RGB Color, глубину цвета изображения — 8 bit, а содержимое фона (**Background Contents**) — Transparent (Прозрачный фон).

- Из группы векторных инструментов выберем **Custom Shape** (Готовая форма). На панели параметров, нажав кнопку **Shape Layers** (Слои формы), установим соответствующий режим рисования.
- Из раскрывающегося списка **Shape** (Форма) на панели параметров выберем нужную форму. Построим изображение формы в площади рисунка, растягивая его при нажатой клавише мыши и удерживая клавишу <Shift>, что позволит получить изображение правильной формы. Размеры формы и ее положение на этом этапе построения значения не имеют. Цвет заливки формы соответствует цвету, установленному в поле **Color** (Цвет) на панели параметров.
- Выберем инструмент **Move Tool** (Перемещение) на панели инструментов и установим флажок **Show Bounding Box** (Показать маркеры) на панели параметров. Вокруг изображения формы появятся маркеры, позволяющие изменять размеры вручную. Гораздо точнее и удобнее это можно сделать с помощью панели параметров, которая появится после щелчка мышью по любому маркеру (после появления двухсторонней стрелки). Прежде чем приступить к изменению параметров, нужно убедиться, что все они заданы в пикселях (px) и если это не так, щелкнуть правой кнопкой по соответствующему полю и в раскрывшемся списке выбрать **pixels** (пиксели). Далее в поля **X** и **Y** вводим значения координат центра формы (50 px), а в поля **W** и **H** — ширину и высоту формы (100 px). Завершается изменение параметров щелчком по кнопке **Commit Transform** (Завершить преобразование) в правой части панели параметров или нажатием клавиши <Enter>.
- Откроем палитру **Layers** (Слои) и убедимся, что появился слой формы Shape1. Слева от названия слоя формы располагаются две миниатюры. Первая из них — закрашенный прямоугольник. Двукратный щелчок по нему откроет диалоговое окно **Color Picker** (Выбор цвета), в котором можно выбрать новый цвет заливки формы. Вторая миниатюра содержит

изображение контура формы без заливки. В нижней части палитры нажмем кнопку **Add a layer style** (Добавить эффект слоя) и выберем эффект **Bevel and Emboss** (Фаска и рельеф). На рис. 3.30 показана палитра **Layers** (Слои) после всех описанных действий.

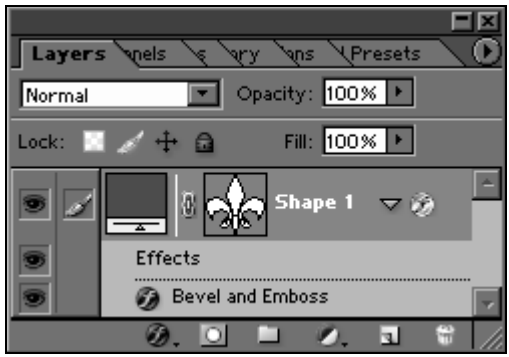


Рис. 3.30. Палитра **Layers** после создания векторной формы и применения к ней эффекта **Bevel and Emboss**

На рис. 3.31 показан внешний вид созданного изображения.

Рис. 3.31. Изображение, подготовленное к имитации вращения вокруг вертикальной оси

Сделаем запись о форме в палитре **Layers** более компактной, спрятав сведения о примененном эффекте. Для этого нужно щелкнуть по треугольной стрелке правее имени формы Shape 1.

- Создадим копию слоя Shape 1, для чего щелкнем по имени слоя правой кнопкой мыши и в открывшемся меню выберем

Duplicate Layer (Копировать слой). В открывшемся диалоговом окне будет предложено имя нового слоя Shape 1 copy, с чем можно согласиться, чтобы не тратить время на переименование. Далее, находясь на новом слое (запись о нем в палитре **Layers** выделена), нужно воспользоваться панелью параметров и уменьшить ширину формы до 90 px.

- Повторим действия, создавая каждый раз копию только что полученной копии и уменьшая ширину формы на 10 px. После проделанных действий изображение будет выглядеть, как показано на рис. 3.32.

Рис. 3.32. Изображение после создания нескольких слоев, с копиями, уменьшенными по ширине

- Теперь можно было бы повторить действия по созданию копий слоев, постепенно увеличивая ширину формы. Но так как у нас уже имеются такие слои, то достаточно создать копии уже имеющихся слоев и переместить их на новые места в соответствии с номерами. Для этого начнем создавать копии слоев, начиная со слоя Shape 1 copy 8. Его копия получит имя Shape 1 copy 10 и расположится над слоем Shape 1 copy 8, копия слоя Shape 1 copy 7 получит имя Shape 1 copy 11 и т. д. до слоя Shape 1 copy 18, который расположится над слоем Shape 1. После описанных преобразований палитра **Layers** (Слои) должна иметь вид, как на рис. 3.33.
- Остается расположить копии слоев в соответствии с номерами и подготовка изображения к созданию GIF-анимации будет завершена. Перемещение слоев в палитре осуществляется перетаскиванием при нажатой клавише мыши. Слой Shape 1 copy 9 следует расположить выше слоя Shape 1 copy 8, над ним слой Shape 1 copy 10 и т. д.

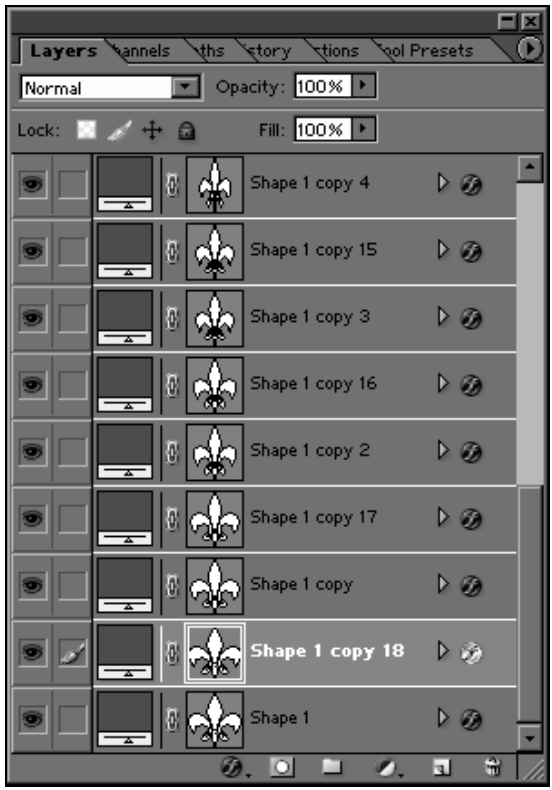


Рис. 3.33. Палитра **Layers** после создания всех копий слоев

- Для создания GIF-анимации перейдем в программу Adobe ImageReady, нажав кнопку **Edit in ImageReady** (Редактирование в программе ImageReady) в нижней части панели инструментов. После загрузки программы ImageReady мы увидим перешедшее в нее из программы Photoshop, только что созданное нами изображение. Если палитра **Animation** (Анимация) не установлена, то ее нужно установить через меню **Window** (Окно). В меню палитры **Animation** (Анимация) выберем команду **Make Frames From Layers** (Создать кадры из слоев), после чего каждый слой изображения формы станет кадром анимационного изображения. Палитра **Animation** (Анимация) с первыми семью кадрами показана на рис. 3.34.

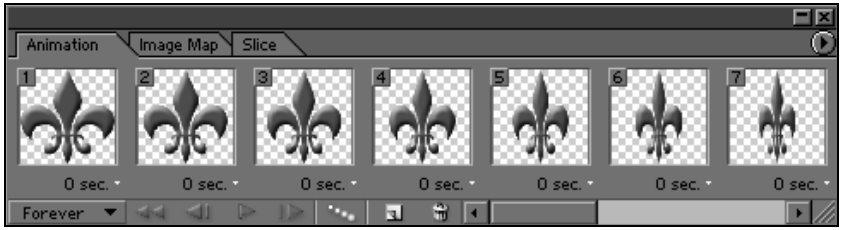


Рис. 3.34. Палитра **Animation** после создания кадров из слоев изображения

- Убедившись в том, что созданы все 19 кадров, можно приступать к воспроизведению анимации. В нижней части палитры **Animation** (Анимация) имеется кнопка в виде треугольной стрелки **Plays/stops animation** (Проигрывание/остановка анимации), нажатие на которую позволит начать воспроизведение, а повторное нажатие остановит его. В одну группу с этой кнопкой входят еще три: **Selects previous frame** (Выделить предыдущий кадр), **Selects next frame** (Выделить следующий кадр) и **Selects first frame** (Выделить первый кадр). Под первым кадром располагается список выбора числа повторных воспроизведений анимации: Forever — непрерывно, Once — однократно, Other — другая. Выбор последнего режима воспроизведения подразумевает открытие диалогового окна **Set Loop Count** (Установка числа повторов) для задания числа повторов (от 1 до 30 000).

  Для каждого кадра имеется возможность задания времени его демонстрации (задержки перехода к следующему кадру), которое располагается в нижней части кадра и по умолчанию равно 0 с. Щелчок по установленному значению раскрывает список, позволяющий выбрать задержку в секундах или вариант Other (Другая), при котором в открывающемся диалоговом окне задается произвольная задержка от 0 до 240 с. Если требуется изменить задержку сразу нескольких кадров, то они предварительно должны быть выделены. Для выделения идущих подряд кадров нужно выделить первый из них и, удерживая клавишу <Shift>, щелкнуть на последнем кадре. Выделение кадров в произвольном порядке осуществляется при нажатой клавише <Ctrl>. Выделение всех кадров осуществляется командой меню палитры **Select all frames** (Выделить все кадры).

При выборе числа повторов и времени задержки кадров следует помнить, что изменяющиеся с большой скоростью и в течение длительного времени изображения могут раздражать посетителей вашей страницы и вынудят их покинуть ее раньше времени. В некоторых случаях будет вполне достаточно повторить анимацию два-три раза. Возможно, также, сохранив большое число повторов, задать большую задержку на один из кадров (например первый кадр) и это сделает анимацию более спокойной. Для нашего примера рекомендуется установить задержку первого кадра 2 с и посмотреть результат.

- Сохранение анимации осуществляется командой **File | Save Optimized As** (Файл | Сохранить оптимизированное изображение как) меню программы ImageReady. В диалоговом окне **Save Optimized As** (Сохранить оптимизированное изображение как) в строке **Тип файла** следует выбрать, будет ли сохраняться только анимированный рисунок — Images Only (*.gif), готовый HTML-документ вместе с рисунком — HTML and Images (*.html) или только HTML-документ — HTML-Only (*.html). Если анимация создается для конкретного документа, то достаточно сохранить только один рисунок, вариант сохранения вместе с HTML-документом можно рекомендовать для предварительного просмотра анимации в окне браузера.

Имитация вращения объекта в плоскости вокруг центра

В предыдущем примере мы рассмотрели способ создания имитации вращения объекта вокруг вертикальной оси. Теперь рассмотрим имитацию вращения объекта вокруг центра.

- Повторим в программе Photoshop действия по созданию нового изображения, описанные в предыдущем примере, сохранив те же размеры изображения 100 × 100 px, но выберем другую форму (рис. 3.35).
- К полученному изображению применим эффект градиентной заливки, для чего обратимся к палитре **Layers** (Слои). Щелкнем по левой кнопке в нижней части палитры — **Add a layer style** (Добавить эффект слоя) и в раскрывшемся меню выберем эффект **Gradient Overlay** (Градиентная заливка). В диало-

говом окне **Layer Style** (Эффект слоя) можно изменить параметры эффекта. В частности из раскрывающегося списка **Gradient** (Градиент) можно выбрать многоцветную заливку по своему вкусу, а из списка **Style** (Стиль) — Radial (Радиальная заливка) Результат выполненных действий показан на рис. 3.35.

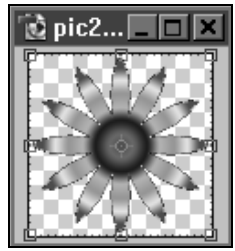


Рис. 3.35. Форма для имитации вращения с радиальной заливкой

- Как и в предыдущем примере создадим копию слоя Shape 1, для чего щелкнем по имени слоя правой кнопкой мыши и в контекстном меню выберем **Duplicate Layer** (Копировать слой). Согласимся с тем, чтобы новый слой назывался Shape 1 copy. Далее, убедившись, что запись о новом слое в палитре **Layers** (Слои) выделена, нужно щелкнуть мышкой по любому маркеру вокруг изображения и на панели параметров инструмента **Move Tool** (Перемещение) в поле **Set rotation** (Установка поворота) установить угол поворота формы 5°.

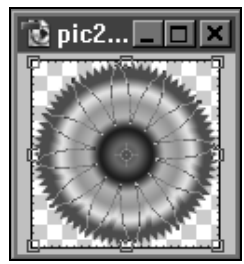


Рис. 3.36. Результат создания пяти копий слоев формы и поворота каждой на 5° относительно предыдущего слоя

- Повторим предыдущие действия еще 4 раза, создавая каждый раз копию только что полученной копии слоя и устанавливая угол поворота равным 5°. После всех описанных действий изображение будет выглядеть, как показано на рис. 3.36.
- Теперь можно переходить к преобразованию полученного многослойного изображения в GIF-анимацию, в точности повторяя действия, описанные в последних пунктах предыдущего примера, а именно:
  - переход в программу ImageReady и создание кадров из слоев изображения;
  - воспроизведение полученной анимации и при желании изменение параметров воспроизведения (например числа повторных воспроизведений);
  - сохранение анимации.

Изменение цвета объекта

В HTML-документах часто применяют анимацию, в котором объект изменяет свой цвет. Рассмотрим порядок действий над созданием данного эффекта.

- В программе Photoshop повторим действия, описанные в предыдущих примерах, по созданию нового изображения. Установим те же размеры изображения 100 × 100 px и выберем одну из готовых векторных форм. Пусть, например, форма будет красного цвета (рис. 3.37).

Рис. 3.37. Форма для имитации изменения цвета

- Используя палитру **Layers** (Слои), создадим копию слоя Shape 1. Изменим цвет заливки формы копии слоя. Двукрат-

ным щелчком по цветному прямоугольнику в соответствующей строке палитры **Layers** (Слои) откроем диалоговое окно **Color Picker** (Выбор цвета), в котором выберем синий цвет. Палитра **Layers** (Слои) после всех преобразований будет выглядеть, как показано на рис. 3.38.

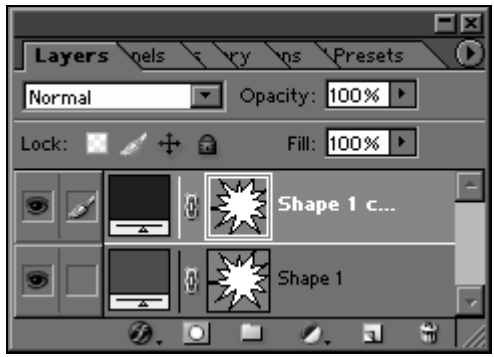


Рис. 3.38. Палитра **Layers** после создания двух слоев формы разного цвета

- Перейдем в программу ImageReady. В палитре **Animation** (Анимация) на первом кадре будет изображение формы синего цвета, так как копия слоя Shape 1 сору синего цвета. Щелкнем по кнопке с изображением глаза в левой части верхнего слоя Shape 1 сору палитры **Layers** (Слои). Этот слой станет невидимым, а изображение формы на первом кадре станет красным.
- В нижней части палитры **Animation** (Анимация) нажмем на вторую справа кнопку **Duplicates current frame** (Копировать текущий кадр), после чего появится новый кадр, являющийся точной копией первого кадра. Выполним повторный щелчок по кнопке с изображением глаза в левой части слоя Shape 1 сору палитры **Layers** (Слои), после чего слой снова станет видимым, а изображение формы на втором кадре станет синим.
- Создадим третий кадр, являющийся копией второго и сделаем невидимым слой Shape 1 сору. Изображение третьего кадра станет красным. После этого палитра **Animation** (Анимация) примет вид, показанный на рис. 3.39.

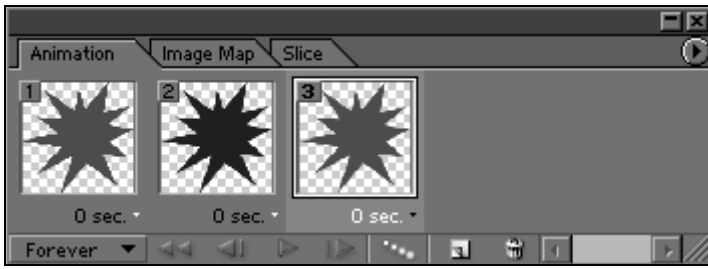


Рис. 3.39. Палитра **Animation** после создания трех кадров

- Между первым и вторым кадрами необходимо поместить промежуточные кадры, на которых цвет первого кадра будет постепенно становиться прозрачным. Таким образом, под первым кадром будет проявляться цвет второго кадра, в результате чего будет создаваться иллюзия плавного перехода от одного цвета к другому. Программа ImageReady позволяет автоматизировать процесс создания промежуточных кадров, для чего в нижней части палитры **Animation** (Анимация) имеется кнопка **Tween** (Промежуток), нажатие на которую открывает одноименное диалоговое окно (рис. 3.40).

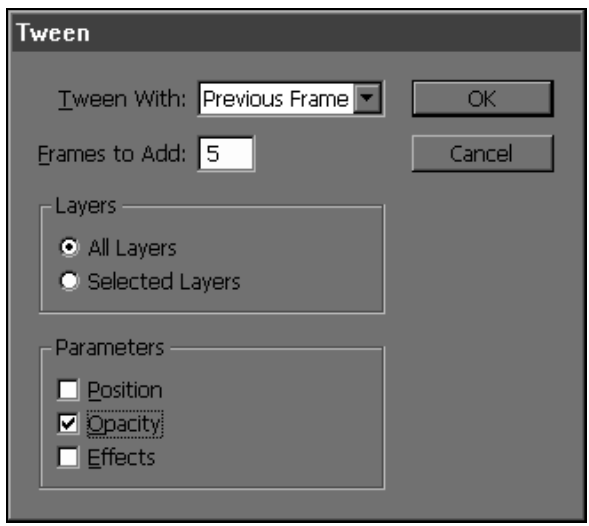


Рис. 3.40. Диалоговое окно **Tween**

Второй кадр нужно предварительно выделить, щелкнув по нему мышью. В списке **Tween With** (Заполнить кадрами) можно выбрать: **Selection** (Между выделенными кадрами), **First Frame** (Начиная с первого кадра), **Previous Frame** (Начиная с предыдущего кадра). В нашем случае нужно выбрать **Previous Frame** (Начиная с предыдущего кадра). В поле **Frames to Add** (Добавить кадры) необходимо указать число промежуточных кадров, например 5. Переключатель **Layers** (Слои) нужно установить в положение **All Layers** (Все слои), а в группе **Parameters** (Параметры) установить флажок **Opacity** (Непрозрачность), сняв остальные флажки. После нажатия кнопки **OK** промежуток между первым и вторым кадрами будет заполнен пятью новыми кадрами. Выделим последний кадр (теперь он стал восьмым) и заполним промежуток между ним и предыдущим кадром также пятью новыми кадрами;

- Анимация готова, ее можно просмотреть и сохранить.

Изменение эффекта

Рассмотрим способ создания анимации с изменением эффектов изображения.

- В программе Photoshop проделаем уже известные операции по созданию нового изображения размером 100 × 100 px, с прозрачным фоном. Построим готовую векторную форму в виде сердца красного цвета.
- Перейдем в программу ImageReady и откроем палитру **Animation** (Анимация). Используя кнопку **Add a layer style** (Добавить эффект слоя), в нижней части палитры **Layers** (Слои) раскроем список эффектов слоя, выберем **Bevel and Emboss** (Фаска и рельеф) и установим следующие параметры:
  - В списке **Style** (Эффект) — Inner Bevel;
  - В списке **Technique** (Техника) — Smooth;
  - Переключатель **Direction** (Направление) в положение Up;
  - В поле **Depth** (Глубина фаски) — 100 %;
  - В поле **Size** (Размер теневой области) — 5;
  - В поле **Soften** (смягчение границ) — 10.

  Значения остальных параметров можно не менять.

- Создадим еще две копии первого кадра в палитре **Animation** (Анимация). Выделим средний кадр и в диалоговом окне **Bevel and Emboss** (Фаска и рельеф) изменим значение параметра Size на 25. После этого палитра **Animation** (Анимация) будет иметь следующий вид (рис. 3.41).

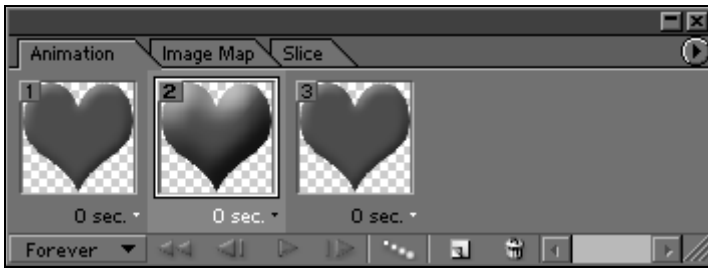


Рис. 3.41. Палитра **Animation** после создания трех кадров с применением эффекта **Bevel and Emboss**

- Убедившись, что выделен средний кадр, поместим между первым и вторым кадрами промежуточные кадры, используя кнопку **Tween** (Промежуток) палитры **Animation** (Анимация). В диалоговом окне **Tween** (Промежуток) из списка **Tween with** (Заполнить кадрами) выберем **Previous Frame** (Начиная с предыдущего кадра). Установим число промежуточных кадров 3, переключатель **Layers** (Слои) в положение **All Layers** (Все слои), а в группе **Parameters** (Параметры) установим флажок **Effects** (Эффекты), сняв остальные флажки. Выделим последний кадр, который теперь стал шестым, и также поместим между ним и предыдущим кадром три промежуточных кадра. Можно изменить время задержки среднего кадра, установив 0,5 с, что сделает анимацию более спокойной.
- Полученную анимацию можно воспроизвести и при желании сохранить.

Перемещение объекта

Для анимации часто применяют перемещение объекта, создавая иллюзию движения.

Прежде всего, необходимо найти подходящий по содержанию рисунок и подготовить его для использования в анимации. Под-

готовка может заключаться в создании прозрачного фона и уменьшении его размеров, если они велики. Обычно размеры рисунка не должны превышать нескольких десятков пикселей, в противном случае даже при небольшом числе кадров размер файла с анимацией будет достаточно большим, что затруднит его практическое использование. Предположим, что такой рисунок имеется (рис. 3.42) и находится в файле bus.gif, тогда его можно открыть непосредственно в программе ImageReady, используя команду **File | Open** (Файл | Открыть).

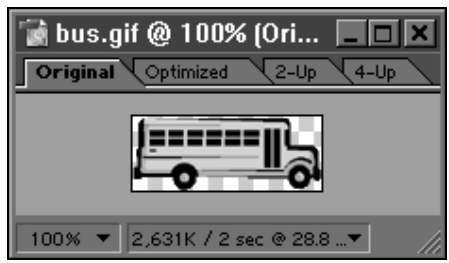


Рис. 3.42. Рисунок, подготовленный к имитации движения (размеры 100 × 39 px) и открытый в программе ImageReady

- В программе ImageReady создадим новое изображение, которое по высоте будет соответствовать высоте приведенного выше рисунка, а по ширине будет превышать его в два раза. Очевидно, что принципиального значения ширина рисунка не имеет. При большей ширине можно будет дольше наблюдать движение объекта, но при этом потребуется и больше кадров. Используя команду **File | New** (Файл | Создать), откроем диалоговое окно **New Document** (Создать документ) и введем значение поля **Width** (Ширина) — 200 px, поля **Height** (Высота) — 39 px. Переключатель **Contents of First Layer** (Заполнение первого слоя) необходимо установить в положение **Transparent** (Прозрачный). Назовем это изображение pic5.
- Сделаем активным рисунок bus.gif. Откроем палитру **Layers** (Слои), подведем указатель мыши к названию слоя Layer 1, нажмем левую кнопку и, не отпуская ее, поместим указатель над изображением pic5, после чего отпустим кнопку мыши. В палитре **Layers** (Слои) изображения pic5 должен появится новый слой — копия слоя изображения bus.gif.

Выбрав инструмент **Move Tool** (Перемещение), можно переместить слой таким образом, чтобы он занял правильное положение по вертикали в границах изображения pic5. При включенном инструменте **Move Tool** (Перемещение) слой можно перемещать не только мышью, но и клавишами управления курсором. При нажатой клавише <Shift> шаг перемещения возрастает с 1 до 10 px.

- Откроем палитру **Animation** (Анимация) и убедимся, что на первом кадре находится изображение pic5. Клавишей управления курсором переместим изображение влево до того момента, когда исчезнут последние пиксели его правой части. Создадим копию первого кадра и клавишей управления курсором переместим изображение вправо также до момента исчезновения последних пикселей, но уже левой части изображения. После этого палитра **Animation** (Анимация) будет содержать два кадра, на которых не будет ничего, кроме прозрачного фона.

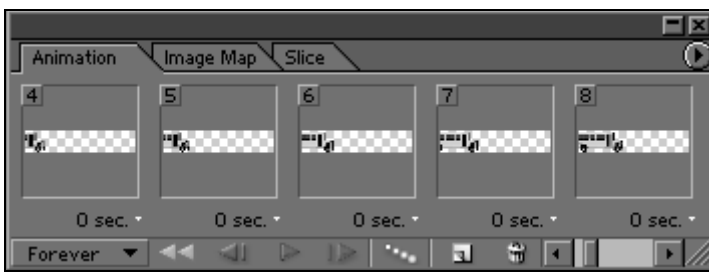


Рис. 3.43. Несколько промежуточных кадров палитры **Animation**, имитирующих движение объекта

- При выделенном втором кадре нажмем кнопку **Tween** (Промежуток) в нижней части палитры **Animation** (Анимация) и откроем диалоговое окно **Tween** (Промежуток). Теперь нам предстоит уже известная операция по размещению между первым и вторым кадрами промежуточных кадров. В диалоговом окне **Tween** (Промежуток) установим число кадров 25, а в группе **Parameters** (Параметры) установим флажок **Position** (Положение), сняв другие флажки. После нажатия кнопки **OK** в палитре **Animation** (Анимация) появятся 25 промежуточ-

ных кадров, демонстрирующих перемещение изображения автобуса слева направо. Кадры с 4 по 8 показаны на рис. 3.43.

- После воспроизведения анимации ее при желании можно сохранить.

3.4.22. Фрагментация изображений

Под *фрагментацией изображения* подразумеваются действия, в результате которых изображение разрезается на прямоугольные фрагменты, сохраняемые в виде отдельных файлов. При желании можно сохранить не только фрагменты изображения, но и HTML-документ, в котором первоначальное изображение будет восстановлено из фрагментов с помощью таблицы. Фрагментацию можно использовать для создания графических меню или активных кнопок (ролловеров), изменяющих свой внешний вид в момент осуществления ссылки.

Фрагментация осуществляется с помощью инструментов  — **Slice** (Фрагмент) и **Slice Select** (Выделение фрагмента), которые имеются как в программе Photoshop, так и в программе ImageReady.

Для этой цели рекомендуется использовать программу ImageReady, которая располагает большими возможностями. В программе ImageReady в дополнение к инструментам **Slice** (Фрагмент) и **Slice Select** (Выделение фрагмента) имеется одноименная палитра **Slice** (Фрагмент), которая может быть открыта командой **Slice** (Фрагмент) меню **Window** (Окно). Для быстрого открытия этой палитры служит кнопка **Bring the Slice Palette Forward** (Поместить палитру Slice на передний план) на панели параметров инструмента **Slice Select** (Выделение фрагмента).

После выбора инструмента **Slice** (Фрагмент), при нажатой кнопке мыши, следует растянуть прямоугольную рамку вокруг нужного фрагмента. В результате выполненных действий изображение разделится на несколько пронумерованных фрагментов. Вокруг обведенного фрагмента появится сплошная рамка. Этот фрагмент называется *пользовательским*. Остальные фрагменты дополняют пользовательский фрагмент до полного изображения. Они обозначаются пунктирными рамками и называются *автофрагментами*.

Панель параметров инструмента **Slice** (Фрагмент) содержит список **Style** (Стиль), который позволяет выбрать один из трех способов построения фрагмента:

- Normal (Обычный) — любое соотношение длины и ширины фрагмента;
- Fixed Aspect Ratio (Фиксированное соотношение) — задается фиксированное соотношение между длиной и шириной;
- Fixed Size (Фиксированный размер) — задаются точные размеры длины и ширины. При выборе двух последних вариантов в поля **Width** (Ширина) и **Height** (Высота) вводятся соответствующие значения.

Можно построить несколько пользовательских фрагментов. После того как фрагмент перестал быть активным, например в результате построения другого фрагмента, его изменение и перемещение осуществляется инструментом **Slice Select** (Выделение фрагмента).

Для перемещения пользовательского фрагмента необходимо указатель мыши инструмента **Slice Select** (Выделение фрагмента) установить внутри фрагмента и, нажав и удерживая кнопку мыши, переместить фрагмент в необходимое место экрана.

Изменение размеров фрагмента осуществляется протаскиванием левой кнопки мыши за маркеры выделения.

Удалить выделенный пользовательский фрагмент можно, нажав на клавишу <Delete>.

При создании кнопок или разделов меню требуется разделение изображения на фрагменты одинакового размера. Для этого удобно воспользоваться командой **Divide Slice** (Разделить фрагмент) меню **Slices** (Фрагменты). Откроется диалоговое окно **Divide Slice** (Разделить фрагмент).

В разделе **Divide Horizontally Into** (Разделить горизонтально) можно выбрать один из двух способов разделения на горизонтальные фрагменты. Установка переключателя в положение **slices down, evenly spaced** (одинаковые фрагменты вниз) позволяет указать число равных по величине горизонтальных фрагментов, а в положение **pixels per slice** (пикселей на фрагмент) — задать высоту горизонтальных фрагментов в пикселях, начиная с верхнего фрагмента.

Раздел **Divide Vertically Into** (Разделить вертикально) предназначен для разделения на вертикальные фрагменты. Число одинаковых фрагментов или ширина вертикальных фрагментов задаются соответственно в полях **slices across, evenly spaced** (одинаковые фрагменты поперек) или **pixels per slice** (пикселей на фрагмент).

Палитра **Slice** (Фрагмент) предназначена для задания параметров пользовательских фрагментов. Возможно задание параметров и для автофрагментов, но при этом они преобразуются в пользовательские фрагменты. Палитра **Slice** (Фрагмент) показана на рис. 3.44.

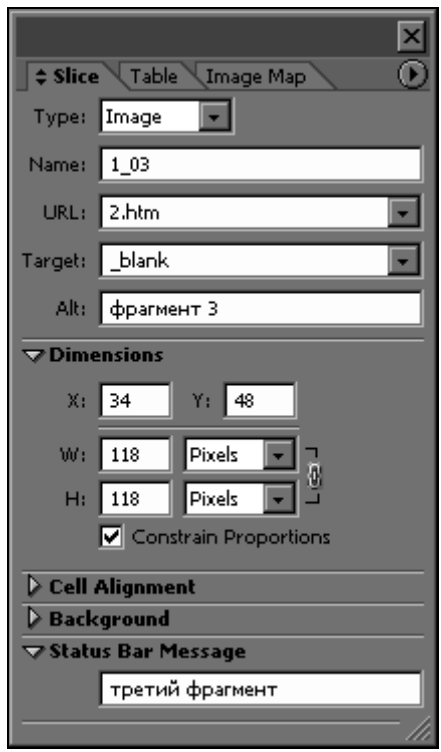


Рис. 3.44. Палитра **Slice**

После предварительного выделения фрагмента в палитре **Slice** (фрагмент) становятся доступны следующие элементы:

- в списке **Type** (Тип) можно выбрать **Image** (Графический), что означает, что фрагмент является частью графического изображения. В случае выбора **No Image** (Неграфический), в появившееся поле **Text** (Текст) можно ввести текст HTML-документа, который разместится вместо выделенного графического фрагмента, при этом флажок **Text is HTML** (Текст HTML) должен быть установлен. Если флажок не установлен, то в поле **Text** (Текст) можно вводить только обычный текст, не содержащий HTML-меток. Результат введения HTML-документа или текста можно увидеть только в окне браузера. Выбрав в списке **Type** (Тип) значение **Table** (Таблица), можно вместо графического фрагмента вставить таблицу, разделив фрагмент на ячейки командой **Divide Slice** (Разделить фрагмент) меню **Slices** (Фрагменты). В дальнейшем для каждой созданной ячейки таблицы возможно изменение параметров с использованием палитры **Slice** (Фрагмент) после выделения ячейки инструментом **Slice Select** (Выделение фрагмента);
- в поле **Name** (Имя) можно задать имя фрагмента (по умолчанию предлагается имя, включающее имя файла рисунка и номер фрагмента, с чем можно согласиться);
- в списке **URL** выбирается имя файла или URL-адрес документа, на которого осуществляется ссылка (перед вводом первого адреса список пуст и адрес вводится в поле вручную);
- в списке **Target** (Цель) можно выбрать одно из значений определяющих окно, в котором откроется адресуемый документ. Эти значения аналогичны значениям параметра `TARGET` метки `<A>`;
- в поле **Alt** можно ввести текст, который будет появляться в качестве всплывающей подсказки, при наведении курсора на данный фрагмент.

Ниже располагаются дополнительные разделы палитры, содержимое которых раскрывается после щелчка по треугольнику слева от названия раздела. Раздел **Dimensions** (Размеры), позволяет изменить координаты левого верхнего угла фрагмента (**X**, **Y**) и его размеры (**W**, **H**). При установленном флажке **Constrain Proportions** (Сохранять пропорции) изменение одного из размеров ведет к пропорциональному изменению другого. Раздел **Cell**

Alignment (Выравнивание ячейки) позволяет выровнять содержимое фрагмента, отнесенного к типу **No Image**. Выравнивание по горизонтали осуществляется выбором из списка **Horiz** одного из следующих значений: **Left** (По левому краю фрагмента), **Center** (По центру фрагмента), **Right** (По правому краю), **Default** (По умолчанию). Для выравнивания по вертикали из списка **Vert** можно выбрать: **Top** (Выравнивание по верхнему краю фрагмента), **Baseline** (Выравнивание базовых линий строк таблицы, помещенной в площадь фрагмента), **Middle** (Центрирование относительно верхнего и нижнего краев фрагмента), **Bottom** (Выравнивание по нижнему краю фрагмента), **Default** (По умолчанию). В разделе **Background** (Фон) выбирается цвет фона содержимого, отнесенного к типу **No Image**, или цвет прозрачных участков фрагмента изображения. В разделе **Status Bar Message** (Сообщение в строке состояния) можно ввести текст сообщения, которое появится в строке состояния браузера при наведении курсора на данный фрагмент.

Сохранить фрагментированное изображение можно с помощью команды **Save Optimized As** (Сохранить оптимизированное изображение как) меню **File** (Файл) программы ImageReady. Так же как и при сохранении анимированных изображений, в диалоговом окне **Save Optimized As** (Сохранить оптимизированное изображение как) в строке **Format:** (Тип файла:) можно выбрать тип файла для сохранения: будет ли сохраняться только набор фрагментов рисунка — Images Only (*.jpg), готовый HTML-документ вместе с фрагментами рисунка — HTML and Images (*.html) или только HTML-документ — HTML-Only (*.html). Здесь можно рекомендовать второй вариант сохранения, что в дальнейшем позволит поместить в создаваемый документ копию файла, содержащего таблицу со вставленными в ячейки фрагментами изображения.

Пример создания ролловера

Ролловер — графический объект, изменяющий свой внешний вид в окне браузера в зависимости от действий пользователя, например наведения на него указателя мыши. Это достигается заменой одного рисунка другим. Ролловер может использоваться в качестве динамической кнопки или раздела меню.

Для создания ролловера необходимо как минимум два рисунка, а также программа-сценарий, которая будет управлять сменой рисунков. Рассмотрим создание графического меню.

- В программе Photoshop откроем рисунок, которому предстоит стать фоновым рисунком меню.
- Используя инструмент **Type** (Текст), напишем на рисунке названия четырех разделов. Чтобы весь текст оказался в одном слое, следует нажимать клавишу <Enter> после ввода названия каждого раздела. В нижней части палитры **Layers** (Слои) раскроем список эффектов слоя, выберем **Bevel and Emboss** (Фаска и рельеф) и создадим эффект выпуклости текста. Подбирая размер шрифта и расстояние между строчками в палитре **Character** (Шрифт), равномерно распределим текст по изображению, чтобы получился результат, показанный на рис. 3.45.

Рис. 3.45. Рисунок с названиями разделов, подготовленный к созданию ролловеров

- Предварительно сохраним рисунок.
- Выделив весь текст, изменим его цвет и сохраним рисунок под другим именем (команда **Save As** (Сохранить как) меню **File** (Файл)).
- Перейдем в программу ImageReady и в меню **Slices** (Фрагменты) выберем команду **Divide Slice** (Разделить фрагмент). В диа-

логовом окне **Divide Slice** (Разделить фрагмент) в разделе **Divide Horizontally Into** (Разделить горизонтально) установим переключатель в верхнее положение и укажем число фрагментов — 4. Рисунок после фрагментации будет выглядеть как показано на рис. 3.46.

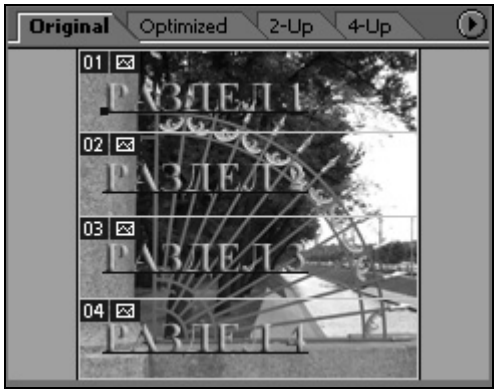


Рис. 3.46. Внешний вид рисунка после фрагментации

- Используя команду **File | Save Optimized As** (Файл | Сохранить оптимизированное изображение как) меню программы ImageReady сохраним рисунок, указав в строке **Format:** (Тип файла:) — HTML and Images (*.html). Будет создан HTML-документ, в котором фрагменты рисунка будут объединены с помощью таблицы, сами же фрагменты будут помещены в отдельную папку Images.
- Откроем в программе ImageReady второй экземпляр рисунка и осуществим его фрагментацию аналогично первому экземпляру.
- Сохраним фрагментированное изображение второго экземпляра, но, в отличие от первого, в строке **Format:** (Тип файла:) выберем Images Only (*.gif).

  В результате в папке Images должно появиться 8 рисунков: 4 фрагмента первого экземпляра и 4 второго. HTML-документ, объединяющий фрагменты одного из экземпляров рисунка в единое целое, показан в листинге 3.3.

Листинг 3.3. HTML-документ, объединяющий фрагменты рисунка в единое изображение с помощью таблицы

```
<html>
<head>
<title>фрагментация</title>
</head>
<body>
<table width="170" height="165" border="0" cellpadding="0"
cellspacing="0">
 <tr>
  <td><img src="images/F_01.gif" width="170" height="41"
alt=""></td>
 </tr>
 <tr>
  <td><img src="images/F_02.gif" width="170" height="42"
alt=""></td>
 </tr>
 <tr>
  <td><img src="images/F_03.gif" width="170" height="41"
alt=""></td>
 </tr>
 <tr>
  <td><img src="images/F_04.gif" width="170" height="41"
alt=""></td>
 </tr>
</table>
</body>
</html>
```

- Откроем созданный HTML-документ в программе Dreamweaver. В режиме **Design** (Конструирование) выделим рисунок первого раздела и удалим его, нажав клавишу <Delete>. То же самое проделаем с остальными рисунками. На экране останется пустая таблица с четырьмя ячейками, в которые предстоит поместить четыре ролловера. Установим указатель мыши в верхнюю ячейку и щелкнем по кнопке **Rollover Image** (Ролловер) из семейства **Images** (Изображения) в группе инструмен-

тов **Common** (Общие). Откроется диалоговое окно **Insert Rollover Image** (Вставка изображения ролловера) (рис. 3.47).

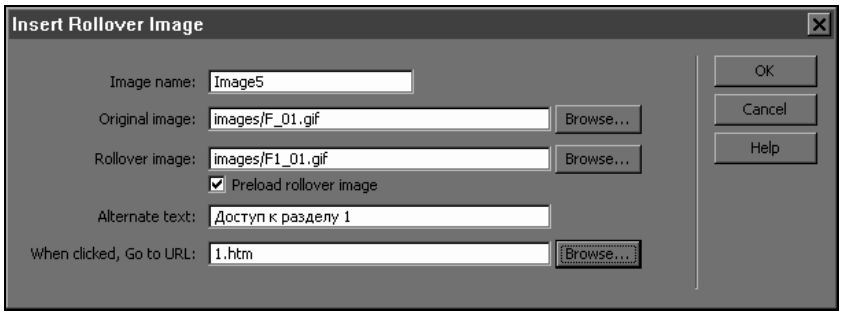


Рис. 3.47. Диалоговое окно **Insert Rollover Image**

В поле **Image name** (Имя изображения) указывается имя ролловера, которое будет использоваться программой-сценарием. Можно согласиться с тем, что программа Dreamweaver предлагает по умолчанию.

Поле **Original image** (Первоначальное изображение) предназначено для ввода пути к файлу рисунка, который должен постоянно отображаться на странице, пока указатель мыши находится за его пределами. В нашем случае это первое изображение из первого комплекта. Для ввода пути к файлу удобнее использовать кнопку **Browse** (Поиск).

В поле **Rollover image** (Ролловер) вводится путь к файлу рисунка, который сменит первоначальное изображение, при наведении на него указателя мыши (первое изображение из второго комплекта).

Установка флажка **Preload rollover image** (Предварительная загрузка ролловера) предусматривает загрузку файла второго рисунка в браузер до того, как в нем может возникнуть необходимость, что позволит в окне браузера практически мгновенно заменить одно изображение другим.

В поле **Alternate text** (Альтернативный текст) вводится текст, который будет появляться в виде всплывающей подсказки.

В поле **When clicked, Go to URL** (После щелчка перейти по адресу) должен находиться URL-адрес или имя файла документа, на который осуществляется ссылка по этому разделу меню.

- После завершения работы с диалоговым окном в ячейке появится первоначальное изображение первого раздела, входящее в ролловер. Устанавливая курсор поочередно в каждую из трех оставшихся ячеек таблицы, нужно проделать описанные выше действия, создав еще три ролловера.
- Теперь можно перейти к просмотру документа в окне браузера и, перемещая курсор по рисункам, убедиться, что текст надписи меняет цвет (рис. 3.48).

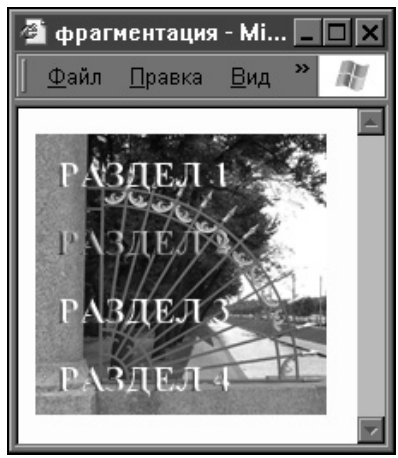


Рис. 3.48. Внешний вид меню, содержащего 4 ролловера. Название раздела изменяет цвет при наведении указателя мыши

В данном примере мы совместили возможности двух программ. Программа ImageReady позволила осуществить фрагментацию изображения и создать HTML-документ с таблицей, а программа Dreamweaver помогла создать ролловеры в ячейках готовой таблицы.

3.4.23. Создание изображений-карт

Изображение-карта это изображение, имеющее чувствительные области, по каждой из которых можно осуществлять ссылку на

другой документ. Например, на рисунке изображена группа музыкантов. Создав несколько чувствительных областей, каждая из которых включает лицо одного из музыкантов, можно осуществлять ссылки на документы, содержащие более подробную информацию о каждом из них. Чувствительные области могут иметь форму прямоугольника, окружности или многоугольника.

Для создания чувствительных областей предназначена группа инструментов, расположенная на панели инструментов программы ImageReady, включающая кнопки **Image Map** (Изображение-карта) и **Image Map Select** (Выделение чувствительной области) —  .

Группа инструментов **Image Map** (Изображение-карта) включает инструменты:

- **Rectangle Image Map** (Прямоугольная область);
- **Circle Image Map** (Область в виде окружности);
- **Polygon Image Map** (Область в виде многоугольника).

Инструмент **Image Map Select** (Выделение чувствительной области) предназначен для выделения, перемещения, изменения размеров и формы чувствительных областей. Изменение параметров выделенной чувствительной области осуществляется с помощью палитры **Image Map** (Изображение-карта), которая может быть открыта командой **Image Map** (Изображение-карта) меню **Window** (Окно). Для быстрого открытия данной палитры можно воспользоваться кнопкой **Bring the Image Map Palette Forward** (Поместить палитру **Image Map** на передний план) на панели параметров инструмента **Image Map Select** (Выделение чувствительной области).

Построение прямоугольных и круглых чувствительных областей осуществляется после выбора соответствующего инструмента путем растягивания рамки вокруг нужной области изображения при нажатой кнопке мыши.

Создание чувствительной многоугольной области осуществляется инструментом **Polygon Image Map** (Многоугольная область). Первый щелчок мышью по изображению устанавливает начальную точку первой стороны многоугольника, каждый последующий щелчок завершает построение одной стороны и начинает

построение следующей. Построение завершается после щелчка по начальной точке первой стороны многоугольной области.

Изображение с тремя чувствительными областями показано на рис. 3.49.

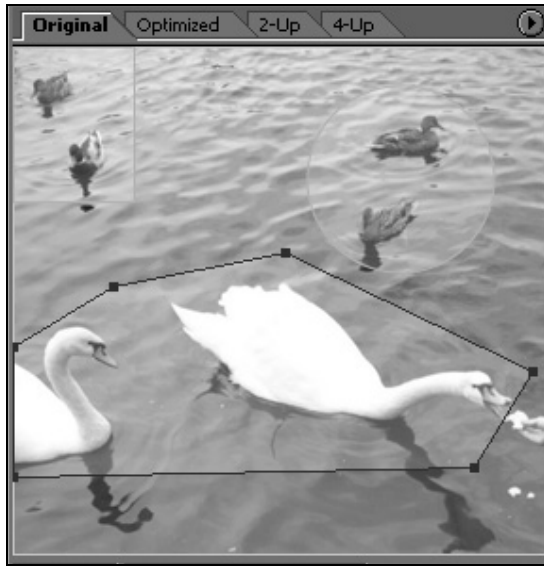


Рис. 3.49. Изображение, содержащее три чувствительные области различной формы

После построения чувствительных областей можно приступить к заданию их параметров. Чувствительная область выделяется инструментом **Image Map Select** (Выделение чувствительной области), после чего в палитре **Image Map** (Изображение-карта) задаются параметры выделенной области. Палитра **Image Map** (Изображение-карта) с содержимым раздела **Dimensions** (Размеры) для круглой области показана на рис. 3.50.

Элементы **Name**, **URL**, **Target**, **Alt** палитры **Image Map** (Изображение-карта) аналогичны соответствующим элементам палитры **Slice** (Фрагмент). В разделе **Dimensions** (Размеры) для круглой области задаются координаты центра круга (**X**, **Y**) и его радиус (**Radius**), а для прямоугольной области — координаты левого верхнего угла (**X**, **Y**), ширина (**W**) и высота (**H**) области.

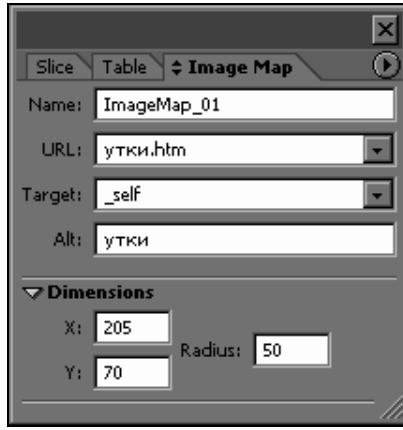


Рис. 3.50. Палитра **Image Map**

В результате создания чувствительных областей в само изображение никакие изменения не вносятся. Все необходимые параметры записываются во фрагменте HTML-документа внутри меток `<MAP>...</MAP>`. С помощью параметра `NAME` метки `<MAP>` карте присваивается имя, которое используется для ее связи с конкретным изображением. Одна и та же карта может быть связана с разными изображениями. Чтобы убедиться в этом сохраним изображение-карту, воспользовавшись командой **Save Optimized As** (Сохранить оптимизированное изображение как) меню **File** (Файл). В диалоговом окне **Save Optimized As** (Сохранить оптимизированное изображение как) в строке **Format** (Тип файла) выберем HTML and Images (*.html). Текст документа, созданного программой ImageReady, приведен в листинге 3.4.

Листинг 3.4. HTML-документ, содержащий изображение-карту Map1, связанную с рисунком природа.gif

```
<html>
<head>
<title>изобр-карта</title>
</head>
<body>
<img src="images/природа.gif" width="283" height="270" bor-
```

```
der="0" alt="" usemap="#Map1">
<map name="Map1">
<area shape="poly" alt="лебеди" coords="0,159, 53,127,
144,109, 275,172, 244,223, 0,228" href="лебеди.htm" tar-
get="_blank">
<area shape="rect" alt="утки" coords="0,0,64,82"
href="утки.htm" target="_self">
<area shape="circle" alt="утки" coords="205,70,50"
href="утки.htm" target="_self">
</map>
</body>
</html>
```

Программа поместила рисунок в отдельную папку images. Карте дано имя Map1, которое с помощью параметра USEMAP метки <IMG> связывает карту с рисунком. В параметре USEMAP перед именем карты должен находиться символ #. Для задания каждой чувствительной области внутри меток <MAP>…</MAP> используется метка <AREA>. Параметрами метки <AREA> являются:

- SHAPE — задает форму чувствительной области и может принимать значения "rect" – прямоугольник, "circle" – окружность, "poly" – многоугольник;
- COORDS — задает координаты чувствительных областей. Координаты перечисляются через запятую в следующей последовательности: X1, Y1, X2, Y2 — координаты левого верхнего и правого нижнего углов прямоугольника, X, Y, R — координаты центра и радиус окружности, X1, Y1, X2, Y2, X3, Y3 — координаты вершин многоугольника. Все координаты отсчитываются от левого верхнего угла изображения вправо (X) и вниз (Y);
- HREF — задает имя файла или URL-адрес, на который осуществляется ссылка (полностью аналогичен параметру HREF метки <A>);
- Назначение параметров ALT и TARGET должно быть понятно из описания одноименных элементов в палитре **Slice** (Фрагмент).

В приведенном примере три чувствительные области разной формы. По двум из них осуществляются ссылки на один и тот же документ утки.htm, по третьей — на документ лебеди.htm.

Глава 4

Использование таблиц для размещения данных и компоновки страниц

4.1. Создание таблиц средствами HTML

Значение *таблиц* для структуризации данных общеизвестно. Существует множество данных, которые удобно представлять в виде таблиц. Например, расписания транспортных средств, репертуары театров и концертных залов, технические характеристики устройств и т. д. В web-дизайне это не единственное применение таблиц. С помощью таблицы можно осуществлять компоновку элементов web-страницы (рисунков, фрагментов текста), размещая их в любых местах страницы в соответствии с желаниями дизайнера. Работа по компоновке страницы в значительной степени облегчается, если для этой цели используется программа Dreamweaver.

Для создания таблиц используются следующие метки:

- `<TABLE>...</TABLE>` — отмечают начало и конец таблицы;
- `<TR>...</TR>` — отмечают начало и конец строки;
- `<TD>...</TD>` — между метками располагаются данные одной ячейки таблицы;
- `<TH>...</TH>` — между метками располагаются данные одной ячейки таблицы, которая является заголовком столбца или

строки. В отличие от метки <TD>, содержимое меток <TH>…</TH> отображается браузером полужирным шрифтом и центрируется по горизонтали.

Рассмотрим следующий пример создания таблицы (листинг 4.1):

Листинг 4.1. Размещение таблицы с параметрами по умолчанию

```
<HTML>
<HEAD>
<TITLE>Таблицы</TITLE>
</HEAD>
<BODY>
<TABLE>
<TR>
<TD></TD><TH>1</TH><TH>2</TH><TH>3</TH>
</TR>
<TR>
<TH>1</TH><TD>ЯЧЕЙКА_1-1</TD><TD>ЯЧЕЙКА_1-2</TD><TD>ЯЧЕЙКА_1-3</TD>
</TR>
<TR>
<TH>2</TH><TD>ЯЧЕЙКА_2-1</TD><TD>ЯЧЕЙКА_2-2</TD><TD>ЯЧЕЙКА_2-3</TD>
</TR>
</TABLE>
</BODY>
</HTML>
```

В окне браузера такая таблица будет выглядеть как на рис. 4.1.

Обратим внимание на то, что по умолчанию таблица не имеет рамок вокруг ячеек и вокруг самой таблицы. Для создания рамок, изменения расстояний между ними, а также между рамками и данными внутри ячеек используются специальные параметры метки <TABLE>.

- BORDER="значение" — позволяет создать рамки вокруг таблицы и вокруг каждой ячейки. Значение параметра определяет толщину рамки вокруг таблицы в пикселях. Толщина рамки вокруг ячеек во всех случаях равна 1 пикселю.

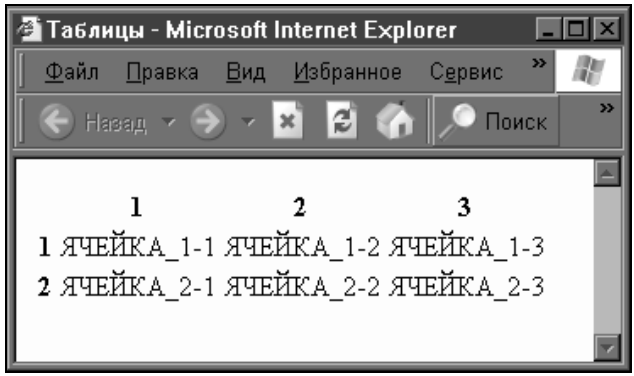


Рис. 4.1. Внешний вид таблицы с параметрами по умолчанию

В предыдущем документе в метку `<TABLE>` введем параметр `BORDER`, например: `<TABLE BORDER=5>`. Результат показан на рис. 4.2.

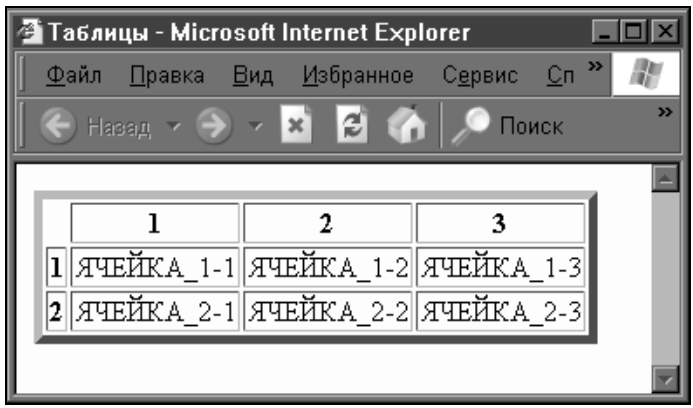


Рис. 4.2. Таблица, имеющая рамку толщиной 5 пикселей. Расстояние между рамками ячеек по умолчанию равно 2 пикселям

Задание `BORDER=0` эквивалентно отсутствию параметра `BORDER`, т. е. в обоих случаях рамка не воспроизводится. Следует также обратить внимание на то, что вокруг ячейки, не содержащей данных, рамка отсутствует.

- `CELLSPACING="значение"` — позволяет задать расстояние между рамками ячеек по горизонтали и вертикали. По умолча-

нию это значение равно 2. В случае CELLSPACING=0 рамки ячеек сливаются, образуя сетку разделительных линий. Вариант такой таблицы приведен на рис. 4.3.

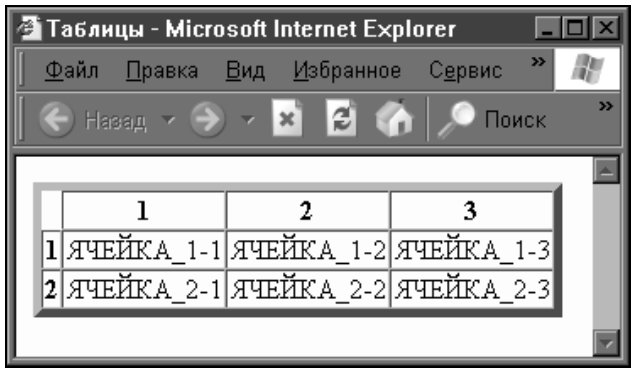


Рис 4.3. Таблица, в которой расстояния между рамками ячеек равны нулю

- CELLPADDING="значение" — позволяет задать расстояние между рамкой ячейки и данными внутри ячейки. По умолчанию это значение равно 1 пикселю. При задании CELLPADDING=0 — текст внутри ячейки будет касаться рамки, что для текстовых таблиц обычно неприемлемо. Зато для некоторых таблиц с рисунками в ячейках может быть недопустимым расстояние между рисунками. В этом случае использование параметра CELLPADDING со значением 0 является обязательным.

Установка размеров таблицы и ее горизонтального расположения в окне браузера осуществляется с помощью следующих параметров метки <TABLE>:

- WIDTH="значение" — задает ширину таблицы. Если после значения ширины установлен знак %, то она задается в процентах от ширины окна браузера, в противном случае — в пикселях;
- HEIGHT="значение" — задает высоту таблицы. При наличии знака % после значения высоты, она будет определяться в процентах относительно высоты окна браузера, в противном случае — в пикселях.

  Пример записи: <TABLE WIDTH=100% HEIGHT=100%>. Результат действия параметров показан на рис. 4.4.

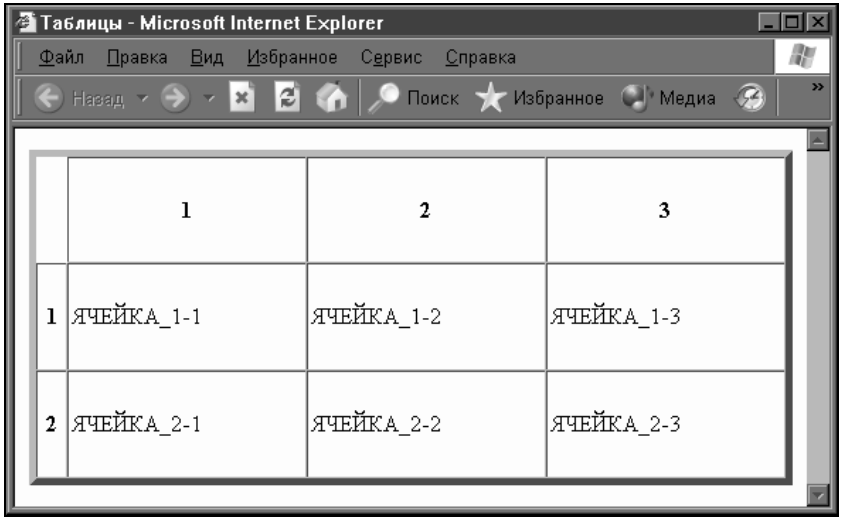


Рис. 4.4. Результат изменения размеров таблицы

Рекомендуется создать аналогичную таблицу и, изменяя размеры окна браузера, просмотреть изменения размеров таблицы.

Примечание

Если размеры, заданные в параметрах `WIDTH` и `HEIGHT`, меньше минимально возможных для данного содержимого ячеек, браузер может не выполнить заданных требований. В этом случае размеры таблицы будут доведены до минимально возможных значений.

- `ALIGN="значение"` — позволяет задать горизонтальное расположение таблицы в окне браузера. Возможны следующие значения параметра ALIGN:
  - `CENTER` — по центру;
  - `LEFT` — слева;
  - `RIGHT` — справа.

  Для значений `LEFT` и `RIGHT` при наличии свободного места предусмотрено обтекание таблицы с противоположной стороны текстом, идущим после метки `</TABLE>`. Пример показан на рис. 4.5.

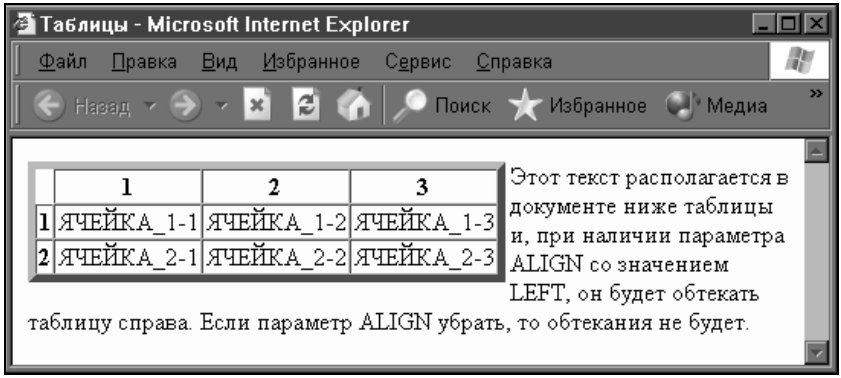


Рис. 4.5. Обтекание таблицы текстом

Если параметр `ALIGN` не задан, то таблица будет располагаться слева, но обтекания текстом не будет.

Цветовое оформление таблицы может осуществляться с помощью параметров:

- `BORDERCOLOR="значение"` — задает цвет рамки таблицы;
- `BORDERCOLORLIGHT="значение"` — задает цвет левой и верхней сторон рамки таблицы, правой и нижней сторон рамки ячейки;
- `BORDERCOLORDARK="значение"` — задает цвет правой и нижней сторон рамки таблицы, левой и верхней сторон рамки ячейки.

Между метками `<TABLE>…</TABLE>` с помощью меток `<CAPTION>…</CAPTION>` можно расположить заголовок таблицы. Для форматирования шрифта заголовка допускается использование внутри `<CAPTION>…</CAPTION>` других меток, например:

```
<CAPTION><H1>Это заголовок таблицы</H1></CAPTION>
```

Метка `<CAPTION>` может иметь параметр `ALIGN`, определяющий положение и выравнивание заголовка. Параметр может принимать значения:

- `TOP` — заголовок располагается над таблицей и выравнивается по центру (действует по умолчанию);
- `LEFT` — заголовок над таблицей слева;
- `RIGHT` — заголовок над таблицей справа;

- BOTTOM — заголовок располагается под таблицей и выравнивается по центру.

Следует отметить, что назначение метки <CAPTION> — это, прежде всего, привязка заголовка к таблице. Поэтому при перемещении таблицы, заголовок переместится вместе с ней.

В метках <TR>, <TD> и <TH> можно использовать следующие параметры:

- ALIGN="значение" — действие параметра распространяется на строку или ячейку, в метке которой он установлен. Значения параметра аналогичны значениям, приведенным выше для таблицы, но в этом случае выравнивается содержимое строки или ячейки;
- VALIGN="значение" — позволяет задать вертикальное выравнивание содержимого строки или ячейки. Параметр может принимать следующие значения:
  - TOP — по верхнему краю,
  - BOTTOM — по нижнему краю,
  - MIDDLE — по середине (действует по умолчанию);
- NOWRAP — запрещает браузеру разбивать текст на строки. Текст в ячейке с параметром NOWRAP вытянется в одну строку, увеличив ширину ячейки до ширины строки текста.

Кроме того, в метках <TR>, <TD> и <TH> можно использовать параметры BORDERCOLOR, BORDERCOLORLIGHT и BORDERCOLORDARK, действие которых будет распространяться на строку или ячейку, в метке которой они установлены.

С целью объединения нескольких ячеек по горизонтали или по вертикали в метках <TD> и <TH> используются параметры:

- COLSPAN="значение" — задает количество ячеек, объединяемых по горизонтали, включая текущую ячейку;
- ROWSPAN="значение" — задает количество ячеек, объединяемых по вертикали, включая текущую ячейку.

Пример таблицы с объединенными ячейками приведен на рис. 4.6.

	1	2	3	4	5
1	Объединились ячейки 1-1, 1-2, 1-3			ЯЧЕЙКА_1-4	ЯЧЕЙКА_1-5
2	ЯЧЕЙКА_2-1	ЯЧЕЙКА_2-2	ЯЧЕЙКА_2-3	Объединились ячейки 2-4, 2-5, 3-4, 3-5	
3	ЯЧЕЙКА_3-1	ЯЧЕЙКА_3-2	ЯЧЕЙКА_3-3		

Рис. 4.6. Таблица, в которой несколько ячеек объединены по горизонтали, а несколько – одновременно по горизонтали и вертикали

Текст документа, позволяющего создать такую таблицу, приведен в листинге 4.2.

Листинг 4.2. Объединение ячеек по горизонтали и вертикали

```
<HTML>
<HEAD>
<TITLE>Таблицы</TITLE>
</HEAD>
  <BODY>
   <TABLE BORDER=5>
    <TR>
<TD></TD><TH>1</TH><TH>2</TH><TH>3</TH><TH>4</TH><TH>5</TH>
    </TR>
    <TR>
      <TH>1</TH>
      <TD COLSPAN=3 ALIGN=CENTER>Объединились ячейки 1-1, 1-
2, 1-3</TD>
      <TD>ЯЧЕЙКА_1-4</TD><TD>ЯЧЕЙКА_1-5</TD>
    </TR>
    <TR>
      <TH>2</TH><TD>ЯЧЕЙКА_2-1</TD>
      <TD>ЯЧЕЙКА_2-2</TD><TD>ЯЧЕЙКА_2-3</TD>
      <TD COLSPAN=2 ROWSPAN=2>Объединились ячейки 2-4, 2-5,
3-4, 3-5</TD>
    </TR>
    <TR>
      <TH>3</TH><TD>ЯЧЕЙКА_3-1</TD>
```

```
      <TD>ЯЧЕЙКА_3-2</TD><TD>ЯЧЕЙКА_3-3</TD>
    </TR>
   </TABLE>
  </BODY>
</HTML>
```

Одна из ячеек одновременно содержит параметры `COLSPAN` и `ROWSPAN`, что позволяет объединить сразу четыре ячейки, две из которых располагаются в одной строке, а две в другой.

Размеры отдельных ячеек можно изменять с помощью известных параметров `WIDTH` и `HEIGHT`, значения которых, задаются как в пикселях, так и в процентах. При этом следует помнить, что изменение ширины ячейки ведет к изменению ширины всего столбца, в котором она находится, а изменение высоты ячейки ведет к изменению высоты всей строки.

4.2. Работа с таблицами в программе Dreamweaver

4.2.1. Добавление таблиц

Установив курсор в то место документа, куда должна быть вставлена таблица, необходимо открыть диалоговое окно **Table** (Таблица). Это можно сделать следующими способами:

- Команда **Table** (Таблица) меню **Insert** (Вставка)**;**
- Кнопка **Table** (Таблица) в группе инструментов **Common** (Общие) панели **Insert** (Вставка).

В диалоговом окне **Table** (Таблица) имеются три раздела для задания параметров таблицы (рис. 4.7). В разделе **Table Size** (Размер таблицы) можно задать:

- **Rows** — число строк таблицы;
- **Columns** — число столбцов таблицы;
- **Table width** — ширину таблицы в процентах (percent) или пикселях (pixels), выбрав единицы измерения из раскрывающегося списка;
- **Border thickness** — толщину рамки вокруг таблицы;

- **Cell padding** — расстояние между границами ячейки и ее содержимым в пикселях;
- **Cell spacing** — расстояние между ячейками в пикселях.

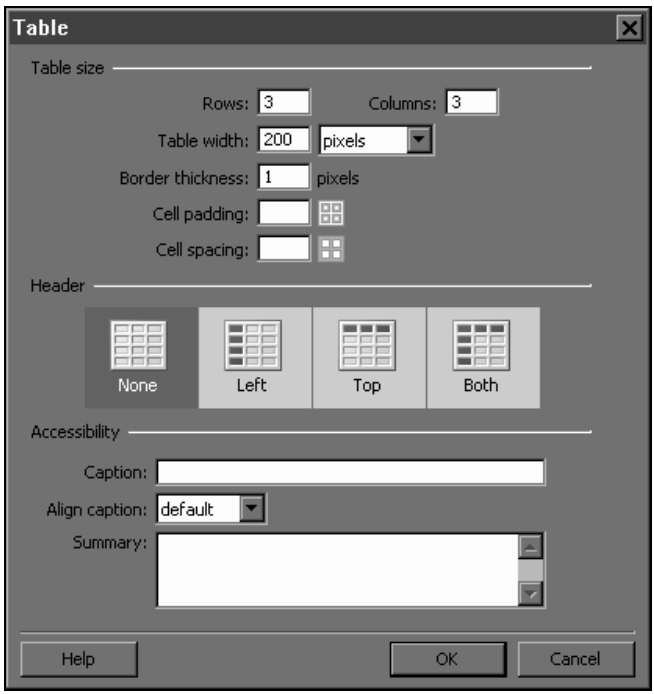


Рис. 4.7. Диалоговое окно **Table** для задания параметров таблицы

В разделе **Header** (Заголовок) можно выбрать группу ячеек, содержимое которых будет воспроизводиться браузером как заголовки, т. е. полужирным шрифтом с центрированием по горизонтали (None — нет заголовков, Left — левый столбец, Top — верхняя строка, Both — левый столбец и верхняя стока).

В разделе **Accessibility** (Дополнительные возможности) возможно:

- в поле **Caption** (Заголовок) поместить заголовок таблицы;
- из списка **Align caption** (Выравнивание заголовка) выбрать место расположения заголовка (default — по умолчанию,

top — над таблицей по центру, bottom — под таблицей, left — над таблицей слева, right — над таблицей справа);

- в текстовое поле **Summary** (Резюме) ввести текст резюме.

Если некоторые из параметров не будут заданы, то они не появятся в тексте HTML-документа, и их значения будут определяться по умолчанию. На начальной стадии создания таблицы не следует придавать слишком большого значения выбору параметров, так как в дальнейшем их можно легко изменить.

После создания таблицы все ячейки будут заполнены неразрывными пробелами ( ). Заполнение таблицы данными осуществляется в визуальном режиме (**Design**). Для этого нужно поместить курсор в соответствующую ячейку и начать заполнение, после чего неразрывный пробел будет автоматически удален из ячейки. Содержимым ячеек таблицы может быть не только текст, но и любые объекты, которые можно размещать на странице, в том числе другая таблица.

4.2.2. Изменение свойств таблицы

Перед выполнением действий по изменению свойств таблицы, она должна быть предварительно выделена. Признаком выделения таблицы служит появление вокруг нее черной рамки с тремя маркерами.

Выделение таблицы можно осуществлять следующими способами:

- Установить курсор внутри таблицы и дать команду меню **Modify** | **Table** | **Select Table** (Изменить | Таблица | Выделить таблицу);
- Указатель мыши поместить на правую или нижнюю сторону таблицы таким образом, чтобы вокруг таблицы появилась красная рамка и произвести щелчок мышью.

После выделения таблицы панель свойств **Properties** (Свойства) должна принять вид, приведенный на рис. 4.8. В случае отсутствия нижней части панели, следует щелкнуть по треугольной стрелке в ее правом нижнем углу.

Чтобы изменить количество строк или столбцов таблицы, в полях **Rows** (Строки) или **Cols** (Столбцы) необходимо ввести соответствующие значения. Строки добавляются или удаляются сни-

зу, а столбцы — справа. Новые значения вступят в силу после щелчка мышью в любом месте панели вне изменяемого поля. Аналогичным образом можно установить значения других параметров таблицы, которые приведут к изменению соответствующих параметров метки <TABLE>.

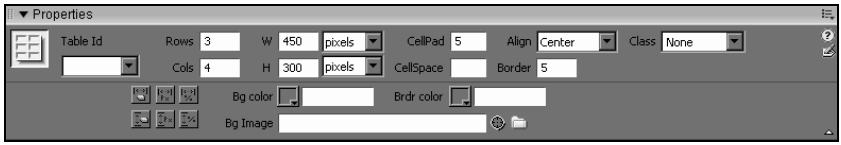


Рис. 4.8. Внешний вид панели **Properties** после выделения таблицы

Значения полей **W** и **H** устанавливают значения параметров WIDTH и HEIGHT (ширина и высота таблицы) в пикселях или процентах.

В поле **CellPad** можно задать расстояние между границами ячейки и ее содержимым, а в поле **CellSpace** — расстояние между ячейками.

Список **Align** позволяет выбрать выравнивание по горизонтали для всей таблицы относительно страницы, в которой она находится. Возможен выбор следующих значений: default — выравнивание по умолчанию, left — по левому краю, center — по центру, right — по правому краю.

Поля **Border**, **Bg Color**, **Brdr Color** предназначены для задания толщины рамки, цвета фона и цвета рамки таблицы, причем выбор цвета можно осуществить из раскрывающейся палитры, нажав кнопку перед соответствующим полем.

Поле **Bg Image** предназначено для задания имени файла фонового рисунка таблицы. Щелкнув по кнопке **Browse for File** (Поиск файла) справа от поля, можно открыть диалоговое окно **Select Image Source** (Выбор источника изображения) и выбрать файл рисунка.

Слева от полей задания фона располагаются три пары кнопок. Первая пара, включающая кнопки **Clear Column Widths** (Удалить ширину столбцов) и **Clear Row Heights** (Удалить высоту строк), предназначена для удаления параметров WIDTH и HEIGHT по всей таблице, включая отдельные ячейки. Кроме этих кнопок можно

воспользоваться командами **Clear Cell Widths** (Удалить ширину ячеек) и **Clear Cell Heights** (Удалить высоту ячеек) меню **Modify | Table** (Изменить | Таблица), действия которых аналогичны. Вторая пара содержит кнопки **Convert Table Widths to Pixels** (Преобразовать ширину таблицы в пиксельное представление) и **Convert Table Heights to Pixels** (Преобразовать высоту таблицы в пиксельное представление). Эти кнопки позволяют преобразовать ширину и высоту как всей таблицы, так и отдельных ячеек из процентного представления в пиксельное представление. При этих преобразованиях в параметрах `WIDTH` и `HEIGHT` не только меняются единицы измерения, но и пересчитываются значения таким образом, чтобы размеры таблицы остались неизменными. Третья пара кнопок **Convert Table Widths to Percent** (Преобразовать ширину таблицы в процентное представление) и **Convert Table Heights to Percent** (Преобразовать высоту таблицы в процентное представление) осуществляет обратное преобразование ширины и высоты из пиксельного представления в процентное. Для этих же целей можно использовать команды меню **Modify | Table** (Изменить | Таблица):

- **Convert Widths to Pixels** — преобразовать ширину в пиксельное представление;
- **Convert Heights to Pixels** — преобразовать высоту в пиксельное представление;
- **Convert Widths to Percent** — преобразовать ширину в процентное представление;
- **Convert Heights to Percent** — преобразовать высоту в процентное представление.

Поле **Table Id** позволяет ввести индивидуальное имя метки, в данном случае метки `<TABLE>`. Введенное в поле значение будет присвоено параметру `ID`, который появится в метке `<TABLE>`. Индивидуальные имена меток используются при создании динамических страниц, внешний вид которых меняется в зависимости от действий пользователя. Используя индивидуальное имя, программа-сценарий обращается к метке, изменяя ее свойства. Подробно эти вопросы будут рассмотрены в *главе 8*.

Список **Class** содержит перечень имен каскадных таблиц стилей (CSS), из которого можно выбрать нужный стиль форматирования таблицы, после чего имя стиля будет присвоено параметру

Class и таблица изменит свой внешний вид. Вопросы создания и использования каскадных таблиц стилей рассмотрены в *главе 7*.

4.2.3. Изменение свойств ячеек таблицы

Изменение свойств ячеек таблицы осуществляется с помощью инструментов панели **Properties** (Свойства), которые появятся после установки курсора в любую ячейку таблицы или выделения ячеек. Для выделения ячеек существует несколько способов.

- Нажать клавишу <Ctrl> и щелкнуть мышкой внутри ячейки. Не отпуская <Ctrl>, можно выделить несколько несмежных ячеек.
- Для выделения прямоугольного фрагмента таблицы курсор установить в ячейку, которая будет левой верхней в выделяемом фрагменте, нажать клавишу <Shift> и щелкнуть по правой нижней ячейке выделяемого фрагмента.
- Для выделения столбца необходимо указатель мыши подвести к верхней границе столбца таким образом, чтобы он принял вид черной стрелки, и произвести щелчок мышью. Для выделения нескольких столбцов указатель следует переместить вдоль верхней границы выделяемых столбцов, не отпуская левую кнопку мыши.
- Для выделения строки указатель подвести к левой ее границе (при этом он должен принять вид черной стрелки) и щелкнуть мышью. Выделение нескольких строк осуществляется путем перемещения указателя мыши вдоль левой границы выделяемых строк при нажатой кнопке мыши.
- Выделение нескольких смежных ячеек можно осуществить путем протаскивания курсора по выделяемым ячейкам при нажатой левой кнопке мыши.

Панель **Properties** (Свойства) в режиме изменения свойств ячеек показана на рис. 4.9.

В верхней половине панели **Properties** (Свойства) содержатся уже известные инструменты форматирования текста, а в нижней — инструменты изменения свойств ячеек таблицы. Если нижняя половина не видна, то нужно щелкнуть по треугольной стрелке в правом нижнем углу панели **Properties** (Свойства).

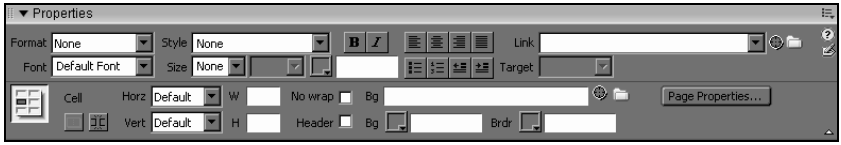


Рис. 4.9. Панель **Properties** в режиме изменения свойств ячеек

Разделение ячейки таблицы осуществляется с помощью кнопки **Splits cell into rows or columns** (Разделить ячейку на строки или столбцы), которая находится под словом **Cell** (Ячейка) в левой части панели. Для разделения ячейки достаточно поместить в нее указатель и нажать кнопку (при выделении нескольких ячеек кнопка не активна). Откроется диалоговое окно **Split Cell** (Разделить ячейку), позволяющее выбрать способ разделения ячейки по строкам (Rows) или столбцам (Columns), а также указать их количество (рис. 4.10).

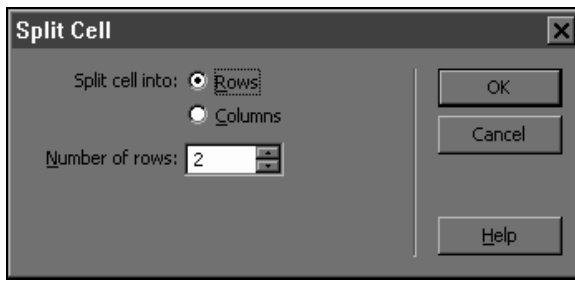


Рис. 4.10. Диалоговое окно **Split Cell**

В меню **Modify** | **Table** (Изменить | Таблица) для этих же целей служит команда **Split Cell** (Разделить ячейку).

Объединение выделенных смежных ячеек осуществляется с помощью кнопки **Merges selected cells using spans** (Объединение выделенных смежных ячеек) из той же группы кнопок. В меню **Modify** | **Table** (Изменить | Таблица) для объединения выделенных ячеек можно воспользоваться командой **Merge Cells** (Объединение ячеек).

Выравнивание содержимого ячеек по горизонтали и вертикали осуществляется путем выбора соответствующего значения из списка **Horz** (Выравнивание по горизонтали) и **Vert** (Выравни-

вание по вертикали). Из списка **Horz** можно выбрать: Left — по левому краю; Center — по центру; Right — по правому краю. Список **Vert** содержит следующие значения: Top — по верхнему краю; Middle — по середине; Bottom — по нижнему краю; Baseline — по базовой линии. Значение Default в обоих списках задает выравнивание по умолчанию.

Поля **W** и **H** предназначены соответственно для задания ширины и высоты ячеек. Значения могут задаваться в процентах относительно ширины всей таблицы или в пикселях.

Флажок **No wrap** запрещает разбиение текста на строки. При установленном флажке в метку `<TD>` соответствующей ячейки будет вставлен параметр `NOWRAP`. При этом текст в ячейке вытянется в одну строку, игнорируя заданные размеры ячейки и всей таблицы.

При установке флажка **Header** (Заголовок) ячейка становится заголовком, т. е. метки `<TD>...</TD>` заменяются метками `<TH>...</TH>`. Текст в такой ячейке центрируется по горизонтали и отображается полужирным шрифтом.

Поля **Bg** предназначены для задания фона ячейки. Верхнее поле **Bg** позволяет задать имя файла фонового рисунка ячейки, а нижнее поле — однородный цвет фона.

Поле **Brdr** служит для задания цвета рамки вокруг ячейки.

4.2.4. Создание таблицы в режиме компоновки страницы

Как уже отмечалось, кроме традиционного `использования` таблиц для вывода регулярных данных, они широко применяются для размещения на странице различных объектов (фрагментов текста, рисунков) в произвольном порядке, предусмотренном дизайном страницы, что было бы невозможно сделать другими средствами языка HTML. Программа Dreamweaver предоставляет для этой цели большие возможности. Для перехода в режим компоновки страницы необходимо нажать кнопку **Layout** (Компоновка) в группе инструментов **Layout** (Компоновка) панели **Insert** (Вставка), после чего становятся активными кнопки **Layout Table** (Компоновка таблицы) и **Draw Layout Cell** (Рисование ячейки) справа от кнопки **Layout** (Компоновка) (рис. 4.11).

Рис. 4.11. Группа инструментов **Layout** панели **Insert**

Нажав на кнопку — **Layout Table** (Компоновка таблицы), можно растянуть прямоугольную рамку контура, внутри которого будут размещаться элементы таблицы. Если не выполнить данные действия, то для размещения объектов станет доступно все рабочее поле программы.

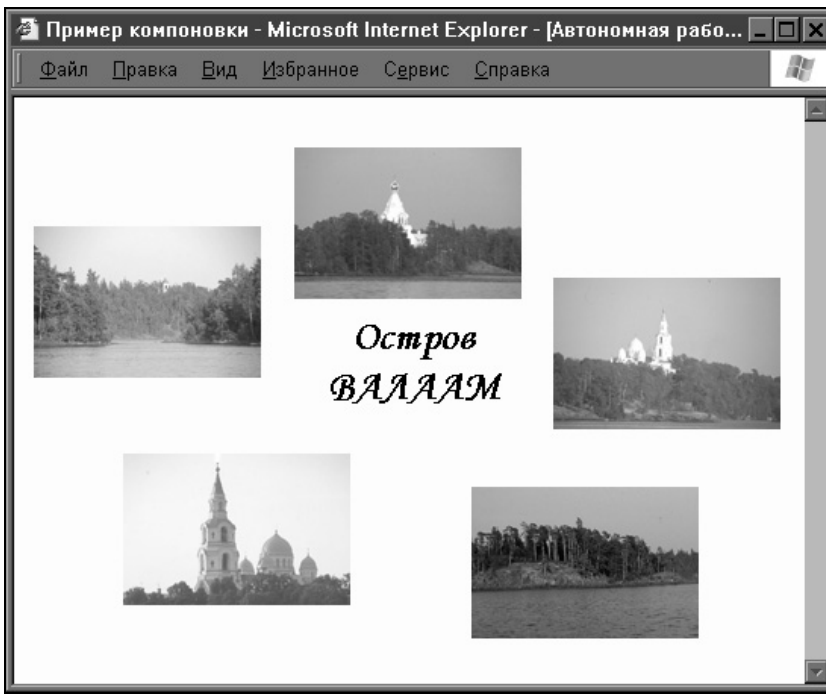


Рис. 4.12. Размещение объектов на странице в произвольных местах с помощью таблицы

После нажатия кнопки — **Draw Layout Cell** (Рисование ячейки) нужно растянуть прямоугольную рамку вокруг области, в которую мы хотим поместить какой-либо объект. Внутри созданной области активизируется мерцающий курсор, предлагающий приступить к набору текста или вставке рисунка. Подведя

указатель мыши к границе созданной области, нажав и удерживая кнопку мыши, ее можно переместить в другое место. Использование появившихся маркеров позволяет изменить размер выбранной области.

Рассмотрим компоновку страницы на следующем примере. Предположим, что размер страницы 500 × 350 пикселей. На странице предполагается разместить 5 рисунков размером 150 × 100 пикселей и текстовый заголовок. В окне браузера страница должна выглядеть как на рис. 4.12.

Начнем с построения рамки, ограничивающей размер страницы. Нажмем на кнопку **Layout Table** (Компоновка таблицы) и в рабочем поле растянем рамку нужного размера. При построении размер рамки указывается в строке состояния. Если после построения требуется изменить размер, то это легко сделать с помощью панели **Properties** (Свойства), которая в этом случае приобретает вид, приведенный на рис. 4.13.

Рис. 4.13. Панель **Properties** в режиме изменения свойств таблицы, используемой для компоновки

Кроме изменения размеров, с помощью панели можно задать цвет фона (палитра **Bg**), изменить расстояние между границами ячейки и ее содержимым (**CellPad**) и расстояние между ячейками (**CellSpace**). В режиме компоновки поля **CellPad** и **CellSpace** обычно имеют нулевые значения.

Нажав кнопку **Draw Layout Cell** (Рисование ячейки), растянем шесть прямоугольных рамок (одну для текста и пять для рисунков). На этом этапе расположение рамок большого значения не имеет. Однако следует обратить внимание, чтобы размеры рамок для рисунков были немного меньше размеров рисунка. В дальнейшем, после вставки рисунков и перемещении их на заданные места, размеры рамок ячеек таблицы установятся в соответствии с размерами рисунков. Панель **Properties** (Свойства) в случае выделения одной из ячеек приобретает вид как на рис. 4.14.

Рис. 4.14. Панель **Properties** в режиме изменения свойств ячеек таблицы, используемой для компоновки

Поля **Width** и **Height** позволяют изменить размеры ячейки, поле **Bg** с кнопкой палитры — цвет фона ячейки, списки **Horz** и **Vert** — выбрать способ выравнивания содержимого ячейки по горизонтали и вертикали. Установка флажка **No wrap** запрещает разбиение текста на строки.

Результат размещения ячеек таблицы показан на рис. 4.15.

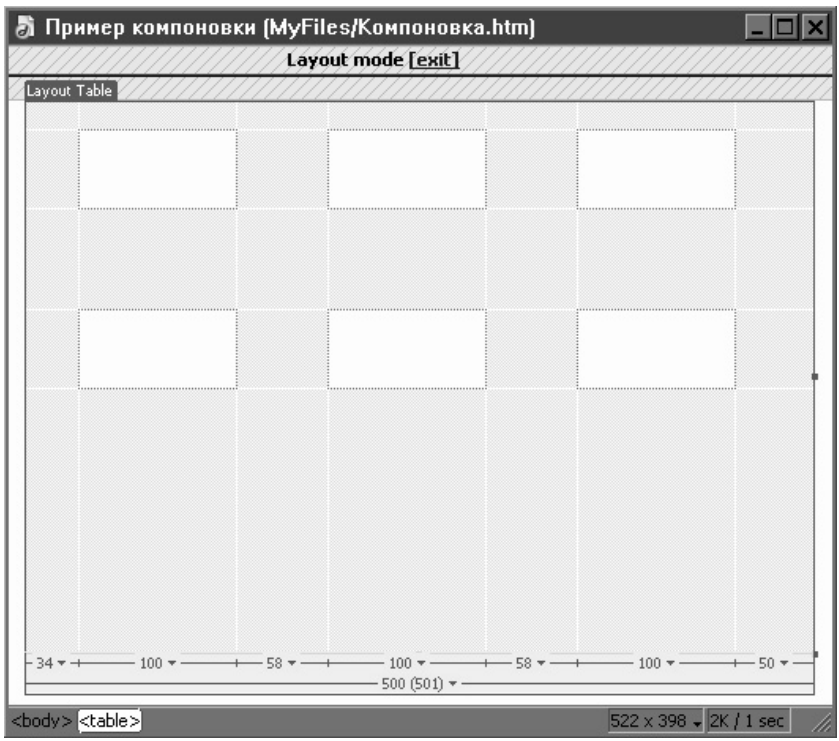


Рис. 4.15. Результат создания таблицы с ячейками для размещения объектов страницы

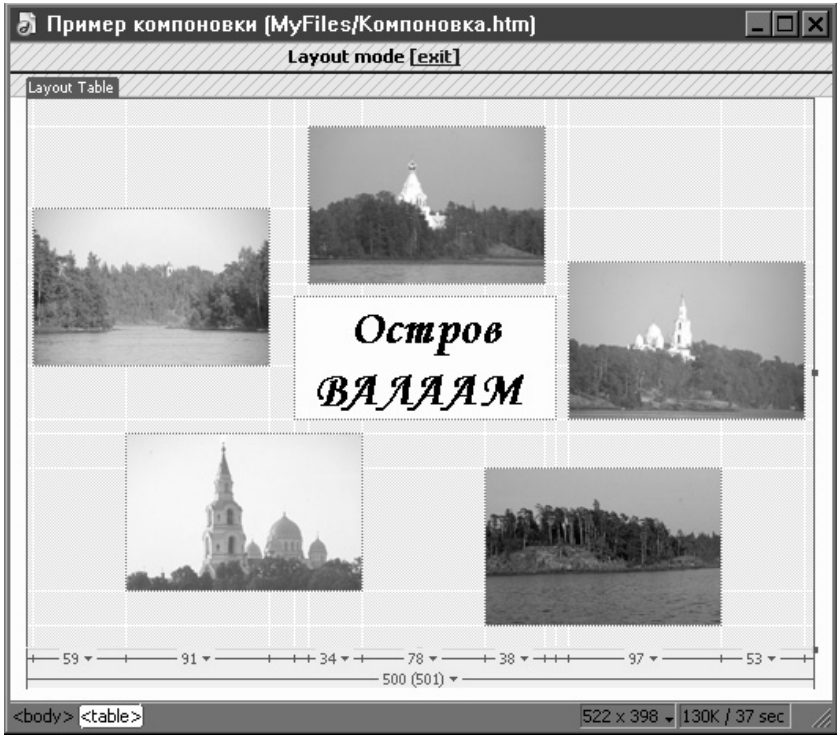


Рис. 4.16. Результат компоновки страницы с использованием группы инструментов **Layout**

Теперь можно приступать к заполнению ячеек. В любые 5 ячеек вставим рисунки, предварительно устанавливая курсор в соответствующую ячейку и используя команду меню **Insert | Image** (Вставка | Рисунок). В последнюю ячейку введем текст. Выделим текст и на панели **Properties** (Свойства) зададим гарнитуру шрифта Monotype Corsiva (список **Font**), формат шрифта Heading 1 (список **Format**). Выделим рамку вокруг ячейки, изменим ее размеры таким образом, чтобы текст располагался в две строки и размеры ячейки не были бы слишком велики. При необходимости можно осуществить центрирование текста по горизонтали и вертикали, выбрав при выделенной рамке из списков **Hors** и **Vert** панели **Properties** (Свойства) значения Center и Middle. После заполнения ячеек общий размер таблицы может из-

мениться, поэтому перед началом перемещения ячеек необходимо выделить рамку вокруг таблицы и на панели **Properties** (Свойства) изменить размеры, если в этом есть необходимость. Для перемещения ячеек можно использовать не только мышь, но и клавиши управления курсором (если ячейка выделена).

Результат компоновки показан на рис. 4.16. Как видим, программа Dreamweaver создала множество пустых ячеек, назначение которых расположить заполненные ячейки в нужных местах страницы. Для создания такой таблицы вручную потребовалось бы много времени.

Глава 5

Компоновка страниц с использованием фреймов

5.1. Создание фреймов средствами HTML

Слово "frame" в переводе с английского обозначает рамку. Однако *фрейм* это не просто рамка, которую можно провести вокруг рисунка, таблицы или фрагмента текста. Фрейм — это заданного размера окно документа прямоугольной формы, входящее в состав окна браузера. В окно браузера могут входить несколько фреймов, в каждом из которых будет находиться самостоятельный документ.

5.1.1. Деление окна браузера на фреймы

Чтобы разделить окно браузера на фреймы, нужно создать специальный документ, в котором необходимо указать параметры фреймов и имена файлов HTML-документов, загружаемых в каждый фрейм в момент открытия документа в окне браузера. Деление на фреймы осуществляется с помощью метки `<FRAMESET>…</FRAMESET>`, а описание свойств содержимого конкретных фреймов — с помощью метки `<FRAME>`. Основными параметрами метки `<FRAMESET>` являются:

- `ROWS="список значений"` — деление на горизонтальные фреймы;

- COLS="список значений" — деление на вертикальные фреймы.

Список значений определяет число фреймов и их размеры. Запись значений производится через запятую. Размеры фреймов могут задаваться в пикселях, процентах или относительных единицах, причем в одном списке можно использовать одновременно все три варианта обозначений. Если размер фрейма задается в процентах, то справа от числа записывается %, в относительных единицах — знак *, для пикселей обозначение отсутствует.

Например:

- <FRAMESET ROWS="30%,40%,30%"> — деление на три горизонтальных фрейма. Первый фрейм имеет высоту 30 % от высоты окна браузера, второй — 40 % и третий — 30 %;
- <FRAMESET COLS="*,6*,2*"> — деление на три вертикальных фрейма. Ширина фреймов задана в относительных единицах. На первый фрейм приходится одна часть ширины окна браузера (отсутствие числа слева от знака * означает число 1), на второй — 6 частей и на третий — 2 части. Таким образом, ширину окна браузера нужно разделить на 9 равных частей и каждый фрейм будет занимать предписанное ему количество этих частей;
- <FRAMESET ROWS="30,*"> — деление на два горизонтальных фрейма, один из которых имеет высоту 30 пикселей, другой занимает оставшуюся часть окна браузера, о чем свидетельствует знак *;
- <FRAMESET COLS="25%,30%,*"> — деление на три вертикальных фрейма, первый из которых имеет ширину 25 % от ширины окна браузера, второй — 30 %, третий занимает оставшуюся часть окна;
- <FRAMESET ROWS="20%,*" COLS="30%,35%,*"> — деление на шесть фреймов. Результат деления на 6 фреймов показан на рис. 5.1.

Примечание

Если будут заданы несколько значений в пикселях и в списке не будет использован знак *, то это равносильно заданию размеров

фреймов в относительных единицах. Например: `<FRAMESET ROWS= "80,160,240">` — деление на три горизонтальных фрейма. Так как фреймы должны занять по высоте все окно браузера, оно будет распределено между ними пропорционально заданным величинам, что в данном случае эквивалентно записи `"*,2*,3*"`. То же самое произойдет, если размеры фреймов будут заданы в процентах, но сумма превысит 100 %. Например, запись `"50%,100%,100%"` эквивалентна записи `"*,2*,2*"`.

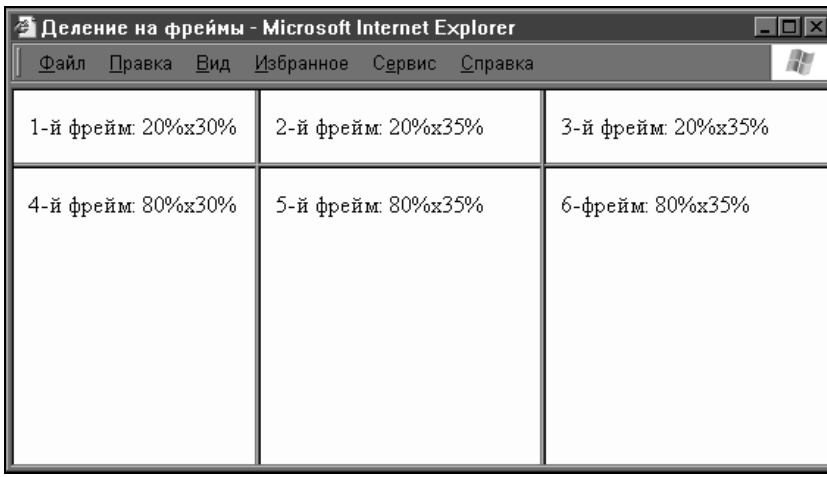


Рис. 5.1. Окно браузера разделено на 6 фреймов

5.1.2. Описание фреймов

Каждый полученный в результате деления фрейм должен быть описан с помощью метки `<FRAME>`. К основным параметрам метки `<FRAME>` можно отнести:

- `SRC="имя файла"` — задается имя файла HTML-документа, который откроется в данном фрейме в момент начальной загрузки;
- `NAME="имя фрейма"` — задается имя фрейма, которое в дальнейшем может быть использовано для загрузки в этот фрейм других документов.

Примечание

При создании документа с фреймовой структурой метка <BODY>...</BODY> не используется.

В качестве примера рассмотрим одну из популярных фреймовых структур, приведенную на рис. 5.2.

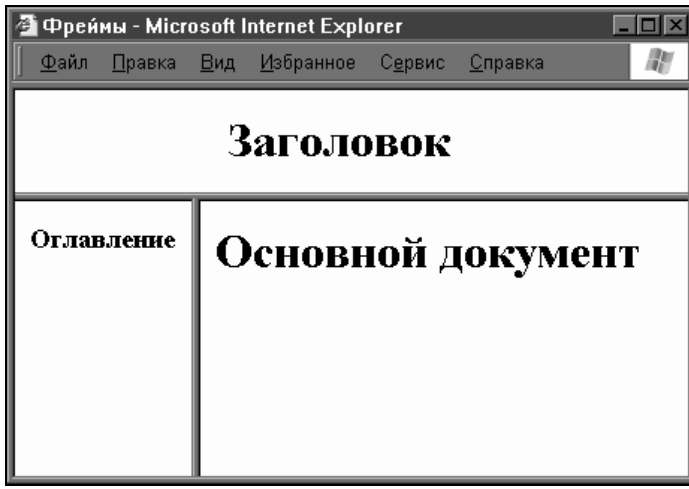


Рис. 5.2. Документ с фреймовой структурой. В каждый из трех фреймов загружен самостоятельный документ

В верхний фрейм загружается файл с заголовком, который может служить заголовком всего сайта, включающего несколько документов. Например, в качестве заголовка может использоваться название фирмы, владеющей сайтом, название учебного заведения, памятника культуры, географического объекта и т. д. В левый фрейм загружается документ с оглавлением, в котором перечислены названия всех документов, входящих в сайт. По каждому пункту оглавления может осуществляться ссылка на соответствующий документ, который будет загружаться в правый (главный) фрейм. Содержимое главного фрейма в процессе работы с оглавлением будет постоянно обновляться. Однако для начальной загрузки для главного фрейма должен существовать документ, который в нашем примере назван "Основным документом".

Текст документа с показанной на рис. 5.2 фреймовой структурой приведен в листинге 5.1.

Листинг 5.1. Текст документа с фреймовой структурой

```
<html>
<head>
<title>Фреймы</title>
</head>
<frameset rows="70,*">
  <frame src="TopDoc.htm" name="topFrame">
  <frameset cols="120,*">
    <frame src="LeftDoc.htm" name="leftFrame">
    <frame src="MainDoc.htm" name="mainFrame">
  </frameset>
</frameset>
<noframes>
<body>
Ваш браузер не поддерживает работу с фреймами
</body>
</noframes>
</html>
```

Первая метка `<FRAMSET>` делит окно браузера на два горизонтальных фрейма, верхний из которых имеет высоту 70 пикселей, а нижний занимает остальную часть окна. Так как мы не предполагаем делить верхний фрейм на другие фреймы, следом за меткой `<FRAMESET>` идет метка `<FRAME>`, с помощью которой задаются его параметры. Нижний фрейм делится на два вертикальных фрейма меткой `<FRAMESET>`, из которых левый имеет ширину 120 пикселей, а правый занимает остальную часть нижнего фрейма. Только после этого можно задать параметры созданных вертикальных фреймов, что и делается с помощью идущих друг за другом меток `<FRAME>`. Далее последовательно закрываются метки `<FRAMESET>`.

Завершается документ парой меток `<NOFRAMES>...</NOFRAMES>`, между которыми размещаются традиционные для обычных до-

кументов метки <BODY>…</BODY> с содержимым, которое будет воспроизводиться только браузерами, не поддерживающими фреймы. С каждым днем все труднее представить себе, что такие существуют в природе, поэтому размещение в документе с фреймовой структурой этих меток скорее дань давним традициям, чем практическая необходимость.

Остается выяснить, что представляют собой документы, загружаемые в каждый из трех фреймов: TopDoc.htm, LeftDoc.htm и MainDoc.htm. В нашем случае они чрезвычайно просты, так как предназначены для вывода небольших фраз. Тексты документов приведены в листингах 5.2, 5.3 и 5.4.

Листинг 5.2. Текст документа, загружаемого в верхний фрейм

```
<html>
<head>
<title>Заголовок</title>
<body>
<h1 align="center">Заголовок</h1>
</body>
</html>
```

Листинг 5.3. Текст документа, загружаемого в левый фрейм

```
<html>
<head>
<title>Оглавление</title>
</head>
<body>
<strong>Оглавление</strong>
</body>
</html>
```

Листинг 5.4. Текст документа, загружаемого в правый фрейм

```
<html>
<title>Основной документ</title>
```

```
</head>
<body>
<h1 align="center">Основной документ</h1>
</body>
</html>
```

Как видим, это обычные HTML-документы, которые хранятся в виде отдельных файлов. Чтобы в параметрах SRC меток <FRAME> можно было указать только имя файла, все документы должны находиться в одной папке.

5.1.3. Дополнительные параметры меток <FRAMESET> и <FRAME>

Дополнительные параметры метки <FRAMESET>:

- FRAMEBORDER="значение" — задает отображение рамок вокруг фреймов, описанных внутри данной пары меток <FRAMESET>…</FRAMESET>. Может принимать значения: "Yes" или "1" — отображать рамки, "No" или "0" — не отображать рамки;
- FRAMESPACING="значение" — позволяет задать расстояние между смежными фреймами в пикселях, что практически эквивалентно изменению толщины рамки;
- BORDERCOLOR="значение" — предназначен для изменения цвета рамки вокруг фрейма. В браузере MS Internet Explorer цвет рамки не меняется, но меняется цвет межрамочного пространства;
- BORDER="значение" — предназначен для изменения толщины рамки вокруг фрейма. Браузер MS Internet Explorer не поддерживает изменение толщины рамки, но при значениях "5" и более изменяется расстояние между рамками соседних фреймов, что может восприниматься как утолщение рамки.
- Дополнительные параметры метки <FRAME>:
- FRAMEBORDER="значение" — задает отображение рамок вокруг фрейма. Может принимать значения: "Yes" или "1" — отображать рамки, "No" или "0" — не отображать рамки. Значение параметра FRAMEBORDER, заданное в метке <FRAME>, явля-

ется более приоритетным по отношению к значениям этого же параметра, заданным в вышестоящих метках <FRAMESET>. Иначе говоря, если в метке <FRAMESET> параметр FRAMEBORDER=1, в метке <FRAME> FRAMEBORDER=0, то вокруг этого фрейма рамка отображаться не будет;

- SCROLLING="значение" — задает отображение полос прокрутки. Может принимать значения: "Yes" — всегда отображать полосы прокрутки; "No" — никогда не отображать полосы прокрутки; "Auto" — отображать полосы прокрутки в случае необходимости, когда документ не помещается во фрейм целиком (действует по умолчанию);
- NORESIZE — запрещает изменение размеров фрейма пользователем;
- MARGINHEIGHT="значение" — позволяет задать размеры полей от границ фрейма до содержимого документа, загруженного во фрейм, сверху и снизу;
- MARGINWIDTH="значение" — позволяет задать размеры полей от границ фрейма до содержимого документа, загруженного во фрейм, слева и справа.

5.1.4. Ссылки из документов, находящихся во фреймах

Если в каком-либо документе находится ссылка, то возникает вопрос, куда будет загружен документ, на который осуществляется ссылка? По умолчанию документ загружается в тот же фрейм. Однако чаще всего требуется загрузить документ в другой фрейм, полное или новое окно. Во всех перечисленных случаях в метке <A> нужно использовать параметр TARGET. Чтобы загрузить документ в определенный фрейм необходимо в качестве значения параметра TARGET использовать имя этого фрейма, указанное в соответствующей метке <FRAME>. Например:

```
<A HREF=doc1.htm TARGET=mainFrame>Первый документ</A>
```

Независимо от того, в каком фрейме находится документ с такой ссылкой, документ doc1.htm загрузится во фрейм с именем mainFrame. Кроме того, параметру TARGET можно присваивать значения: "_blank" — документ, на который осуществляется

ссылка, откроется в новом окне браузера; "_self" — документ откроется в том же фрейме, что и исходный файл (по умолчанию); "_parent" — документ откроется в родительском фрейме. Фрейм будет родительским по отношению к загружаемому в него документу с фреймовой структурой. Такая сложная конструкция используется редко, поэтому в большинстве случаев значение "_parent" эквивалентно значению "_top"; _"top" — документ займет все окно браузера, например:

```
<A HREF=doc2.htm TARGET=_top>Второй документ</A>
```

Независимо от фрейма, в котором располагается документ с данной ссылкой, документ doc2.htm откроется в полном окне.

5.2. Создание фреймов в программе Dreamweaver

5.2.1. Создание нового документа с фреймовой структурой

Если работа с программой Dreamweaver начинается со стартовой страницы, то в разделе **Create New** (Создать новый документ) следует нажать кнопку **More** (Больше) в нижней части раздела и открыть диалоговое окно **New Document** (Новый документ).

При отсутствии стартовой страницы для открытия диалогового окна **New Document** (Новый документ) можно использовать команду меню **File | New** (Файл | Создать), которая дублируется кнопкой **New** (Создать) на панели **Standard** (Стандартная). Для этих же целей можно использовать сочетание клавиш <Ctrl>+<N>.

В диалоговом окне **New Document** (Новый документ) на вкладке **General** (Общая) в списке **Category** (Категория) нужно выбрать **Framesets** (Фреймы). Правее списка **Category** (Категория) откроется список **Framesets** (Фреймы), в котором выбирается нужная структура фреймов, отображаемая в окне **Preview** (Предварительный просмотр). Если нужную структуру найти не удалось, то необходимо выбрать наиболее подходящую структуру, а требуемые изменения можно будет произвести позже. После нажа-

тия кнопки **Create** (Создать) осуществляется переход в рабочее окно для продолжения работы над документом.

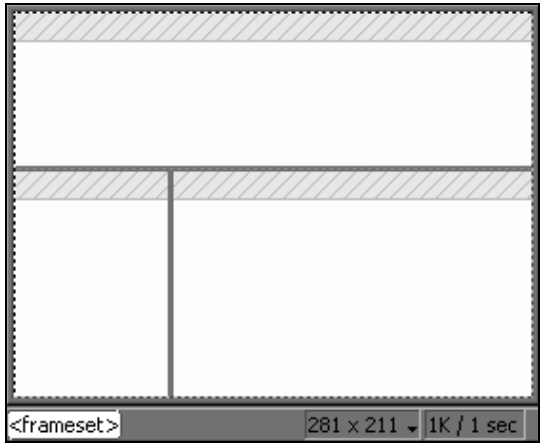
5.2.2. Создание документа с фреймовой структурой на базе имеющегося HTML-документа

В этом разделе будет решаться следующая задача. Имеется уже созданный документ с обычной структурой. Требуется создать документ с фреймовой структурой таким образом, чтобы имеющийся документ стал содержимым одного из фреймов. Для этого необходимо открыть документ в рабочем окне программы и в группе инструментов **Layout** (Компоновка) панели **Insert** (Вставка) нажать кнопку **Frames** (Фреймы) —  и из списка структур фреймов выбрать наиболее подходящую. Открытый в окне документ будет помещен в тот фрейм, который закрашен.

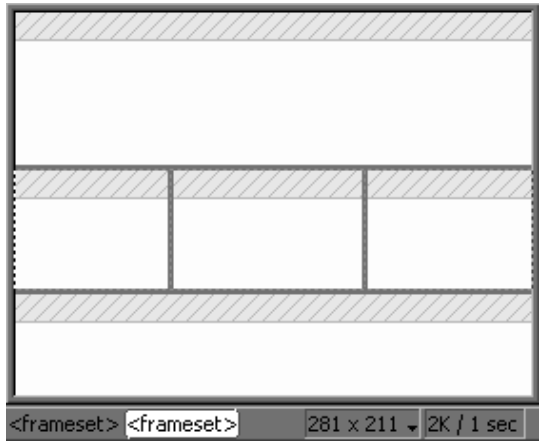
5.2.3. Изменение фреймовой структуры документа

Если при создании документа не удалось подобрать нужную структуру, то, выбрав наиболее подходящую, можно в рабочем окне внести необходимые изменения. Предположим, что нами выбрана структура фрейма, показанная на рис. 5.3, *a*, а на рис. 5.3, *b* показана структура, к которой мы хотим перейти.

Наиболее простой и эффективный способ изменения структуры документа с фреймами заключается в перетаскивании границы фрейма мышкой. Начнем с добавления еще одного горизонтального фрейма. Установим указатель мыши на горизонтальную линию, разделяющую фреймы, и при появлении двунаправленной стрелки щелкнем мышкой. Вокруг документа появится пунктирная линия, а в селекторе меток строки состояния выделится метка `<FRAMESET>`. Поместим курсор на нижнюю границу документа и с появлением двунаправленной стрелки перетащим горизонтальную границу нового фрейма. Перейдя в режим **Code** (Код), можно убедится, что в списке значений параметра `ROWS` метки `<FRAMESET>` появилась третья запись.



a)



b)

Рис. 5.3. Фреймовые структуры:
a — выбранная из числа существующих,
b — желаемая структура

Для создания еще одного вертикального фрейма щелкнем по одной из вертикальных разделительных линий, после чего выделится пунктирной линией группа вертикальных фреймов, а в селекторе меток строки состояния выделится вторая метка `<FRAMESET>`. Теперь поместим указатель мыши на правую грани-

цу вертикального фрейма и перетащим границу нового фрейма. Удалить ненужную границу фрейма можно, перетащив ее в обратном направлении за пределы документа.

Мы рассмотрели наиболее простые варианты, когда новые фреймы добавлялись к уже существующим фреймам, становясь крайними. Как же быть, если нужно разделить фрейм, находящийся в окружении других фреймов (например средний фрейм). Пусть необходимо разделить его на два горизонтальных фрейма. Фрейм должен быть предварительно выделен, для чего нужно щелкнуть по нему, удерживая клавишу <Alt>. В селекторе меток строки состояния выделится метка `<FRAME>`. Это означает, что теперь необходимо добавить новую метку `<FRAMESET>`. Для этого подведем указатель мыши к верхней или нижней границе фрейма и, удерживая клавишу <Alt>, перетащим новую границу. Используя описанные приемы, можно создать структуру документа любой сложности.

Существуют и другие способы изменения фреймовой структуры документа.

- Установив курсор внутри фрейма, можно воспользоваться командой меню **Modify | Frameset** (Изменить | Фрейм) и из открывшегося меню выбрать одну из команд: **Split Frame Left** (Добавить фрейм слева), **Split Frame Right** (Добавить фрейм справа), **Split Frame Up** (Добавить фрейм сверху), **Split Frame Down** (Добавить фрейм снизу).
- Установив курсор внутри фрейма, в группе инструментов **Layout** (Компоновка) панели **Insert** (Вставка) нажать кнопку **Frames** (Фреймы) —  и выбрать одну из структур фреймов. Если в этом фрейме уже создан документ, то он будет помещен в тот фрейм, который на кнопке показан закрашенным.

Оба перечисленных способа позволяют создать новые фреймы, используя еще одну метку `<FRAMESET>`, что нельзя считать оптимальным.

5.2.4. Изменение параметров документа с фреймовой структурой

Предположим, что документ с фреймовой структурой уже создан. Дальнейшая задача заключается в том, чтобы, не изменяя полученной структуры, изменить свойства отдельных фреймов.

Изменение параметров фреймов осуществляется с помощью панели **Properties** (Свойства), что приведет к изменению параметров меток `<FRAMESET>`. Внешний вид документа в рабочем окне программы Dreamweaver показан на рис. 5.4. Фрагмент текста документа, определяющий его структуру, приведен в листинге 5.5.

Листинг 5.5. Фрагмент документа с фреймовой структурой

```
<frameset rows="70,*">
 <frame src="TopDoc.htm" name="topFrame">
 <frameset cols="200,200,*">
  <frame src="Doc1.htm" name="Frame1">
  <frame src="Doc2.htm" name="Frame2">
  <frame src="Doc3.htm" name="Frame3">
 </frameset>
</frameset>
```

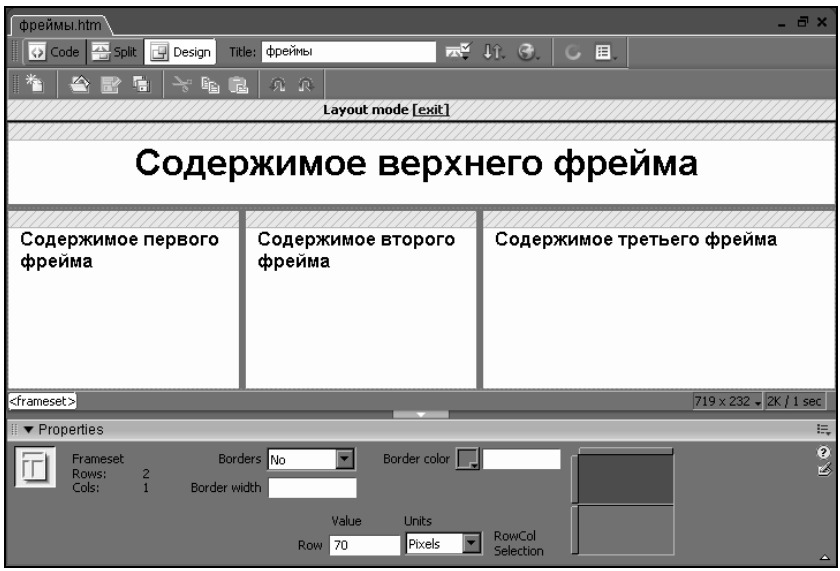


Рис. 5.4. Внешний вид документа в рабочем окне программы Dreamweaver и панели **Properties** со свойствами верхнего фрейма

Чтобы внести изменения в деление документа на горизонтальные фреймы, а следовательно, в первую метку <FRAMESET>, нужно в рабочем окне щелкнуть по горизонтальной разделительной линии. При этом панель свойств **Properties** (Свойства) примет вид как на рис. 5.4. Обратим внимание на то, что в селекторе меток строки состояния выделилась метка <FRAMESET>. В правой части панели свойств представлена схема деления, из которой видно, что деление осуществлено на два фрейма, из которых активным является верхний фрейм. Это значит, что все изменения параметров будут относиться именно к этому фрейму.

В разделе **Row** (Строка) находятся поле **Value** (Значение) и список **Units** (Единицы измерения). Из списка **Units** (Единицы измерения) нужно выбрать единицу измерения высоты фрейма: Pixels — пиксели, Percent — проценты или Relative — относительные единицы. В поле **Row** (Строка) ввести ее значение. В нашем случае нужно установить высоту фрейма 70 пикселей.

В списке **Borders** (Рамки) можно выбрать одно из трех значений: Default — по умолчанию (параметр FRAMEBORDER не устанавливается), Yes — отображать рамки вокруг фреймов, No — не отображать рамки.

В поле **Border width** вводится толщина рамки, а в поле **Border color** — ее цвет.

Перейдем к изменению параметров второго горизонтального фрейма. Чтобы сделать его активным, нужно в схеме деления на фреймы панели свойств **Properties** (Свойства) щелкнуть по нижнему прямоугольнику. Для этого фрейма в списке **Units** (Единицы измерения) необходимо выбрать относительные единицы (Relative), а в поле **Value** (Значение) установить 1. В результате данных изменений вторым значением параметра ROWS станет знак "*", то есть по высоте второй фрейм займет оставшуюся часть окна браузера.

Второй горизонтальный фрейм делится на три вертикальных второй меткой <FRAMESET>. Чтобы перейти к изменению параметров вертикальных фреймов нужно щелкнуть по любой из вертикальных линий, разделяющих эти фреймы. При этом в селекторе меток строки состояния выделится вторая метка

<FRAMESET>, а схема деления на фреймы панели **Properties** (Свойства) изменит свой вид (рис. 5.5).

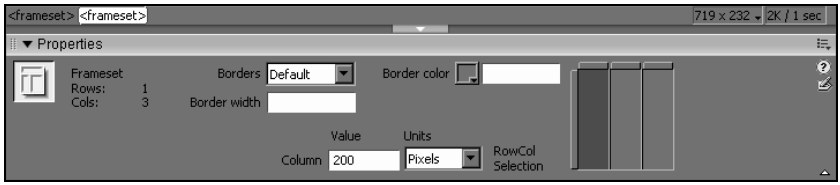


Рис. 5.5. Панель свойств **Properties** со свойствами левого вертикального фрейма

Изменение параметров вертикальных фреймов ничем не отличается от только что описанной процедуры редактирования горизонтальных фреймов.

5.2.5. Изменение содержимого отдельных фреймов

Изменения содержимого отдельных фреймов предполагают изменения параметров меток <FRAME>. Для этого фрейм должен быть предварительно выделен. Выделение осуществляется щелчком внутри фрейма при нажатой клавише <Alt>. Вокруг выделенного фрейма появится пунктирная рамка, а панель **Properties** (Свойства) будет содержать свойства содержимого этого фрейма.

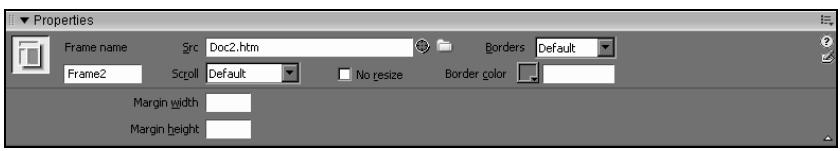


Рис. 5.6. Панель свойств **Properties** со свойствами содержимого фрейма Frame2

Щелчком по треугольной стрелке в правом нижнем углу панели свойств можно скрывать/показывать нижнюю половину панели.

Поле **Frame Name** (Имя фрейма) предназначено для изменения имени фрейма.

В поле **Src** помещается имя файла документа, первоначально загружаемого во фрейм. Необходимый файл позволяет найти диалоговое окно **Select HTML File** (Выбор HTML-файла), которое откроется после щелчка по кнопке **Browse for file** (Поиск файла), расположенной справа от поля **Src**.

Раскрывающийся список **Borders** (Рамки) содержит три варианта управления отображением рамки вокруг фрейма (значения параметра `FRAMEBORDER`):

- Yes — отобразить рамку, отделяющую фрейм от других фреймов;
- No — скрыть рамку;
- Default — по умолчанию.

Примечание

При выборе отображения рамки по умолчанию параметр `FRAMEBORDER` не устанавливается. Установка этого параметра в предшествующей метке `<FRAMESET>` со значением `"No"` приводит к тому, что рамка будет скрыта, в противном случае — отображена.

Раскрывающийся список **Scroll** содержит четыре варианта управления отображением полос прокрутки (значением параметра `SCROLLING`):

- Yes — всегда отображать полосы прокрутки;
- No — никогда не отображать полосы прокрутки;
- Auto — отображать полосы прокрутки в случае необходимости, когда документ не помещается во фрейм целиком;
- Default — по умолчанию (параметр `SCROLLING` не устанавливается, что соответствует установке Auto).

Установка флажка **No Resize** (Не изменять размер) приведет к установке параметра `NORESIZE`, т. е. запрету изменения пользователем размера фрейма в окне браузера.

Поле **Border Color** (Цвет рамки) и кнопка (слева от него), раскрывающая палитру, предназначены для установки цвета рамки (браузером MS Internet Explorer не поддерживается).

Поля **Margin Width** (Ширина поля) и **Margin Height** (Высота поля) позволяют задать расстояния от границ фрейма до содержи-

мого документа, загруженного во фрейм, по горизонтали и вертикали соответственно.

5.2.6. Создание HTML-документа для каждого фрейма

В момент открытия в окне браузера документа с фреймовой структурой в каждом фрейме должен находиться HTML-документ. До сих пор мы подробно рассматривали способы создания документа с фреймовой структурой и изменения его параметров, в том числе говорили о возможности указания имен файлов документов, которые будут открываться в соответствующих фреймах. Предполагалось, что такие документы уже созданы и сохранены в виде отдельных файлов. Однако вовсе не обязательно HTML-документы готовить заранее. Программа Dreamweaver предоставляет возможность создания HTML-документов для каждого фрейма непосредственно в документе с фреймовой структурой. Для этого достаточно щелкнуть по фрейму и с появлением мерцающего курсора приступить к работе.

Работа по созданию, форматированию и редактированию HTML-документа внутри фрейма, практически ничем не отличается от подобной работы в отдельном окне. При необходимости просмотра текста создаваемого документа следует нажать кнопку **Code** (Код) на панели **Document** (Документ). В случае, если возникнет необходимость вернуться к тексту документа с фреймовой структурой, нужно щелкнуть по любой из линий, разделяющих фреймы, и также нажать кнопку **Code** (Код) на панели **Document** (Документ).

5.2.7. Сохранение документа с фреймовой структурой и связанных с ним документов

Сохранение создаваемых документов как обычно осуществляется командами меню **File** (Файл). Особенность работы заключается в том, что если выделен какой-либо фрейм или группа фреймов, то сохраняется документ с фреймовой структурой, для чего используется команда **Save Frameset** (Сохранить документ с фрей-

мами) или **Save Frameset As** (Сохранить документ с фреймами как). Если же курсор находится в одном из фреймов, т. е. ведется работа с HTML-документом, находящимся во фрейме, то будет сохраняться только этот документ, для чего используются команды **Save Frame** (Сохранить документ во фрейме) и **Save Frame As** (Сохранить документ во фрейме как). По команде **Save All** (Сохранить все) при первом сохранении начнется процесс сохранения всех документов, для чего несколько раз (по числу сохраняемых документов) будет открываться диалоговое окно **Save As** (Сохранить как), а сохраняемый документ в рабочем окне будет обводиться рамкой. При повторных сохранениях командой **Save All** (Сохранить все) будут сохранены изменения, внесенные во все документы, без каких-либо предупреждающих сообщений. Команды **Save Frameset** (Сохранить документ с фреймами) и **Save Frame** (Сохранить документ во фрейме) дублируются кнопкой **Save** (Сохранить) на панели **Standard** (Стандартная), а команда **Save All** (Сохранить все) — кнопкой **Save All** (Сохранить все).

Глава 6

Создание интерактивных страниц с использованием форм

Формы предназначены для создания интерактивных страниц, т. е. страниц, позволяющих посетителю сайта отправить информацию его владельцу. С помощью форм можно осуществлять прием заказов, тестирование, проведение опросов и т. д.

6.1. Создание форм средствами HTML

Формы размещаются в документе с помощью метки `<FORM>...</FORM>`. Метка `<FORM>` является контейнером, внутри которого могут размещаться другие метки, предназначенные для создания конкретных элементов.

Метка `<INPUT>` предназначена для размещения внутри формы элементов нескольких типов. Для указания типа выбранного элемента в метке `<INPUT>` используется параметр `TYPE`, который может принимать следующие значения:

- `TYPE=TEXT` — создается однострочное текстовое поле, предназначенное для ввода или вывода информации. В этом случае в метке `<INPUT>` можно использовать дополнительные параметры:
  - `MAXLENGTH="значение"` — максимальное количество символов в строке (по умолчанию не ограничивается);

- SIZE="значение" — максимальное количество отображаемых символов;
- VALUE="значение" — начальное содержимое текстового поля;
- NAME="значение" — имя элемента;

❒ TYPE=PASSWORD — однострочное текстовое поле, аналогичное предыдущему, но предназначенное для ввода пароля. Отличие состоит в том, что все вводимые символы отображаются "*";

❒ TYPE=FILE — поле для ввода имени файла, который необходимо переслать вместе с формой. Справа от строки будет добавлена кнопка **Brows** (Обзор) для поиска и выбора файла. Дополнительные параметры:

- MAXLENGTH="значение" — максимальная длина пути к файлу;
- SIZE="значение" — максимальное количество отображаемых символов;
- NAME="значение" — имя элемента;

❒ TYPE=HIDDEN — скрытое текстовое поле. Содержимое поля не демонстрируется браузером и не может быть изменено пользователем. Оно пересылается на сервер вместе с формой и может быть обработано серверной программой. Дополнительные параметры:

- VALUE="значение" — содержимое поля;
- NAME="значение" — имя элемента;

❒ TYPE=CHECKBOX — поле для установки "флажка". В качестве дополнительных параметров можно использовать:

- VALUE="текст" — текст, который будет передан, если флажок установлен;
- CHECKED — этому параметру не присваивается какого-либо значения, но при его наличии элемент появится в окне браузера с уже установленным флажком;
- NAME="значение" — имя элемента;

❒ TYPE=RADIO — создается элемент переключателя, существующий в составе группы. Для объединения в группу все элементы должны иметь одинаковые имена, которые указываются в параметре NAME. Дополнительные параметры:

- VALUE="текст" — текст, который будет передан, если выбран данный элемент,
- CHECKED — при наличии этого параметра, элемент появится в окне браузера уже выбранным;

❐ TYPE=SUBMIT — кнопка передачи данных. После ее нажатия осуществляется пересылка всех данных формы, а именно имен элементов, заданных в параметрах NAME и текстов, заданных в параметрах VALUE. Дополнительные параметры:

- VALUE="надпись" — позволяет изменить надпись на кнопке,
- NAME="имя" — в случае использования этого параметра имя кнопки будет отправлено вместе с другими данными, но обычно этого не требуется;

❐ TYPE=IMAGE — в качестве кнопки SUBMIT будет использовано графическое изображение, т. е. щелчок по изображению в окне браузера приведет к отправке формы по назначению. Дополнительные параметры:

- SRC="имя" — имя файла рисунка;
- ALT="текст" — текст, появляющийся в виде всплывающей подсказки;

Допускается использование и других параметров метки <IMG> (*см. главу 3*);

❐ TYPE=RESET — кнопка сброса. При ее нажатии отменяются все сделанные установки. Дополнительный параметр VALUE позволяет изменить надпись на кнопке;

❐ TYPE=BUTTON — простая кнопка. Дополнительные параметры:

- VALUE="значение" — надпись на кнопке;
- NAME="значение" — имя кнопки.

Метка <SELECT>…</SELECT> предназначена для создания списка, из которого можно выбрать одно или несколько значений. Параметры метки <SELECT>:

❐ NAME="значение" — имя элемента;

❐ SIZE="значение" — число одновременно видимых элементов. При значении SIZE меньшем, чем число элементов списка, для перемещения по списку будет отображаться полоса прокрутки;

- MULTIPLE — при установке этого параметра из списка можно выделить сразу несколько элементов. В окне браузера элементы, идущие подряд, можно выделить, удерживая клавишу <Shift>, а в произвольном порядке — <Ctrl>.

Элементы списка задаются с помощью метки <OPTION>…</OPTION>, размещаемой внутри метки <SELECT>. В метке <OPTION> можно использовать параметр SELECTED. Такой элемент списка в окне браузера будет отмечен как выбранный. Например:

```
<SELECT SIZE="1" NAME="country" multiple>
<OPTION selected>Испания</OPTION>
<OPTION>Аргентина</OPTION>
<OPTION SELECTED>Марокко</OPTION>
</SELECT>
```

Метка <TEXTAREA>…</TEXTAREA> создает поле для ввода многострочного текста. Параметры метки <TEXTAREA>:

- NAME="имя_элемента";
- ROWS="значение" — число видимых строк;
- COLS="значение" — число видимых символов в строке.

Текст, размещенный между метками <TEXTAREA>…</TEXTAREA>, будет начальным содержимым поля ввода.

Метка <LABEL>…</LABEL> позволяет создать подпись к элементу формы. Текст подписи должен располагаться между открывающей и закрывающей меткой.

Метка <FIELDSET>…</FIELDSET> предназначена для объединения нескольких элементов формы в группу, вокруг которой создается рамка. В разрыв рамки можно вставить название группы, для чего используется метка <LEGEND>…</LEGEND>. Пример в листинге 6.1.

Листинг 6.1. Объединение нескольких элементов в группу

```
<html>
<head>
<title>Объединение в группу</title>
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
```

```
</head>
<body>
<fieldset style="width:150">
<legend>Городской транспорт</legend>
<form name="form1" method="post" action="">
<input type="checkbox" name="ch1" value="автобус">
<label>Автобус</label> <br>
<input type="checkbox" name="ch2" value="трамвай">
<label>Трамвай</label><br>
<input type="checkbox" name="ch3" value="троллейбус">
<label>Троллейбус</label><br>
<input type="checkbox" name="ch4" value="метро">
<label>Метро</label>
</form>
</fieldset>
</body>
</html>
```

Для изменения ширины рамки в метке `<FIELDSET>` используется параметр `STYLE` и свойство каскадной таблицы стилей (CSS) width. Подробно о каскадных таблицах стилей и их свойствах будет рассказано в *главе 7*. Документ в окне браузера будет выглядеть как на рис. 6.1.

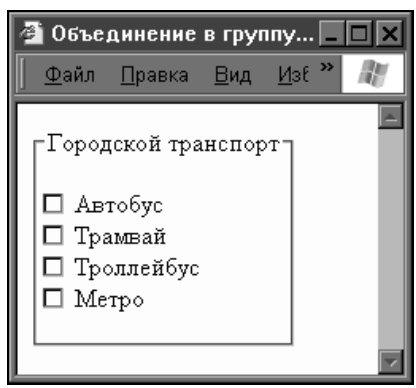


Рис. 6.1. Вид в окне браузера нескольких элементов, объединенных в группу и содержащих название группы

6.2. Пересылка содержимого формы

Существуют различные способы пересылки и обработки содержимого формы. В частности, содержимое формы может быть передано на сервер, где программа-сценарий поместит содержимое полей формы в базу данных. Здесь мы рассмотрим более простой способ использования формы — пересылку ее содержимого по заданному адресу электронной почты. Действия над формой зависят от значений параметров метки `<FORM>`:

- `ACTION="URL"` — позволяет задать адрес получателя данных из формы. Например, если написать `ACTION="mailto:q@qqq"`, то форма будет отправлена по электронному адресу, указанному после "mailto:";
- `NAME="имя формы"` — позволяет задать имя формы;
- `METHOD="метод"` — позволяет задать метод пересылки данных. Возможны два метода пересылки: `GET` (действует по умолчанию) и `POST`. От выбранного метода зависит, в каком месте пересылаемых данных будут находиться данные формы. При пересылке данных по электронной почте необходимо использовать метод `POST`. В этом случае сообщение будет отправлено без открытия почтовой программы (при наличии подключения к сети Интернет), а в теме сообщения будет написано "Форма отправлена из Microsoft Internet Explorer";
- `ENCTYPE="значение"` — определяет формат кодирования данных. Для пересылки данных по почте необходимо задать значение этого параметра `"text/plain"`. В этом случае текст формы будет воспроизведен в окне сообщения почтовой программы.

  Пример формы, отправляемой по почте, приведен в листинге 6.2.

Листинг 6.2. Текст документа, содержащего элементы формы, со значениями, передаваемыми по электронной почте

```
<html>
<head>
<title>Форма</title>
</head>
```

```
<body>
<p><b><font size="5">Закажите у нас партию
фруктов</font></b></p>
<form method="POST" action="mailto:q@qqq" enc-
type="text/plain">
<table border="0" width="450" cellpadding="0" cellspac-
ing="0">
  <tr>
    <td><b>Введите название вашей фирмы</b></td>
    <td><input type="text" name="Firma" size="25"></td>
  </tr>
  <tr>
    <td><b>Введите адрес вашей фирмы</b></td>
    <td><input type="text" name="Adres" size="25"></td>
  </tr>
</table>
<p>
<table border="0" width="450" cellpadding="0" cellspac-
ing="0">
  <tr>
    <td><b>Выберете способ доставки</b></td>
    <td><input type="radio" name="Dostavka"
value="Автомобиль" checked><b>Автомобиль</b></td>
  </tr>
  <tr>
    <td></td>
    <td><input type="radio" name="Dostavka" value="Железная
дорога"><b>Железная дорога</b></td>
  </tr>
</table>
<p>
<table border="0" width="450" cellpadding="0" cellspac-
ing="0">
  <tr>
    <td></td>
    <td><p align="center"><b>Количество в тоннах</b></td>
    <td><p align="center"><b>Страна производитель</b></td>
  </tr>
```

```
  <tr>
    <td><input type="checkbox" name="Zakaz_1"
value="Апельсины"><b>Апельсины</b></td>
    <td><p align="center"><input type="text"
name="Kol_Zakaz_1" size="6"></td>
    <td><p align="center">
      <select size="1" name="Strana" multiple>
        <option>Испания</option>
        <option>Аргентина</option>
        <option>Марокко</option>
      </select></td>
  </tr>
  <tr>
    <td><input type="checkbox" name="Zakaz_2"
value="Груши"><b>Груши</b></td>
    <td><p align="center"><input type="text"
name="Kol_Zakaz_2" size="6"
    <td><p align="center">
      <select size="1" name="Strana" multiple>
        <option>Испания</option>
        <option>Аргентина</option>
        <option>Марокко</option>
      </select></td>
  </tr>
</table>
<p><b>Свои пожелания перечислите в этом окне</b></p>
<textarea rows="3" name="Prosba" cols="50"></textarea>
<p>
<input type="submit" value="Отправить">
<input type="reset" value="Отменить"></p>
</form>
</body>
</html>
```

Размещение элементов формы на странице осуществляется с помощью таблицы. Располагая элементы формы в соответствии с требованиями внешнего вида страницы, следует помнить, что

все они должны находится между метками `<FORM>...</FORM>`, так как кнопки SUBMIT и RESET действуют только на те элементы, которые находятся с ними в одной форме.

Вид страницы в окне браузера показан на рис. 6.2.

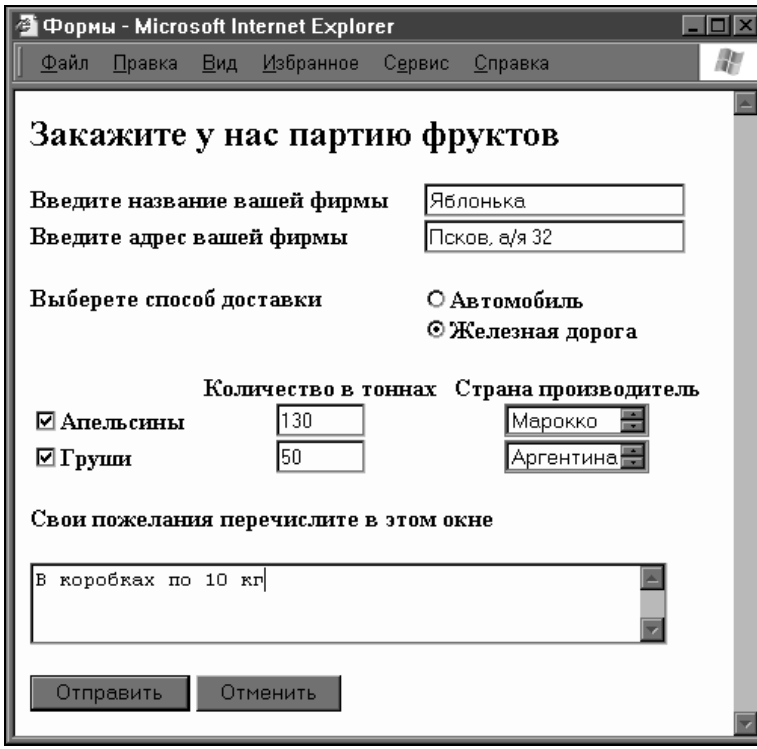


Рис. 6.2. Внешний вид страницы с формами в окне браузера

Почтовое отправление, содержащее данные формы, приведено на рис. 6.3.

Из текста почтового сообщения становится очевидным, что имена элементов формы несут смысловую нагрузку и должны выбираться таким образом, чтобы полученное сообщение могло быть легко прочитано. То же самое относится и к значениям параметров `VALUE` элементов формы. В данном случае форма отправляется по почте, и имена ее элементов могли бы быть напи-

саны с использованием кириллицы. Однако при обработке формы с помощью программы-сценария для имен необходимо использовать только латинские буквы и цифры. Поэтому лучше создавать форму с учетом данного требования, чтобы не пришлось в дальнейшем ее переделывать. Для этого достаточно заменить символы кириллицы похожими по звучанию латинскими буквами, что и сделано в нашем примере.

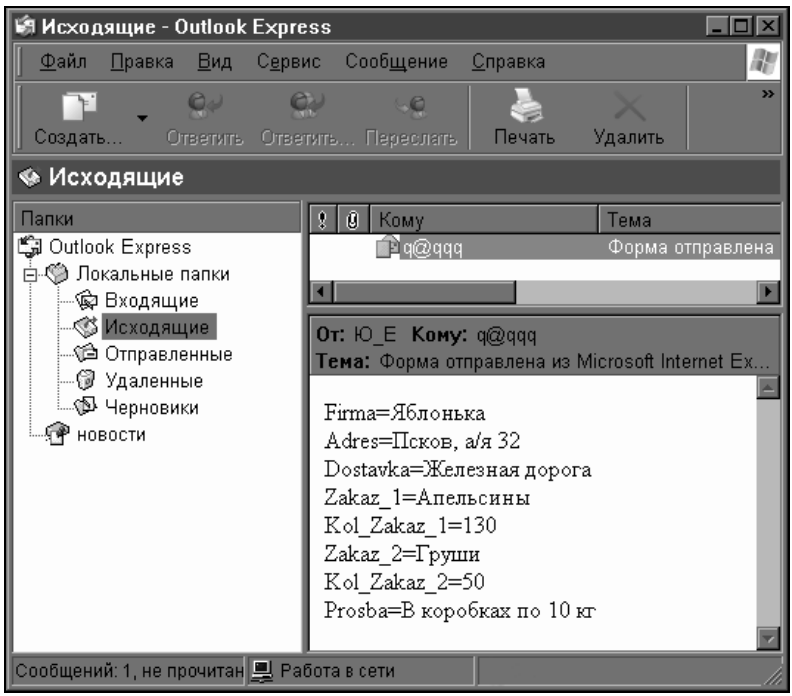


Рис. 6.3. Текст почтового сообщения с данными формы

6.3. Создание форм в программе Dreamweaver

Для создания форм на web-странице, редактируемой в программе Dreamweaver, необходимо воспользоваться группой инструментов **Forms** (Формы) панели **Insert** (Вставка) (рис. 6.4).

Рис. 6.4. Группа инструментов **Forms** панели **Insert**

Кнопки на вкладке **Forms** (Формы) располагаются в следующем порядке:

- **Form** (Форма) — предназначена для создания контейнера будущей формы. После щелчка по кнопке **Form** на странице появится прямоугольник, ограниченный прерывистой линией, а в тексте документа появится пара меток `<FORM>...</FORM>`. Курсор будет мерцать внутри прямоугольного контейнера, поэтому достаточно щелкнуть по кнопке с изображением одного из элементов формы и он появится внутри контейнера. Одновременно с созданием контейнера панель свойств **Properties** (Свойства) приобретет вид, приведенный на рис. 6.5.

Рис. 6.5. Панель свойств с полями для изменения свойств формы

На панели свойств **Properties** (Свойства) можно изменить свойства формы.

В поле **Form Name** (Имя формы) указывают новое имя формы, а в поле **Action** (Действие) — URL-адрес получателя данных из формы.

В списке **Method** (Метод) можно выбрать метод пересылки данных.

Список **Target** (Цель) позволяет выбрать окно или фрейм, в который будет загружен документ, присланный с сервера в ответ на пересылку формы: _self — текущее окно или фрейм, _parent — родительский фрейм, _top — полное окно, _blank — новое окно.

В списке **Enctype** выбирается значение параметра `ENCTYPE`, т. е. формат кодирования данных при пересылке, а при отсутствии нужного формата в списке его нужно вписать вручную. Для пересылки данных по почте необходимо выбрать значение text/plain.

В списке **Class** (Класс) задается значение параметра `CLASS`, т. е. стиль форматирования формы (*см. главу 7*). Для всех описанных ниже элементов список **Class** (Класс) имеет то же назначение;

- **Text Field** (Текстовое поле) — создает однострочное текстовое поле. В текст документа вставляется метка `<INPUT>` с параметром `TYPE=TEXT`. На панели свойств **Properties** (Свойства) появится все необходимое для изменения свойств этого элемента формы, и даже возможность превращения его в другой элемент (рис. 6.6).

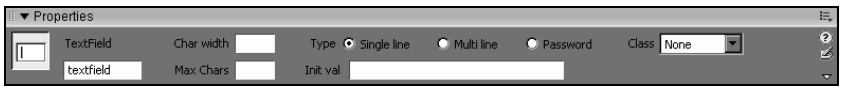


Рис. 6.6. Панель свойств с полями для изменения свойств текстового поля

В поле **TextField** можно изменить имя элемента формы, в поле **Char width** — установить максимальное количество отображаемых символов, в поле **Max Chars** – максимальное количество символов в строке (по умолчанию не ограничивается), в поле **Init val** — задать начальное содержимое текстового поля.

Переключатель **Type** (Тип) может находиться в одном из трех положений:

- **Single line** — однострочное текстовое поле;
- **Multi line** — многострочное текстовое поле;
- **Password** — однострочное текстовое поле для ввода пароля (вводимые символы отображаются звездочками).

Выбор положения **Multi line** приведет не только к смене внешнего вида элемента, но и замене метки `<INPUT>` на `<TEXTAREA>`;

- **Hidden Field** (Скрытое поле) — создает скрытое однострочное текстовое поле. Это поле не отображается браузером и поэтому скрыто от пользователя, но оно может содержать текстовую информацию, которая будет передана вместе с формой. Панель **Properties** (Свойства) содержит только два поля:

- **HiddenField** — для изменения имени элемента формы,
- **Value** (значение) — для ввода передаваемого текста;

❐ **Textarea** (Текстовая область) — создает многострочное текстовое поле. В текст документа вставляется метка `<TEXTAREA>`. Панель свойств **Properties** (Свойства) приобретает вид, показанный на рис. 6.7.

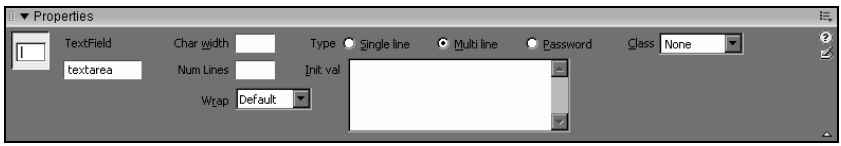


Рис. 6.7. Панель свойств с полями для изменения свойств многострочного текстового поля

Напомним, для того чтобы была видна нижняя половина панели **Properties** (Свойства), нужно щелкнуть по треугольной стрелке в правом нижнем углу панели. Несмотря на увеличение размера панели свойств по сравнению с элементом **Text Field** (Текстовое поле), она не претерпела серьезных изменений. Вместо поля **Max Chars** появилось поле **Num Lines** — число отображаемых строк и дополнительно появился список **Wrap**, позволяющий выбрать способ перехода на новую строку. В списке можно выбрать следующие значения:

- Default — по умолчанию;
- Off — выключен автоматический переход на новую строку;
- Virtual — текст автоматически переходит на новую строку при достижении максимальной ширины, но символ перехода не вставляется в текст и он будет передан одной строкой (действует по умолчанию);
- Physical — текст автоматически переходит на новую строку и вставляется символ перехода;

❐ **Checkbox** (Флажок) — создает поле для установки флажка. В текст документа будет вставлена метка `<INPUT>` с параметром `TYPE=CHECKBOX`. Панель свойств **Properties** (Свойства) для этого элемента содержит следующие поля: **Check Box name** (Имя флажка) — для изменения имени элемента и **Checked**

value — для ввода текста, который будет передан вместе с формой, если флажок установлен. Кроме того, на панели свойств имеется переключатель **Initial State** (Начальное состояние), позволяющий выбрать начальное состояние поля: Checked — флажок установлен, Unchecked — флажок снят;

- **Radio Button** (Элемент переключателя) — создает элемент переключателя, существующий в составе группы. В текст документа вставляется метка `<INPUT>` с параметром `TYPE=RADIO`. На панели **Properties** (Свойства) появятся следующие поля: **Radio Button** (Элемент переключателя) — изменение имени элемента, **Checked value** — ввод текста, который будет передан вместе с формой в случае выбора данного элемента. Переключатель **Initial State** (Начальное состояние) позволяет отметить данный элемент как выбранный (Checked) или нет (Unchecked);
- **Radio Group** (Переключатель) — открывает одноименное диалоговое окно для создания переключателя, содержащего несколько элементов, и оформления его внешнего вида (рис. 6.8).

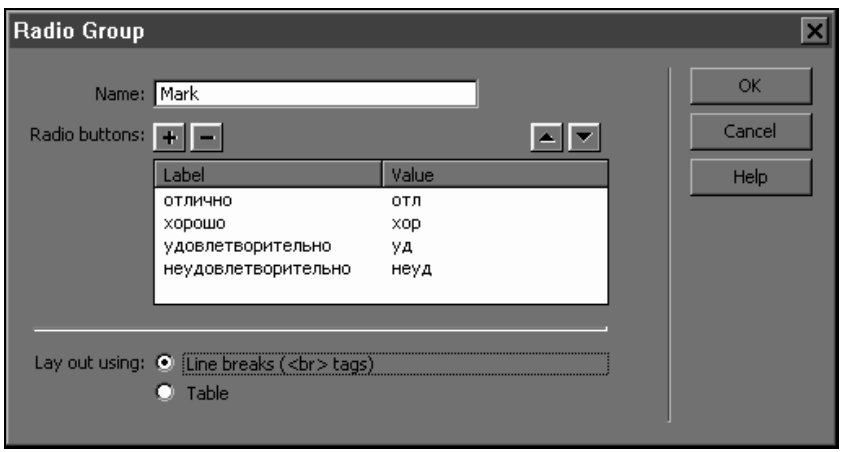


Рис. 6.8. Диалоговое окно для создания переключателя **Radio Group**

В поле Name (Имя) вносится имя переключателя, которое будет присвоено каждому элементу группы. Для добавления

нового элемента необходимо нажать кнопку с изображением плюса, для удаления выделенного — кнопку с изображением минуса. В столбце **Label** (Подпись) содержатся подписи к элементам переключателя, а в столбце **Value** (Значение) — текст, передаваемый вместе с формой, в случае выбора данного элемента. Кнопки с треугольными стрелками позволяют изменить порядок следования элементов. Переключатель **Lay out using** (Расположение) позволяет выбрать способ оформления группы. Выбор **Line breaks (
 tags)** означает, что элементы переключателя будут расположены в столбец с помощью меток `<BR>`, а выбор Table позволяет расположить группу с помощью таблицы. Результат показан на рис. 6.9;

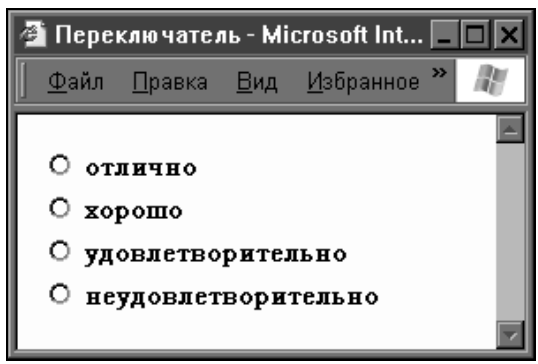


Рис. 6.9. Результат создания переключателя **Radio Group**

- **List/Menu** (Список/Меню) — создает список, из которого можно выбрать одно или несколько значений. В документ список будет вставлен с помощью метки `<SELECT>`. Изменение параметров списка осуществляется с помощью панели свойств **Properties** (Свойства) (рис. 6.10). Данная панель содержит специальное поле **List/Menu** (Список/Меню) для задания имени списка. Переключатель **Type** (Тип) предназначен для выбора типа списка. Тип **Menu** отличается тем, что до раскрытия списка видимым является только один элемент, и после его раскрытия можно выбрать также только один элемент. Выбор типа **List** позволяет изменить число видимых элементов списка, задав их число в поле **Height** (Высота). Установка флажка **Selections Allow multiple** (Разрешить много-

кратный выбор) разрешает одновременное выделение нескольких элементов.

Рис. 6.10. Панель свойств с полями для изменения свойств списка значений

В окне браузера элементы, идущие подряд, можно выделить, удерживая клавишу <Shift>. Для выделения необходимых элементов в произвольном порядке используют клавишу <Ctrl>.

Независимо от выбранного типа списка его заполнение осуществляется в диалоговом окне **List Values** (Список значений) (рис. 6.11), которое открывается после щелчка по одноименной кнопке.

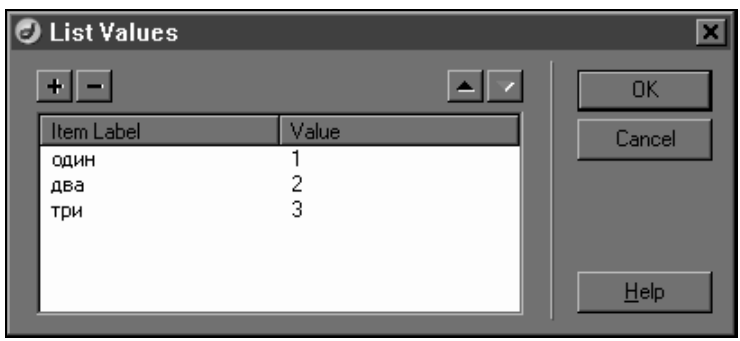


Рис. 6.11. Диалоговое окно **List Values**

Добавление элемента списка осуществляется после нажатия кнопки с изображением плюса, удаление — кнопки с изображением минуса. В столбце **Item Label** (Текст элемента) перечисляется содержимое списка, а в столбце **Value** (Значение) — текст, передаваемый в случае выбора данного элемента. Изменение порядка следования элементов осуществляется кнопками с треугольными стрелками. После нажа-

тия **OK** содержимое списка появится в поле **Initially Selected** (Предварительно отобранные);

- **Jump Menu** (Список перехода) — открывает диалоговое окно **Insert Jump Menu** (Вставка списка перехода) (рис. 6.12), позволяющее создать раскрывающийся список аналогичный **List/Menu**. Следует иметь в виду, что при выборе элементов списка в окне браузера будут осуществляться переходы на другие документы, которые создаются не путем создания гиперссылок, а с помощью программы-сценария на языке JavaScript, которая автоматически вставляется в текст документа.

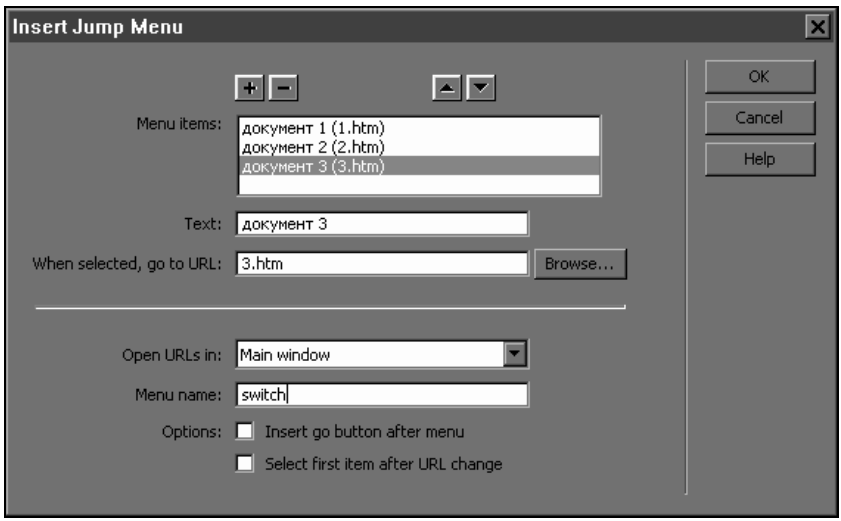


Рис. 6.12. Диалоговое окно **Insert Jump Menu**

Добавление новых элементов списка осуществляется нажатием кнопки со знаком плюс, удаление — кнопки со знаком минус. Значение элемента вписывается в поле **Text** (Текст), а в поле **Menu items** (Список элементов), оно будет продублировано автоматически. В поле **When selected, go to URL** (Перейти к документу после выделения) вводится имя файла документа, на который должен осуществиться переход. Список **Open URLs In** (Открывать адресуемые документы в) содержит единственное значение Main Window — главное окно. В поле

Menu Name (Имя списка) указывается имя списка. В группе **Options** (Выбор) можно установить флажки **Insert Go Button After Menu** (Вставить кнопку GO после списка) и **Select First Item After URL Change** (Выделить первое значение после адресации). Изменение параметров на панели свойств **Properties** (Свойства) аналогично работе со списком **List/Menu** (Список/Меню);

- **Image Field** (Графическое поле) — вставляет графическое изображение, которое будет выполнять роль кнопки SUBMIT, т. е. щелчок по изображению в окне браузера приведет к отправке формы по назначению. В текст документа будет вставлена метка `<INPUT>` с параметром `TYPE=IMAGE`. Панель свойств имеет следующие поля:
  - **ImageField** — имя элемента;
  - **W** (Width) — ширина рисунка;
  - **H** (Height) — высота рисунка;
  - **Src** — имя файла рисунка;
  - **Alt** — текст, появляющийся в виде всплывающей подсказки;
  - **Align** (Выравнивание) — список значений, определяющих способ выравнивания изображения, известный по выравниванию обычных изображений.

  Щелчок по кнопке **Edit Image** (Редактирование изображения) на панели свойств **Properties** (Свойства) откроет изображение в окне графического редактора;
- **File Field** (Поле файла) — позволяет создать поле для ввода имени локального файла, который требуется переслать вместе с формой. Рядом с полем будет выводиться кнопка **Browse** (Обзор) для поиска и выбора файла. В текст документа будет вставлена метка `<INPUT>` с параметром `TYPE=FILE`. Панель свойств **Properties** (Свойства) в этом случае будет содержать поля:
  - **FileField name** — для изменения имени элемента;
  - **Char width** — для задания максимального количества отображаемых символов;

- **Max chars** — для задания максимальной длины пути к файлу;

- **Button** (Кнопка) — создает кнопку. В текст документа будет вставлена метка `<INPUT>` с параметром `TYPE`, значения которого зависят от выбранного положения переключателя **Action** (Действие) на панели свойств **Properties** (Свойства). Выбор **Submit form** означает создание кнопки для отправки содержимого формы к месту назначения. Параметру `TYPE` будет присвоено значение `"submit"`. Переключатель в положении **Reset form** позволяет создать кнопку сброса содержимого формы, параметру `TYPE` присваивается значение `"reset"`. Положение переключателя **None** означает, что выбрана обычная кнопка, а параметру `TYPE` будет присвоено значение `"button"`. Обычная кнопка используется только совместно с программой-сценарием, которая должна запускаться при ее нажатии. Подобные примеры будут рассмотрены в *главе 8*. Следует отметить наличие на панели **Properties** (Свойства) еще двух полей: **Button Name** (Имя кнопки) — для изменения имени кнопки и **Label** (Надпись) — надписи на кнопке;

- **Label** (Надпись) — вставляет в текст документа метку `<LABEL>...</LABEL>`. Текст внутри метки используется в качестве подписи к элементу формы;

- **Fieldset** (Составное поле) — создает элемент, объединяющий несколько элементов формы в одну группу при помощи рамки и обобщающей надписи. Текст надписи указывается в диалоговом окне **Fieldset** (Составное поле), которое открывается после нажатия кнопки. Объединяемые объекты предварительно должны быть выделены. Результат объединения можно увидеть в окне браузера.

Глава 7

Каскадные таблицы стилей — расширение возможностей форматирования

Каскадные таблицы стилей (Cascade Style Sheets, CSS) — наборы меток HTML, к которым добавляются определения, изменяющие их свойства. Например:

```
P {font-size: 40pt; color: green; font-family: "Comic
Sans MS"}
```

В приведенном примере каскадная таблица стилей содержит определения метки абзаца `<P>`, в соответствии с которыми устанавливается: размер шрифта — 40 pt (пунктов), цвет шрифта — green и гарнитура — Comic Sans MS.

Шрифт, который будет располагаться в области действия метки `<P>...</P>`, будет иметь параметры форматирования, соответствующие определениям каскадной таблицы стилей.

7.1. Назначение каскадных таблиц стилей

Рассмотрим назначение каскадных таблиц стилей.

- Они позволяют получить результаты, которые не могут быть получены средствами HTML. Так, в приведенном выше примере установлен размеры шрифта 40 pt, тогда как в HTML

с помощью параметра SIZE можно задать максимальный размер равный 7, что соответствует 36 pt.

- В сочетании с программами сценариев (JavaScript или VBScript) каскадные таблицы позволяют динамически изменять стиль отображения документа в окне браузера в зависимости от каких-либо действий пользователя (щелчок мышью, перемещение указателя и т. д.).
- Каскадные таблицы позволяют задать единый стиль оформления разных страниц документа и быстро поменять его, путем изменения нужного определения в таблице стилей.

В заключение хотелось бы отметить, что таблицы стилей назвали каскадными потому, что в одном документе можно использовать несколько таблиц, а браузер выстроит их каскадом, в соответствии с приоритетами использования.

7.2. Связь каскадной таблицы стилей с HTML-документом

Существуют четыре способа связи CSS с HTML-документом:

1. Таблица стилей существует в виде отдельного текстового файла с расширением CSS (желательно, чтобы он находился в той же папке, что и HTML-документ) и подключается к нему с помощью метки <LINK>, помещаемой в раздел <HEAD>:

   ```
   <LINK REL="stylesheet" TYPE="text/css" HREF="mystyle.css">
   ```

   Параметру REL присвоено значение "stylesheet". Это означает, что подключаемая таблица стилей будет использоваться браузером по умолчанию.

   Параметр TYPE предназначен для указания языка стилей (в нашем случае CSS).

   Параметр HREF позволяет задать имя CSS-файла.

   Для примера создадим CSS-файл в Блокноте и поместим в него строку определений для метки <P>, приведенную в начале главы. Сохраним файл под именем mystyle.css. Далее в Блокноте создадим HTML-документ, приведенный в листинге 7.1.

Листинг 7.1. HTML-документ, в котором форматирование текста осуществляется каскадной таблицей стилей, находящейся в отдельном файле mystyle.css

```
<HTML>
<HEAD>
<LINK REL="stylesheet" TYPE="text/css" HREF="mystyle.css">
<TITLE>
каскадные таблицы стилей
</TITLE>
</HEAD>
<BODY>
<p>Это абзац (font-size: 40pt; color: green; font-family:
"Comic Sans MS")</p>
</BODY>
</HTML>
```

Сохраним документ под именем mydoc.htm в той же папке, что и mystyle.css, и, открыв его в браузере, увидим результат (рис. 7.1).

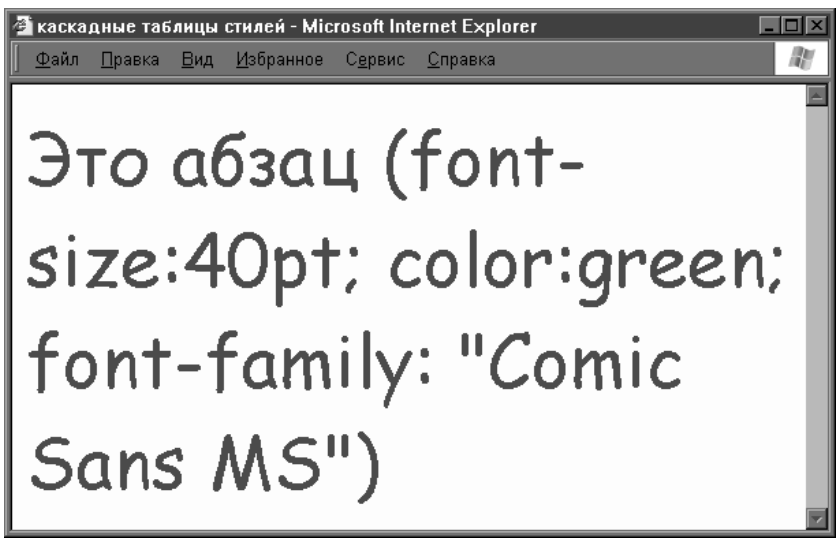


Рис. 7.1. Результат форматирования текста с использованием каскадной таблицы стилей

Достоинство метода подключения CSS заключается в том, что одна таблица может быть подключена к большому количеству документов, придав им единый стиль оформления.

2. Таблица стилей непосредственно размещается в разделе <HEAD> документа, внутри блока, отмеченного метками <STYLE>… </STYLE>. Например:

```
<HEAD>
  <STYLE TYPE="text/css">
    P {font-size: 48pt; color: red; background-color:
blue}
    B {font-size: 96pt; color: blue; font-weight: 800}
  </STYLE>
…
</HEAD>
```

В этом примере изменены свойства двух меток. Кроме размера и цвета шрифта, изменен цвет фона для метки <P> и толщина шрифта для метки <B>.

3. Файл с таблицей стилей импортируется в начало блока <STYLE> или другой таблицы стилей с помощью инструкции "@import: url (имя_файла.css)". Например:

```
<HEAD>
  <STYLE TYPE="text/css">
   @import: url (mystyle.css);
    P {font-size: 48pt; color: red; background-color:
blue}
    B {font-size: 96pt; color: blue; font-weight: 800}
  </STYLE>
…
</HEAD>
```

Добавив перед таблицей стилей имя файла, содержащего другую таблицу стилей, мы создали тот самый каскад, который и дал название таблицам стилей. Причем в созданном ранее файле mystyle.css были установлены значения свойств метки <P> "font-size" и "color", которые противоречат значениям этих же свойств, заданным непосредственно в блоке <STYLE>. Браузер будет считать более приоритетными те значения, ко-

торые следуют ниже по тексту. В нашем случае это значения, заданные непосредственно в блоке <STYLE>.

4. Каждая метка может иметь параметр STYLE, в котором указываются свойства и их значения. Например:

```
<B STYLE="font-size: 48pt; color: yellow">
```

Такой способ задания свойств каскадной таблицы стилей придает максимальный приоритет значениям свойств по сравнению с другими способами, так как они задаются непосредственно в самой метке. Недостатком способа является высокая трудоемкость создания и коррекции документа, увеличение его объема, поэтому его целесообразно использовать только для изменения свойств отдельных элементов, число которых невелико.

7.3. Задание одинаковым меткам разных параметров форматирования

Вполне возможна ситуация, когда часть одних и тех же меток применительно к различным частям текста должна иметь разные свойства.

С этой целью метки (в зависимости от стиля форматирования) можно разбить на классы, каждому из которых присвоить персональное имя, которое записывается справа от имени метки и отделяется от него точкой. Таким образом, для одной метки можно определить несколько стилей форматирования. Например:

```
<STYLE TYPE="text/css">
   P.white_black {font-size: 38pt; color: white; background-
color: black}
   P.black_gray {font-size: 46pt; color: black; background-
color: gray}
</STYLE>
```

В этом примере метки <P> мы разбили на два класса, имеющих разные параметры форматирования, один из которых получил имя "white_black", а другой — "black_gray".

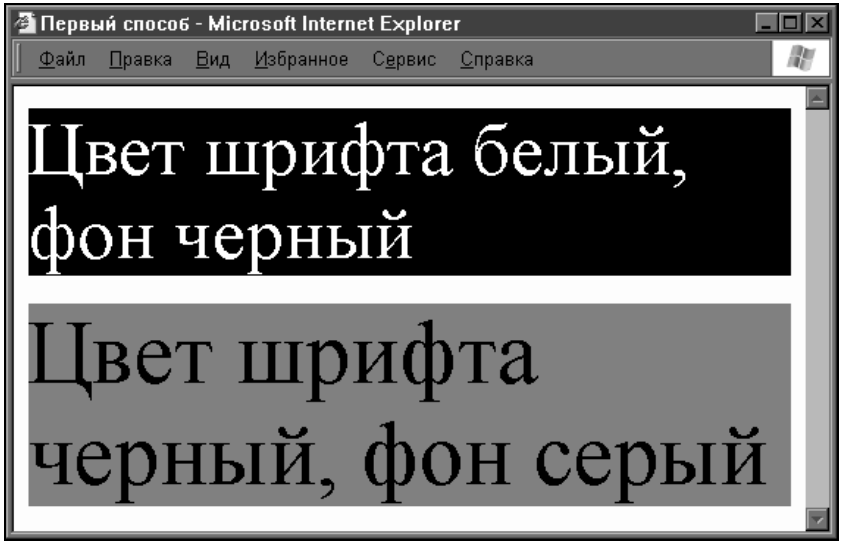


Рис. 7.2. Результат форматирования шрифта, с использованием двух стилей

В тексте документа выбор стиля (класса) осуществляется с помощью параметра `CLASS` следующим образом:

```
<P CLASS=white_black>Цвет шрифта белый, фон черный</P>
<P CLASS=black_gray>Цвет шрифта черный, фон серый</P>
```

Полный текст документа приведен в листинге 7.2.

Листинг 7.2. Текст документа, в котором используются два стиля форматирования шрифта

```
<HTML>
<HEAD>
<STYLE>
P.white_black {font-size: 38pt; color: white; background-
color: black}
P.black_gray {font-size: 46pt; color: black; background-
color: gray}
</STYLE>
<TITLE>
```

```
Первый способ
</TITLE>
</HEAD>
<BODY>
<P class="white_black">Цвет шрифта белый, фон черный</P>
<P class="black_gray">Цвет шрифта черный, фон серый</P>
</BODY>
</HTML>
```

Открыв документ в окне браузера, увидим результат (рис. 7.2).

7.4. Создание стиля форматирования, применимого к различным меткам

В один и тот же класс могут входить не обязательно одинаковые метки, но тогда в описании стиля не нужно указывать имя метки. Здесь требуется указать только имя класса с предшествующей точкой. Например:

```
<STYLE TYPE="text/css">
   .white_black {font-size: 38pt; color: white; background-
color: black}
   .black_gray {font-size: 46pt; color: black; background-
color: gray}
</STYLE>
Далее в тексте документа можно, в частности, написать:
<EM class="white_black">Курсивный шрифт</EM><BR>
<INS class="black_gray">Подчеркнутый шрифт</INS>
```

Полный текст документа приведен в листинге 7.3.

Листинг 7.3. Применение одного стиля форматирования к различным меткам

```
<HTML>
<HEAD>
<STYLE>
.white_black {font-size: 38pt; color: white; background-
color: black}
```

```
.black_gray {font-size: 46pt; color: black; background-color:
gray}
</STYLE>
<TITLE>
Первый способ
</TITLE>
</HEAD>
<BODY>
<EM class="white_black">Курсивный шрифт</EM><BR>
<INS class="black_gray">Подчеркнутый шрифт</INS>
</BODY>
</HTML>
```

7.5. Вложение меток

Часто метки располагаются одна внутри другой. Если внутренняя (дочерняя) метка не имеет собственного стиля форматирования, то на нее будет распространяться стиль форматирования родительской метки, т. е. той метки, внутри которой она находится. Если же стиль дочерней метки противоречит родительскому стилю, то на форматируемый текст будет действовать ее собственный стиль, что объясняется большей близостью к форматируемому тексту. Пример записи приведен в листинге 7.4.

Листинг 7.4. Вложение меток с разными стилями

```
<HTML>
<HEAD>
<STYLE TYPE="text/css">
.white {font-size: 38pt; color: white}
.black_gray {font-size: 46pt; color: black; background-color:
gray}
</STYLE>
<TITLE>
Вложения меток
```

```
</TITLE>
</HEAD>
<BODY>
<INS CLASS="black_gray"><EM CLASS="white">
Курсивом</EM>написано только одно слово</INS>
</BODY>
</HTML>
```

В окне браузера документ будет воспроизведен как на рис. 7.3.

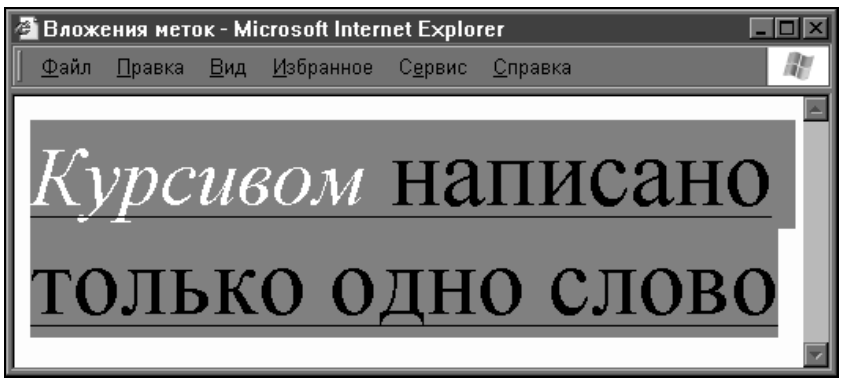


Рис. 7.3. Результат форматирования текста вложенными метками с разными стилями

Из примера видно, что такие элементы форматирования шрифта, как: его цвет, цвет фона, подчеркивание, — это результат действия стиля родительской метки `<INS>`, распространившегося на весь текст. Действие метки `<EM>` повлияло только на одно слово, причем заданные в ней цвет текста и его размер являются более приоритетными по отношению к аналогичным свойствам, заданным в родительской метке.

Кроме разбиения на классы, существует еще одна интересная возможность задания для одних и тех же меток различных стилей форматирования. Запись `H1 EM {font-size: 48px; color: gray}` означает, что определения относятся к метке `<EM>`, но действовать они будут только тогда, когда она будет дочерней по отношению к метке `<H1>`. Имена родительской и дочерней меток разделяются пробелом. Пример в листинге 7.5.

Листинг 7.5. Пример действия стиля только в случае одновременного использования двух меток

```
<HTML>
<HEAD>
<STYLE TYPE="text/css">
H1 EM {font-size: 48px; color: gray}
</STYLE>
<TITLE>
Вложения меток
</TITLE>
</HEAD>
<BODY>
<H1>Заголовок<EM>первого</EM>уровня</H1>
<EM>Курсивный текст</EM>
</BODY>
</HTML>
```

Результат в окне браузера приведен на рис. 7.4.

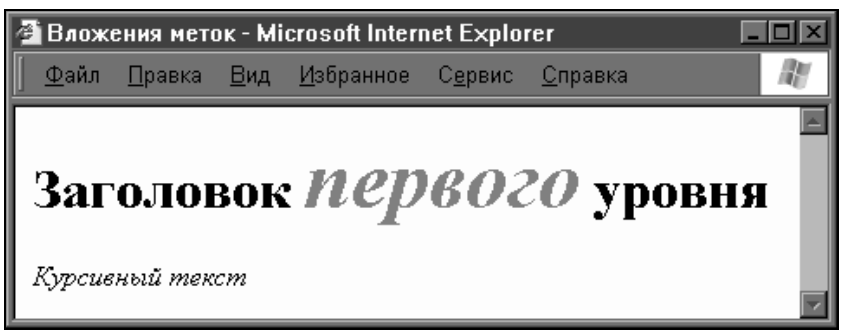


Рис. 7.4. Результат действия стиля только в случае одновременного использования двух меток

Действия метки `<EM>` оказались различными. В первом случае в сочетании с меткой `<H1>` меткой `<EM>` были изменены начертание, размер и цвет шрифта, во втором — только начертание.

7.6. Единицы измерения в каскадных таблицах стилей

По сравнению с HTML в CSS можно использовать существенно больше единиц измерения и применять их не только к размерам объектов, но и к размеру шрифта. В приведенных выше примерах мы задавали размеры шрифта в пунктах (pt) и пикселях (px).

Единицы измерения принято подразделять на *абсолютные* и *относительные*. Абсолютные единицы приведены в табл. 7.1.

Таблица 7.1. *Абсолютные единицы измерения*

Наименование	Обозначение	Соотношение с другими единицами
Миллиметр	mm	1 mm = 0,001 m (метра)
Сантиметр	cm	1 cm = 10 mm
Дюйм	in	1 in = 2,54 cm
Пункт	pt	1 pt = 1/72 in
Пика	pc	1 pc = 12 pt
Пиксель	px	Размер пикселя на экране монитора по горизонтали и вертикали зависит от установленного разрешения экрана (800 × 600, 1024 × 768 и т. д.) и размера экрана

Большинство приведенных в таблице единиц нам хорошо известно и, казалось бы, нет ничего удобнее при создании web-страниц использовать единицы, к которым давно привык, например сантиметры. Однако следует иметь в виду, что сантиметр на экране монитора будет отличаться от сантиметра на обычной линейке. То же самое можно сказать и про миллиметр, дюйм, пункт и пику, которые жестко связаны с сантиметром известными соотношениями.

Объясняется это тем, что перед выводом на экран все величины преобразуются в пиксели, для чего используется масштаб изображения. В операционной системе Windows XP, чтобы посмотреть масштаб изображения, требуется открыть диалоговое окно

Свойства: Экран (открывается двукратным щелчком по значку **Экран** в папке **Панель управления**). Перейдя на вкладку **Параметры**, нужно щелкнуть по кнопке **Дополнительно**. Откроется диалоговое окно **Свойства: Модуль подключения монитора**, в котором на вкладке **Общие** и можно увидеть масштаб изображения. Обычно масштаб изображения составляет 96 пикселей/дюйм. Например, если задан размер 5 сантиметров (cm), то на экране монитора это будет составлять 5/2,54*96 = 189 пикселей. Так как размер пикселя зависит от размера экрана монитора и установленного разрешения экрана (800 × 600, 1024 × 768 и т. д.), то становится понятно, что все абсолютные величины являются таковыми лишь для монитора с конкретными характеристиками и практически всегда будут отличаться от реальных сантиметров, дюймов и т. д. В связи с этим наиболее удобной единицей для жесткой компоновки страницы, т. е. задания абсолютных размеров объектов и их координат, является пиксель. Задавая размеры в пикселях, придется ориентироваться только на разрешение экрана, а остальные параметры, такие как размер экрана и масштаб изображения, на внешний вид страницы влиять не будут.

Относительные единицы приведены в табл. 7.2.

Таблица 7.2. *Относительные единицы измерения*

Наименование	Обозначение	Соотношение с другими единицами
Процент	%	Вычисляется относительно родительского элемента
–	em	Вычисляется относительно размера шрифта элемента

Для гибкой компоновки страниц, когда размеры объекта зависят от размера окна браузера, можно использовать проценты. Например, если для компоновки страницы использована таблица, размеры которой заданы в процентах, относительно размеров окна браузера, то, задав размеры рисунков, находящихся в ячейках таблицы, также в процентах, но относительно размеров ячеек, можно получить страницу с гибкой компоновкой, которая

будет практически одинаково выглядеть на мониторах с разными параметрами.

Весьма полезной может быть и относительная единица em, с помощью которой можно задавать размеры объектов и их координаты относительно размера шрифта. *Примеры использования относительных единиц будут рассмотрены* в листинге 7.6 и в *разделе 8.19.*

7.7. Основные свойства каскадной таблицы стилей

До сих пор мы использовали только несколько свойств, с помощью которых показывали способы создания и размещения каскадных таблиц стилей. Теперь более подробно рассмотрим основные свойства каскадной таблицы стилей, которые для удобства использования разделим на несколько категорий.

7.7.1. Свойства форматирования шрифтов

В табл. 7.3 приведены основные свойства форматирования шрифтов.

Таблица 7.3. *Свойства форматирования шрифтов*

№ п/п	Обозначение	Изменяемое свойство	Примечания
1	font-family	гарнитура шрифта	
2	font-style	начертание шрифта	Допустимые значения: "normal" – обычный; "italic" – курсивный; "oblique" – наклонный
3	font-weight	толщина шрифта (степень жирности)	Возможны 9 значений, расположенных по возрастанию жирности: 100, 200, ..., 900

Таблица 7.3 *(окончание)*

№ п/п	Обозначение	Изменяемое свойство	Примечания
4	font-size	размер шрифта	Абсолютное или относительное значение
5	font-variant	изображение строчных букв, как прописных	Допустимые значения: "normal" – обычный; "small-caps" – строчные как прописные
6	font	все перечисленные свойства	Записываются через пробел

Примеры использования свойств форматирования шрифта приведены в листинге 7.6.

Листинг 7.6. Примеры использования различных свойств форматирования шрифта и вложенности меток

```
<HTML>
<HEAD>
<STYLE TYPE="text/css">
P {font-family: "Times New Roman"; font-style: oblique; font-
weight: 800; font-size: 3cm}
U {font-size: 1.5em; font-variant: small-caps}
</STYLE>
<TITLE>
Свойства форматирования шрифтов
</TITLE>
</HEAD>
<BODY>
<P>внутри <U>абзаца</U></P>
<U>После Абзаца</U>
</BODY>
</HTML>
```

Рис. 7.5. Результат использования свойств форматирования шрифта и вложенности меток

7.7.2. Свойства цвета и фона

Основные свойства цвета и фона приведены в табл. 7.4.

Таблица 7.4. *Свойства цвета и фона*

№ п/п	Обозначение	Изменяемое свойство	Примечания
1	color	Цвет текста	Цвет устанавливается указанием его названия: color: blue или с использованием цветовой модели RGB: color: #0000FF color: rgb (0,0,255) В первом случае используется шестнадцатеричный код, во втором – десятичный

Таблица 7.4 *(окончание)*

№ п/п	Обозначение	Изменяемое свойство	Примечания
2	background-color	Цвет фона	Устанавливается аналогично цвету текста
3	background-image	Рисунок в качестве фона	Указывается имя файла рисунка: background-image: url (имя файла) Отказ от фона, заданного в родительской метке: background-image: none
4	background-repeat	Повторяемость для фона, заданного изображением	Повторяемость по горизонтали: background-repeat: repeat-x Повторяемость по вертикали: background-repeat: repeat-y Изображение не повторяется: background-repeat: no-repeat По умолчанию действует повторяемость по горизонтали и вертикали одновременно
5	background-attachment	Неподвижность фона при прокрутке содержимого окна браузера	Устанавливается значение fixed: background-attachment: fixed
6	background-position	Положение фонового изображения относительно левого верхнего угла элемента, в котором оно находится	Задается в процентах от размера элемента (первое число – смещение по горизонтали, второе – по вертикали): background-position: 20 % 15 % или в пикселях: background-position:50 px 70 px

7.7.3. Свойства форматирования текста

В табл. 7.5 содержатся основные свойства форматирования текста.

Таблица 7.5. *Свойства форматирования текста*

№ п/п	Обозначение	Изменяемое свойство	Примечания
1	text-align	Выравнивание текста относительно элемента, в котором он находится	Возможны значения: left – по левому краю, right – по правому краю, center – центрирование, justify – по ширине
2	vertical-align	Положение элемента по вертикали относительно родительского элемента	Возможны значения: middle – базовая линия элемента выравнивается на уровне средней точки родителя, top – совмещаются верхние края элементов, bottom – совмещаются нижние края элементов
3	text-indent	Отступ первой строки абзаца	Задается в абсолютных единицах: text-indent: 10 px или в процентах относительно ширины элемента text-indent: 5 %
4	text-decoration	Подчеркивание, надчеркивание или перечеркивание текста	Возможны значения: underline – подчеркивание; overline – надчеркивание; line-through – перечеркивание; none – отказ от применения

Таблица 7.5 *(окончание)*

№ п/п	Обозначение	Изменяемое свойство	Примечания
5	text-transform	Трансформация текста	Возможны значения: capitalize – первые буквы каждого слова выводятся заглавными; uppercase – весь текст выводится заглавными буквами; lowercase – весь текст выводится строчными буквами; none – нет трансформации текста (по умолчанию)
6	letter-spacing	Дополнительное расстояние между символами, которое добавится к расстоянию, установленному по умолчанию	Абсолютное или относительное значение
7	word-spacing	Расстояние между словами	Возможны положительные и отрицательные значения. При отрицательных значениях возможно наложение слов
8	line-height	Расстояние между строками	При отсутствии единиц измерения, расстояние между строками вычисляется браузером как произведение заданного значения и размера шрифта. Например: DIV {font-size: 16 px; line-height: 1.5} Задано расстояние между строками 24 px

Свойство `text-decoration` со значением `none` можно в частности использовать для снятия подчеркивания гипертекста.

В качестве примера рассмотрим текст документа, в котором использованы некоторые свойства форматирования текста (листинг 7.7).

Листинг 7.7. Использование свойств форматирования текста

```
<HTML>
<HEAD>
<TITLE>форматирование текста</TITLE>
</HEAD>
<BODY>
<P style="font-size: 14pt; text-indent: 20px; text-align:
justify">Пример возможностей форматирования текста. Текст выравнивается по ширине, задан отступ первой строки.<SPAN
style="letter-spacing: 3px">Этот фрагмент имеет увеличенное
расстояние между символами.</SPAN><SPAN style="text-
transform:capitalize">Первые буквы этого фрагмента выводятся
заглавными буквами.</SPAN>
</P>
</BODY>
</HTML>
```

Внешний вид документа в окне браузера показан на рис. 7.6.

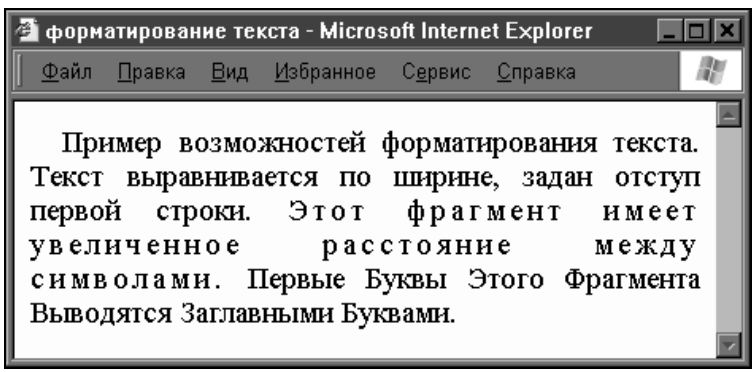


Рис. 7.6. Результат применения свойств форматирования текста

Следует обратить внимание на особенности применения свойств `text-align` и `vertical-align`. Первое из них позволяет выровнять по горизонтали содержимое того элемента, в котором оно задано. В нашем примере это содержимое абзаца, выравниваемое по ширине. Что касается свойства `vertical-align`, то оно позволяет выровнять по вертикали сам элемент, в котором оно задано, относительно родительского элемента. В качестве примера рассмотрим следующий документ (листинг 7.8).

Листинг 7.8. Пример использования вертикального выравнивания

```
<HTML>
<HEAD>
<STYLE>
<!--
.p1 {font-size:18 pt; background-color: yellow}
-->
</STYLE>
<TITLE>
вертикальное выравнивание
</TITLE>
</HEAD>
<BODY>
<P style="font-size: 32pt; color: red; background-color:
blue">Абзац
<SPAN class=p1 style="vertical-align: middle">Середина</SPAN>
<SPAN class=p1 style="vertical-align: top">По верху</SPAN>
<SPAN class=p1 style="vertical-align: bottom">По низу</SPAN>
</P>
</BODY>
</HTML>
```

Выравнивание по вертикали задано в трех фрагментах, отмеченных метками `<SPAN>`. Родительским элементом является абзац, в который входят все три фрагмента.

На рис. 7.7 показан вид документа в окне браузера.

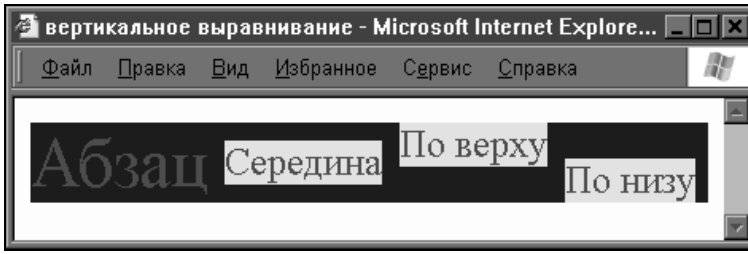


Рис. 7.7. Результат вертикального выравнивания отдельных фрагментов внутри абзаца

7.7.4. Свойства блоков

Блок это прямоугольный фрагмент страницы, который может иметь определенные размеры, отступы между содержимым блока и его границами, поля, рамку и фоновую заливку. Содержимым блока могут быть любые элементы, размещаемые на странице. Блок может быть снабжен собственными полосами прокрутки. Свойства блоков приведены в табл. 7.6.

Таблица 7.6. *Свойства блоков*

№ п/п	Обозначение	Изменяемое свойство	Примечания
1	width	Ширина блока	Абсолютное или относительное значение
2	height	Высота блока	Абсолютное или относительное значение
3	margin-top, margin-right, margin-bottom, margin-left	Размеры полей вокруг блока: верхнего, правого, нижнего, левого	Абсолютное или относительное значение
4	margin	Размеры полей вокруг блока – сокращенная форма записи	Абсолютное или относительное значение. Задаются через пробел в той последовательности, в которой они перечислены в п. 3. Например: margin: 5 px 10 px 10 px 15 px

Таблица 7.6 *(продолжение)*

№ п/п	Обозначение	Изменяемое свойство	Примечания
			В случае равенства всех полей задается только одно значение: margin: 10 px В случае попарного равенства противоположных полей (верхнего и нижнего, правого и левого) задаются только два значения: margin: 5 px 10 px
5	padding-top, padding-right, padding-bottom, padding-left	Размеры отступов содержимого блока от его краев: верхнего, правого, нижнего, левого	Абсолютное или относительное значение
6	padding	Размеры отступов содержимого блока от его краев – сокращенная форма записи	Абсолютное или относительное значение. Формат записи аналогичен свойству margin
7	border-top-style, border-right-style, border-bottom-style, border-left-style	Стиль рамки вокруг блока – сверху, справа, снизу, слева	Возможны следующие значения: none – нет рамки (по умолчанию), dotted – пунктирная; dashed – прерывистая; solid – непрерывная; double – двойная; groove – углубленная; ridge – выпуклая;

Таблица 7.6 *(продолжение)*

№ п/п	Обозначение	Изменяемое свойство	Примечания
			insert – эффект углубления блока; outset – эффект выпуклости блока;
8	border-style	Стиль рамки вокруг блока – сокращенная форма записи	Формат записи аналогичен свойству margin
9	border-top-width, border-right-width, border-bottom-width, border-left-width	Толщина рамки вокруг блока: сверху, справа, снизу, слева	Абсолютные значения, а также ключевые слова: thin – тонкая рамка; medium – средняя; thick – толстая
10	border-width	Толщина рамки вокруг блока – сокращенная форма записи	Формат записи аналогичен свойству margin
11	border-top-color, border-right-color, border-bottom-color, border-left-color	Цвет рамки вокруг блока: сверху, справа, снизу, слева.	Цвет можно задавать как шестнадцатеричным кодом, например: #00FF00 или #0F0, так и используя названия цветов, например: green
12	border-color	Цвет рамки вокруг блока – сокращенная форма записи	Формат записи аналогичен свойству margin
13	border-top, border-right, border-bottom, border-left	Свойства каждой из сторон рамки в отдельности – сокращенная форма записи	Позволяет задать для каждой из сторон сразу три свойства рамки в любой последовательности. Например: border-top: red dotted 10 px

Таблица 7.6 *(окончание)*

№ п/п	Обозначение	Изменяемое свойство	Примечания
14	border	Свойства всех сторон рамки при условии их равенства	Позволяет задать три свойства рамки, одинаковые для всех сторон: border: blue solid 5 px
15	overflow	Действия при переполнении блока	Возможны следующие значения: visible – содержимое блока, не поместившееся внутри, остается видимым (по умолчанию); hidden – содержимое блока, оказавшееся за его пределами, будет скрыто; scroll – независимо от объема содержимого блок снабжается полосами прокрутки; auto – полосы прокрутки устанавливаются только при переполнении блока

В качестве примера создания блоков рассмотрим следующий документ (листинг 7.9)

Листинг 7.9. Создание блоков с различными параметрами форматирования

```
<html>
<head>
<title>Блоки</title>
<style type="text/css">
<!--
```

```
.st1 {background-color: #00FFFF; margin: 20px; padding: 10px;
height: 50px; width: 100px; border: medium solid #0000FF;
letter-spacing: 8px; font-weight: 700}
.st2 {background-color: #CCFFFF; padding: 10px; height: 50px;
width: 100px; border: solid #00FF00; letter-spacing: 8px;
font-weight: 700}
.st3 {background-color: #00FFFF; margin: 10px; padding: 10px;
height: 50px; width: 100px; border: medium solid #0000FF;
letter-spacing: 8px; font-weight: 700}
-->
</style>
</head>
<body leftmargin=0 topmargin=0>
<div class="st2">Блок1</div>
<div class="st2">Блок2</div>
<div class="st3">Блок3</div>
<div class="st1">Блок4</div>
</body>
</html>
```

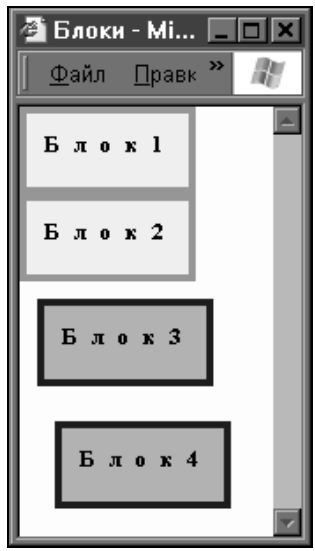


Рис. 7.8. Внешний вид блоков с различными свойствами форматирования в окне браузера

В документе были использованы три стиля. Стиль st1 предусматривает создание поля вокруг блока размером 20 px, стиль st3 — 10 px, а стиль st2 не предусматривает создания полей. Стиль st2 применен к блокам 1 и 2, стиль st1 — к блоку 4, а стиль st3 — к блоку 3. Внешний вид документа в окне браузера показан на рис. 7.8.

По умолчанию страница имеет поля и для того, чтобы при отсутствии собственных полей блоки примыкали к сторонам браузера, нужно задать нулевые значения этих полей. Это делается с помощью параметров LEFTMARGIN и TOPMARGIN метки <BODY>.

7.7.5. Позиционирование объектов

Различают статическое, относительное и абсолютное *позиционирование объектов*. Позиционирование задается свойством position, которое может приобретать следующие значения: static — статическое (по умолчанию), relative — относительное, absolute — абсолютное. Для относительного и абсолютного позиционирования используются еще четыре свойства left и right, задающие смещение объекта по горизонтали, top и bottom — по вертикали. На практике необходимо использовать по одному значению каждой пары.

Статическое позиционирование действует по умолчанию и предусматривает размещение очередного объекта на свободное место вслед за предыдущим. Причем разные объекты будут вести себя по-разному. Например, абзацы будут располагаться друг под другом, даже если ширина абзаца ограничена. Рисунки будут располагаться в одной строке, если для этого достаточно места и предшествующие метки <IMG> не содержат параметра ALIGN. Пример документа, в котором используется статическое позиционирование, приведен в листинге 7.10.

Листинг 7.10. Пример статического позиционирования четырех объектов

```
<body>
<p style="font-size: 14pt; background-color: yellow; width:
150px">Это первый абзац</p>
```

```
<p style="font-size: 14pt; background-color: yellow; posi-
tion: static; width: 150px">Это второй абзац</p>
<img src="1.jpg" width="151" height="100">
<img src="2.jpg" width="142" height="100">
</body>
```

В документе представлено четыре объекта: два абзаца и два рисунка. Все они позиционированы статически. Для второго абзаца позиционирование задано явно, для остальных объектов — по умолчанию. Внешний вид документа в окне браузера показан на рис. 7.9.

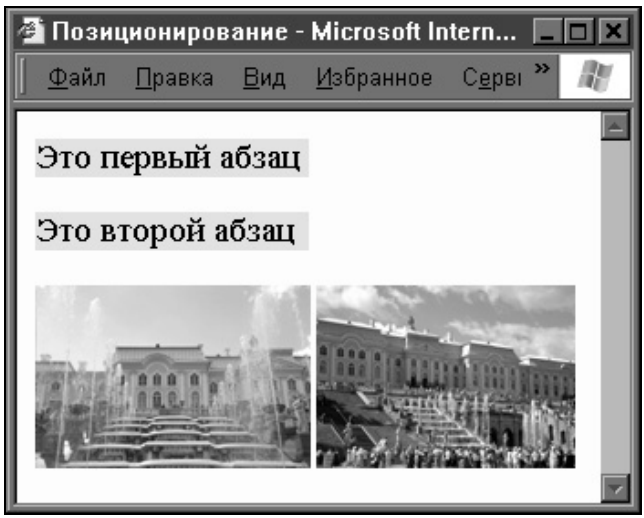


Рис. 7.9. Результат статического позиционирования двух абзацев и двух рисунков

Относительное позиционирование определяет положение объекта относительно того места, которое он бы занял, если бы был позиционирован статически. Значение свойства `left` определяет смещение объекта слева направо, `top` — сверху вниз, `right` — справа налево, `bottom` — снизу вверх относительно места статического позиционирования. Очевидно, что если ни одно из свойств `left`, `top`, `right` или `bottom` не задано, то относительное позиционирование эквивалентно статическому. Последующие объекты позиционируются относительно места статиче-

ского положения объекта независимо от заданных для него смещений. В качестве примера будем использовать тот же документ, что и в предыдущем случае, изменив позиционирование первого рисунка на относительное (листинг 7.11).

Листинг 7.11. Пример относительного позиционирования одного объекта и статического позиционирования трех объектов

```
<body>
<p style="font-size: 14pt; background-color: yellow; width:
150px">Это первый абзац</p>
<p style="font-size: 14pt; background-color: yellow; width:
150px">Это второй абзац</p>
<img src="1.jpg" width="151" height="100" style="position:
relative; left: 80px; top: -50px">
<img src="2.jpg" width="142" height="100">
</body>
```

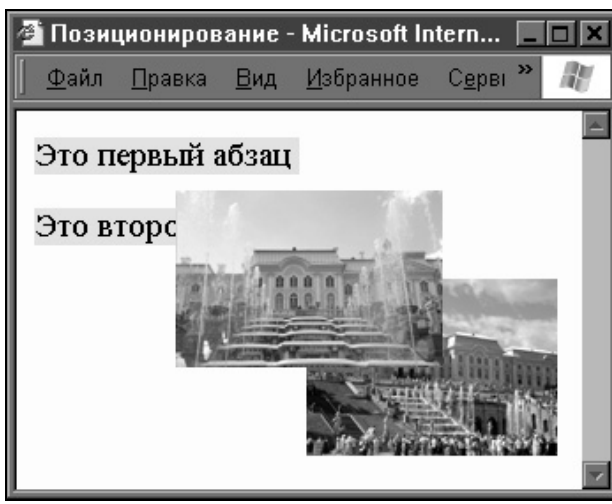


Рис. 7.10. Результат относительного позиционирования одного объекта и статического позиционирования трех объектов

Рисунок смещен вверх на 50 px (отрицательное значение) и вправо на 80 px относительно того положения, которое он занимал

при статическом позиционировании. Особо следует обратить внимание на второй рисунок, который сохранил свое положение. Он позиционирован относительно положения статического позиционирования первого рисунка (рис. 7.10).

Абсолютное позиционирование предусматривает позиционирование объекта с помощью свойств `left`, `top`, `right` или `bottom`. Отсчет смещения объекта по горизонтали, заданного свойством `left`, осуществляется от левого края страницы. Если объект является дочерним, то смещение произойдет от левого края родительского объекта при условии, что родительский объект также имеет абсолютное позиционирование. Аналогично смещение, заданное свойством `right`, отсчитывается от правого края страницы или правого края родительского объекта. Смещение, заданное свойством `top` или `bottom`, отсчитывается соответственно от верхнего края страницы (родительского объекта) или нижнего края страницы (родительского объекта). Объект, позиционированный абсолютно, выпадает из общего потока объектов и его положение никак не влияет на положение следующих за ним объектов.

В приведенном ниже документе первый рисунок получил абсолютное позиционирование (листинг 7.12).

Листинг 7.12. Пример абсолютного позиционирования одного объекта и статического позиционирования трех объектов

```
<body>
<p style="font-size: 14pt; background-color: yellow; width:
150px">Это первый абзац</p>
<p style="font-size: 14pt; background-color: yellow; width:
150px">Это второй абзац</p>
<img src="1.jpg" width="151" height="100" style="position:
absolute; left: 90; top: 50">
<img src="2.jpg" width="142" height="100">
</body>
```

В отличие от предыдущего примера, смещение первого рисунка по горизонтали отсчитывается от левого края страницы, а по вертикали — от верхнего края страницы. Кроме того, он выпал из общего потока статически позиционированных объектов, ос-

вободив место для второго рисунка. Результат показан на рис. 7.11.

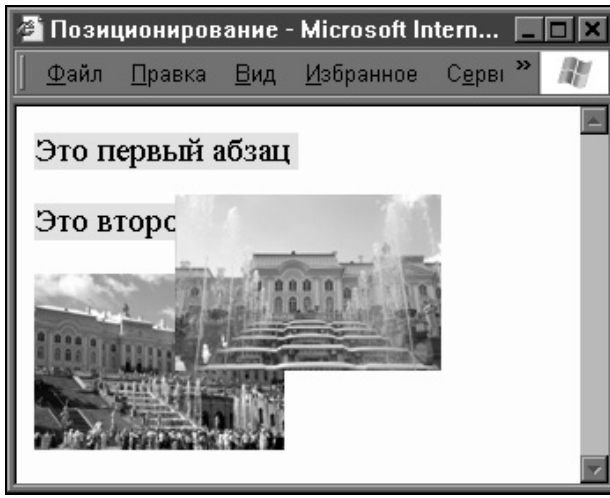


Рис. 7.11. Результат абсолютного позиционирования одного объекта и статического позиционирования трех объектов

7.7.6. Использование слоев

В результате позиционирования одни объекты могут накладываться на другие, перекрывая их. Например, это может быть воплощением художественного замысла автора, расположившего рисунки так, чтобы они частично перекрывали друг друга. В этом случае использование слоев не потребуется, все рисунки по умолчанию будут располагаться в нулевом слое, создавая стопку объектов. В нижней части стопки располагается объект, описанный первым, а сверху — последним. Применение слоев потребуется тогда, когда предполагается динамическое изменение внешнего вида документа, для чего будет использована программа-сценарий. Слой объекта задается с помощью свойства `z-index` следующим образом:

```
z-index: 2
```

В приведенном примере свойству `z-index` присвоено значение `"2"`. Значениями свойства `z-index` могут быть только целые

числа и "auto". Значение "auto" эквивалентно значению "0". Объект с более высоким значением свойства z-index располагается поверх объектов с меньшим значением, перекрывая их. В случае равенства значений выше располагается объект, описанный в документе ниже по тексту.

В программе-сценарии должно быть предусмотрено динамическое изменение свойства z-index, в результате чего объект будет изменять свое положение относительно других объектов, например, перемещаясь поверх остальных. Примеры программ-сценариев, использующих изменение свойства z-index, будут рассмотрены в *главе 9*.

7.7.7. Видимость объектов

Возможность скрывать и вновь показывать объекты представляет интерес только при создании страниц, содержимое которых изменяется динамически в процессе просмотра. Управление внешним видом таких страниц осуществляется с помощью программ-сценариев. Примеры таких программ будут рассмотрены в *главах 8* и *9*. Здесь же нам предстоит предварительно познакомиться с двумя свойствами, позволяющими скрывать объекты или делать их видимыми:

- Display — может принимать значения: "block" — объект отображен на странице, "none" — объект скрыт. Таким образом, в случае применения значения "none" объект полностью удаляется со страницы, и на его место перемещается следующий за ним объект. При выборе значения "block" объект отображается, а последующие объекты смещаются, чтобы освободить для него место. В качестве примера рассмотрим документ в листинге 7.13.

Листинг 7.13. Изменение видимости объекта с использованием свойства display

```
<body>
<p style="font-size:14pt; background-color: yellow; width:
150px">Это первый абзац</p>
<img src="1.jpg" width="151" height="100" style="display:
block">
```

```
<p style="font-size:14pt; background-color: yellow; width:
150px">Это второй абзац</p>
</body>
```

На рис. 7.12 показаны результаты открытия в окне браузера двух документов: первый — с установленным значением "block" свойства display, второй — после изменения значения свойства display на "none".

Рис. 7.12. Результаты открытия одинаковых документов с разными значениями свойства display: block и none

- visibility — может принимать значения: "inherit" — унаследованная видимость, "hidden" — объект скрыт, visible – объект отображен. При выборе "inherit" видимость объекта определяется видимостью родительского объекта. В отличие от свойства display, скрытие объекта не сопровождается его удалением со страницы, поэтому следующие за ним объекты в процессе скрытия/отображения всегда будут оставаться на своих местах. Пример использования свойства visibility приведен в листинге 7.14.

Листинг 7.14. Изменение видимости объекта с использованием свойства visibility

```
<body>
<p style="font-size: 14pt; background-color: yellow; width:
```

```
150px">Это первый абзац</p>
<img src="1.jpg" width="151" height="100" style="visibility:
visible">
<p style="font-size: 14pt; background-color: yellow; width:
150px">Это второй абзац</p>
</body>
```

На рис. 7.13 показаны результаты открытия двух документов со значениями свойства `visibility: visible` и `hidden`.

Рис. 7.13. Результаты открытия одинаковых документов, но с разными значениями свойства visibility: visible и hidden

7.7.8. Изменение внешнего вида указателя мыши

Изменение внешнего вида указателя мыши осуществляется с помощью свойства `cursor`, которое может принимать следующие значения:

- `default` — указатель, используемый в операционной системе по умолчанию;
- `move` — маркер перемещения;
- `hand` или `pointer` — рука;

- `wait` — песочные часы;
- `help` — стрелка со знаком вопроса;
- `crosshair` — перекрестье;
- `text` — указатель для ввода текста (в виде буквы I);
- `e-resize`, `ne-resize`, `n-resize`, `nw-resize`, `w-resize`, `sw-resize`, `s-resize`, `se-resize` — стрелки изменения размера объекта различного направления.

Пример:

```
<p style="cursor: help">Текст</p>
```

Следует иметь в виду, что вид некоторых указателей зависит от персональной настройки операционной системы и может отличаться от стандартных.

7.7.9. Изменение цвета элементов полосы прокрутки

Для полосы прокрутки предусмотрена возможность изменения цвета отдельных ее элементов. Рассмотрим свойства, позволяющие установить данные изменения (рис. 7.14).

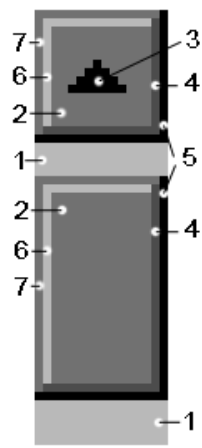


Рис. 7.14. Элементы полосы прокрутки, цвета которых можно изменять

1. `Scrollbar-track-color` — цвет фона полосы прокрутки;
2. `Scrollbar-base-color` — базовый цвет полосы;
3. `Scrollbar-arrow-color` — цвет треугольных стрелок;
4. `Scrollbar-shadow-color` — цвет тени на полосе;
5. `Scrollbar-darkshadow-color` — цвет темной тени на полосе;
6. `Scrollbar-highlight-color` — цвет светлого блика на полосе;
7. `Scrollbar-3dlight-color` — цвет крайнего блика на полосе.

В большинстве случаев можно обойтись гораздо меньшим количеством свойств. Дело в том, что при изменении только базового цвета полос прокрутки (свойство `scrollbar-base-color`), фон полосы, тени и блики будут заполнены оттенками базового цвета. Изменив еще и цвет треугольных стрелок (свойство `scrollbar-arrow-color`), можно вполне обойтись только двумя свойствами. Для изменения цвета полос прокрутки всей страницы необходимо разместить соответствующие свойства в метке `<BODY>`. Например:

```
<BODY style="scrollbar-base-color: green; scrollbar-arrow-
color: yellow">
```

Цвет полос прокрутки можно менять и в отдельных блоках. Для этого необходимые свойства размещаются в метке, образующей блок. Например:

```
<DIV style="overflow: auto; scrollbar-base-color: red;
scrollbar-arrow-color: blue; width: 300; height:
200">Содержимое блока</DIV>
```

7.8. Работа с каскадными таблицами стилей в программе Dreamweaver

Работа с каскадными таблицами стилей может включать:

- создание новой таблицы стилей. В результате таблица может быть размещена либо только в том документе, для которого создается, либо сохранена в виде отдельного файла. В последнем случае она может использоваться и другими документами;

- редактирование таблицы стилей. Вносятся изменения в уже созданную таблицу стилей;
- присоединение к документу существующих таблиц стилей. Подключаются таблицы стилей, созданные ранее для других документов и сохраненные в виде отдельных файлов;
- применение стиля к определенному фрагменту документа. Документ может содержать несколько таблиц стилей, находящихся как внутри документа, так и в присоединенных к нему файлах. Применить стиль означает указать, на какой фрагмент документа будет действовать тот или иной стиль.

Все перечисленные виды работ с каскадными таблицами стилей подробно описаны в *разделе 7.8.1*. Однако на этом возможности программы Dreamweaver для работы с CSS далеко не исчерпываются. Программа располагает удобными инструментами, работающими со слоями. Слои играют важную роль в создании динамических страниц, т. е. страниц изменяющих свой внешний вид в зависимости от действий пользователя. Примеры создания таких страниц рассмотрены в *главе 8*. Другим важным направлением использования слоев является компоновка элементов web-страницы. Мы уже знакомы с двумя способами компоновки страниц. Для этой цели использовались таблицы (*глава 4*) и фреймы (*глава 5*). Компоновке страниц с использованием слоев посвящен отдельный раздел настоящей главы.

7.8.1. Создание новой таблицы стилей

Приступить к созданию новой таблицы стилей можно несколькими способами:

- воспользоваться командой меню **Text | CSS Styles | New** (Текст | CSS стили | Создать);
- открыть панель **CSS Styles** (CSS стили), используя команду меню **Window | CSS Styles** (Окно | CSS стили), входящую в состав инструментальных панелей (рис. 7.15).

  Щелкнуть по кнопке **New CSS Style** (Создать CSS стиль) с изображением знака "+" в нижней части панели;
- на панели свойств **Properties** (Свойства) в режиме работы с текстом (рис. 2.8) раскрыть список **Style** (Стиль) и выбрать

команду **Manage Styles** (Управление стилями). В открывшемся диалоговом окне **Edit Style Sheet** (Редактирование таблицы стилей) (рис. 7.16), а при наличии списка только внутренних стилей — **<style>** нужно нажать кнопку **New** (Создать).

Рис. 7.15. Инструментальная панель **CSS Styles**

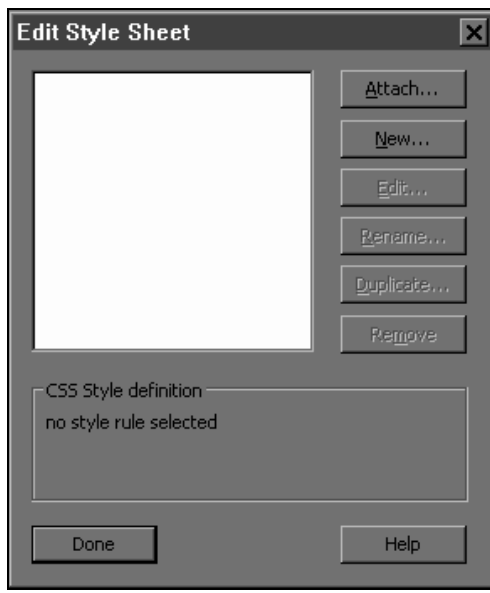


Рис. 7.16. Диалоговое окно **Edit Style Sheet**

Все перечисленные способы приводят к открытию диалогового окна **New CSS Style** (Создать CSS стиль), в котором необходимо

воспользоваться переключателем **Selector Type** (Выбор типа) и выбрать тип создаваемой каскадной таблицы стилей (рис. 7.17).

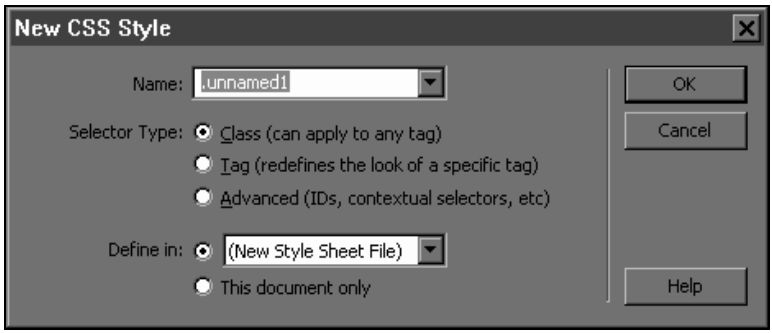


Рис. 7.17. Диалоговое окно **New CSS Style**

При выборе **Tag (redefines the look of a specific tag)** (Метка (изменение свойств метки)), мы получим возможность изменять свойства любой метки HTML. Верхний раскрывающийся список станет называться **Tag** (Метка). В нем можно выбрать метку, свойства которой мы хотим изменить. Переключатель **Define in** (Определить в) позволяет выбрать будет ли каскадная таблица стилей храниться в отдельном файле (переключатель в верхнем положении) или она будет размещаться в разделе `<HEAD>` документа внутри блока, отмеченного метками `<STYLE>...</STYLE>` — переключатель в положении **This document only** (Только этот документ).

При создании каскадной таблицы стилей в виде отдельного файла верхний список переключателя **Define In** (Определить в) содержит единственную запись New Style Sheet File (Новый файл таблицы стилей). При повторных открытиях окна **New CSS Style** (Создать CSS стиль) список переключателя будет содержать имена файлов, сохраненных ранее. Создаваемая таблица стилей может быть внедрена в существующий файл или сохранена в новом файле.

Установив переключатель **Selector Type** (Выбор типа) в положение **Class (can apply to any tag)** (Класс (может применяться к любой метке)), получаем возможность создания класса стилей, применяемого к различным меткам. Имя класса нужно вписать

в строку **Name** (Имя), а при повторных обращениях его можно выбрать из списка. Так же как и в предыдущем случае, с помощью переключателя **Define In** (Определить в) можно сохранить класс стилей в отдельном файле либо поместить его внутри блока `<STYLE>...</STYLE>`, создаваемого документа.

Перевод переключателя **Selector Type** (Выбор типа) в положение **Advanced (IDs, contextual selectors, etc)** (Дополнительные возможности) позволяет из верхнего списка **Selector** (Список выбора) выбрать для редактирования один из четырех так называемых псевдоклассов. К ним относятся: a:link — свойства обычной гиперссылки, a:active — свойства активной (последней посещенной) гиперссылки, a:visited — свойства посещенной гиперссылки, a:hover — свойства гиперссылки, на которой установлен курсор.

Если переключатель **Define In** (Определить в) был установлен в верхнее положение, то после завершения работы в окне **New CSS Style** (Создать CSS стиль) нам будет предложено сохранить создаваемую таблицу стилей в виде отдельного файла. Откроется диалоговое окно **Save Style Sheet File As** (Сохранить файл таблицы стилей как), в котором необходимо дать файлу имя и указать папку, в которой его нужно сохранить. После этого откроется диалоговое окно **CSS Style definition for** (Определение CSS стиля для), в котором выбранным свойствам можно задать значения.

Если же было выбрано нижнее положение переключателя **Define In** (Определить в), то сразу после завершения работы в окне **New CSS Style** (Создать CSS стиль) откроется диалоговое окно **CSS Style definition for** (Определение CSS стиля для) (рис. 7.18).

Свойства стилей для удобства разбиты на 8 категорий, которые перечислены в разделе **Category** (Категория):

1. **Type** (Текст) — форматирование текста. Позволяет изменить следующие параметры форматирования текста:
   - **Font** (Шрифт) — гарнитуру шрифта;
   - **Size** (Размер) — размер шрифта;
   - **Style** (Стиль) — начертание шрифта. Из списка можно выбрать normal (обычное), italic (курсивное), oblique (наклонное);

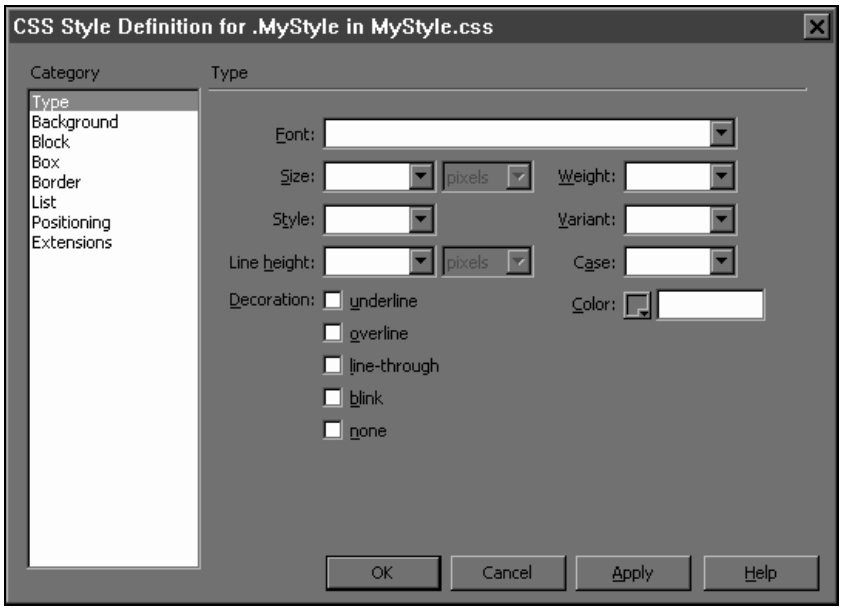


Рис. 7.18. Диалоговое окно **CSS Style definition for...**

- **Line Height** (Высота строки) — расстояние между строками;
- **Decoration** (Оформление) — оформление шрифта. Устанавливая соответствующие флажки можно задать: underline (подчеркивание), overline (надчеркивание), line-through (зачеркивание), blink (мигание) — не поддерживается браузером MS Internet Explorer, none (отсутствие оформления);
- **Weight** (Толщина) — толщина шрифта (степень жирности);
- **Variant** (Вариант) — изображение строчных букв как заглавных (small-caps) или обычный шрифт (normal);
- **Case** (Регистр) — смена регистра символов. Возможны варианты: capitalize (первые буквы всех слов прописные), uppercase (весь текст прописными буквами), lowercase (весь текст строчными буквами), none (отсутствие смены регистра);
- **Color** (Цвет) — цвет текста.

2. **Background** (Фон). Позволяет задать следующие параметры фона:
   - **Background Color** — цвет фона;
   - **Background Image** (Фоновое изображение) — имя файла фонового графического изображения;
   - **Repeat** (Повторение) — повторяемость фонового изображения. Возможен выбор значений: no-repeat (не повторять), repeat (повторять по горизонтали и вертикали — по умолчанию), repeat-x (повторять по горизонтали), repeat-y (повторять по вертикали);
   - **Attachment** (Привязка) — взаимодействие фона с содержимым страницы при перемещении содержимого с помощью полос прокрутки (скроллинга): fixed (содержимое страницы перемещается относительно неподвижного фона), scroll (содержимое страницы перемещается вместе с фоном по умолчанию);
   - **Horizontal Position** (Горизонтальная позиция) — позиционирование фоновой картинки по горизонтали относительно границ объекта, в котором задан фон: left (выравнивание по левому краю), center (центрирование), right (выравнивание по правому краю), value (задание произвольного значения смещения);
   - **Vertical Position** (Вертикальная позиция) — позиционирование фоновой картинки по вертикали: top (выравнивание по верхнему краю), center (центрирование), bottom (выравнивание по нижнему краю), value (задание произвольного значения смещения).
3. **Block** (Блок) — форматирование текста. Дополняет категорию **Type** новыми возможностями:
   - **Word spacing** — расстояние между словами: normal (обычное), value (произвольное значение);
   - **Letter spacing** — расстояние между буквами: normal (обычное), value (произвольное значение);
   - **Vertical alignment** — положение элемента по вертикали, относительно родительского элемента. Наибольший практический интерес представляют значения: top (совмеща-

ются верхние края элементов), middle (базовая линия элемента выравнивается на уровне средней точки родителя), bottom (совмещаются нижние края элементов);

- **Text align** — выравнивание текста по горизонтали относительно элемента, в котором он находится. Возможные значения: left (по левому краю), right (по правому краю), center (по центру), justify (по ширине);
- **Text indent** (Текстовый отступ) — отступ первой строки абзаца;
- **Whitespace** (Пробел) — выбор способа обработки пробелов и строк. Возможны значения: normal (при демонстрации документа пробелы идущие подряд сокращаются до одного по умолчанию), pre (сохраняется предварительное форматирование текста, включая пробелы), nowrap (запрещает переход на новую строку);
- **Display** — отображение или скрытие объекта. Наибольший практический интерес представляют значения: block (отображать объект), none (скрыть объект).

4. **Box** (Контейнер). Позволяет оформить фрагмент текста в виде прямоугольного блока, задав: размеры блока, способы выравнивания внутри блока, отступы и т. д. Возможно задание следующих параметров:
   - **Width**, **Height** — ширина и высота блока;
   - **Padding** (Отступ) — группа параметров, задающих отступы между границами блока и его содержимым. Группа включает следующие поля для задания параметров: top (отступ от верхней границы), right (от правой границы), bottom (от нижней границы), left (от левой границы). При установленном флажке **Same for All** (Одинаковые для всех) все отступы будут одного размера;
   - **Float** (Плавающий) — блок становится "плавающим". В зависимости от заданного значения он будет примыкать к левому краю страницы (left) или к правому краю (right). Значение none отменяет действие свойства **float** (плавающий). "Плавающий" блок влияет на положение следующих за ним объектов. Если позволит ширина страницы, объекты будут обтекать "плавающий блок", располагаясь

на одной горизонтальной линии с ним. Таким образом, можно расположить на одной горизонтальной линии несколько абзацев, которые в обычных условиях должны были бы расположиться друг под другом. Например:

```
<body>
<p style="float: left; width: 100px; background-color:
#FFFF00">Первый абзац </p>
<p style="float: left; width: 100px; background-color:
#00FFFF">Второй абзац</p>
<p style="float: left; width: 100px; background-color:
#FF00FF">Третий абзац</p>
</body>
```

Результат применения свойства **float** со значением left к трем абзацам показан на рис. 7.19.

Рис. 7.19. Результат применения свойства float со значением left к трем абзацам

- **Clear** (Очищать) — используется совместно со свойством **float**. Запрещает обтекание объекта слева (left), справа (right), с обеих сторон (both). Значение none отменяет действие свойства **clear**;
- **Margin** (Поле) — группа параметров, задающих расстояния между границами блока и соседними элементами страницы. Значения параметров те же, что и в группе **Padding** (Отступ).

5. **Border** (Рамка). В этой категории можно задавать три параметра рамки:
   - **Style** — стиль рамки. Из раскрывающегося списка можно выбрать: none (нет рамки по умолчанию), dotted (пунктирная), dashed (прерывистая), solid (непрерывная), double (двойная), groove (углубленная), ridge (выпуклая), insert

(эффект углубления блока), outset (эффект выпуклости блока);

- **Width** — толщина рамки. Список содержит значения: thin (тонкая рамка), medium (средняя), thick (широкая), value (произвольное значение);
- **Color** — цвет рамки.

Если снят флажок **Same for All** (Одинаковые для всех), то значение параметра можно задавать отдельно для каждой из сторон рамки: top (верхняя), left (левая), right (правая), bottom (нижняя).

6. **List** (Список). Возможна установка следующих параметров списка:
   - **Type** (Тип) — тип маркера или номера. Варианты выбора: disc (закрашенная окружность), circle (не закрашенная окружность), square (закрашенный квадрат), decimal (арабские цифры), lower-roman (маленькие римские цифры), upper-roman (большие римские цифры), lower-alpha (маленькие латинские буквы), upper-alpha (большие латинские буквы), none (без маркеров или номеров);
   - **Bullet Image** (Изображение маркера) — имя файла графического изображения, используемого в качестве маркера. Нажатие кнопки **Browse** (Поиск) открывает диалоговое окно **Select Image Source** (Выбор источника изображения) для поиска файла рисунка;
   - **Position** (Положение) — положение маркера в списке: inside (внутри), outside (снаружи). Разница заметна только при наличии элементов списка, занимающих более одной строки. В первом случае вторая и последующие строки будут начинаться под маркером, во втором — выравниваются по первой строке.
7. **Positioning** (Позиционирование). Основное назначение данной категории – создание блоков и размещение их на странице, но в отличие от категории **Box** (Контейнер) она содержит больше возможностей. *Знакомясь со свойствами данной категории, можно обращаться к разделам 7.7.5, 7.7.6 и 7.7.7, в которых даны более подробные разъяснения и приведены примеры.*

- **Type** (Тип) — тип позиционирования. Из списка можно выбрать один из типов позиционирования: absolute (абсолютное), relative (относительное), static (статическое);
- **Width** (Ширина), **Height** (Высота) — размеры блока;
- **Placement** (Размещение) группа параметров, определяющих положение блока. Горизонтальное положение блока определяется параметрами Left (величина смещения слева направо) и Right (справа налево), а вертикальное параметрами Top (величина смещения сверху вниз) и Bottom (снизу вверх). Имеет смысл выбора только по одному значению из каждой пары для абсолютного и относительного типов позиционирования. При абсолютном позиционировании смещение отсчитывается от краев страницы или родительского блока, при относительном — от положения объекта в случае его статического позиционирования;
- **Visibility** (Видимость) — список содержит следующие значения: inherit (унаследованная видимость), visible (видимый), hidden (скрытый);
- **Z-Index** (Номер слоя) — поле, позволяющее задать номер слоя, в котором будет располагаться блок;
- **Overflow** (Переполнение) — список действий при переполнении блока, т. е. в тех случаях, когда содержимое блока превышает его размеры: visible (содержимое блока, не поместившееся внутри, остается видимым по умолчанию), hidden (содержимое блока, оказавшееся за его пределами, будет скрыто), scroll (независимо от объема содержимого блок снабжается полосами прокрутки), auto (полосы прокрутки устанавливаются только при переполнении блока);
- **Clip** (Область просмотра) — группа параметров, задающих прямоугольную область просмотра, за пределами которой содержимое блока будет скрыто. Параметры Top и Bottom отсчитываются от верхней границы блока и задают расстояния до верхней и нижней границ области просмотра. Параметры Right и Left отсчитываются от левой границы блока и задают расстояния до правой и левой границ области просмотра.

8. **Extensions** (Дополнения) — дополнительные возможности. В этой категории практический интерес представляют следующие свойства:
   - **Cursor** (Курсор) — позволяет изменить внешний вид курсора, при наведении его на данный объект. Из списка можно выбрать следующие значения: hand (рука), crosshair (перекрестье), text (в виде буквы I), wait (песочные часы), help (стрелка со знаком вопроса), e-resize, ne-resize, n-resize, nw-resize, w-resize, sw-resize, s-resize, se-resize (стрелки изменения размера объекта различного направления), default (по умолчанию), auto (форма зависит от типа элемента);
   - **Filter** (Фильтр) — создание эффектов с использованием статических и динамических фильтров. Следует обратить внимание, что программа Dreamweaver MX 2004 для задания фильтров использует устаревший синтаксис, который применялся в браузерах MS Internet Explorer версий 4.0 и 5.0. Начиная с версии 5.5, Microsoft изменила синтаксис задания фильтров и расширила их количество (*см. главу 9*). В браузерах MS Internet Explorer 5.5 и 6.0 фильтры, заданные с использованием старого синтаксиса также работают. Читателю необходимо самому определиться, будет ли он с целью ускорения работы задавать фильтры в программе Dreamweaver, рискуя, что в новых версиях браузеров они работать не будут, или будет использовать ручное задание фильтров, описанное в *главе 9*.

7.8.2. Редактирование таблицы стилей

Для редактирования уже имеющейся таблицы стилей можно воспользоваться следующими способами:

- Воспользоваться командой меню **Text | CSS Styles | Manage Styles** (Текст | CSS стили | Управление стилями) или раскрыть список **Style** (Стиль) на панели свойств **Properties** (Свойства) в режиме работы с текстом и выбрать команду **Manage Styles** (Управление стилями). Откроется диалоговое окно **Edit Style Sheet** (Редактирование таблицы стилей), которое будет содержать список файлов присоединенных внешних таблиц стилей (рис. 7.20). Следует отметить, что стили, размещенные внутри самого документа, скрыты в окне за меткой <style>.

Двукратный щелчок по <style> раскрывает одноименное диалоговое окно со списком внутренних стилей. Если внешние стили отсутствуют, то сразу же раскроется окно <**style**> (рис. 7.20). Редактируемый стиль нужно выделить и нажать кнопку **Edit** (Правка);

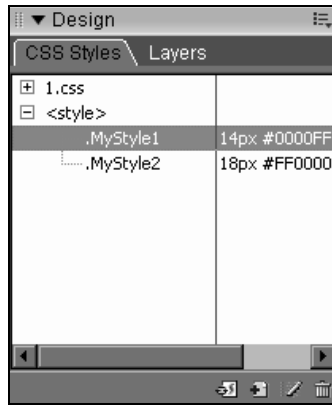


Рис. 7.20. Диалоговые окна **Edit Style Sheet** (открывается при наличии присоединенных внешних таблиц стилей и стилей внутри самого документа) и <**style**> (содержит список внутренних стилей)

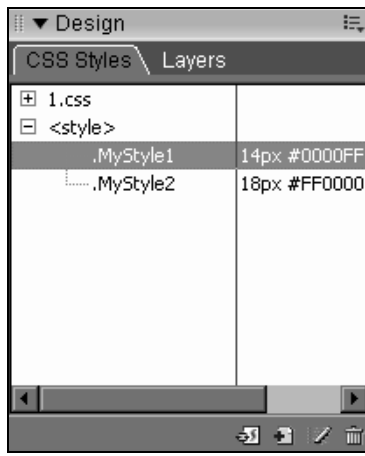


Рис. 7.21. Инструментальная панель **CSS Styles**

- Открыть панель **CSS Styles** (CSS стили), используя команду меню **Window | CSS Styles** (Окно | CSS стили), входящую в состав инструментальных панелей;
- Выделить редактируемый стиль и щелкнуть по кнопке **Edit Style** (Правка стиля) в нижней части панели (вторая справа).

Во всех перечисленных случаях откроется диалоговое окно **CSS Style definition for** (Определение CSS стиля для), в котором необходимо приступить к редактированию стиля. Работа по редактированию стиля ничем не отличается от его создания.

7.8.3. Присоединение внешней таблицы стилей

Если каскадная таблица стилей была сохранена в виде отдельного файла с расширением CSS, то она может быть присоединена к любому HTML-документу одним из следующих способов:

- Командой меню **Text | CSS Styles | Manage Styles** (Текст | CSS стили | Управление стилями) или командой **Manage Styles** (Управление стилями) списка **Style** (Стиль) на панели свойств **Properties** (Свойства) можно открыть диалоговое окно **Edit Style Sheet** (Редактирование таблицы стилей) (**<style>**) и нажать кнопку **Attach** (Присоединить);
- Открыть панель **CSS Styles** (CSS стили), используя команду меню **Window | CSS Styles** (Окно | CSS стили), и щелкнуть по кнопке **Attach Style Sheet** (Присоединить таблицу стилей) в нижней части панели (первая слева).

При использовании любого способа откроется диалоговое окно **Attach External Style Sheet** (Присоединить внешнюю таблицу стилей), приведенное на рис. 7.22.

Нажав кнопку **Browse** (Поиск) необходимо отыскать нужный файл, определив для него путь в строке **File/URL** (файл/URL-адрес). Переключатель **Add As** (Добавить как) можно установить в положение **Link** (связь CSS файла с HTML-документом будет установлена с помощью метки `<LINK>`) или **Import** (файл будет импортирован с помощью инструкции "@import"). После подключения, имя файла каскадной таблицы стилей появится в списке стилей. Фразой "Dreamweaver has sample style sheets to get you started" ("Dreamweaver располагает образцами таблиц сти-

лей") программа напоминает о наличии типовых стилей, к которым можно обратиться, щелкнув по гиперссылке.

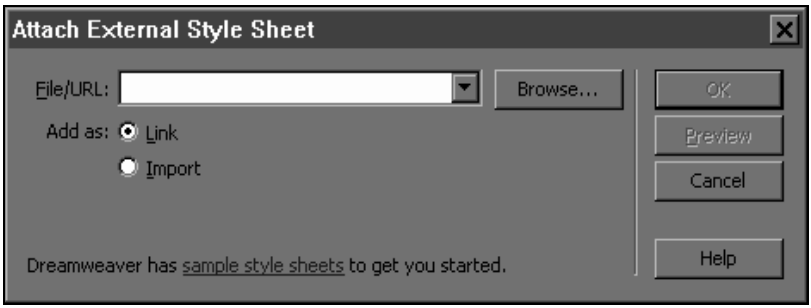


Рис. 7.22. Диалоговое окно **Attach External Style Sheet**

7.8.4. Применение стиля форматирования или отказ от его применения

Для применения стиля к целому абзацу курсор должен быть установлен в любом месте абзаца. Если требуется применить стиль к определенной части абзаца или одновременно к нескольким абзацам, то они предварительно должны быть выделены. Имя стиля выбирается из списка стилей **Style** (Стиль) на панели свойств **Properties** (Свойства) или из списка стилей в меню **Text | CSS Styles** (Текст | CSS стили). Стили можно применять не только к тексту, но и к другим объектам, например рисункам. Для отказа от применения стиля нужно выбрать команду **None** (Не применять) списка стилей одним из вышеприведенных способов.

Пример создания текстового блока

Приступим к созданию нового HTML-документа.

- Одним из способов *(см. раздел 7.8.1)* откроем диалоговое окно **New CSS Style** (Создать CSS стиль). В диалоговом окне установим переключатель **Selector Type** (Выбор типа) в положение **Class (can apply to any tag)** (Класс (может применяться к

любой метке)), переключатель **Define In** (Определить в) в положение **This Document Only** (Только этот документ). В строку **Name** (Имя) впишем имя стиля, например, "style1".

- После нажатия **OK,** откроется диалоговое окно **CSS Style definition for .style1** (Определение стиля для .style1). Воспользуемся категорией **Type** (Текст), выбрав в списке **Size** (Размер) значение 14, а в списке **Weight** (Толщина) — 600. В категории **Background** (Фон) раскроем палитру **Background Color** (Цвет фона) и выберем желтый цвет. Перейдем в категорию **Box** (Прямоугольный контейнер). В полях **Width** (Ширина) и **Height** (Высота) установим ширину (300 px) и высоту (70 px) блока, а в группе параметров **Padding** (Отступ) зададим отступы между границами блока и его содержимым на 10 px, поместив это значение в строку **Top** (флажок **Same for All** (Одинаковые для всех) должен быть установлен). Перейдем в категорию **Border** (Рамка). В группе параметров **Style** (Стиль рамки) выберем из списка значение solid, в группе **Width** (Толщина рамки) — medium и в группе **Color** (Цвет) укажем в палитре синий цвет. В категории **Positioning** (Позиционирование) и в списке **Type** (Тип) выберем значение absolute, в списке **Overflow** (Переполнение) — auto. В группе параметров **Placement** (Размещение) установим **Left** (Слева) — 20, **Top** (Сверху) — 20. Нажав кнопку **OK**, в визуальном режиме получим заготовку блока, готовую для заполнения.
- В качестве метки контейнера блока используется метка абзаца `<P>`. После щелчка мышью внутри блока и появления мигающего курсора введем следующий текст: `Этот текст оформлен в виде отдельного блока, размеры которого 300 × 70 px. К блоку применено абсолютное позиционирование, он имеет фоновую заливку, рамку и полосу прокрутки. Все это осуществлено с помощью CSS.` Применим к тексту форматирование по ширине, нажав для этого кнопку **Justify** (По ширине) на панели свойств **Properties** (Свойства).

Текст полученного документа представлен в листинге 7.15.

Листинг 7.15. Создание текстового блока с использованием свойств CSS

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

```
<title>текстовый блок</title>
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
<style type="text/css">
<!--
.style1 {
   font-size: 14px;
   font-weight: 600;
   background-color: #FFFF00;
   padding: 10px;
   height: 70px;
   width: 300px;
   border: medium solid #0000FF;
   position: absolute;
   left: 20px;
   top: 20px;
   overflow: auto;
   }
-->
</style>
</head>
<body>
<p class="style1">Этот текст … </p>
</body>
</html>
```

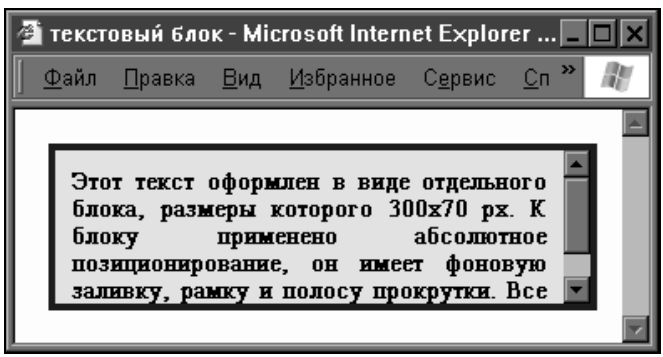


Рис. 7.23. Вид текстового блока в окне браузера

Для просмотра документа в браузере нажмем <F12>. На рис. 7.23 показан вид блока в окне браузера.

В текстовый блок, созданный путем применения стилей к метке абзаца <P>, можно поместить и рисунок, сохранив все параметры блока.

- В визуальном режиме установим курсор перед текстом и дадим команду меню **Insert | Image** (Вставка | Рисунок). В открывшемся диалоговом окне выберем имя файла рисунка. Если, начиная работу над документом, мы указали, к какому сайту он относится, то после нажатия **OK** нам будет предложено скопировать файл рисунка в папку сайта, с чем нужно согласиться.
- Вставленный рисунок выделен, следовательно, панель свойств **Properties** (Свойства) содержит свойства рисунка. Панель следует раскрыть до полного размера, щелкнув по треугольной стрелке в правом нижнем углу. В списке **Align** (Выравнивание) выберем left, в поле **H Space** (Отступ по горизонтали) введем значение 4.

Вид блока с рисунком 150 × 100 px в окне браузера приведен на рис. 7.24.

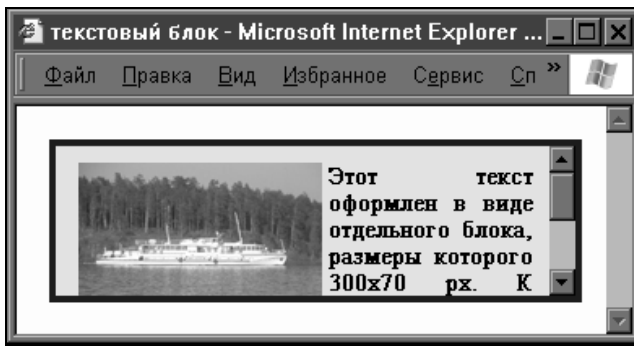


Рис. 7.24. Вид в окне браузера блока, включающего рисунок и текст

7.8.5. Работа со слоями

Как уже отмечалось, категория **Positioning** (Позиционирование) диалогового окна **CSS Style definition for** (Определение CSS сти-

ля для) располагает возможностями по работе со слоями. В частности, для прямоугольного блока, размеры которого задаются в этой же категории, можно указать номер слоя, в котором он должен находиться, и задать видимость слоя.

В программе Dreamweaver для этих целей есть более удобный инструмент. Чтобы отобразить этот инструмент, необходимо перейти в стандартный режим: нажать кнопку **Standard** (Стандартный режим) в группе инструментов **Layout** (Компоновка) панели **Insert** (вставка) (рис. 7.25), после чего становится активной кнопка **Draw Layer** (рисование слоя) .

Рис. 7.25. Группа инструментов **Layout** панели **Insert**

Щелкнем по кнопке **Draw Layer** (Рисование слоя). На странице указатель мыши примет вид небольшого крестика. Установив его на то место, где предполагается размещение левого верхнего угла слоя, нужно нажать кнопку мыши и, не отпуская ее, растянуть прямоугольник до нужного размера. Отпустив кнопку, подвести указатель к границе слоя, чтобы он принял вид перекрестья со стрелками, и щелкнуть мышью. Вокруг слоя появятся маркеры, а панель свойств **Properties** (Свойства) изменит свой вид (рис. 7.26).

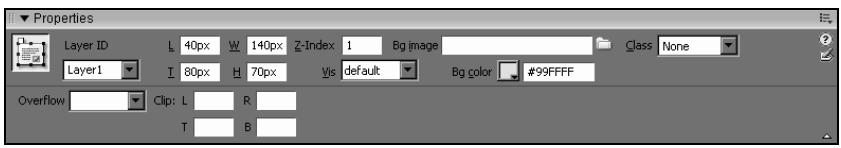


Рис. 7.26. Панель свойств **Properties** в режиме изменения свойств слоя

Изменять размеры слоя и перемещать его можно как с помощью мыши, так и соответствующих полей на панели свойств **Properties** (Свойства). Поля **L** (Left) и **T** (Top) определяют положение левого верхнего угла слоя относительно левого и верхнего краев страницы, а поля **W** (Width) и **H** (Height) — ширину и высоту слоя.

Рассмотрим предназначение других элементов панели свойств **Properties** (Свойства) для работы со слоями.

- Поле **Z-Index** содержит номер слоя. Для слоя, построенного первым, будет задано значение 1, для последующих слоев оно будет увеличиваться на единицу.
- Список **Vis** (Visibility) позволяет выбрать один из четырех вариантов видимости слоя: default (по умолчанию), inherit (унаследованная видимость), visible (видимый), hidden (скрытый).
- Поле **Bg Image** позволяет задать имя файла фонового изображения слоя, а поле **Bg Color** — однородный цвет фона.
- Поле **Layer ID** содержит индивидуальное имя слоя, которое используется программой-сценарием при изменении его свойств.
- В списке **Overflow** (Переполнение) выбираются действия при переполнении слоя: visible (не поместившаяся часть содержимого слоя остается видимой по умолчанию); hidden (не поместившаяся часть содержимого слоя будет скрыта); scroll (слой снабжается полосами прокрутки); auto (полосы прокрутки устанавливаются только при переполнении слоя).
- Поля группы **Clip** задают прямоугольную область просмотра, за пределами которой содержимое слоя будет скрыто. В поля **T** (Top) и **B** (Bottom) вводятся расстояния от верхней границы слоя до верхней и нижней границ области просмотра. В поля **R** (Right) и **L** (Left) – расстояния от левой границы слоя до правой и левой границ области просмотра.

Для удаления слоя его необходимо выделить и нажать клавишу <Delete>.

Создавая большое количество слоев, удобно пользоваться инструментальной панелью **Layers** (Слои) (рис. 7.27), которая открывается по команде меню **Window | Layers** (Окно | Слои).

С помощью инструментальной панели **Layers** (Слои) можно изменять параметры отображения слоя в документе. Это позволяет выполнить значок с изображением глаза. Щелчок по полю левее имени слоя устанавливает изображение закрытого глаза, при этом слой становится невидим. Второй щелчок позволяет открыть глаз — слой делается видимым. Третьим щелчком глаз

удаляется (слой видим). Установка изображения глаза означает, что в метку слоя `<DIV>` вводится параметр `VISIBILITY` со значением hidden (скрытый) – глаз закрыт или visible (видимый) – глаз открыт. После удаления изображения глаза параметр `VISIBILITY` также удаляется.

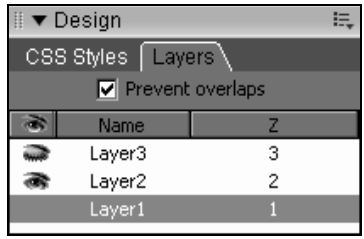


Рис. 7.27. Инструментальная панель **Layers** с тремя слоями

Захватив имя слоя мышью, его можно переместить выше или ниже других слоев, при этом номер слоя (значение свойства `Z-Index`) будет автоматически изменено. Номер слоя можно изменить вручную, щелкнув по нему мышью. После изменения номера нужно щелкнуть по имени слоя, и он переместится в соответствии с новым номером.

Имя слоя может быть изменено после двукратного щелчка по нему мышью. Выделение имени слоя на панели **Layers** (Слои) сопровождается выделением соответствующего слоя на странице.

Установка флажка **Prevent Overlaps** (Предотвращать перекрытия) запрещает наложение слоев друг на друга.

Слои можно вкладывать друг в друга. Слой, выполняющий функции контейнера, будет являться *родительским* по отношению к вложенным в него слоям, называемым *дочерними* слоями. Вложения слоев не следует понимать буквально. Вложенный слой может располагаться за пределами родительского слоя и иметь размеры, значительно превышающие размеры родительского слоя. Зависимость дочернего слоя от родительского проявляется в следующем:

- координаты дочернего слоя отсчитываются от левого верхнего угла родительского слоя, а не всей страницы. В связи

с этим перемещение родительского слоя приводит к перемещению дочерних слоев;

- дочерние слои наследуют некоторые свойства родительских слоев. Например, если свойство Visibility (Видимость) дочернего слоя имеет значение inherit, то его видимость будет зависеть от видимости родительского слоя;
- независимо от номера слоя дочерние слои располагаются поверх родительского, а их расположение относительно других слоев определяется номером родительского слоя. Номер дочернего слоя определяет лишь его положение относительно других дочерних слоев. Например, пусть имеется три слоя: S1, S2 и S3 с номерами 1, 2 и 3 соответственно. Слой S2 содержит дочерние слои D1 и D2 с номерами слоев 1 и 2. Слои расположатся в следующей последовательности, начиная с нижнего слоя: S1, S2, D1, D2, S3.

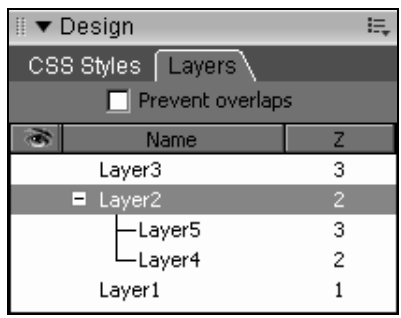


Рис. 7.28. Инструментальная панель **Layers** содержащая пять слоев, два из которых вложены в слой Layer2

Вложение слоев можно осуществлять следующими способами:

- расположить курсор внутри родительского слоя и дать команду меню **Insert | Layout Objects | Layer** (Вставка | Объекты компоновки | Слой);
- используя инструментальную панель **Layers** (Слои), захватить мышью имя слоя и, удерживая клавишу <Ctrl>, наложить его на имя родительского слоя, отпустить кнопку мыши.

Результат вложения слоев Layer4 и Layer5 в слой Layer2 показан на рис. 7.28.

Используя механизм вложения слоев, можно управлять свойствами нескольких дочерних слоев, изменяя свойства только родительского слоя. Данная возможность может потребоваться при создании динамических страниц, для которых необходима программа-сценарий.

Пример компоновки страницы с использованием слоев

Приведем пример компоновки страницы с использованием слоев.

- Откроем пустой шаблон для создания HTML-документа.
- Используя кнопку **Draw Layer** (Рисование слоя), в левом верхнем углу рабочего поля изобразим слой, размеры которого можно сделать немного меньше размеров рисунка, помещаемого на этот слой. Аналогично разместим еще три слоя по диагонали рабочего поля (рис. 7.29).
- В центральной части изобразим еще один слой для размещения в нем текста. Щелкнем мышью за пределами созданных слоев и напишем слово `ПЕТРОДВОРЕЦ`.
- Осуществим форматирование текста, используя панель свойств **Properties** (Свойства): **Font** (Гарнитура) — Monotype Corsiva, **Size** (Размер) — 7, **B** (Начертание) — Bold (полужирное).
- Вырежем текст, воспользовавшись кнопкой **Cut** (Вырезать) на панели инструментов **Standard** (Стандартная) и через буфер обмена вставим в предназначенный для него слой, используя кнопку **Paste** (Вставить) на этой же панели.
- В остальные слои вставим предназначенные для них рисунки, используя команду меню **Insert | Image** (Вставка | Рисунок).
- Командой меню **Window | Layers** (Окно | Слои) откроем панель инструментов **Layers** (Слои). В списке слоев должно находиться пять слоев от Layer1 до Layer5. Захватив мышью имя слоя Layer2 и удерживая клавишу <Ctrl>, наложим его на имя слоя Layer1. Теперь слой Layer2 стал дочерним по отношению к слою Layer1. Аналогично слой Layer3 сделаем дочерним по отношению к слою Layer4. Перемещая дочерние

слои, добьемся того, чтобы они заняли нужное положение относительно родительских слоев. Перемещать выделенный слой можно не только мышью, но и клавишами управления курсором.

- После этого можно приступить к изменению положения родительских слоев Layer1, Layer4 и слоя с текстом Layer5 относительно друг друга, добиваясь требуемой компоновки. Благодаря вложенности слоев, дочерние слои Layer2 и Layer3 будут перемещаться вместе с родительскими, что упрощает процесс компоновки страницы.
- Результат проделанных действий изображен на рис. 7.29.

Рис. 7.29. Результат использования слоев для компоновки страницы

Глава 8

Создание динамических страниц

Динамические страницы изменяют свой внешний вид в зависимости от определенных обстоятельств (времени суток, разрешения экрана монитора) или действий пользователя (например манипуляций мышью). Для изменения внешнего вида страниц используются *программы-сценарии*. В этой главе мы познакомимся с наиболее популярным языком написания программ-сценариев — языком программирования JavaScript.

8.1. Структура и размещение программы

Текст программы размещается в HTML-документе с помощью *метки-контейнера* `<SCRIPT>…</SCRIPT>`. *Операторы* языка JavaScript могут размещаться внутри метки-контейнера как на разных строчках, так и на одной строке. При расположении операторов на одной строке, они разделяются точкой с запятой. Например:

```
<SCRIPT> оператор 1; оператор 2; оператор 3; … </SCRIPT>
```

или

```
<SCRIPT>
оператор 1
оператор 2
оператор 3
```

```
...
</SCRIPT>
```

Рекомендуется располагать операторы на разных строчках, так как в этом случае программа легче читается, что способствует более быстрому поиску и устранению возможных ошибок.

Метка <SCRIPT> может располагаться в любом месте документа, причем можно использовать сразу несколько меток с текстами разных программ. Однако при выборе места расположения текста программы следует руководствоваться следующими соображениями. Программа будет выполняться по мере загрузки ее в браузер, и все данные, необходимые для ее работы, к этому моменту должны быть уже загружены. Исключения составляют *подпрограммы-функции*, которые могут входить в состав программы. Подпрограммы-функции выполняются только после вызова их на выполнение из HTML-документа или другой программы, поэтому рекомендуется располагать их в разделе <HEAD> с таким расчетом, чтобы к моменту возможного вызова они уже были загружены. *Подробно подпрограммы-функции будут рассмотрены в разделе 8.5.*

8.2. Типы данных

Любая программа предназначена для обработки данных. Программы на JavaScript позволяют обрабатывать данные следующих простых типов:

- *Строки символов.* Могут содержать любые буквы, цифры и другие символы, которые заключаются в кавычки (") или апострофы ('). Следует помнить, что внутри строки, заключенной в апострофы, можно использовать только кавычки, а строки, заключенной в кавычки, только апострофы.

  Например:

  ```
  "программирование на JavaScript";

  'газета "Московские новости"';

  "автомобиль 'Жигули'".
  ```

- *Целые числа.* Допускается использование не только десятичных чисел, но также восьмеричных и шестнадцатеричных.

Восьмеричные числа записываются с префиксом 0, а шестнадцатеричные — с префиксом 0x или 0X.

Например:

`512` — десятичное число;

`03047` — восьмеричное число (для записи используются только цифры 07);

`0x63BF` — шестнадцатеричное число (для записи используются цифры 09 и латинские буквы от A до F).

- *Вещественные числа.* Существуют две формы записи вещественных чисел: с точкой, отделяющей целую часть от дробной, и экспоненциальная, в которой латинская буква "e" отделяет мантиссу от порядка числа. Чтобы определить значение числа, записанного в экспоненциальной форме, нужно мантиссу умножить на 10 в степени, равной порядку числа.

  Например, число 4,95 в экспоненциальной форме может быть записано следующими способами: `495e-2`, `0.495E01`, `0.00495e+3`.

- *Логический тип.* Данные этого типа могут принимать только два значения `true` и `false`, т. е. "истина" и "ложь".

8.3. Литералы и переменные. Оператор присваивания

Конкретные строки символов, числа и логические значения называются *литералами* или *константами.* Примеры литералов: `367`, `5.39E-5`, `0631`, `0X3DF` — числовые литералы, `"Русский музей"` — строковый литерал; `true`, `false` — логические литералы.

Кроме литералов в программах используются *переменные.* Переменные предназначены для хранения исходных, промежуточных или результирующих данных, необходимых для работы программы. Переменной называется область памяти, которая имеет имя. В именах переменных следует использовать только латинские буквы, цифры и символ подчеркивания (_). Начинаться имя должно с буквы или символа подчеркивания. Следует помнить, что язык JavaScript чувствителен к регистру букв в именах переменных. Поэтому имена переменных `Name` и `name` будут раз-

ными именами. В качестве имен переменных нельзя использовать так называемые служебные слова, т. е. слова, имеющие в языке JavaScript определенное предназначение (например слова true и false).

Присвоить переменной конкретное значение можно с помощью *оператора присваивания.*

```
имя_переменной = значение
```

Например:

```
X = 8.31e+5
firstName = "Максим"
Firstname = "Надежда"
```

После выполнения оператора присваивания в памяти выделяется место, за которым закрепляется имя переменной, а присвоенное значение помещается в эту память. В программе одной и той же переменной можно неоднократно присваивать новые значения, причем язык JavaScript позволяет одной переменной присваивать значения разных типов. Повторное присваивание переменной нового значения приведет к потере значения, присвоенного ранее. Рассмотрим действие оператора присваивания на конкретном примере программы на JavaScript, вставленной в HTML-документ (листинг 8.1).

Листинг 8.1. HTML-документ с программой, включающей несколько операторов присваивания

```
<html>
<head>
<title>оператор присваивания</title>
</head>
<body>
  <form name=f1>
    <input type="text" name="T1" size="10"><p>
    <input type="text" name="T2" size="10"><p>
    <input type="text" name="T3" size="10"><p>
    <input type="text" name="T4" size="10"><p>
    <input type="text" name="T5" size="10">
  </form>
```

```
<script>
X = 8.31e+5
firstName = "Максим"
Firstname = "Надежда"
f1.T1.value=firstName; f1.T2.value=Firstname
f1.T3.value=X
X=firstName
f1.T4.value=X
X=true
f1.T5.value=X
</script>
</body>
</html>
```

Результат работы программы приведен на рис. 8.1.

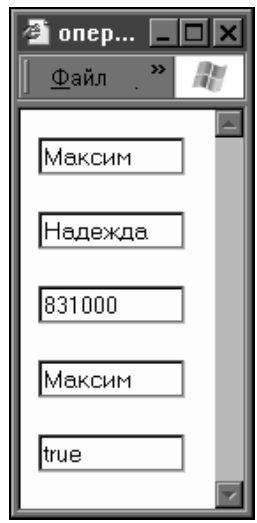


Рис. 8.1. Результат работы программы, приведенной в листинге 8.1

Документ содержит пять элементов формы, а именно текстовых строк, в которые последовательно выводятся значения переменных `firstName`, `Firstname` и `X`. Причем переменная `X` дважды

меняет не только свое значение, но и тип данных. Запись `f1.T1.value = firstName` означает, что свойству `value` элемента `T1` формы `F1` присваивается значение переменной `firstName`, а свойство `value` и определяет значение текстовой строки.

Попробуйте изменить место расположения программы, переместив ее выше открывающей метки `<FORM>`. Нажав кнопку обновить в окне браузера, убедитесь в возникновении ошибки. Как уже отмечалось выше, такое расположение программы в документе не допустимо, так как элементы формы, с которыми работает программа, будут загружаться после программы.

8.4. Выражения

Выражение — совокупность переменных и литералов, соединенных знаками операций, в результате выполнения которых получается единственное результирующее значение. В программах на JavaScript можно использовать арифметические выражения, логические выражения, выражения сравнения и строковые выражения.

- *Арифметические выражения.*

  Операции, используемые в арифметических выражениях, приведены в табл. 8.1 в соответствии с приоритетами их выполнения

Таблица 8.1. *Арифметические операции*

Операции	Описание операций
++, -- , -	Инкремент, декремент, унарный минус
*, / , %	Умножение, деление, деление по модулю
+, -	Сложение, вычитание

Операции сложения, вычитания, умножения и деления являются обычными бинарными операциями, т. е. в операции участвуют два операнда и определяется один результат. Эти операции не требуют каких-либо разъяснений.

Операция деления по модулю, также относится к бинарным операциям. Результатом операции является остаток от деле-

ния первого операнда на второй. Например, в результате выполнения операции `ostatok=23%5` переменной `ostatok` будет присвоено значение 3.

Операции инкремент, декремент и унарный минус относятся к числу унарных операций, т. е. в операции участвует только один операнд. Операция инкремент увеличивает значение операнда на единицу, а декремент — уменьшает на единицу. В качестве операндов операций инкремент и декремент можно использовать только переменные.

Например:

```
X=2
++X
```

В результате переменная `X` будет иметь значение 3. Допускается запись знаков операции и после операнда `X++`, что в данном случае не приведет к изменению результата.

В том случае, когда операция используется в операторе присваивания, положение знака операции влияет на ее результат следующим образом. Если знак операции предшествует операнду, то сначала выполнится операция, а потом переменной слева от знака "=" будет присвоено полученное значение, в противном случае сначала будет выполнено присваивание, а потом выполнится операция.

Например:

```
X=2
Y=X++
```

результат: `Y=2, X=3`;

```
X=2
Y=++X
```

результат: `Y=3, X=3`.

Все приведенные рассуждения справедливы и для операции декремент, за исключением того, что значения переменных будут не увеличиваться, а уменьшаться на единицу.

Операция унарный минус изменяет знак операнда на противоположный.

Приоритетность выполнения операций убывает от верхней строки к нижней строке. Операции, находящиеся в одной

строке, имеют равный приоритет и выполняются последовательно слева направо. Приоритет выполнения операций может быть изменен с помощью круглых скобок.

- *Выражения сравнения.*

  В выражениях сравнения используются операции:

  - `>` — больше;
  - `<` — меньше;
  - `==` — равно;
  - `>=` — больше или равно;
  - `<=` — меньше или равно;
  - `!=` — не равно.

  В качестве операндов операции сравнения можно использовать данные любых типов. Однако для практических целей, как правило, имеет смысл сравнивать числовые данные с числовыми данными, а строковые — со строковыми данными. При сравнении строк сравниваются коды первых символов, а в случае их равенства — коды вторых символов и так далее до первого несовпадения, причем соотношение длин строк значения не имеет. При полном совпадении всех символов равными считаются только строки одинаковой длины, в противном случае меньшей будет более короткая строка. Результат операции сравнения — логическое значение `true`, если утверждение справедливо, и `false` — если нет.

  Например:

  ```
  5>7
  ```

  результат: `false`;

  ```
  Z=13
  W=15
  Z<=W
  ```

  результат: `true`.

  Очевидно, что сравнение друг с другом литералов не имеет практического значения. Сравнение друг с другом переменных или переменных с константами играет в программировании важную роль, так как результат сравнения, от которого может зависеть дальнейший ход программы, в свою очередь,

зависит от значений переменных, которые они приобрели к моменту сравнения.

Кроме перечисленных операций сравнения, существуют еще две операции, требующие дополнительных разъяснений:

- `===` — строго равно;
- `!==` — строго не равно.

При выполнении обычных операций равно (`==`) и не равно (`!=`) в случае сравнивания разнотипных данных, они будут преобразовываться к одному типу.

Например:

```
12=="12"
```

результат: `true`.

При выполнении операций строго равно (`===`) и строго не равно (`!==`) преобразование типов не производится. В случае сравнения разнотипных данных результатом всегда будет `false`.

- *Логические выражения.*

В логических выражениях используются операции:

- `&&` — логическая операция И;
- `||` — логическая операция ИЛИ;
- `!` — логическая операция НЕ.

Операнды всех логических операций должны быть логического типа, результат операции также логический, т. е. `true` или `false`. Операции И и ИЛИ относятся к числу бинарных и результаты этих операций в зависимости от значений операндов приведены в табл. 8.2.

Таблица 8.2. *Результаты выполнения логических операций И и ИЛИ*

X	Y	X && Y	X || Y
true	true	true	true
true	false	false	true
false	true	false	true
false	false	false	false

Примеры записи логических операций И, ИЛИ:

```
true && false
```

результат: `false`;

```
true || false
```

результат: `true`;

```
X>5 && X<10
```

результат: `true` при `X` больше 5, но меньше 10.

Использование в качестве операндов логических литералов не имеет практического значения, и первые два примера приведены лишь для демонстрации записи операций.

Операция НЕ относится к числу унарных операций, она изменяет значение операнда на противоположное.

Примеры логической операции НЕ:

```
!true
```

результат: `false`;

```
!false
```

результат: `true`;

```
X=true
!X
```

результат: `false`.

- *Строковые выражения.*

  В строковых выражениях операндами являются строки символов. Основная операция над строками — операция соединения строк (конкатенации). Соединение строк обозначается знаком плюс (+). Например:

  ```
  STR="Java"+"Script"
  ```

  В результате операции переменной `STR` будет присвоено значение "JavaScript".

- *Смешанные выражения.*

  В смешанных выражениях допускается использование операций разного типа, из числа приведенных выше. При записи таких выражений следует учитывать приоритетность выполнения операций и в случае необходимости (для повышения приоритета) использовать круглые скобки.

Все рассмотренные операции приведены в табл. 8.3 в порядке убывания приоритетов. Операции, расположенные в одной строке, имеют равный приоритет и выполняются последовательно слева направо.

Таблица 8.3. *Приоритетность выполнения операций*

Операции	Описание операций
+ +, --, -, !	Инкремент, декремент, унарный минус, логическая операция НЕ
*, /, %	Умножение, деление, деление по модулю
+, -	Сложение, вычитание
<, >, <=, >=	Меньше, больше, меньше или равно, больше или равно
==, !=, ===, !==	Равно, не равно, строго равно, строго не равно
&&	Логическая операция И
||	Логическая операция ИЛИ

В качестве примера рассмотрим программу (листинг 8.2), включающую два смешанных выражения. Результаты выражений будем выводить в элементы формы — текстовые строки.

Листинг 8.2. HTML-документ, включающий программу, содержащую смешанные выражения

```
<html>
<head>
<title>выражения</title>
</head>
<body>
  <form name=f1>
    <input type="text" name="T1" size="10"><p>
    <input type="text" name="T2" size="10">
  </form>
```

```
<script>
X = 4
Y = 3
f1.T1.value=2*X+1>3*Y&&Y+X<8
f1.T2.value=2*(X+1)>3*Y&&Y+X<8
</script>
</body>
</html>
```

В первом выражении операции будут выполняться в следующем порядке:

- Две операции умножения `2*X=8, 3*Y=9`;
- Две операции сложения `8+1=9, Y+X=7`;
- Две операции сравнения `9>9=false, 7<8=true`;
- Логическая операция И `false&&true=false`.

Во втором выражении изменен порядок выполнения операций за счет применения скобок. Операция сложения в скобках выполнится в первую очередь, поэтому результатом первого умножения станет 10. Результат первого сравнения изменится на `true`. Следовательно, результатом всего выражения также будет являться `true`.

8.5. Подпрограммы-функции

Подпрограммой называется часть программы, оформленная специальным образом и выполняемая только после вызова ее из любого места программы (или из другой подпрограммы), а также при выполнении каких-либо действий с объектами браузера или HTML-документа. Подпрограмма, написанная один раз, может вызываться на выполнение многократно. Кроме того, облегчается отладка программы, состоящей из отдельных подпрограмм, каждая из которых может решать самостоятельную задачу. В принципе программа может состоять из одних подпрограмм. В программах на языке JavaScript используются *подпрограммы-функции*, которые в дальнейшем будем называть просто *функциями*. Функция размещается в теле программы, т. е. внутри метки-контейнера `<SCRIPT>...</SCRIPT>` и имеет следующую структуру:

```
function имя_функции (список параметров)
{
тело функции
}
```

При отсутствии списка параметров круглые скобки обязательны. Слово `function` следует писать строчными буквами.

В отличие от обычной программы, операторы которой выполняются друг за другом по мере загрузки документа в браузер, функция будет реализовываться только после вызова ее пользователем. Вполне возможен вариант, что в течение всего сеанса работы пользователя с документом, функция не будет вызвана ни разу. Вызов функции на выполнение осуществляется путем указания имени функции в тексте программы либо путем присваивания имени какому-либо *событию*. Событие задается в виде параметра в метке, по отношению к которой совершается событие, а значением этого параметра должно быть имя вызываемой функции. Пример использования функции в HTML-документе приведен в листинге 8.3.

Листинг 8.3. Использование функции в HTML-документе

```
<html>
<head>
<title>функции</title>
<script>
function PR1()
{
F1.T1.value="Здравствуйте, " + F1.T1.value + "!"
}
</script>
</head>
<body>
<form name="F1">
<h4 align="center">Введите свое имя и нажмите кнопку</h4>
    <p align=center>
```

```
<input type="text" name="T1" size="30">
    <p align=center>
<input type="button" value="Принять" onMouseDown=PR1()>
</form>
</body>
</html>
```

В приведенном примере действием, вызывающим функцию на выполнение, является нажатие кнопки. В метке `<INPUT TYPE="BUTTON">` используется событие onMouseDown (нажатие кнопки мыши), которому присваивается имя функции `PR1()`.

Функция содержит только одну строку, в которой изменяется содержимое (параметр `VALUE`) текстового поля `T1`, находящегося внутри формы `F1`. Параметру `VALUE` присваивается сумма трех слагаемых: строки символов `"Здравствуйте, "`, содержимого текстового поля `T1` до нажатия кнопки (текст, введенный пользователем) и строки `"!"`. Предположим, пользователь ввел в строку имя Николай. После нажатия кнопки в окне браузера можно будет увидеть результат, приведенный на рис. 8.2.

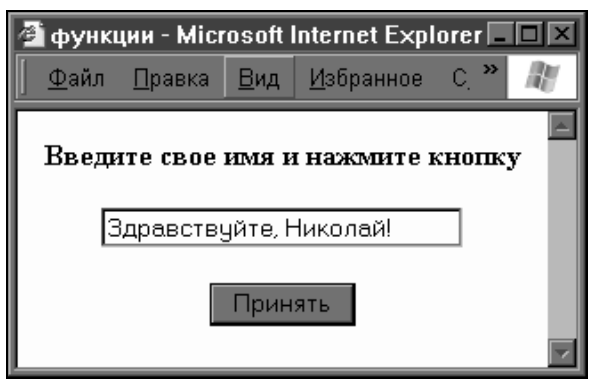


Рис. 8.2. Результат вызова функции в окне браузера после нажатия кнопки **Принять**

Некоторые события, которые можно использовать для вызова функций на выполнение, приведены в табл. 8.4.

Таблица 8.4. *Список событий, используемых для вызова функций*

Название событие	Описание события
onMouseDown	Нажатие кнопки мыши над объектом
onMouseUp	Отпускание кнопки мыши над объектом
onMouseMove	Перемещение указателя мыши над объектом
onMouseOut	Перемещение указателя мыши из объекта
onMouseOver	Нахождение указателя мыши над объектом
onClick	Щелчок по объекту
onLoad	Загрузка документа в браузер
onUnLoad	Закрытие документа

В названиях событий допускается использование как заглавных, так и строчных букв, но не допускается использование пробелов.

Что же касается текстов программ на JavaScript, то используемые в них имена объектов и имена функций чувствительны к регистру букв. Например, если при вызове функции `PR1()` в тексте документа написать onMouseDown=`Pr1()`, то функция `PR1()` выполняться не будет, а выполнится функция `Pr1()`, если, конечно, она существует. Аналогично, если в тексте документа имеются объекты с именами `T1` и `t1`, то программой они будут восприниматься как разные объекты.

В этом разделе мы познакомились только с функциями без параметров, т. е. с функциями, у которых как в заголовке функции, так и в строке вызова функции в круглых скобках после имени функции отсутствуют списки параметров. Механизм параметров играет исключительно важную роль, и к использованию функций с параметрами мы вернемся после знакомства с основными понятиями объектного программирования.

8.6. Локальные и глобальные переменные

Локальной называется переменная, которая доступна только в той функции, в которой она объявлена. Изменение значений локальной переменной не повлечет за собой изменение значений пере-

менных с такими же именами в других частях программы. Объявить локальную переменную можно, используя оператор `var`.

Например:

```
function F1()
{
var x1=0, y1=5, shag, Z
…
}
```

Внутри функции `F1()` объявлены четыре локальные переменные, причем двум из них одновременно присвоены начальные значения.

Глобальной называется переменная, значение которой доступно в любом месте программы, включая функции. Если внутри функции изменено значение глобальной переменной и после этого вызвана на выполнение другая функция, в которой используется та же переменная, то в ней она будет иметь измененное значение. Глобальная переменная будет объявлена после того, как оператором присваивания ей будет присвоено определенное значение, причем это может быть сделано в любом месте программы, включая функции. Кроме того, объявить глобальную переменную можно также, используя оператор `var`, но только вне функций. Например:

```
<SCRIPT>
var x1, y1=5
R=10
function F2()
{FL=0
…
}
…
</SCRIPT>
```

В примере объявлены четыре глобальные переменные: две — оператором `var`, а две другие (`R` и `FL`) — после присваивания им значений оператором присваивания, причем одна из них (`FL`) объявлена внутри функции.

Практическое использование локальных и глобальных переменных покажем на следующем примере (листинг 8.4).

Листинг 8.4. Использование локальных и глобальных переменных

```
<html>
<head>
<title>Локальные и глобальные переменные</title>
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
<script>
var X=0
function fct1()
{X++
f1.textfield.value=X
}
function fct2()
{var X=50
f1.textfield2.value=X
}
function fct3()
{X=X+Y
f1.textfield3.value=X
}
</script>
</head>
<body>
<form name="f1" method="post" action="">
  <table width="300" border="0">
    <tr>
      <td height="40" align="center">
        <input name="textfield" type="text" size="6">
      </td>
      <td align="center">
        <input name="textfield2" type="text" size="6">
      </td>
      <td align="center">
        <input name="textfield3" type="text" size="6">
      </td>
```

```
    </tr>
    <tr>
      <td height="50" align="center">
        <input type="button" name="Button1" value="Кнопка 1"
onClick=fct1()>
      </td>
      <td align="center">
        <input type="button" name="Button2" value="Кнопка 2"
onClick=fct2()>
      </td>
      <td align="center">
        <input type="button" name="Button3" value="Кнопка 3"
onClick=fct3() onMouseOver=fct4()>
      </td>
     </tr>
   </table>
<script>
function fct4()
{Y=300}
</script>
 </form>
</body>
</html>
```

В программе две глобальные переменные: `X` (объявлена оператором `var` вне функций) и `Y` (объявлена внутри функции `fct4()` оператором присваивания). Следует обратить внимание на то, что функция `fct4()` оторвана от остальной части программы. Внутри функции `fct2()` имеется локальная переменная `X` (объявлена оператором `var`), т. е. переменная с тем же именем, что и одна из глобальных переменных.

Для демонстрации результатов работы программы в документе созданы три текстовых поля и три кнопки, которые упорядочены с помощью таблицы.

Откроем документ в окне браузера. Щелкнем несколько раз по Кнопке 1. В текстовое поле над кнопкой будут выводиться числа 1, 2, 3, ... Это выполняется функцией `fct1()`, в которой глобальная переменная `X`, обнуленная в начале программы, увели-

чивается на 1 при каждом вызове функции. Щелкнем по Кнопке 2. В текстовом поле над кнопкой появится число 50, т. е. значение локальной переменной `X` функции `fct2()`.

Повторные щелчки не изменят результата. Вновь щелкнем несколько раз по Кнопке 1. В поле над кнопкой продолжится вывод следующих по порядку чисел. Как видим, изменение значения локальной переменной `X` никак не повлияло на значение глобальной переменной `X`.

Теперь подведем курсор к Кнопке 3. Мы пока не видим результата действия, но в тексте документа предусмотрен вызов функции `fct4()`, в которой глобальной переменной `Y` присваивается значение 300. Чтобы убедиться в этом, щелкнем по Кнопке 3, и в текстовом поле над кнопкой появится сумма значений глобальных переменных `X` и `Y`. Повторные щечки по Кнопке 1 будут приводить к изменению значений переменной X на 1, а по Кнопке 3 — на 300. Один из промежуточных результатов можно увидеть на рис. 8.3.

Рис. 8.3. Промежуточный результат изменения значений глобальной переменной X (поля над кнопками 1 и 3) и локальной переменной X (поле над кнопкой 2)

8.7. Основные понятия объектного программирования

Язык программирования JavaScript относится к языкам *объектного программирования*. Основными понятиями таких языков яв-

ляются: *объект*, *свойство* и *метод*. Программный объект должен иметь уникальное имя, он может обладать определенными свойствами, которые в программе могут изменяться, и над ним могут производиться определенные действия. Действия над объектами производятся с помощью методов — программ небольшого объема, написанных разработчиками языка.

В частности, объектами могут быть HTML-метки, а их свойствами — параметры этих меток. В листинге 8.3. приведен пример программного изменения содержимого текстового поля. В функции `PR1()` объектом является текстовое поле, уникальное имя которого T1 задано в параметре `NAME` метки `<INPUT>`. В качестве свойства метки используется параметр `VALUE`. Изменить свойство метки в программе можно следующим образом:

```
имя_метки.параметр="значение"
```

Следует заметить, что метка `<INPUT>` находится внутри формы `F1`, поэтому изменение свойств объектов формы является исключением из общего правила и требует указания также имени формы, что и сделано в функции `PR1()`. Кроме того, особенностью использования элементов формы является наличие параметра `NAME`. В других метках использование этого параметра может не предусматриваться, поэтому в этих случаях для задания уникального имени метки следует использовать параметр `ID`.

Рассмотрим следующий пример (листинг 8.5).

Листинг 8.5. Пример изменения свойства объекта, имя которого задано параметром ID

```
<HTML>
<HEAD>
<TITLE>
Программирование на JavaScript
</TITLE>
<SCRIPT>
function PR()
{
Pic1.src="2.jpg"
}
```

```
</SCRIPT>
</HEAD>
<BODY>
<IMG ID=Pic1 src=1.jpg onClick=PR()>
</BODY>
</HTML>
```

В этом документе единственным объектом является рисунок с уникальным именем `Pic1`. В функции изменяется свойство этого объекта `src`, значением которого является имя файла рисунка. Таким образом, при вызове функции `PR()` файл рисунка 1.jpg будет заменен файлом 2.jpg и можно будет наблюдать замену одного рисунка другим. Произойдет же это событие после щелчка мышью по рисунку, так как параметр `onClick` установлен в метке `<IMG>`. В метке `<IMG>` также используется параметр `ID`, значение которого определяет имя рисунка. Следует обратить внимание на то, что в программе значение свойства обязательно должно заключаться в кавычки.

Изменение свойств каскадной таблицы стилей имеет следующую особенность:

```
имя_объекта.style.свойство="значение"
```

Свойства каскадной таблицы стилей, содержащие дефис, записываются следующим образом. Дефис удаляется, а часть свойства после дефиса присоединяется к предшествующей части с большой буквы. В качестве примера рассмотрим следующий документ (листинг 8.6).

Листинг 8.6. Пример изменения свойств каскадной таблицы стилей

```
<HTML>
<HEAD>
<TITLE>Программирование на JavaScript</TITLE>
<SCRIPT>
function PR1()
{
P1.style.fontSize="24pt"
P1.style.color="blue"
```

```
P1.style.backgroundColor="magenta"
}
function PR2()
{
P1.style.backgroundColor="green"
P1.style.fontSize="20 pt"
}
</SCRIPT>
</HEAD>
<BODY>
<P id=P1 style="BACKGROUND-COLOR: yellow; font-size: 16pt"
onclick=PR1() onmouseout=PR2()>
Это абзац, щелчком мыши по которому изменяется размер шрифта
до 24 pt, цвет шрифта на синий, цвет фона на пурпурный, а
перемещением курсора за пределы абзаца изменяется цвет фона
на зеленый, а размер шрифта до 20 pt.
</P>
</BODY>
</HTML>
```

События, которые вызывают на выполнения функции `PR1()` и `PR2()`, и действия этих функций описаны в тексте документа. Остается лишь добавить, что здесь объектом является абзац с уникальным именем `P1`. В метке абзаца <P> с помощью свойств каскадной таблицы стилей задаются начальный размер текста и цвет фона. Начальный цвет текста по умолчанию будет черным. Возвращение к начальным характеристикам текста после вызова любой из функций в данном случае возможно только после повторной загрузки документа.

8.8. Подпрограммы-функции с параметрами

Эффективность применения *механизма параметров* рассмотрим на конкретном примере. В документе расположим таблицу, содержащую одну строку и три столбца. В каждую ячейку таблицы поместим определенный текст. При перемещении указателя мыши в ячейку таблицы должен изменяться цвет фона и текста

в ячейке, а при переходе в другую ячейку или за пределы таблицы цвет фона и текста должен восстанавливаться. Текст такого документа будет иметь следующий вид (листинг 8.7).

Листинг 8.7. Пример использования нескольких функций без параметров для изменения цвета фона и текста в ячейках таблицы

```
<html>
<head>
<title>функция без параметров</title>
<SCRIPT>
function PR1_Color ()
{
S1.style.color="red"
S1.style.backgroundColor="cyan"
}
function PR2_Color ()
{
S2.style.color="red"
S2.style.backgroundColor="cyan"
}
function PR3_Color ()
{
S3.style.color="red"
S3.style.backgroundColor="cyan"
}
function PR1_Black ()
{
S1.style.color="black"
S1.style.backgroundColor="white"
}
function PR2_Black ()
{
S2.style.color="black"
S2.style.backgroundColor="white"
}
```

```
function PR3_Black ()
{
S3.style.color="black"
S3.style.backgroundColor="white"
}
</SCRIPT>
</head>
<body>
<table border="1" width="400">
  <tr>
    <td valign="top" id="S1" onMouseMove=PR1_Color() onMouse-
Out=PR1_Black()>
    Завтрак
    <ul>
      <li>макароны с сыром</li>
      <li>какао</li>
      <li>пирожок с капустой</li>
    </ul>
    </td>
    <td width=33% valign="top" id="S2" onMouse-
Move=PR2_Color() onMouseOut=PR2_Black()>
    Обед
    <ul>
      <li>винегрет</li>
      <li>борщ</li>
      <li>бифштекс</li>
      <li>компот</li>
    </ul>
    </td>
    <td width=33% valign="top" id="S3" onMouse-
Move=PR3_Color() onMouseOut=PR3_Black()>
    Ужин
    <ul>
      <li>овощное рагу</li>
      <li>кефир</li>
    </ul>
```

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

Из текста документа видно, что для решения поставленной задачи нам пришлось использовать шесть функций. Функция `PR1_Color()` изменяет цвет фона и текста в ячейке с индивидуальным именем `S1`, а функция `PR1_Black()` восстанавливает белый цвет фона и черный цвет текста в этой же ячейке. Соответственно первая из них вызывается на выполнение в случае события onMouseMove по отношению ячейке `S1`, а вторая — события onMouseOut. Очевидно, что остальные четыре функции имеют то же предназначение, но по отношению к ячейкам `S2` и `S3`. Нетрудно также заметить, что содержимое всех шести функций в значительной степени совпадает, а отличие состоит либо в именах ячеек, либо в задаваемых цветах текста и фона. Используя механизм параметров можно все шесть функций заменить одной, а нужные имена ячеек и значения цветов передавать с помощью параметров. Вот как будет выглядеть текст такого документа (листинг 8.8).

Листинг 8.8. Пример использования одной функции с параметрами для изменения цвета фона и текста в ячейках таблицы

```
<html>
<head>
<title>функция с параметрами</title>
<SCRIPT>
function PR1(PN,col,bgcol)
{
PN.style.color = col
PN.style.backgroundColor = bgcol
}
</SCRIPT>
</head>
<body>
```

```
<table border="1" width="400">
  <tr>
    <td valign="top" id="S1" onMouseMove=PR1(S1,"red","cyan")
onMouseOut=PR1(S1,"black","white")>
    Завтрак
    <ul>
      <li>макароны с сыром</li>
      <li>какао</li>
      <li>пирожок с капустой</li>
    </ul>
    </td>
    <td width=33% valign="top" id="S2" onMouse-
Move=PR1(S2,"red","cyan") onMouseOut=PR1(S2,"black","white")>
    Обед
    <ul>
      <li>винегрет</li>
      <li>борщ</li>
      <li>бифштекс</li>
      <li>компот</li>
    </ul>
    </td>
    <td width=33% valign="top" id="S3" onMouse-
Move=PR1(S3,"red","cyan") onMouseOut=PR1(S3,"black","white")>
    Ужин
    <ul>
      <li>овощное рагу</li>
      <li>кефир</li>
    </ul>
    </td>
  </tr>
</table>
</body>
</html>
```

Единственная функция `PR1` имеет три параметра: `PN` — имя объекта, `col` — цвет шрифта, `bgcol` — цвет фона. Параметры в заголовке функции называются *параметрами-переменными*. Вызывая

функцию на выполнение, в круглых скобках после ее имени необходимо перечислить *параметры-значения*, которые мы хотим присвоить параметрам-переменным в заголовке функции. Например, при вызове функции `onMouseMove=PR1(S2,"red","cyan")`, имя объекта `PN` внутри функции будет присвоено значение `S2`, а переменным `col` и `bgcol` будут присвоены значения `"red"` и `"cyan"`. Так как `"red"` и `"cyan"` — строки символов, то они должны записываться в кавычках. Таким образом, использование механизма параметров позволило значительно сократить объем программы, сделать ее более наглядной и облегчить поиск и устранение возможных ошибок.

8.9. Оператор условного перехода

Оператор условного перехода является одним из важнейших операторов не только языка JavaScript, но и многих других языков программирования. Оператор позволяет на определенном этапе работы программы осуществить анализ возникшей к этому моменту ситуации и (в зависимости от результатов анализа) приступить к выполнению тех или иных операторов. Оператор записывается следующим образом:

```
if (условие) {группа операторов 1}; else { группа операторов 2}
```

В качестве условия можно использовать любое выражение, результат которого является логическим, т. е. true (истина) или false (ложь). Оператор работает следующим образом. Проверяется условие: если его результат true, то выполняется "группа операторов 1", а "группа операторов 2" не выполняется; если результат false, то выполняется "группа операторов 2", а "группа операторов 1" не выполняется. Дополнительно проиллюстрировать работу оператора можно с помощью блок-схемы, приведенной на рис. 8.4.

Конструкция `else` может использоваться не всегда. В упрощенной форме оператор условного перехода можно записать так:

```
if (условие) {группа операторов 1}
```

В этом случае, если результат условия — true, то выполняется "группа операторов 1", а если false, то "группа операторов 1" не выполняется. На рис. 8.5 представлена блок-схема, показывающая работу оператора в упрощенной форме.

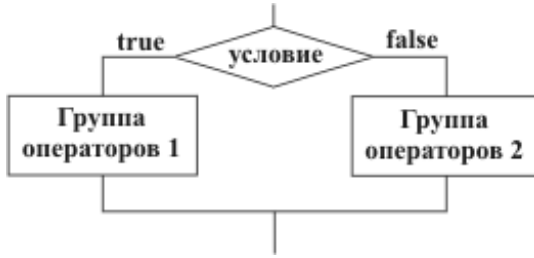


Рис. 8.4. Блок-схема работы оператора условного перехода (полная форма записи)

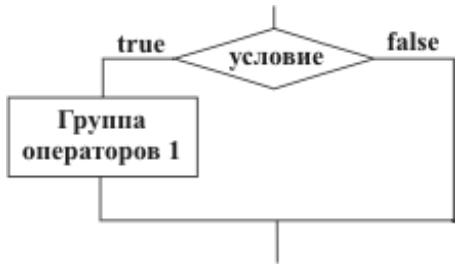


Рис. 8.5. Блок-схема работы оператора условного перехода (упрощенная форма записи)

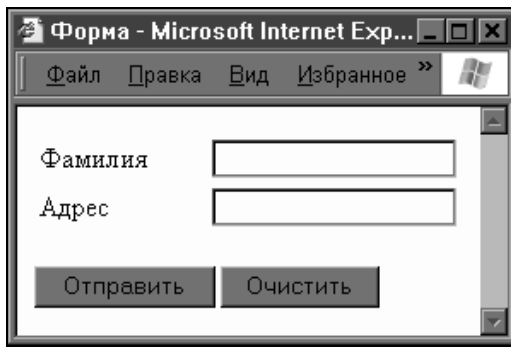


Рис. 8.6. Внешний вид формы с полями, обязательными для заполнения

Если в "группу операторов 1" или "группу операторов 2" входит только один оператор, то соответствующие фигурные скобки можно не ставить.

Приведем примеры программ, использующих оператор условного перехода.

Проверка заполнения формы перед отправкой.

Программа проверяет, заполнены ли обязательные поля формы перед ее отправкой. Внешний вид формы показан на рис. 8.6.

Текст документа, с включенной в него программой, приведен в листинге 8.9.

Листинг 8.9. HTML-документ с программой проверки заполнения полей формы

```
<html>
<head>
<title>Форма</title>
</head>
<script>
function PR()
{
if (F1.T1.value=="" || F1.T2.value=="") alert ("Заполните все
поля")
}
function PR1()
{
alert ("Готово к отправке")
}
</script>
<body>
<form method="POST" action="" name="F1">
  <table border="0" width="250" style="font-size: 12pt">
    <tr>
      <td width="150">Фамилия</td>
      <td width="100">
        <input type="text" name="T1" size="20">
      </td>
```

```
    </tr>
    <tr>
      <td width="150">Адрес</td>
      <td width="100">
         <input type="text" name="T2" size="20">
      </td>
    </tr>
  </table>
  <p><input type="submit" value="Отправить" name="B1" on-
MouseOver=PR() onclick=PR1()>
  <input type="reset" value="Очистить" name="B2">
</form>
</body>
</html>
```

Программа включает две функции. В функции `PR()` используется упрощенная форма оператора условного перехода. Условием является логическое выражение, включающее две операции сравнения, объединенные логической операцией ИЛИ. Если хотя бы одно из полей не заполнено, то результатом операции сравнения, следовательно, и всего выражения, будет являться true. В этом случае метод `alert()` создаст диалоговое окно с предупреждающим текстом. Вызов функции `PR()` осуществляется при наведении указателя мыши на кнопку "Отправить" (событие `onMouseOver`). Таким образом, пользователь не успеет щелкнуть по кнопке, как откроется диалоговое окно.

Функция `PR1()` играет вспомогательную, но очень важную роль. Дело в том, что после заполнения всех полей и щелчка по кнопке "Отправить" поля очищаются, а так как указатель в этот момент находится над кнопкой, то функция `PR()` будет вызвана вновь, что приведет к повторному появлению диалогового окна с предупреждающим текстом. Чтобы избавиться от этого, функция `PR1()` в момент щелчка по кнопке откроет другое диалоговое окно с подтверждением правильности действий пользователя, указатель будет отведен от кнопки "Отправить" для нажатия кнопки **OK** в диалоговом окне и проблема будет устранена.

Вычисления с использованием выбранных полей формы.

Программа осуществляет вычисления, используя поля формы, выбранные пользователем, и выводит результат также в поле формы. Пользователь, устанавливая соответствующие флажки, выбирает предметы, которые хотел бы заказать, вводит число предметов и нажимает кнопку "Сумма". Вычисляется только сумма предметов, отмеченных флажком. Результат работы программы показан на рис. 8.7.

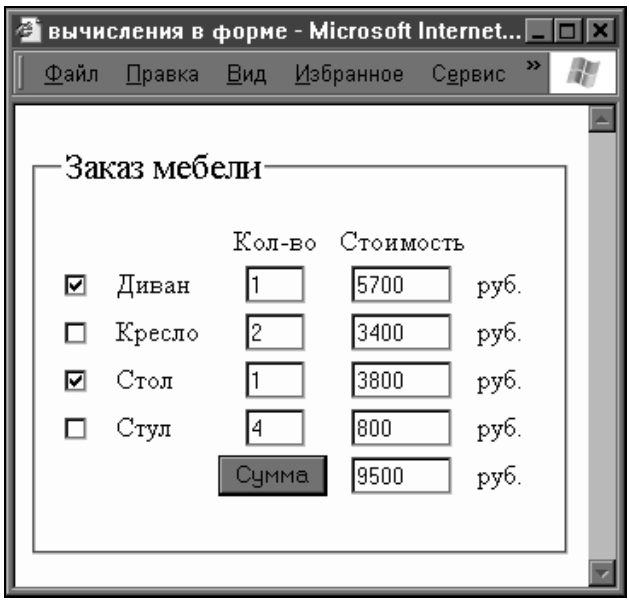


Рис. 8.7. Результат работы программы определения суммарной стоимости элементов, выбранных пользователем

Текст документа приведен в листинге 8.10.

Листинг 8.10. Пример программы, вычисляющей сумму содержимого полей формы, выбранных пользователем

```
<html>
<head>
<title>вычисления в форме</title>
```

```
<script>
divan=5700
kreslo=3400
stol=3800
stul=800
function nznach()
{F1.ST1.value=divan
 F1.ST2.value=kreslo
 F1.ST3.value=stol
 F1.ST4.value=stul
}
function sum()
{if (F1.FL1.checked) k1=F1.KOL1.value; else k1=0
 if (F1.FL2.checked) k2=F1.KOL2.value; else k2=0
 if (F1.FL3.checked) k3=F1.KOL3.value; else k3=0
 if (F1.FL4.checked) k4=F1.KOL4.value; else k4=0
 s1=k1*divan
 s2=k2*kreslo
 s3=k3*stol
 s4=k4*stul
 F1.SUM.value=s1+s2+s3+s4
}
</script>
</head>
<body onLoad=nznach()>
<fieldset style="padding: 10; width: 310">
<legend style="font-size: 16pt">Заказ мебели</legend>
<form method="POST" action="" name="F1">
<table width="290" border="0" style="font-size: 12pt">
  <tr>
    <td width="30"> </td>
    <td width="60"> </td>
    <td width="60" align="center">Кол-во</td>
    <td width="80"align="center">Стоимость</td>
    <td> </td>
```

```
  </tr>
  <tr >
    <td> <input type="checkbox" name="FL1" value="checkbox">
</td>
    <td>Диван</td>
    <td align="center">
        <input name="KOL1" type="text" size="2">
      </td>
    <td align="center">
        <input name="ST1" type="text" size="6">
     </td>
      <td> руб.</td>
  </tr>
  <tr>
    <td> <input type="checkbox" name="FL2" value="checkbox">
</td>
    <td>Кресло</td>
    <td align="center">
        <input name="KOL2" type="text" size="2">
    </td>
    <td align="center">
        <input name="ST2" type="text" size="6">
     </td>
<td> руб.</td>
  </tr>
  <tr>
    <td> <input type="checkbox" name="FL3" value="checkbox">
</td>
    <td>Стол</td>
    <td align="center">
        <input name="KOL3" type="text" size="2">
    </td>
    <td align="center">
        <input name="ST3" type="text" size="6">
     </td>
     <td> руб.</td>
```

```
  </tr>
  <tr>
    <td> <input type="checkbox" name="FL4" value="checkbox">
</td>
    <td>Стул</td>
    <td align="center">
        <input name="KOL4" type="text" size="2">
     </td>
     <td align="center">
        <input name="ST4" type="text" size="6">
     </td>
     <td> руб.</td>
  </tr>
    <tr>
    <td colspan="3" align="right">
       <input name="KN1" type="button" value="Сумма" on-
Click=sum()>
    </td>
    <td align="center">
        <input name="SUM" type="text" size="6">
     </td>
     <td> руб.</td>
  </tr>
</table>
</form>
</fieldset>
</body>
</html>
```

Программа начинается несколькими операторами присваивания. Переменным `divan`, `kreslo`, `stol` и `stul` присваиваются значения стоимостей соответствующих предметов. Переменные являются глобальными и могут использоваться в любой функции. При необходимости изменения стоимости предмета, достаточно внести изменения только в этом месте.

Для расчетов стоимости заказа используются значения переменных, а не содержимое полей "Стоимость", которые могут

быть изменены пользователем. Функция `nznach()` предназначена для заполнения полей "Стоимость" сразу после загрузки документа, поэтому вызывается на выполнение событием `onLoad`. Функция `sum()` вызывается после нажатия кнопки "Сумма" (событие `onClick`). Операторами условного перехода осуществляется проверка установки каждого из четырех флажков выбора предмета. Свойство `checked` объекта `checkbox` ("флажок") принимает значение true, если флажок установлен, и false — в противном случае. Переменным `k1…k4`, относящимся к предметам, отмеченным флажками, будут присвоены значения соответствующих полей группы "Кол-во", в противном случае им будет присвоено нулевое значение. Далее вычисляется стоимость покупки каждого предмета с учетом количества, и полученные значения присваиваются переменным `s1…s4`. В последней строке в поле с именем `SUM` выводится сумма заказа.

Движение изображения по экрану.

Изображение движется внутри условного прямоугольника. Достигнув края прямоугольника, оно "отражается" от него и продолжает движение до достижения другого края и т. д. Например, логотип фирмы можно заставить пробежаться по странице и остановить в нужном месте или можно изобразить на странице падающие снежинки, движущиеся автомобили и т. д. Текст документа следующий (листинг 8.11).

Листинг 8.11. Пример программы перемещающей изображение внутри условного прямоугольника

```
<html>
<head>
<title>Динамические картинки</title>
<script>
z=0
x=50
y=50
sx=3
sy=2
```

```
function nach()
{
i1.style.left=x
i1.style.top=y
i1.style.display="block"
si=setInterval("dv()",100)
}
function dv()
{
z++
x+=sx
y+=sy
if (x<50 || x>350) sx =-sx
if (y<50 || y>200) sy=-sy
i1.style.left=x
i1.style.top=y
if (z>200) clearInterval(si)
}
</script>
</head>
<body>
<p>
<img id=i1 border="0" style="position: absolute; display:
none" src="1.jpg">
<p><font size="5" color="#0000FF" onclick=nach()>Начать
движение</font></p>
</body>
</html>
```

Программа начинается с присваивания начальных значений переменным: `z` — счетчик числа шагов, `x` — положение изображения относительно левого края страницы, `y` — положение изображения относительно верхнего края страницы, `sx` — шаг перемещения изображения по горизонтали, `sy` — шаг перемещения изображения по вертикали. Перемещаемое изображение, имеющее индивидуальное имя `i1`, в момент загрузки документа скрыто. Чтобы сделать его видимым и заставить

перемещаться, щелчком по текстовой строке "Начать движение" вызывается функция `nach()` (событие `onclick`). Очевидно, что можно предусмотреть и другие способы начала движения, например можно вызвать функцию после загрузки документа (событие `onLoad`) и т. д. В функции `nach()` задается: начальное положение изображения по горизонтали (значение свойства `i1.style.left`) и по вертикали (значение свойства `i1.style.top`), видимость изображения и вызов метода `setInterval()` на выполнение. Метод имеет два параметра, которые записываются в круглых скобках через запятую. Первый параметр ("dv()") — имя вызываемой функции, второй (100) — временной интервал (в мс). Таким образом, метод `setInterval()` постоянно вызывает заданную функцию на выполнение через указанный временной интервал. Чтобы остановить этот процесс нужно использовать идентификатор и метод `clearInterval()` (идентификатор). В нашем примере идентификатором является переменная `si`, которой присваивается значение самим же методом `setInterval()`. Метод `clearInterval()` используется в функции `dv()`. Функция `dv()` вызывается методом `setInterval()` через каждые 100 мс. При каждом вызове счетчик числа шагов z увеличивается на 1, координата изображения x увеличивается на шаг перемещения по горизонтали `sx`, а `y` — на шаг перемещения по вертикали `sy`. Далее с помощью операторов условного перехода осуществляются проверки нахождения координат изображения в заданных интервалах. Например, если значение координаты x больше 350, то шаг перемещения по горизонтали станет отрицательным и при следующем вызове функции `dv()` координата x начнет уменьшаться, а изображение перемещаться по горизонтали влево. Изображение изменит свое положение после присваивания свойствам `i1.style.left` и `i1.style.top` новых значений. Оператор условного перехода в конце функции сравнивает значение счетчика числа шагов с максимально возможным значением и в случае превышения его вызывает метод `clearInterval()`, вследствие чего работа метода `setInterval()` будет прервана. Скорость движения изображения регулируется изменением временного интервала в методе `setInterval()` и изменением шагов `sx` и `sy`.

8.10. Окно браузера как объект программирования

До сих пор в качестве объектов в наших программах фигурировали метки HTML. Однако возможности языка JavaScript гораздо шире. В частности, объектом программирования может являться окно браузера (объект `Window`). Объект имеет множество свойств и методов. К методам объекта `Window` относятся уже рассмотренные нами методы: `alert()`, `setInterval()`, `clearInterval()`, а также еще один — метод `open()`, с которым познакомимся в данном разделе.

Метод `open()` предназначен для открытия нового окна. Его структура может выглядеть следующим образом:

```
open ("имя файла", "имя окна", "параметры окна")
```

- имя файла — имя файла документа, который должен загрузиться в раскрываемом окне или его адрес, если документ находится на другом сервере;
- имя окна — имя открываемого окна. Используя это имя после открытия окна, можно загружать в него другие документы;
- параметры окна — группа параметров, перечисляемых через запятую, определяющих внешний вид окна.

Часть параметров позволяет задать размеры окна и его положение на экране. К ним относятся: `width` — ширина окна, `height` — высота окна, `top` — смещение по вертикали относительно верхней границы окна монитора, `left` — смещение по горизонтали относительно левой границы окна монитора. Все четыре параметра задаются в пикселях.

Другая часть параметров позволяет установить, будет ли отображаться тот или иной элемент браузера. Каждый из них может принимать одно из двух значений: 1 — отобразить элемент, 0 — скрыть элемент (допускается также использование значений yes/no). По умолчанию действует значение 0 (элемент скрыт). Эта часть параметров включает: `toolbar` — отображение панели инструментов (кнопки), `directories` — отображение панели инструментов (ссылки), `location` — отображение адресной

строки, `menubar` — отображение меню, `status` — отображение строки состояния, `resizable` — разрешение изменения размеров окна пользователем.

Приведем пример использования программы для открытия документа в новом окне. Текст документа показан в листинге 8.12.

Листинг 8.12. Пример использования программы для открытия документа в новом окне и управления параметрами окна

```
<HTML>
<HEAD>
<TITLE>новое окно</TITLE>
<SCRIPT>
function NW()
{
open("P1.jpg", "w1", "top=150, left=300, width=155,
height=230")
}
</SCRIPT>
</HEAD>
<BODY>
<H1 onclick=NW()>Открытие рисунка в новом  окне браузера</H1>
</BODY>
</HTML>
```

Для открытия рисунка P1.jpg в новом окне вызывается функция `NW()`, в которой и располагается метод `open()`. Для открываемого окна заданы только его размеры и положение в окне монитора. Параметры, определяющие внешний вид окна, не заданы и будут определяться по умолчанию. Внешний вид документа вместе с новым окном, открытым после щелчка по тексту, показан на рис. 8.8.

Следует обратить внимание, что по умолчанию на открываемой в браузере странице существуют поля, которые создают белую рамку вокруг рисунка. Задавая размер открываемого окна, приходится это учитывать и устанавливать размеры окна несколько больше, чем размер рисунка. Чтобы избавиться от полей, следует открывать в новом окне не файл рисунка, а HTML-документ

без полей, содержащий только этот рисунок. Пример текста такого документа приведен в листинге 8.13.

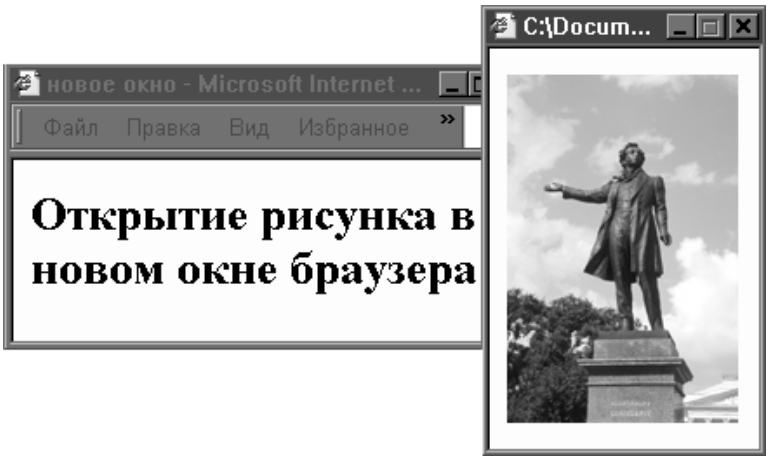


Рис 8.8. Результат открытия изображения в новом окне

Листинг 8.13. Текст документа, создающего страницу без полей

```
<html>
<head>
<title>Открытие окна</title>
</head>
<body topmargin="0" leftmargin="0">
<img src="P1.jpg">
</body>
</html>
```

Параметрами `topmargin` и `leftmargin` устанавливаются нулевые поля страницы. Изменим значения параметров метода `open()` предыдущей программы, изменив имя открываемого файла и размеры окна:

```
open("P1nb.htm", "w1", "top=150, left=300, width=133,
height=200")
```

Результат показан на рис. 8.9.

Рис. 8.9. Результат открытия в новом окне документа, не содержащего полей

Программа может быть использована и для открытия окон с разными документами, причем они могут быть и разного размера. Для этого достаточно ввести механизм параметров. Рассмотрим следующий пример (листинг 8.14).

Листинг 8.14. Пример открытия нового окна с разными документами и разного размера

```
<HTML>
<HEAD>
<TITLE>новое окно</TITLE>
<SCRIPT>
function NW(FN,W,H)
{
open(FN, "w1", "top=150, left=300, width="+W+", height="+H)
}
</SCRIPT>
</HEAD>
<BODY>
<H1 onclick=NW("P1.htm","150","250")>Документ 1</H1>
```

```
<H1 onclick=NW("P2.htm","200","300")>Документ 2</H1>
</BODY>
</HTML>
```

Параметры метода `open()` записываются как строки символов, т. е. в кавычках. Поэтому значения параметров, передаваемых в функцию `NW()` при ее вызове, должны относиться к типу данных "строки символов" и записываться в кавычках. В свою очередь переменные `FN`, `W` и `H` в заголовке функции `NW()` также принадлежат к строчному типу данных, так как через них осуществляется передача значений. Это объясняет, почему переменная `FN` в методе `open()` записывается без кавычек, а переменные `W` и `H` объединяются с другими частями в единую строку при помощи знака "+".

Проверив работу программы, убедимся, что прежде чем открыть новое окно документа, следует закрыть окно предыдущего. Объясняется это тем, что открываемое окно имеет конкретное имя w1 и открыть второе окно с таким же именем невозможно. Для многих решаемых задач такое положение дел можно считать вполне нормальным. Например, вы создаете картинную галерею и на странице размещаете уменьшенные репродукции картин. Щелчок по любой из них открывает увеличенное изображение картины в новом окне. Перед тем как перейти к демонстрации следующей картины, вполне естественно желание закрыть изображение предыдущей. И все-таки можно придумать ситуацию, когда необходимо открыть сразу несколько новых окон с разными документами. В этом случае в программе необходимо предусмотреть передачу еще одного параметра с именем окна (листинг 8.15).

Листинг 8.15. Пример открытия нескольких окон документов разного размера

```
<HTML>
<HEAD>
<TITLE>новое окно</TITLE>
<SCRIPT>
function NW(FN, W, H,WN)
```

```
{
open(FN, WN, "top=150, left=300, width="+W+", height="+H)
}
</SCRIPT>
</HEAD>
<BODY>
<H1 onclick=NW("P1.htm","150","250","w1")>Документ 1</H1>
<H1 onclick=NW("P2.htm","200","300","w2")>Документ 2</H1>
</BODY>
</HTML>
```

8.11. Перетаскивание объектов

В этом разделе будет решаться следующая задача. На странице находится несколько элементов, позиционированных абсолютно. Это, в частности, могут быть рисунки или текстовые блоки. Требуется написать программу, которая позволит после нажатия левой кнопки мыши над элементом перемещать его по странице. После отпускания кнопки объект останется на новом месте. Текст программы приведен в листинге 8.16.

Листинг 8.16. Пример программы, позволяющей перетаскивать объекты в окне браузера

```
<html>
<head>
<title>Перетаскивание</title>
<script>
obj=""
function cursor_down()
{obj=event.srcElement
if (obj != "")
{x0=obj.style.pixelLeft
y0=obj.style.pixelTop
cur_x=event.clientX
```

```
cur_y=event.clientY
del_x=cur_x-x0
del_y=cur_y-y0}
}
function cursor_up()
{obj=""}
function cursor_move()
{if (obj!="")
{obj.style.left=event.clientX-del_x
obj.style.top=event.clientY-del_y}
event.returnValue=false
}
</script>
</head>
<body onmousedown=cursor_down() onmousemove=cursor_move() on-
mouseup=cursor_up()>
<img border="0" style="position: absolute; left: 303; top:
189;cursor=move" src="1.jpg">
<img border="0" style="position: absolute; left: 174; top:
55;cursor=move" src="2.jpg">
<img border="0" style="position: absolute; left: 406; top:
17;cursor=move;" src="3.jpg">
<img border="0" style="position: absolute; left: 104; top:
190;cursor=move" src="4.jpg">
 <b>
    <font size="6">
       <span style="position: absolute; left: 105; top:
190;cursor=move">Здесь находится текст, который можно пере-
местить так же, как и любой другой элемент с абсолютным пози-
ционированием.
       </span>
    </font>
 </b>
</body>
</html>
```

В программе используются три функции: `cursor_down()`, `cursor_up()` и `cursor_move()`. Первая из них вызывается на выполнение после нажатия кнопки мыши в любом месте документа (событие `onMouseDown` в метке `<BODY>`), вторая — после отпускания кнопки (событие `onMouseUp`), третья — при перемещении указателя (событие `onMouseMove`). В программе используется одна глобальная переменная `obj` с начальным значением "пустая строка" (`""`). В первой строке функции `cursor_down()` переменной `obj` присваивается значение ссылки на элемент, над которым была нажата кнопка. Для этого используется объект `event`. До сих пор объекты, с которыми мы имели дело, были вполне конкретными. Это HTML-метки, рисунки, блоки, окна. Объект `event` носит абстрактный характер и существует для того, чтобы получить дополнительные сведения о каком-либо событии. Каждое событие порождает объект `event`, а его свойства несут в себе определенную информацию. К свойствам объекта `event` относятся: `srcElement` — ссылка на элемент, относительно которого произошло событие; `clientX` — горизонтальная координата указателя мыши в момент события; `clientY` — вертикальная координата указателя в момент события.

Всегда ли переменной `obj` будет присвоено значение ссылки на элемент? Очевидно нет, ведь нажать кнопку мыши можно в таком месте страницы, где нет никаких элементов. Тогда значением переменной `obj` останется "пустая строка". Поэтому далее в функции `cursor_down()` проверяется значение переменной `obj` и если оно не является пустой строкой, то выполняется группа операторов в фигурных скобках.

Первые два оператора в фигурных скобках предназначены для определения координат выбранного элемента. Для этого используются два свойства каскадной таблица стилей pixelLeft и pixelTop, определяющих горизонтальную и вертикальную координаты левого верхнего угла элемента, которые присваиваются переменным `x0` и `y0`.

Следующие два оператора предназначены для определения координат указателя мыши в момент выбора элемента, для чего используются свойства объекта `event` — `clientX` и `clientY`. В результате переменным `cur_x` и `cur_y` будут присвоены координаты указателя мыши в момент нажатия клавиши.

Далее определяются разницы между соответствующими координатами левого верхнего угла выбранного объекта и указателя мыши `del_x` и `del_y`, использование которых в дальнейшем позволит нам перемещать элемент именно за ту точку, в которой произошло нажатие клавиши.

В функции `cursor_move()` осуществляется уже известная проверка того, что выбран конкретный элемент и только в этом случае выполняются два оператора в фигурных скобках. Их назначение — присвоить свойствам `left` и `top`, перемещаемого объекта, значения координат указателя мыши с учетом поправок `del_x` и `del_y`. Если не вводить поправки, то в момент нажатия кнопки мыши координаты левого верхнего угла перемещаемого элемента станут равными координатам указателя, элемент перескочит на новое место, и дальнейшее его перемещение будет осуществляться за левый верхний угол.

В последней строке функции `cursor_move()` свойству `returnValue` объекта `event` будет присвоено значение `false`. В этом случае отменяется действие обработчика события, предусмотренное по умолчанию, с тем, чтобы последующее движение указателя мыши вызвало его вновь. Этим будет обеспечен многократный вызов функции `cursor_move()` в процессе движения указателя.

Функция `cursor_up()` вызывается на выполнение после отпускания кнопки мыши и содержит единственный оператор, который присваивает переменной `obj` значение пустой строки. После чего никакие действия, предусмотренные в функции `cursor_move()`, не будут выполняться, пока не будет выбран новый объект. Указатель изменяет свою форму над элементами, которые можно перемещать, для чего используется свойство cursor каскадной таблицы стилей.

Перемещение элемента может сопровождаться выводом координат перемещаемого элемента. В простейшем случае это могут быть координаты левого верхнего угла элемента. Для этого необходимо поместить в документ две текстовые строки. Например, это можно сделать следующим образом:

```
<body onmousedown=cursor_down() onmousemove=cursor_move() on-
mouseup=cursor_up()>
```

```
<p align="right">
<b> X </b>
<input type="text" name="T1" size="6">
<b> Y </b>
<input type="text" name="T2" size="6"></p>
```

Функцию `cursor_move()` также необходимо дополнить двумя строками:

```
function cursor_move()
{if (obj!="")
{obj.style.left=event.clientX-del_x
 obj.style.top=event.clientY-del_y
T1.value=obj.style.pixelLeft
T2.value=obj.style.pixelTop}
event.returnValue=false}
```

8.12. Создание раскрывающегося графического меню

В качестве разделов и пунктов меню будем использовать графические изображения, которые должны быть заранее подготовлены в графическом редакторе (*см. раздел 3.4.14*). Следует отметить, что вместо изображений можно использовать и обычный текст, и хотя это приведет к изменению приводимых ниже документов, эти изменения не будут носить принципиальный характер.

Для всех рассматриваемых в данном разделе примеров в качестве разделов меню будут использованы 3 рисунка (r1.jpg, r2.jpg, r3.jpg) и 4 комплекта по 2 рисунка для четырех пунктов меню. В каждом комплекте один рисунок предназначен для создания пассивного пункта меню (pp1.jpg, , pp4.jpg), второй — активного пункта (pp1d.jpg, , pp4d.jpg).

8.12.1. Горизонтальное меню

Внешний вид горизонтального меню, которое нам предстоит создать, показан на рис. 8.10.

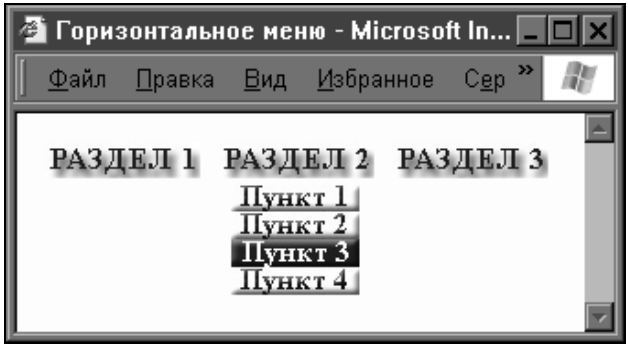


Рис 8.10. Внешний вид в окне браузера горизонтального раскрывающегося меню

Список пунктов раскрывается после щелчка по рисунку с названием раздела. Перемещение указателя мыши по пунктам приводит к выделению соответствующего пункта, что достигается сменой файла рисунка. Щелчок по названию пункта либо выход за пределы списка приведут к его закрытию. Список будет также закрыт при переходе в другой раздел или за пределы меню. Пространство под названиями разделов, в котором раскрываются списки пунктов, до их раскрытия остается пустым и не может быть занято объектами, расположенными ниже меню. Текст документа приведен в листинге 8.17.

Листинг 8.17. Пример создания горизонтального раскрывающегося меню

```
<HTML>
<HEAD><TITLE>Горизонтальное меню</TITLE>
<SCRIPT>
function prR()
{event.cancelBubble="true"
}
function prH(stolbx)
{stolbx.style.visibility="hidden"
}
function prV(stolbx)
```

```
{stolbx.style.visibility="visible"
}
function prS(sx,fn)
{sx.src=fn
}
</SCRIPT>
<BODY onmousemove=prH(stolb1),prH(stolb2),prH(stolb3)>
<SPAN style="WIDTH: 95px; TEXT-ALIGN: center"  onmouse-
move=prH(stolb2),prR()>
<IMG  id=p1 src="r1.jpg" onclick=prV(stolb1)>
<SPAN id=stolb1 style="VISIBILITY: hidden; WIDTH: 70px" on-
click=prH(stolb1),prR()>
<IMG id=p11 src="pp1.jpg" onmousemove=prS(p11,"pp1d.jpg") on-
mouseout=prS(p11,"pp1.jpg")><br>
<IMG id=p12 src="pp2.jpg" onmousemove=prS(p12,"pp2d.jpg") on-
mouseout=prS(p12,"pp2.jpg")><br>
<IMG id=p13 src="pp3.jpg" onmousemove=prS(p13,"pp3d.jpg") on-
mouseout=prS(p13,"pp3.jpg")><br>
<IMG id=p14 src="pp4.jpg" onmousemove=prS(p14,"pp4d.jpg") on-
mouseout=prS(p14,"pp4.jpg")>
</SPAN>
</SPAN>
<SPAN style="WIDTH: 95px; TEXT-ALIGN: center" onmouse-
move=prH(stolb1),prH(stolb3),prR()>
<IMG  src="r2.jpg" onclick=prV(stolb2)>
<SPAN id=stolb2 style="VISIBILITY: hidden; WIDTH: 70px" on-
click=prH(stolb2),prR()>
<IMG id=p21 src="pp1.jpg" onmousemove=prS(p21,"pp1d.jpg") on-
mouseout=prS(p21,"pp1.jpg")><br>
<IMG id=p22 src="pp2.jpg" onmousemove=prS(p22,"pp2d.jpg") on-
mouseout=prS(p22,"pp2.jpg")><br>
<IMG id=p23 src="pp3.jpg" onmousemove=prS(p23,"pp3d.jpg") on-
mouseout=prS(p23,"pp3.jpg")><br>
<IMG id=p24 src="pp4.jpg" onmousemove=prS(p24,"pp4d.jpg") on-
mouseout=prS(p24,"pp4.jpg")>
</SPAN></SPAN>
<SPAN style="WIDTH: 95px; TEXT-ALIGN: center" onmouse-
move=prH(stolb2),prR()>
<IMG  src="r3.jpg" onclick=prV(stolb3)>
```

```
<SPAN id=stolb3 style="VISIBILITY: hidden; WIDTH: 70px" on-
click=prH(stolb3),prR()>
<IMG id=p31 src="pp1.jpg" onmousemove=prS(p31,"pp1d.jpg") on-
mouseout=prS(p31,"pp1.jpg")><br>
<IMG id=p32 src="pp2.jpg" onmousemove=prS(p32,"pp2d.jpg") on-
mouseout=prS(p32,"pp2.jpg")><br>
<IMG id=p33 src="pp3.jpg" onmousemove=prS(p33,"pp3d.jpg") on-
mouseout=prS(p33,"pp3.jpg")><br>
<IMG id=p34 src="pp4.jpg" onmousemove=prS(p34,"pp4d.jpg") on-
mouseout=prS(p34,"pp4.jpg")>
</SPAN></SPAN>
</BODY>
</HTML>
```

Меню создано с помощью блоков, хотя можно было бы использовать и таблицу. Все разделы меню имеют одинаковую структуру и отличаются только индивидуальными именами, поэтому ограничимся рассмотрением первого раздела.

Раздел создается с помощью метки-контейнера `<SPAN>`. Задается ширина блока, которая выбирается несколько большей, чем ширина рисунка с изображением названия раздела, что влияет на расстояние между разделами. Содержимым контейнера, кроме рисунка, является другой контейнер, также созданный меткой `<SPAN>` с индивидуальным именем `stolb1`. Контейнер включает 4 рисунка с изображениями названий пунктов меню. Каждый рисунок имеет индивидуальное имя, которое используется в качестве параметра при вызове функции смены изображений `prS()`. Вторым параметром является имя файла рисунка.

Функция `prS()` вызывается на выполнение двумя событиями. Событие `onMouseMove` (перемещение курсора) вызывает функцию, передавая имя файла рисунка активного пункта, а событие `onMouseOut` (выход за пределы рисунка) — пассивного пункта.

Контейнер с названиями пунктов скрыт (`VISIBILITY: hidden`) и становится видимым только после щелчка по рисунку с названием раздела, в результате чего вызывается функция `prV()`. В качестве параметра функции передается имя контейнера. Действие функции `prV()` очевидно, она содержит одну строку, в которой свойству `VISIBILITY` присваивается значение `visible`. Так же проста и функция `prH()`, которая делает заданный объ-

ект скрытым. Она вызывается на выполнение после щелчка по любому пункту меню контейнера `stolb1`.

Чтобы стало понятно назначение функции `prR()`, следует сказать несколько слов о том, как распространяется событие в иерархии объектов. В нашем документе можно проследить следующую иерархию. На верхнем уровне располагается сама страница (содержимое метки `<BODY>`). На следующем уровне располагается контейнер раздела, в котором располагаются два объекта более низкого уровня — рисунок с названием раздела и контейнер с рисунками пунктов. Любое действие в отношении объекта нижнего уровня создаст соответствующее событие не только в отношении этого объекта, но и всех объектов, расположенных выше по иерархии. Например, если щелкнуть мышью по элементу контейнера `stolb1`, то наступит событие `onclick` и произойдет вызов функции `prH()`. Это же событие наступит и в отношении вышестоящего контейнера, в котором для этого случая предусмотрен вызов функции `prV`(), совершающей противоположное действие. Запретить прохождения события к вышестоящим объектам можно с помощью свойства `cancelBubble` объекта `event`, которому нужно присвоить значение `true`, что и делается в единственной строке функции `prR()`. Вторично функция `prR()` вызывается событием `onMouseMove` контейнера раздела. В случае перемещения курсора, над каким-либо разделом, предусматривается скрытие смежных с ним контейнеров с названиями пунктов. В первом разделе это осуществляется вызовом функции `pH(stolb2)`, во втором разделе она вызывается дважды с разными значениями параметра `pH(stolb1)`, pH(stolb3) и в третьем разделе — `pH(stolb2)`. Аналогичные действия предусмотрены и в метке `<BODY>`, что обеспечивает скрытие списка пунктов после перемещения курсора за пределы меню. Если не запретить прохождение события `onMouseMove` от контейнера раздела ко всей станице, то любое перемещение указателя мыши внутри меню будет сопровождаться скрытием списка пунктов. Вот почему вместе с вызовом функции `prH()` вызывается и функция `prR()`.

8.12.2. Вертикальное меню

Внешний вид вертикального меню показан на рис. 8.11.

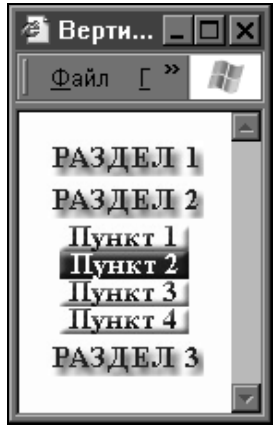


Рис 8.11. Внешний вид вертикального раскрывающегося меню в окне браузера

Кроме вертикального расположения разделов, особенность меню заключается в том, что раскрытие пунктов сопровождается смещением разделов, располагающихся ниже. Несмотря на многие сходства документа с предыдущим, его текст целесообразно привести полностью (листинг 8.18).

Листинг 8.18. Пример создания вертикального раскрывающегося меню

```
<HTML>
<HEAD><TITLE>Вертикальное меню</TITLE>
<SCRIPT>
function prR()
{event.cancelBubble="true"
}
function prH(stolbx)
{stolbx.style.display="none"
}
function prV(stolbx)
{stolbx.style.display="block"
}
```

```
function prS(sx,fn)
{sx.src=fn
}
</SCRIPT>
<BODY onmousemove=prH(stolb1),prH(stolb2),prH(stolb3)>
<DIV style="WIDTH: 95px; TEXT-ALIGN: center" onmouse-
move=prH(stolb2),prR()>
<IMG  id=p1 src="r1.jpg" onclick=prV(stolb1)>
<SPAN id=stolb1 style="DISPLAY:none; WIDTH: 70px" on-
click=prH(stolb1),prR()>
<IMG id=p11 src="pp1.jpg" onmousemove=prS(p11,"pp1d.jpg") on-
mouseout=prS(p11,"pp1.jpg")><br>
<IMG id=p12 src="pp2.jpg" onmousemove=prS(p12,"pp2d.jpg") on-
mouseout=prS(p12,"pp2.jpg")><br>
<IMG id=p13 src="pp3.jpg" onmousemove=prS(p13,"pp3d.jpg") on-
mouseout=prS(p13,"pp3.jpg")><br>
<IMG id=p14 src="pp4.jpg" onmousemove=prS(p14,"pp4d.jpg") on-
mouseout=prS(p14,"pp4.jpg")>
</SPAN>
</DIV>
<DIV style="WIDTH: 95px; TEXT-ALIGN: center" onmouse-
move=prH(stolb1),prH(stolb3),prR()>
<IMG  src="r2.jpg" onclick=prV(stolb2)>
<SPAN id=stolb2 style="DISPLAY:none; WIDTH: 70px" on-
click=prH(stolb2),prR()>
<IMG id=p21 src="pp1.jpg" onmousemove=prS(p21,"pp1d.jpg") on-
mouseout=prS(p21,"pp1.jpg")><br>
<IMG id=p22 src="pp2.jpg" onmousemove=prS(p22,"pp2d.jpg") on-
mouseout=prS(p22,"pp2.jpg")><br>
<IMG id=p23 src="pp3.jpg" onmousemove=prS(p23,"pp3d.jpg") on-
mouseout=prS(p23,"pp3.jpg")><br>
<IMG id=p24 src="pp4.jpg" onmousemove=prS(p24,"pp4d.jpg") on-
mouseout=prS(p24,"pp4.jpg")>
</SPAN>
</DIV>
<DIV style="WIDTH: 95px; TEXT-ALIGN: center" onmouse-
move=prH(stolb2),prR()>
<IMG  src="r3.jpg" onclick=prV(stolb3)>
<SPAN id=stolb3 style="DISPLAY:none; WIDTH: 70px" on-
```

```
click=prH(stolb3),prR()>
<IMG id=p31 src="pp1.jpg" onmousemove=prS(p31,"pp1d.jpg") on-
mouseout=prS(p31,"pp1.jpg")><br>
<IMG id=p32 src="pp2.jpg" onmousemove=prS(p32,"pp2d.jpg") on-
mouseout=prS(p32,"pp2.jpg")><br>
<IMG id=p33 src="pp3.jpg" onmousemove=prS(p33,"pp3d.jpg") on-
mouseout=prS(p33,"pp3.jpg")><br>
<IMG id=p34 src="pp4.jpg" onmousemove=prS(p34,"pp4d.jpg") on-
mouseout=prS(p34,"pp4.jpg")>
</SPAN>
</DIV>
</BODY>
</HTML>
```

Каждый раздел создается меткой-контейнером `<DIV>`, а не `<SPAN>`, как в предыдущем примере. Это позволяет размещать каждый раздел с новой строки, располагая их вертикально. Кроме того, в функциях `prH()` и `prV()` вместо свойства `VISIBILITY` используется свойство `DISPLAY`, что и приводит при раскрытии списка к изменению положения разделов, расположенных ниже. В остальном можно руководствоваться пояснениями к предыдущему документу.

8.12.3. Компактное меню

Особенностью этого меню является то, что для его размещения на странице требуется меньше места, чем в предыдущих случаях. Это объясняется тем, что контейнер со списком пунктов позиционируется абсолютно и накладывается на список разделов (рис. 8.12).

Такое меню может быть создано как на базе горизонтального, так и вертикального меню. Для перехода к компактному меню достаточно контейнеры с названиями пунктов позиционировать абсолютно:

```
<SPAN id=stolb1 style="DISPLAY: none; WIDTH: 70px; POSITION:
absolute; LEFT:35; TOP:32" onclick = prH(stolb1),prR()>
```

Для остальных контейнеров нужно будет только изменить координаты позиционирования.

Рис. 8.12. Внешний вид компактного раскрывающегося меню в окне браузера

8.13. Поиск информации на странице

Если страница имеет достаточно большие размеры, то в ней можно предусмотреть поиск группы слов, отдельного слова и даже части слова. Для этого на странице нужно расположить элементы формы — тестовую строку и кнопку. Программа-сценарий, которая будет выполняться после нажатия кнопки и осуществлять поиск информации, введенной в текстовую строку, создается на базе объекта `TextRange` и его методов. Текст документа приведен в листинге 8.19.

Листинг 8.19. Пример программы поиска информации на странице

```
<html>
<head>
<title>поиск</title>
<script>
str_t=""
function poisk()
{str=T1.value
if (str_t!=str)
{txt=document.body.createTextRange()
```

```
str_t=str}
rezalt=txt.findText(str)
if (rezalt)
{txt.select()
txt.scrollIntoView()
txt.collapse(false)}
else
{alert("Поиск завершен")
str_t=""}
}
</script>
</head>
<body>
  <p><input type="text" name="T1" size="20">
  <p><input type="button" value="Найти" name="B3" on-
click=poisk()>
Текст, в котором ведется поиск.
</body>
</html>
```

Переменной `str_t` уже на этапе загрузки документа присваивается значение "пустая строка". Функция `poisk()` вызывается на выполнение после нажатия кнопки "Найти". Содержимое текстового поля `T1` присваивается переменной `str`. В операторе условного перехода сравниваются переменные `str_t` и `str`. Неравенство этих переменных означает, что поиск только начинается, и это требует выполнения операторов в фигурных скобках. В первом из них используется метод `createTextRange()` объекта `TextRange`, который создает экземпляр объекта `TextRange` из того объекта, к которому применяется метод. В нашем случае метод применяется к объекту `body` (телу документа), который является составной частью объекта `document` (всего документа). В результате переменной `txt` будет присвоено все текстовое содержимое метки `<BODY>`. Далее переменной `str_t` будет присвоено значение переменной `str`, что при повторном нажатии кнопки "Найти" для поиска того же текста, исключит выполнение этих двух операторов. Непосредственный поиск осуществляется методом `findText()` объекта `TextRange` в содержимом тек-

стовой переменной `txt`, причем, если поиск будет успешным, переменной `rezalt` будет присвоено значение `true`, в противном случае — `false`. Назначение следующего оператора условного перехода — выполнить группу из трех операторов только в случае успешного поиска. Во всех трех операторах используются методы объекта `TextRange`. В первом из них используется метод `select()`, который выделяет найденный фрагмент. Во втором — метод `scrollIntoView()`, который вызывает прокрутку документа в окне браузера таким образом, чтобы найденный фрагмент появился в окне. В третьем операторе используется метод `collapse()` с параметром `false`, который исключает из текстовой строки `txt` весь просмотренный текст, включая найденный фрагмент. С этого момента строка `txt` содержит только непросмотренную часть текста. Если после этого функция `poisk()` будет вызвана повторно для поиска того же фрагмента, то поиск продолжится только в непросмотренной части текста. В случае если фрагмент не был найден, будут выполняться операторы после `else`, т. е. появится диалоговое окно с текстом "Поиск завершен", и переменной `str_t` вновь будет присвоено значение "пустая строка". В случае повторного поиска он начнется с начала текста.

8.14. Дата и время в HTML-документах

Для работы с датой и временем предназначен объект `Date`. Объект `Date` относится к числу *встроенных объектов*. Такие объекты имеют две особенности. Первая заключается в том, что они существуют независимо от содержания конкретного документа. Вторая особенность — необходимость создания конкретного экземпляра объекта, к которому и применяются многочисленные методы объекта `Date`. Конкретный экземпляр объекта `Date` создается оператором `new` с помощью конструктора `Date()` следующим образом:

```
имя_экземпляра_объекта = new Date()
```

Например: `ND = new Date()`.

Познакомимся с некоторыми методами объекта `Date`:

- `getDate()` — возвращает текущий день месяца (131). Пример записи:

  ```
  ND = new Date()
  Dm = ND.getDate()
  ```

  Переменной `Dm` будет присвоено значение текущего дня месяца;
- `getDay()` — возвращает порядковый номер текущего дня недели (06), начиная с воскресенья;
- `getMonth()` — возвращает номер текущего месяца (011), начиная с января;
- `getFullYear()` — возвращает текущий год;
- `getHours` — возвращает текущий час суток (023);
- `getMinutes()` — возвращает текущую минуту (059);
- `getSeconds()` — возвращает текущую секунду (059);
- `toLocaleString()` — преобразует значение даты в строку символов в формате, соответствующем настройке операционной системы.

В дальнейшем при создании программ нам понадобится свойство `innerText`, которое применяется в основном для меток-контейнеров и позволяет заменить их текстовое содержимое, а при его отсутствии — добавить текст. Примеры использования свойства `innerText` и методов объекта `Date` приведены в листинге 8.20.

Листинг 8.20. Примеры использования свойства innerText и методов объекта Date

```
<head>
<title>объект Date</title>
<script>
dat=new Date()
function D()
{d1=dat.getDate()
str1.innerText=d1
```

```
d2=dat.getDay()
str2.innerText=d2
d3=dat.getFullYear()
str3.innerText=d3
d4=dat.getHours()
str4.innerText=d4
d5=dat.getMinutes()
str5.innerText=d5
d6=dat.getSeconds()
str6.innerText=d6
d7=dat.getMonth()
str7.innerText=d7
d8=dat.toLocaleString()
str8.innerText=d8
}
</script>
</head>
<body onLoad=D() style="font-size=12pt; font-weight=700">
<span id="str1"></span> - день месяца<br>
<span id="str2"></span> - день недели<br>
<span id="str3"></span> - полный год<br>
<span id="str4"></span> - час суток<br>
<span id="str5"></span> - минуты<br>
<span id="str6"></span> - секунды<br>
<span id="str7"></span> - месяц<br>
<span id="str8"></span> - полная дата и время<br>
</body>
</html>
```

В качестве конкретного экземпляра объекта `Date` создан объект `dat`, к которому применяются методы объекта `Date`. Свойство `innerText` поочередно применяется к контейнерам с индивидуальными именами `str1,…,str8`, что приводит к добавлению текста в пустые контейнеры.

Вид документа в окне браузера показан на рис. 8.13.

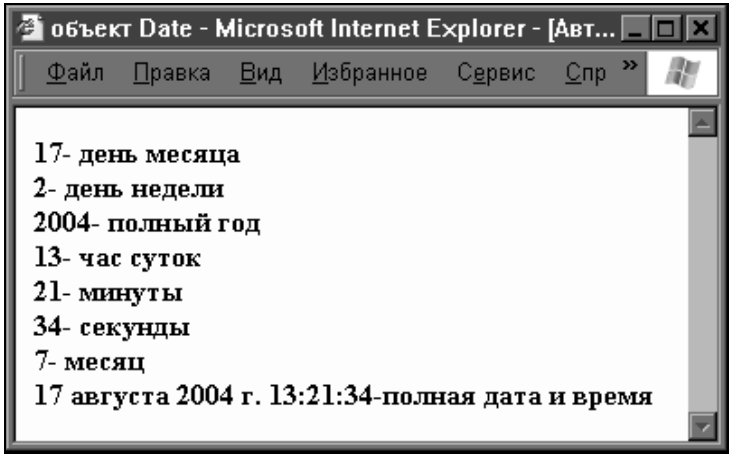


Рис. 8.13. Результат использования методов объекта Date

8.15. Понятие массива

Массив относится к числу классических понятий программирования. Массивом называется множество переменных, имеющих одинаковое имя, относящихся к одному типу данных и отличающихся индексом (номером), который записывается в квадратных скобках после имени. Конкретный экземпляр массива создается оператором `new` с помощью конструктора `Array()`:

```
имя_массива = new Array(список элементов)
```

Например: `v_sport = Array ("легкая атлетика", "хоккей", "теннис", "баскетбол")`. Создан массив `v_sport`, содержащий 4 элемента, каждый из которых относится к типу данных "строки символов".

Обращение к элементу массива осуществляется по имени с указанием в скобках номера элемента (нумерация ведется от 0).

`v_sport[1] = "футбол"` — значение второго элемента "хоккей" будет изменено на "футбол",
`sp=v_sport[2]` — переменной sp будет присвоено значение "теннис".
Чтобы добавить элемент в массив, достаточно осуществить обычное присваивание: `v_sport[5] = "волейбол"`.

После этого массив `v_sport` будет иметь 6 элементов, но элемент `v_sport[4]` не определен.

Существует несколько методов, применяемых к массивам. Для примера рассмотрим только два из них:

- `reverse()` — изменяет порядок следования элементов массива на противоположный. Например:

  ```
  m1 = new Array (0,1,2,3)
  m2 = m1.reverse()
  ```

  В результате применения метода элементы массива `m2` будут располагаться в порядке убывания значений, причем создания массива `m2` оператором `new` не требуется;

- `sort()` — сортирует элементы массива. Например:

  ```
  S1=new Array ("клен","вяз","дуб","осина")
  S2=S1.sort
  ```

  В массиве `S2` элементы будут располагаться в алфавитном порядке.

К массивам применимо свойство `length`, позволяющее определить число элементов в массиве. Например:

```
S1 = new Array ("клен","вяз","дуб","осина")
kol_el = S1.length
```

Переменной `kol_el` будет присвоено значение "4".

Рассмотрим несколько практических примеров, в которых будут использованы методы объекта `Date` и массивы.

8.15.1. Электронный календарь

Рассмотрим пример создания документа, представляющего собой "электронный календарь". Текст документа представлен в листинге 8.21.

Листинг 8.21. Пример создания "электронного календаря"

```
<html>
<head>
<title>Календарь</title>
```

```
<script>
dn_ned=new Array("Воскресенье", "Понедельник", "Вторник",
"Среда", "Четверг", "Пятница", "Суббота")
function vr()
{dat_vrem=new Date() //создается экземпляр объекта Date()
dn=dat_vrem.getDay() //метод getDay() возвращает порядковый
номер дня недели
str=dat_vrem.toLocaleString() //метод toLocaleString() преоб-
разует значение даты и времени в строку
vrdn.innerText=dn_ned[dn] // в метку <span> с именем vrdn
вставляется название дня недели
vrd.innerText=str
setTimeout("vr()",1000)
}
</script>
</head>
<body onLoad="vr()">
<h1 align="center">ЭЛЕКТРОННЫЙ КАЛЕНДАРЬ
<p><span id="vrdn"></span>
<p><span  id="vrd"></span>
</h1>
</body>
</html>
```

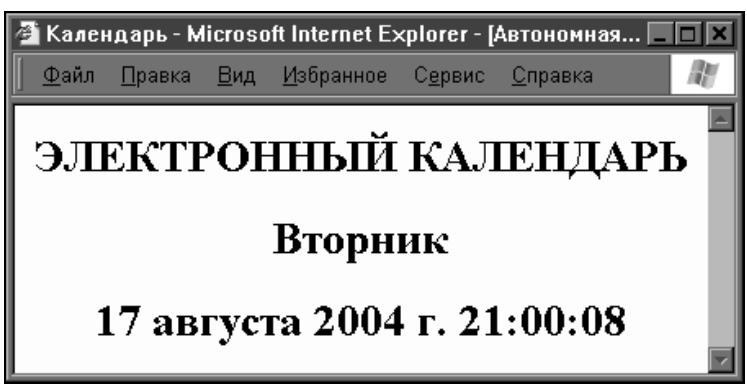


Рис. 8.14. "Электронный календарь" в окне браузера

В тексте программы имеются комментарии, поясняющие назначение отдельных операторов. Можно лишь добавить, что первоначально функция `vr()` вызывается после загрузки документа. В дальнейшем, используя метод `setTimeout()`, функция `vr()` вызывает саму себя через каждую секунду. По понятным причинам прекращение действия метода `setTimeout()` не предусматривается. Результат показан на рис. 8.14.

8.15.2. Электронные часы

По конечному результату программа подобна предыдущей, но ее текст содержит существенные отличия (листинг 8.22).

Листинг 8.22. Пример создания "электронных часов"

```
<html>
<head>
<title>Дата и время</title>
<script>
function vr()
{dat_vrem=new Date()
chas=dat_vrem.getHours()
minut=dat_vrem.getMinutes()
if (chas<10) nch="0"; else nch=""
if (minut<10) nmin="0"; else nmin=""
vrem.innerText=nch+chas+" "+":"+nmin+minut
tm1=setTimeout("vr()",1000)}
</script>
</head>
<body onLoad="vr()">
<h1 align="center">ЭЛЕКТРОННЫЕ ЧАСЫ
<p>
<span id="vrem"></span>
</h1>
</body>
</html>
```

Текущие час и минута определяются раздельно, объединяясь путем суммирования строк. Предусматривается добавление предшествующего нуля к значениям часов и минут, если они меньше 10. Несмотря на то, что сведения о секундах не выводятся, обновление результатов по-прежнему осуществляется через каждую секунду, чем обеспечивается высокая точность хода. Результат показан на рис. 8.15.

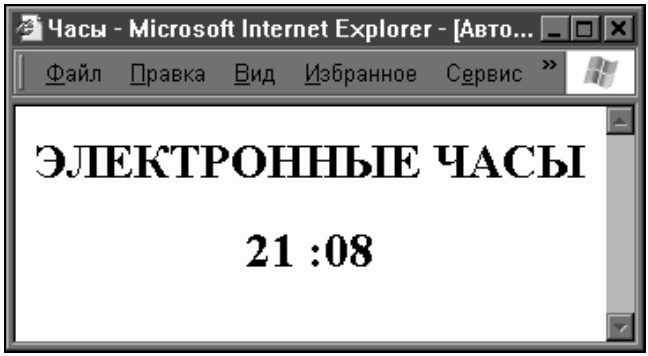


Рис. 8.15. "Электронные часы" в окне браузера

8.15.3. Обновление содержимого в зависимости от времени суток

Вывод текущего времени осуществляется, как и в предыдущей программе, но при этом дополнительно выводится текст, который меняется в зависимости от времени суток. Сутки разделены на четыре временных интервала, принадлежность к которым текущего времени проверяется четырьмя операторами условного перехода. Очевидно, что текущее время может оказаться только в одном из интервалов, после чего переменной `privet` будет присвоено соответствующее значение (листинг 8.23).

Листинг 8.23. Обновление содержимого контейнера в зависимости от времени суток

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

```
<title>Обновление содержимого</title>
<script>
function vr1()
{dat_vrem=new Date()
chas=dat_vrem.getHours()
if (chas>=0 && chas<6) privet="Доброй ночи!"
if (chas>=6 && chas<12) privet="С добрым утром!"
if (chas>=12 && chas<19) privet="Добрый день!"
if (chas>=19 && chas<=23) privet="Добрый вечер!"
minut=dat_vrem.getMinutes()
if (chas<10) nch="0"; else nch=""
if (minut<10) nmin="0"; else nmin=""
priv.innerText=privet+" На ваших часах "+ nch+chas+"
"+":"+nmin+minut
tm1=setTimeout("vr1()",1000)}
</script>
</head>
<body onLoad="vr1()">
<h1 align="center">
<span id="priv"></span>
</h1>
</body>
</html>
```

Результат приведен на рис. 8.16.

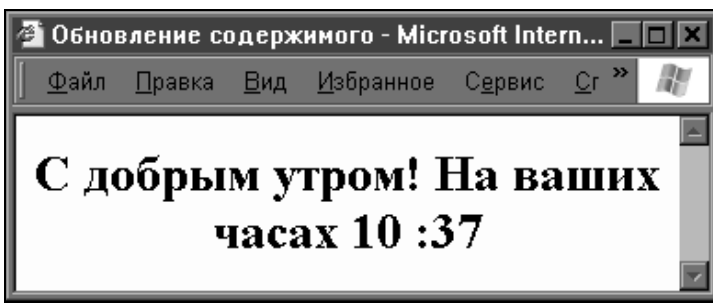


Рис. 8.16. Результат обновления содержимого контейнера в зависимости от времени суток

8.16. Создание всплывающей анимированной подсказки

Рассмотрим создание всплывающей анимированной подсказки на примере текста документа, представленного в листинге 8.24.

Листинг 8.24. Создание всплывающей анимированной подсказки

```
<html>
<head>
<title>Подсказка</title>
</head>
<script>
function str(txt)
{p1.innerHTML="<marquee>"+txt+"</marquee>"
p1.style.left=event.clientX
p1.style.top=event.clientY
p1.style.visibility="visible"}
function nstr()
{p1.style.visibility="hidden"}
</script>
<body>
<p><b onmouseover='str("Это подсказка")' onmouse-
out="nstr()"><font size=5 color="#000080">Анимированная
подсказка.</font></b></p>
<div id=p1 style="position: absolute;visibility: hidden;
width: 100;
background-color: #00FFFF; color: red; border: 1px black
solid"></div>
</body>
</html>
```

Для размещения подсказки создан невидимый пустой блок с именем `p1`, имеющий абсолютное позиционирование, ширину 100 px, фоновую заливку и черную непрерывную рамку толщиной 1 px. Функция `str()` вызывается при нахождении указателя мыши над элементом (в нашем случае это текст "Анимирован-

ная подсказка"). В качестве передаваемого параметра используется текстовая строка, содержимое которой и будет текстом подсказки. Введение параметра позволяет использовать одну и ту же функцию `str()` для создания множества подсказок с разным содержанием. Свойство `innerHTML` позволяет вставить в метку-контейнер `<DIV>` с именем `p1` метку бегущей строки `<MARQUEE>`, в которой находится нужный текст (параметр `txt`). Далее при помощи уже известных свойств `clientX` и `clientY` объекта `event` определяются координаты указателя в момент вызова функции `str()`, которые присваиваются соответствующим координатам контейнера `p1`. После этого контейнер делается видимым и работа функции `str()` завершается. Назначение функции `nstr()` — скрыть подсказку, после того как указатель выйдет за пределы элемента. Результат показан на рис. 8.17.

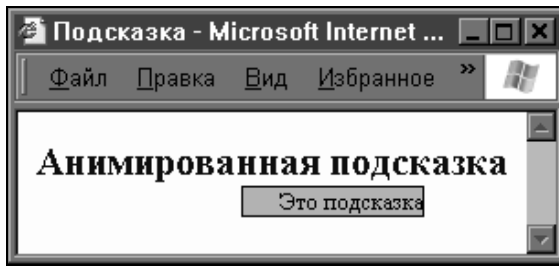


Рис. 8.17. Результат создания анимированной подсказки

8.17. Вывод текста в строку состояния браузера

Для вывода текста в строку состояния браузера используются свойства `status` или `defaultStatus` объекта `window`:

`window.status = "текст"` — содержимое строки "текст" выводится в строку состояния;

`window.defaultStatus = "текст"` — строка "текст" выводится в строку состояния по умолчанию.

Разница между этими свойствами станет более понятной после рассмотрения следующего примера (листинг 8.25).

Листинг 8.25. Пример вывода текста в строку состояния браузера

```
<html>
<head>
<title>Строка состояния</title>
<script>
window.defaultStatus='Вы открыли лучший в мире сайт!!!'
function strs()
{window.status='Ну, один из лучших!!!'}
</script>
</head>
<body>
<p align="center"><font size=5 onmouse-
over=strs()>Заголовок</font></p>
</body>
</html>
```

Открытие документа в окне браузера сопровождается появлением в строке состояния текста, заданного по умолчанию. При наведении указателя мыши на заголовок вызывается функция `strs()`, в которой свойству `status` присваивается определенное значение. Это значение будет находиться в строке состояния до перемещения указателя за пределы заголовка. После этого восстановится значение, предусмотренное по умолчанию.

8.18. Вывод в строку состояния браузера бегущего текста

Для создания эффекта движения строки применяется искусственный прием, смысл которого заключается в следующем. Текст выводится в строку состояния многократно и перед каждым выводом смещается на один символ влево. Для работы с текстовыми строками будут использоваться свойство `length` и метод `substr()` объекта `String`.

Текст документа приведен в листинге 8.26.

Листинг 8.26. Пример создания бегущей строки в строке состояния браузера

```
<html>
<head>
<title>Строка состояния</title>
<script>
str_p="                                                                    "
str="Добро пожаловать на наш сайт!!!"
st=str_p+str
L_str_p=str_p.length
pos=0
function bstr()
{pos++
window.status=st.substr(pos)
if (pos!=L_str_p)
{setTimeout("bstr()",100)}}
</script>
</head>
<body onLoad=bstr()>
<p align="center"><font size=5>Заголовок</font></p>
</body>
</html>
```

Переменной `str_p` присваивается значение символьной строки, содержащей 60 символов пробела. Визуально это можно определить только по положению закрывающих кавычек. Переменной `str` присваивается значение выводимого в строку состояния текста. Переменная `st` — результат суммирования строк `str_p` и `str`, именно она и будет выводиться в строку состояния. Отсюда понятно, что число символов пробела в строке `str_p` выбирается с таким расчетом, чтобы при первом выводе строка состояния казалась пустой.

Далее с помощью свойства `length` определяется длина строки `str_p` и присваивается переменной `L_str_p`. Следует обратить внимание на то, что ее длина в данном случае известна и, как было сказано выше, равна 60 символам. Однако если из каких-

то соображений ее длину нужно будет изменить, то это достаточно будет сделать только в одном месте, а значение переменной L_str_p всегда будет равно ее фактической длине.

Переменная pos играет роль счетчика числа вызовов функции bstr() и одновременно определяет номер позиции, с которой из строки st метод substr() вырезает подстроку, выводимую в строку состояния. Переменная pos с каждым шагом увеличивается на единицу, число пробелов в вырезаемой подстроке убывает на единицу, а текст при этом смещается на один символ влево.

Многократный вызов функции bstr() осуществляется методом setTimeout() через каждые 100 мс. Это продолжается до тех пор, пока переменная pos не достигнет значения равного числу удаляемых пробелов, после чего вызов функции bstr() прекращается.

8.19. Автоматическая компоновка страницы в зависимости от разрешения экрана

Ранее уже рассматривались различные способы компоновки web-страницы, в частности упоминалось о возможности гибкой компоновки, когда размер страницы и ее элементов задается в процентах от размеров окна браузера. Здесь мы рассмотрим возможность жесткой компоновки, при которой размеры элементов страницы и их координаты вычисляются относительно размера шрифта, заданного абсолютно. С целью адаптации внешнего вида страницы к различным разрешениям экрана размер шрифта будет изменен программой-сценарием после определения разрешения экрана. Рассмотрим возможный вариант такой программы на простом примере. Страница содержит пять элементов: текстовый блок и четыре рисунка. Компоновка элементов показана на рис. 8.18.

Адаптация к разрешению монитора достигается за счет изменения координат абсолютно позиционированных элементов и их размеров. Программа предназначена для адаптации только к разрешениям монитора 800 × 600 и 1024 × 768. При желании можно уве-

личить число проверок разрешений монитора, определить дополнительные масштабирующие коэффициенты и текстовые стили, что позволит без особого труда расширить диапазон адаптации. Полный текст документа приведен в листинге 8.27.

Рис. 8.18. Результат компоновки элементов страницы с использованием абсолютного позиционирования

Листинг 8.27. Пример адаптации координат и размеров объектов к разрешению монитора

```
<html>
<head>
<title>Изменение размера</title>
<style type="text/css">
.ris1 {position: absolute;
       left: 0.25em;
       top: 0.25em;
       width: 6.25em;
       height: 4.17em;
}
.ris2 {position: absolute;
       left: 3.83em;
       top: 2.83em;
       width: 6.25em;
```

```
        height: 4.17em;
}
.ris3 {position: absolute;
        left: 15.5em;
        top: 0.25em;
        width: 6.25em;
        height: 4.17em;
}
.ris4 {position: absolute;
        left: 11.96em;
        top: 2.83em;
        width: 6.25em;
        height: 4.17em;
}
.txt {font: 700 24px "Monotype Corsiva";
        text-align: center;
}
.txt1 {font: 900 31px "Monotype Corsiva";
        text-align: center;
}
</style>
</head>
<body id=b1>
<script>
if (screen.width>1000) {b1.className="txt1"};else
{b1.className="txt"}
</script>
<div style="position: absolute; left: 6.54em; top: 0.33em;
width: 8.9em; height:1.7em">
Добро пожаловать в Петродворец</div>
<img src=k1.jpg class="ris1">
<img src=k2.jpg class="ris2">
<img src=k3.jpg class="ris3">
<img src=k4.jpg class="ris4">
</body>
</html>
```

Абсолютными единицами задан только размер шрифта. Для его форматирования созданы две таблицы стилей `txt` и `txt1`. Размер шрифта, заданный стилем `txt`, определяет внешний вид шрифта при разрешении монитора 800 × 600, а стилем `txt1` — при разрешении монитора 1024 × 768 и вычисляется путем умножения размера, заданного в `txt`, на масштабирующий коэффициент. Масштабирующий коэффициент вычисляется следующим образом 1024 : 800 = 1,28.

Для абсолютного позиционирования и задания размеров рисунка созданы четыре таблицы стилей (`ris1`,..., `ris4`). Все размеры заданы в относительных единицах em, это означает, что все они будут отсчитываться от размера шрифта. Так же осуществлено позиционирование и самого текстового блока.

Программа содержит только один оператор условного перехода, в котором свойство `width` (ширина) объекта `screen` (экран) сравнивается с 1000. Если значение превышено, то свойству `className` объекта с индивидуальным именем `b1` присваивается значение `txt1`, в противном случае — значение `txt`. Индивидуальное имя `b1` присвоено метке `<BODY>`, поэтому этим присваиванием мы определяем стиль форматирования текста всего документа. В общем случае текст может содержать несколько фрагментов с разными стилями форматирования. Тогда следует абсолютно задать размер только одного фрагмента текста, а размеры текста остальных фрагментов должны быть заданы в относительных единицах.

Работу по компоновке страницы рекомендуется осуществлять в программе Dreamweaver, предварительно установив разрешение монитора 800 × 600 (*см. раздел 7.8.2 "Пример компоновки страницы с использованием слоев"*). В результате компоновки координаты и размеры абсолютно позиционированных элементов будут заданы в пикселях. Оставив в пикселях только размер шрифта, необходимо остальные размеры перевести в относительные единицы путем деления абсолютной единицы размера на размер шрифта. Создав таблицу стилей с увеличенным размером текста и вставив в документ программу-сценарий, можно перейти к его просмотру в браузере. При изменении разрешения монитора на 1024 × 768 внешний вид страницы изменится, но после нажатия кнопки **Обновить** восстановится вновь.

Глава 9

Добавление статических и динамических эффектов

Для создания различных эффектов путем воздействия как на отдельные элементы web-страницы, так и на всю страницу в целом предназначены *фильтры*. Фильтры разработаны компанией Microsoft и поддерживаются браузером MS Internet Explorer, начиная с версии 4.0. Для браузера MS Internet Explorer версии 5.5 был изменен не только синтаксис задания фильтров, но и принцип функционирования динамических фильтров, появились новые фильтры. Несмотря на то, что браузеры версий 5.5 и 6.0 поддерживают старый синтаксис и старые фильтры, не рекомендуется использовать их при создании новых страниц. В настоящей главе приведены сведения только о новых фильтрах, в которых используется новый синтаксис.

Фильтры подразделяются на *статические* и *динамические*. Статические фильтры создают эффекты, не изменяющиеся во времени, динамические — позволяют получить анимационный эффект. Фильтры задаются в параметре `STYLE` следующим образом:

```
FILTER: progid:DXImageTransform.Microsoft.имя_фильтра (параметры)
```

Параметры со значениями перечисляются через запятую. Если параметры не указаны, то фильтр будет применен со значениями параметров, предусмотренными по умолчанию. При отсутствии параметров круглые скобки можно не писать.

Фильтры применимы не ко всем меткам, поэтому в некоторых случаях используют метки контейнеры `<DIV>…</DIV>` или

<SPAN>…</SPAN>, к которым фильтры можно применить, если они позиционированы абсолютно или указаны размеры элемента.

9.1. Статические фильтры

Рассмотрим статические фильтры:

- `Shadow (color=значение, direction=значение, strength=значение)` — применяется для создания тени. Параметром `color` устанавливается цвет тени, `direction` — угол поворота тени в градусах, `strength` — глубина тени в пикселях;
- `DropShadow (color=значение, offx=значение, offy= значение, positive=значение)` — применяется для создания отброшенной тени. Назначение параметра `color` аналогично описанному выше, параметры `offx` и `offy` предназначены для задания смещения тени в пикселях по горизонтали и вертикали соответственно. Отсчет по оси X — вправо, по оси Y — вниз. Параметр `positive` может принимать два значения: `true` (по умолчанию) — обычная тень, `false` — тень в виде маски, закрашенной заданным цветом и прозрачными буквами;
- `Glow (color=значение, strength=значение)` — создает эффект свечения вокруг объекта. Параметр `color` аналогичен предыдущим случаям, параметр `strength` позволяет задать ширину свечения в пикселях.

  Так выглядит текст документа, в котором используются перечисленные выше фильтры (листинг 9.1).

Листинг 9.1. Текст документа, в котором к фрагментам текста применены фильтры Shadow, DropShadow и Glow

```
<html>
<head>
<title>Фильтры</title>
</head>
<body>
<div style="font-size: 48; position: absolute; top: 10; left: 5;
filter: progid:DXImageTransform.Microsoft.Shadow(color=green,
```

```
direction=60, strength=7)">
Фильтр Shadow - тень
</div>
<div style="font-size: 48;  position: absolute; top: 70;
left:5;
 filter:
progid:DXImageTransform.Microsoft.DropShadow(color=blue,
offx=5,offy=5)">
Фильтр DropShadow - отброшенная тень
</div>
<div style="font-size: 48;  position: absolute; top: 190;
left:5;
 filter:
progid:DXImageTransform.Microsoft.DropShadow(color=blue,
offx=5,offy=5, positive=false)">
Фильтр DropShadow - отброшенная тень
</div>
<div style="font-size: 48; position: absolute; top: 310;
left: 5;
filter: progid:DXImageTransform.Microsoft.Glow(color=red,
strength=5)">
Фильтр Glow - свечение
</div>
</body>
</html>
```

Для задания каждого фильтра используется метка-контейнер `<DIV>`. В параметре `STYLE`, кроме задания соответствующего фильтра, заданы размеры шрифта, абсолютное позиционирование текстового блока и параметры позиционирования. Напомним, что абсолютное позиционирование является одним из условий применения фильтра к содержимому контейнера `<DIV>`. Результаты действия трех перечисленных фильтров показаны на рис. 9.1.

- `MaskFilter (color=значение)` — создает маску прямоугольной формы, закрашенную цветом, заданным в параметре `color`, а пиксели текста становятся прозрачными. Пример задания фильтра в документе:

  ```
  <div style="position: absolute; top:10; left:2; filter:
  ```

```
progid:DXImageTransform.Microsoft.MaskFilter(color=red)
">
Фильтр MaskFilter - маска
</div>
```

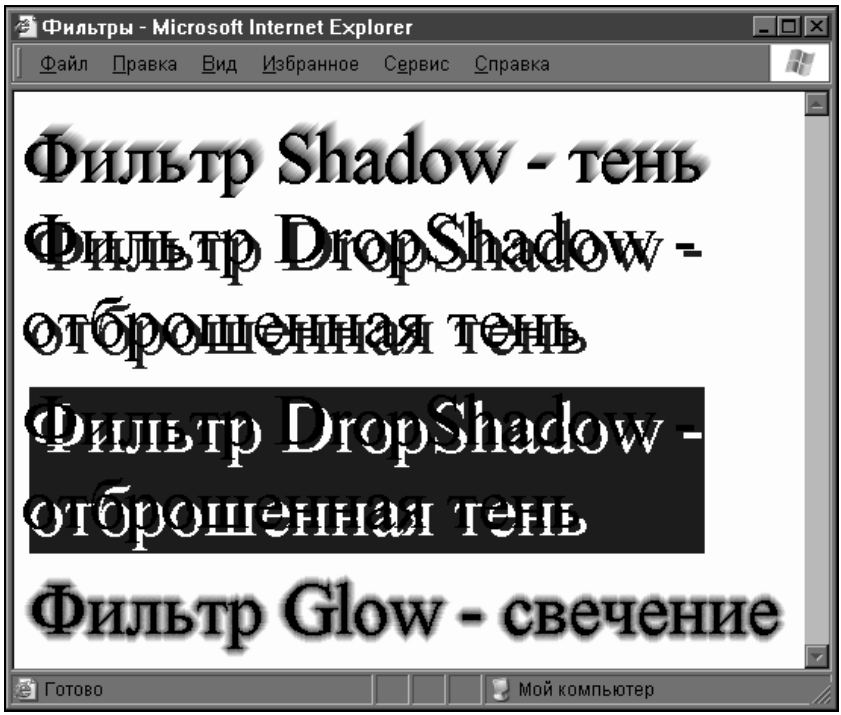


Рис. 9.1. Результаты применения к фрагментам текста фильтров Shadow, DropShadow и Glow

- `Blur (PixelRadius=значение, MakeShadow=значение)` — создает эффект плоского размытия объекта. Применяется как к тексту, так и к рисункам. Параметр `PixelRadius` задает ширину области размытия в пикселях, параметр `MakeShadow` может принимать значения: `true` — объект отображается как тень и `false` — обычное отображение (действует по умолчанию). Пример задания фильтра применительно к тексту:

```
<div style="position: absolute; top:60; left:2; filter:
progid:DXImageTransform.Microsoft.Blur(PixelRadius=3,
```

```
MakeShadow=true)">
Фильтр Blur - плоское размытие
</div>
```

- `MotionBlur (direction=значение, strength=значение, add=значение)` — создает эффект объемного размытия текста или рисунков. Параметр `direction` задает направление размытия в градусах, параметр `strength` — ширину размытия в пикселях. Параметр `add` может принимать значения `true` или `false`. В первом случае исходное изображение будет добавлено к размытому изображению, во втором случае воспроизводится только размытое изображение. Пример задания фильтра:

```
<div style="position: absolute; top: 160; left: 2; filter:
progid:DXImageTransform.Microsoft.MotionBlur(direction=
60, strength=7, add=true)">
Фильтр MotionBlur - объемное размытие
</div>
```

  На рис. 9.2. показаны результаты действия фильтров `MaskFilter`, `Blur` и `MotionBlur`. Для фильтра `Blur` задан параметр `MakeShadow=true`, для фильтра `MotionBlur` показаны два варианта применения фильтра к тексту: первый — параметр `Add=true`, второй — `Add=false`.

- `Wave (Strength=значение, Freq=значение, Phase=значение, Add=значение)` — создает волновые искажения текста или рисунков. Параметр `Strength` позволяет задать ширину искажения в пикселях, параметр `Freq` — количество пиков, `Phase` — сдвиг вдоль направления распространения волны в процентах от длины волны. Назначение параметра `Add` такое же, как в предыдущем случае. Пример применения фильтра к тексту:

```
<div style="font-size: 60; position: absolute; top: 11;
left: 71;
filter:
progid:DXImageTransform.Microsoft.wave(add=true,
strength=8, freq=15, phase=30)">
Фильтр wave - волновые искажения
</div>
```

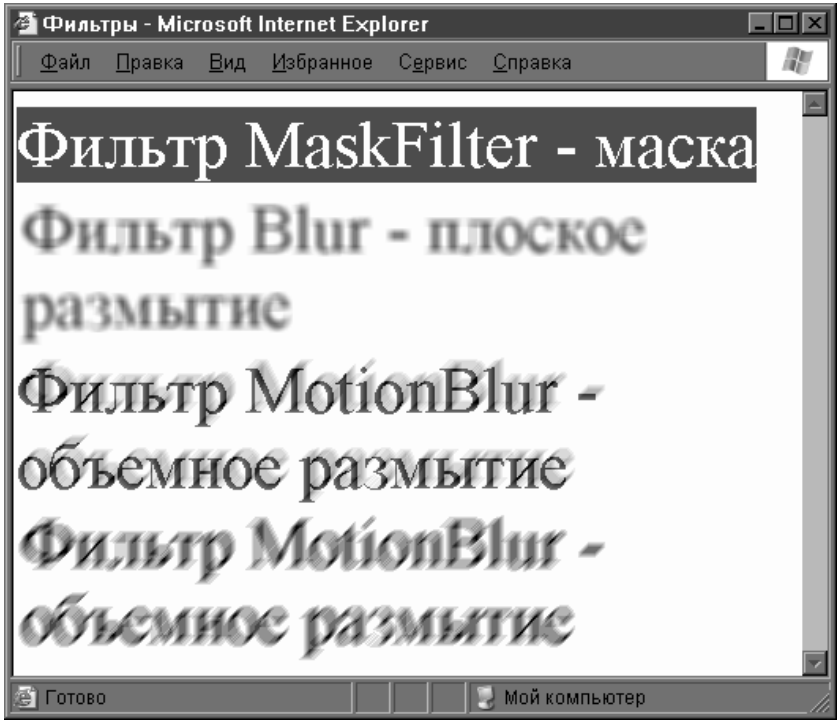


Рис. 9.2. Результаты применения фильтров MaskFilter, Blur (параметр MakeShadow = true), MotionBlur (параметр Add = true), MotionBlur (параметр Add = false)

- `Gradient (startColorStr=значение, endColorStr= значение, GradientType=значение)` — создает градиентную фоновую заливку отдельного блока или всей страницы. Параметры `startColorStr` и `endColorStr` предназначены для задания начального и конечного цвета градиентной заливки. Параметр `GradientType` может принимать два значения: 1 — заливка по горизонтали, 0 — заливка по вертикали (по умолчанию). Пример применения фильтра ко всей странице:

```
<body style="filter:
progid:DXImageTransform.Microsoft.Gradient (startColorStr=yellow, endColorStr=green, GradientType= 0)">
```

Пример применения фильтра к текстовому блоку:

```
<div style="font-size: 36; position: absolute; top:
```

```
100; left: 10; filter:
progid:DXImageTransform.Microsoft.Gradient(startColorSt
r=magenta, endColorStr=cyan, GradientType=1);
width:500; height:85">
Фильтр Gradient, примененный к текстовому блоку
</div>
```

- `Emboss` — создает эффект выпуклости объекта. Применяется как к тексту, так и к рисункам. В случае применения к тексту требуется задание фоновой заливки текстового блока. Однако независимо от заданного цвета фона эффект будет создаваться с использованием оттенков серого цвета. Результат будет наиболее контрастным, если задать белый цвет фона. Пример:

```
<div style="font-size: 36; position: absolute; top:195;
left:   10;   background-color:   white;   width=500;
height=85;
filter:progid:DXImageTransform.Microsoft.Emboss" >
Фильтр Emboss - создание выпуклости
</div>
```

- `Engrave` — создает эффект вдавленности объекта. Применяется аналогично фильтру `Emboss`. Пример:

```
<div style="font-size: 36; position: absolute; top:290;
left:   10;   background-color:   white;   width=500;
height=85; filter:
progid:DXImageTransform.Microsoft.Engrave">
Фильтр Engrave - создание вдавленности
</div>
```

  Результаты действия фильтров `Wave`, `Gradient`, `Emboss` и `Engrave` показаны на рис. 9.3. Ко всей странице применена вертикальная градиентная фоновая заливка, а к одному из блоков — горизонтальная градиентная заливка.

- `Alpha (opacity=значение, style=значение)` — создает эффект прозрачности изображения. Параметр `opacity` позволяет задать степень непрозрачности изображения в процентах от 0 (полная прозрачность) до 100 (полная непрозрачность) — действует по умолчанию. С помощью параметра `style` задается способ применения фильтра. Параметр может принимать следующие значения: 0 — равномерно по всему изображению, 1 — вдоль отрезка прямой, 2 — от центра эллипса, вписанного в границы изображения до краев изображения,

3 — от центра изображения до его краев. При выборе значения "1" необходимо задать значения еще четырех параметров: `startX`, `startY`, `finishX` и `finishY` — координаты начала и конца линии относительно левого верхнего угла рисунка, вдоль которой изменяется прозрачность. Пример применения фильтра `Alpha` к рисунку:

```
<img style="position: absolute; left: 47; top: 29; filter: progid:DXImageTransform.Microsoft.Alpha (opacity=100, style=1, startX=0, startY=0, finishX=150, finishY=225)" src="8.jpg" width="150" height="225">
```

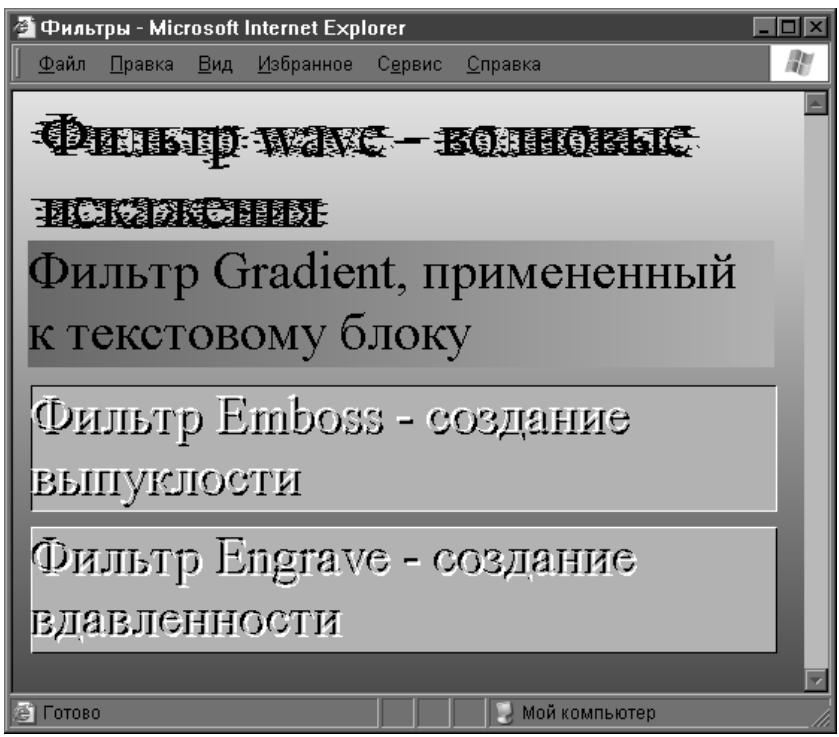


Рис. 9.3. Результаты применения фильтров Wave, Gradient, Emboss и Engrave

Результаты действия фильтра `Alpha` при использовании различных параметров показаны на рис. 9.4 и 9.5.

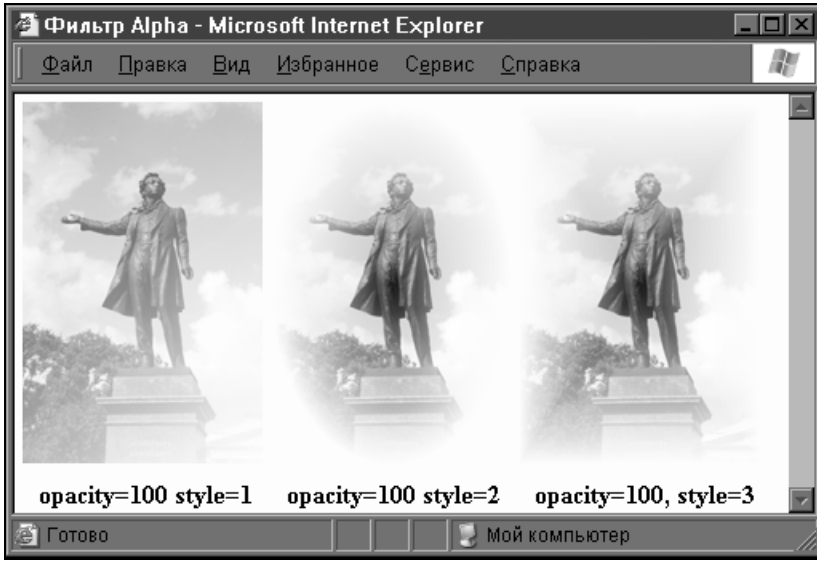


Рис. 9.4. Результаты применения фильтра Alpha к рисунку при неизменном параметре opacity и изменении параметра style

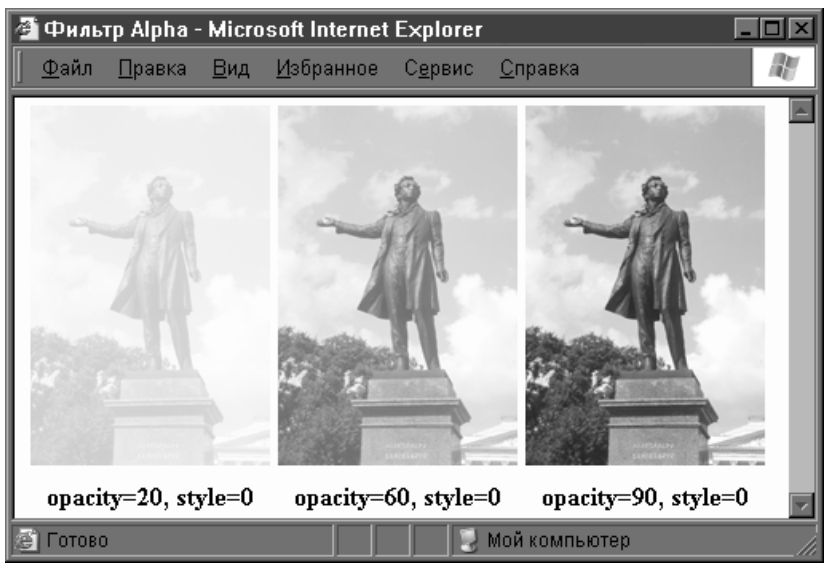


Рис. 9.5. Результаты применения фильтра Alpha к рисунку при неизменном параметре style и изменении параметра opacity

- `BasicImage (Rotation=значение, Mirror=значение, Grayscale=значение, XRay=значение, Invert=значение, Opacity=значение)` — может использоваться для создания различных эффектов, которые задаются значениями параметров:
  - `Rotation` — поворот объекта по часовой стрелке. Может принимать 4 значения: 0 — нет поворота (по умолчанию), 1 — поворот на 90°, 2 — на 180°, 3 — на 270°;
  - `Mirror` — зеркальное отображение объекта по горизонтали. Возможные значения: 0 — нет отображения (по умолчанию), 1 — есть зеркальное отображение;
  - `Grayscale` — преобразование цветного изображения в черно-белое. Может принимать значения: 0 — не применять этот эффект (по умолчанию), 1 — применить эффект;
  - `XRay` — преобразование цветного объекта в черно-белое и одновременно в негативное. Принимает значения: 0 — не производить преобразование (по умолчанию), 1 — преобразовать;
  - `Invert` — объект, оставаясь цветным, становится негативным. Может принимать значения: 0 — эффект не применяется (по умолчанию), 1 — применить эффект;
  - `Opacity` — позволяет задать степень непрозрачности объекта. Может иметь дробные значения от 0.00 (полная прозрачность) до 1.00 (полная непрозрачность по умолчанию).

  Допускается применение к одному объекту нескольких эффектов, но если какие-то эффекты не используются, то соответствующие параметры можно не писать.

  Пример применения фильтра `BasicImage` в листинге 9.2.

Листинг 9.2. Применение фильтра BasicImage к текстовому блоку и рисунку

```
<HTML>
<HEAD>
<TITLE>Фильтр BasicImage</TITLE>
</HEAD>
```

```
<BODY bgColor=#00ffff>
<DIV style="font-size=16; filter:
progid:DXImageTransform.Microsoft.BasicImage ( Rotation=3);
position: absolute; top: 20; left: 5; width=150" ><B><p
align=justify>Здесь располагается текстовый блок, развернутый
на 270 градусов с помощью фильтра BasicImage</B></DIV>
<IMG style= "position: absolute; left: 110px; top: 5"
src="1.jpg">
<IMG style="filter:
progid:DXImageTransform.Microsoft.BasicImage (Mirror=1); po-
sition: absolute; left: 230; top: 5" src="1.jpg">
</BODY>
</HTML>
```

Документ содержит блок с текстом и два одинаковых рисунка. Фильтр применен к блоку, который развернут на 270° (`Rotation=3`) и второму рисунку, который отображен зеркально (`Mirror=1`).

Результат действия фильтра показан на рис. 9.6.

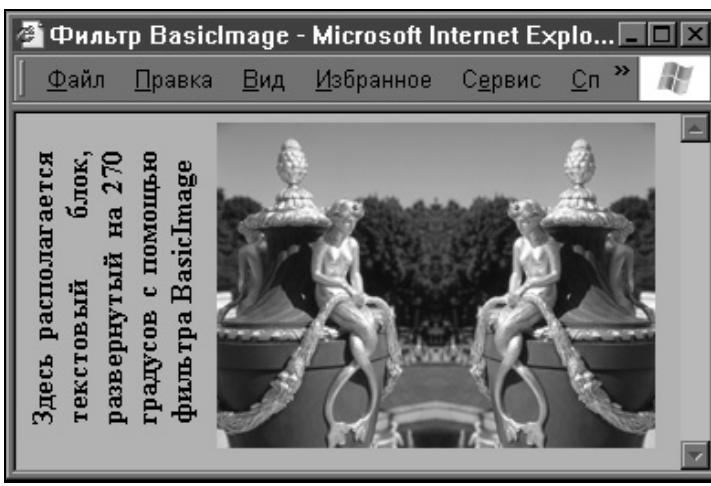


Рис. 9.6. Результат применения фильтра BasicImage к текстовому блоку (Rotation=3) и второму рисунку (Mirror=1)

- `Chroma (color=значение)` — делает прозрачным цвет изображения, заданный параметром `color`. Фильтр `Chroma` может сделать прозрачным один цвет, но при многократном приме-

нении фильтра к одному и тому же объекту можно сделать прозрачными несколько цветов. Не рекомендуется применять фильтр к рисункам в формате JPEG, так как результат фильтрации в этом случае трудно назвать удовлетворительным. Пример применения фильтра к рисунку приведен в листинге 9.3.

Листинг 9.3. Применение фильтра Chroma к рисунку

```
<html>
<head>
<title>Фильтр Chroma</title>
</head>
<body bgcolor="#00FFFF">
<img  src="chroma.gif"><br>
<img
style="filter:progid:DXImageTransform.Microsoft.Chroma(color=
yellow)" src="chroma.gif"><br>
<img
style="filter:progid:DXImageTransform.Microsoft.Chroma(color=
yellow) progid:DXImageTransform.Microsoft.Chroma(color=blue)"
src="chroma.gif">
</body>
</html>
```

Рисунок содержит фоновую заливку желтого цвета, буквы синего цвета, красный контур вокруг букв повторен три раза. В первом случае к нему не применяется фильтр, во втором случае фильтр делает прозрачным желтый цвет, а в третьем — он применен дважды, делая прозрачным как желтый, так и синий цвет. Результат показан на рис. 9.7.

- `AlphaImageLoader (src=значение, sizingmethod= значение)` — размещает изображение, заданное параметром `src`, между фоном и содержимым объекта, на которое распространяется действие фильтра. Параметр `sizingmethod` может принимать три значения: `Crop` — вставляемое изображение обрезается по границе объекта, `Image` — объект обрезается по границе изображения, `Scale` — изображение масштабируется до границ объекта.

Рис. 9.7. Результаты применения фильтра Chroma к верхнему рисунку: однократного (color=yellow) и двукратного (color=yellow и color=blue)

Рассмотрим два примера применения фильтра с различными значениями параметра `sizingmethod`. В обоих случаях фильтр применяется к блоку, включающему текст желтого цвета на зеленом фоне. Размеры блока 300 × 150 пикселей. В параметре src фильтра указано имя файла рисунка с размерами 200 × 215 пикселей:

```
<div style="font-size=16; color=yellow; width: 300;
height: 150; background-color: green; padding: 5; fil-
ter: progid:DXImageTransform.Microsoft.AlphaImageLoader
(src=r1.jpg, sizingmethod=crop)">
<p align="justify"><b>Этот блок включает … </b>
</div>
```

Второй пример полностью аналогичен первому. Различие между ними состоит в значении параметра `sizingmethod`. В первом случае параметр `sizingmethod` имеет значение `Crop`, и так как размеры рисунка по высоте больше размеров блока, рисунок обрезается по вертикали. Во втором — параметру `sizingmethod` задано значение `Image`, поэтому рисунок воспроизводится полностью, а блок обрезается по горизонтали.

В обоих случаях рисунок располагается между содержимым блока (в данном случае текстом) и фоном (рис. 9.8).

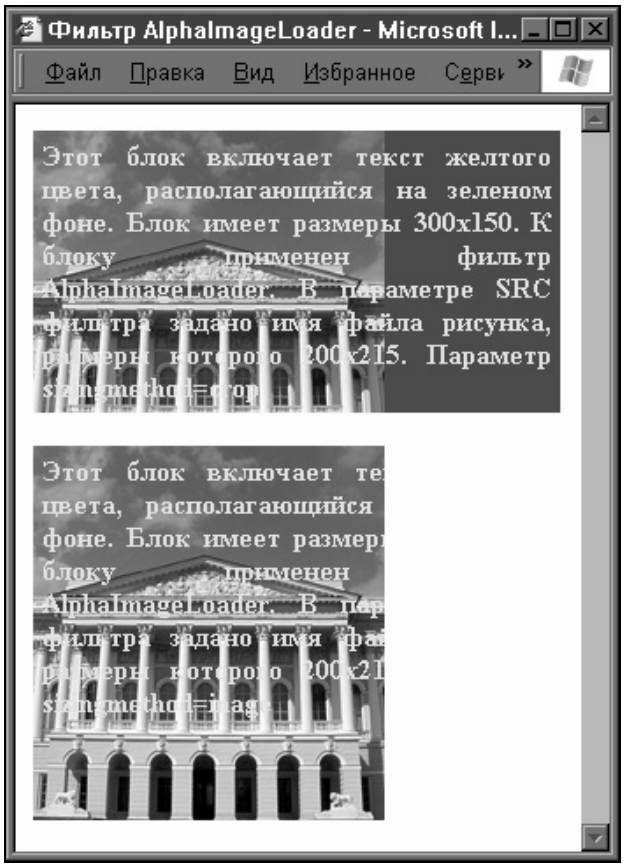


Рис. 9.8. Применение фильтра AlphaImageLoader к текстовому блоку с параметром sizingmethod = Crop (верхний рисунок) и с параметром sizingmethod = Image

- `Pixelate (maxsquare=значение)` — объединяет соседние пикселы изображения в квадратные ячейки с однородной заливкой, создавая эффект мозаики. Размер квадратных ячеек в пикселях задается параметром `maxsquare`. Результаты фильтрации изображения показаны на рис. 9.9.

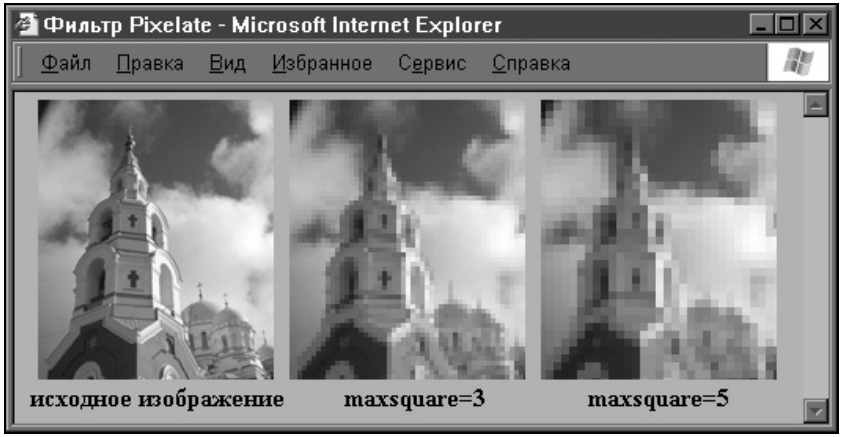


Рис. 9.9. Результаты применения фильтра Pixelate с различными значениями параметра maxsquare

- `Light` — создает эффект одного или нескольких источников света. Управление фильтром осуществляется программой на JavaScript. В программе можно использовать несколько методов, но важнейшим из них является `AddCone ()`, имеющий 10 параметров. Первый и второй параметры — координаты источника по осям X и Y в пикселях относительно левого верхнего угла объекта; третий — номер слоя; четвертый и пятый — координаты точки, в которую направлен источник; шестой, седьмой и восьмой — составляющие цвета в формате RGB; девятый — интенсивность источника (до 100); десятый — половина угла распространения света. В случае использования нескольких источников, каждый из них следует помещать в отдельном слое.

  В качестве примера использования фильтра `Light` рассмотрим следующий документ (листинг 9.4).

Листинг 9.4. Пример создания эффектов одного и двух источников света с помощью фильтра Light применительно к двум одинаковым изображениям

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

```
<title>Фильтр Light</title>
</head>
<body bgcolor="#00FFFF">
<img id=i1 style="filter:
progid:DXImageTransform.Microsoft.light" src="1.jpg"><p>
<img id=i2 style="filter:
progid:DXImageTransform.Microsoft.light" src="1.jpg">
```

Рис. 9.10. Результаты создания эффектов одного и двух источников света фильтром Light

```
<script>
i1.filters[0].addCone(-5,-3,1,500,350,255,255,255,7,20)
i2.filters[0].addCone(15,0,1,350,250,255,255,255,7,20)
i2.filters[0].addCone(280,0,2,-50,300,255,255,255,7,20)
</script>
</body>
</html>
```

Фильтр применен к двум изображениям, в качестве которых используется один и тот же рисунок, размеры которого 30 × 202 пикселя. Размеры рисунка приводятся не случайно. Все координаты источников света должны выбираться с учетом размеров рисунка. Точка расположения источника и точка, в которую он направлен, могут располагаться и за пределами рисунка.

Второе изображение с индивидуальным именем `i2` содержит два источника света, первый из которых располагается в слое 1, а второй — в слое 2. Программа на языке JavaScript, управляющая работой фильтра, должна обязательно располагаться после меток `<IMG>` с описаниями фильтров с тем, чтобы к началу работы программы эти метки уже были загружены.

Может возникнуть желание непременно расположить программу в более привычном месте, а именно в блоке `<HEAD>`. Тогда ее нужно оформить в виде функции и вызывать на выполнение с помощью параметра обработки события `onLoad`, поместив его в метку `<BODY>`. На рис. 9.10 показаны эффекты создания одного и двух источников света.

9.2. Динамические фильтры (переходы)

Динамические фильтры часто называют *переходами*. Такое наименование не случайно. С одной стороны, в документе они описываются аналогично статическим фильтрам, хотя и требуют для управления дополнительной программы на JavaScript (впрочем, управляющая программа нужна и для работы статического фильтра Light). С другой — результатом работы динамического

фильтра является осуществление перехода с применением эффекта из одного состояния объекта в другое, что позволяет называть такие фильтры переходами.

Как уже отмечалось, динамические фильтры задаются аналогично статическим фильтрам, но требуется программа-сценарий для управления фильтром, в которой могут использоваться следующие методы:

- `Apply ()` — фиксирует видимое изображение с тем, чтобы оно не было заменено новым без применения эффекта после замены имени одного изображения на другое;
- `Play ()` — применяет фильтр;
- `Stop ()` — мгновенно завершает действие фильтра.

Текст программы-сценария для всех динамических фильтров может быть одинаковым, поэтому он будет приведен только при описании фильтра `Barn`.

9.2.1. Описание динамических фильтров

Рассмотрим виды динамических фильтров, а также примеры их применения.

- `Barn (motion=значение, orientation=значение, duration=значение)` — создает эффект вертикальных (`orientation=vertical`) или горизонтальных (`orientation=horizontal`) задергивающихся (`motion=in`) или раскрывающихся (`motion=out`) штор. Параметр `duration` предназначен для задания времени перехода в секундах из одного состояния изображения в другое и используется во всех динамических фильтрах. Пример применения фильтра для замены одного рисунка другим с применением эффекта приведен в листинге 9.5.

Листинг 9.5. Пример применения фильтра Barn для замены одного рисунка другим с применением эффекта

```
<html>
<head>
<title>Динамический фильтр</title>
<script>
```

```
nomber_1="1.jpg"
nomber_2="2.jpg"
s=0
function pr1()
{
pic1.filters[0].apply()
if (s==0) {pic1.src=nonber_2; s=1}
else {pic1.src=nonber_1; s=0}
pic1.filters[0].play()
}
</script>
</head>
<body>
  <img id=pic1 style="position: absolute; top: 26; left: 190;
filter: progid:DXImageTransform.Microsoft.Barn (motion=in,
orientation=vertical, duration=5)"
      src=1.jpg>
<div style="position: absolute; top: 300; left:250">
<input type="button" value="Изменить" name="KN1" on-
click="pr1()">
</div>
</body>
</html>
```

К графическому изображению с уникальным именем `pic1`, заданным в параметре `ID`, применен фильтр `Barn` со временем перехода 5 с.

В начале программы-сценария переменным `nomber_1` и `nomber_2` присваиваются имена файлов рисунков, которые должны сменять друг друга с применением эффекта. Переменной `s`, которая играет роль "флажка", присваивается начальное значение 0. Функция `pr1()` предназначена для управления переходом от одного рисунка к другому. В первой строке функции указывается: имя объекта, к которому применен фильтр (`pic1`), имя фильтра (`filters[0]`), которое в данном случае означает, что из семейства примененных фильтров выбирается с порядковым номером 0, и применяе-

мый метод (`apply()`). Метод `apply()` зафиксирует изображение того рисунка, который к моменту вызова функции является видимым.

Далее при помощи оператора условного перехода и "флажка" организовано поочередное изменение имени файла изображения. Например, при первом вызове функции `pr1()` имя файла изображения получит значение `nomber_2`, а флажок изменит свое значение на единицу. При следующем вызове функции значением выражение `s==0` будет `false`, следовательно, имя файла изображения получит значение `nomber_1`, а флажок снова приобретет значение 0.

Следующая запись `pic1.filters[0].play()` означает применение фильтра. Так как имя файла изображения перед этим было изменено, то после применения фильтра появится новое изображение с применением эффекта перехода. Вызов функции осуществляется после щелчка по кнопке с надписью **Изменить**.

- `Blinds (bands=значение, direction=значение, duration=значение)` — создает эффект жалюзи:
  - горизонтальных, открывающихся вниз (`direction=up`);
  - горизонтальных, открывающихся вверх (`direction=down`);
  - вертикальных, открывающихся вправо (`direction=right`);
  - вертикальных, открывающихся влево (`direction=left`).

  Параметр `bands` позволяет задать количество полос.
- `CheckerBoard (direction=значение, squaresX=значение, squaresY=значение, duration=значение)` — создает эффект "шахматной доски". Параметр `direction` позволяет задать направление раскрытия клеток и может принимать значения: `up`, `down`, `right`, `left`. С помощью параметров `squaresX` и `squaresY` можно задать количество клеток по горизонтали и вертикали.
- `Fade (overlap=значение, duration=значение)` — создает эффект плавного исчезновения старого изображения и появления нового. Параметр `overlap` может принимать значения от 0.0 до 1.0 и влияет на степень одновременного присутствия двух изображений. При `overlap=1` с началом исчезновения

старого изображения уже появляется новое. При `overlap=0` старое изображение исчезает полностью и после этого начинает появляться новое.

- `GradientWipe (gradientsize=значение, wipestyle=значение, motion=значение, duration=значение)` — создает эффект развертки нового изображения на фоне старого. Параметр `gradientsize` может принимать значения от 0.0 до 1.0 и позволяет регулировать степень размытости границы между старым и новым изображениями. Значение 0 соответствует максимально резкому переходу, а значение 1 — максимально размытой границе. Параметр `wipestyle` может принимать два значения: 0 — развертка слева направо, 1 — сверху вниз. Параметр `motion` также может принимать только два значения: `forward` — развертка в заданном направлении, `reverse` — развертка в направлении, обратном заданному, в параметре `wipestyle`, т. е. справа налево или снизу вверх.

- `Inset (duration=значение)` — создает эффект диагональной развертки из левого верхнего угла в правый нижний. Других параметров не имеет.
- `Iris (irisstyle=значение, motion=значение, duration=значение)` — создает эффекты переходов различной формы и направленных внутрь изображения или наружу. Параметр `irisstyle` позволяет изменять форму перехода и может принимать значения: `diamond` — ромбовидный, `circle` — в виде окружности, `cross` — X-образный, `plus` — крестообразный, `square` — прямоугольный, `star` — звездообразный. Параметр `motion` задает направление перехода: `in` — внутрь изображения, `out` — наружу.
- `Pixelate (maxsquare=значение, duration=значение)` — создает эффект перехода путем объединения соседних пикселей изображения в квадратные ячейки с однородной заливкой. Параметр `maxsquare` позволяет задать максимальный размер ячейки в пикселях и может принимать значения от 2 до 50.
- `RadialWipe (wipestyle=значение, duration=значение)` — создает эффект радиальной развертки. Параметр `wipestyle` может принимать значения: `clock` — вращение радиуса, напоминающее вращение стрелки часов, `wedge` — вращение

двух радиусов в виде расширяющегося клина, `radial` — вращение радиуса относительно левого верхнего угла изображения.

- `RandomBars (orientation=значение, duration=значение)` — создает эффект перехода путем появления случайных горизонтальных или вертикальных полос. Параметр `orientation` может принимать значения: `horizontal` — горизонтальные полосы, `vertical` — вертикальные полосы.
- `RandomDissolve (duration=значение)` — создает эффект поточечного проявления нового изображения на месте старого. Других параметров не имеет.
- `Slide (slidestyle=значение, bands=значение, duration=значение)` — создает эффект сдвига старого изображения и появления нового. Параметр `slidestyle` может принимать значения: `hide` — двигается только старое изображение, `push` — двигаются оба изображения в одну сторону, `swap` — оба изображения двигаются в разные стороны. Параметру `bands` можно присвоить значения от 1 до 50, что приведет к делению изображений во время перехода на соответствующее число горизонтальных полос.
- `Spiral (gridsizeX=значение, gridsizeY=значение, duration=значение)` — создает эффект смены изображений путем развертки по спирали. Параметры `gridsizeX` и `gridsizeY` позволяют задать, на сколько частей будет поделено изображение по горизонтали для определения ширины вертикального витка и на сколько — по вертикали для определения ширины горизонтального витка, и могут принимать значения от 1 до 100.
- `Stretch (stretchstyle=значение, duration=значение)` — создает эффект растягивания изображений. Параметр `stretchstyle` может принимать значения: `hide` — новое изображение растягивается от левого края, а старое остается неподвижным; `push` — новое изображение растягивается от левого края, а старое сжимается к правому краю; `spin` — новое изображение растягивается от центра к левому и правому краям.
- `Strips (motion=значение, duration=значение)` — создает эффект развертки по диагонали. Параметр `motion` задает на-

правление развертки: `leftup` — влево-вверх, `leftdown` — влево-вниз, `rightup` — вправо-вверх, `rightdown` — вправо-вниз.

- `Wheel (spokes=значение, duration=значение)` — создает эффект "мельницы". Параметр `spokes` задает число секторов, в которых одновременно происходит смена изображения, и может принимать значения от 2 до 20.
- `Zigzag (gridsizeX=значение, gridsizeY=значение, duration=значение)` — создает эффект зигзагообразной замены одного изображения другим. Параметры `gridsizeX` и `gridsizeY` позволяют задать размеры прямоугольника, который, перемещаясь зигзагообразно, формирует новое изображение. Параметр `gridsizeX` задает количество частей, на которое нужно разбить изображение по горизонтали для определения горизонтального размера прямоугольника, а параметр `gridsizeY` — вертикального размера прямоугольника. Параметры могут принимать значения от 1 до 100.

9.2.2. Примеры использования динамических фильтров

Одновременное использование нескольких фильтров применительно к различным изображениям

Сформулируем задачу следующим образом. В момент загрузки страницы начинают появляться три изображения с использованием разных эффектов, а через 3 с возникают еще три, также с использованием разных эффектов. В реальной ситуации, возможно, и не потребуется воспроизводить каждый рисунок со своим эффектом, но в данном примере важно показать, как реализовать такие возможности, применив при этом только одну функцию для управления фильтром (листинг 9.6).

Листинг 9.6. Пример одновременного применения различных фильтров к нескольким изображениям

```
<HTML>
<HEAD>
```

```
<TITLE>Пример1</TITLE>
<script>
function pr()
{
pr1(pic1)
pr1(pic2)
pr1(pic3)
setTimeout("pr1(pic4)", 3000)
setTimeout("pr1(pic5)", 3000)
setTimeout("pr1(pic6)", 3000)
}
function pr1(pic)
{
pic.style.visibility="hidden"
pic.filters[0].apply()
pic.filters[0].play()
pic.style.visibility="visible"
}
</script>
</HEAD>
<BODY onload = pr()>
<IMG ID=pic1 SRC="1.jpg" width=100 style="position:absolute;
left:30; top:10; filter: progid: DXImageTrans-
form.Microsoft.Stretch (stretchstyle=spin, duration=3)">
<IMG ID=pic2 SRC="2.jpg" width=100 style="position:absolute;
left:80; top:70; filter: progid: DXImageTrans-
form.Microsoft.GradientWipe (GradientSize=0, wipestyle=1, mo-
tion=reverse, duration=3)">
<IMG ID=pic3 SRC="3.jpg" width=100 style="position:absolute;
left:5; top:150; filter:
progid:DXImageTransform.Microsoft.Strips (motion=rightup, du-
ration=3)">
<IMG ID=pic4 SRC="4.jpg" width=100 style="position:absolute;
left:250; top:70;visibility:hidden; filter:
progid:DXImageTransform.Microsoft.RandomDissolve (dura-
tion=3)" >
<IMG ID=pic5 SRC="5.jpg" width=100 style="position:absolute;
left:170; top:20; visibility:hidden; filter:
progid:DXImageTransform.Microsoft.Checkerboard (direc-
```

```
tion=right, SquaresX=5, SquaresY=5, duration=3)">
<IMG ID=pic6 SRC="6.jpg" width=100 style="position:absolute;
left:170; top:170; visibility:hidden; filter:
progid:DXImageTransform.Microsoft.Iris(irisstyle=cross, mo-
tion=out, duration=3)">
</BODY>
</HTML>
```

Чтобы осуществить многократный вызов функции `pr1()` с различными значениями параметра, создана вспомогательная функция `pr()`, в которой вызов функции `pr1()` с именами объектов `pic1`, `pic2`, `pic3` осуществляется непосредственно, а с именами `pic4`, `pic5`, `pic6` — через метод `setTimeOut()` с задержкой 3 с. В свою очередь, вызов функции `pr()` осуществляется при загрузке документа. В верхней строке функции `pr1()` предусмотрено скрытие объекта (`pic.style.visibility="hidden"`) с тем, чтобы в дальнейшем он появился с заданным эффектом (`pic.style.visibility="visible"`). Так как выполнение функции `pr1()` связано с загрузкой документа, то пользователь не заметит скрытия объектов `pic1`, `pic2`, `pic3`, а увидит только их постепенное появления с использованием эффекта. Что же касается объектов `pic4`, `pic5` и `pic6`, то для них необходимо предусмотреть предварительное скрытие, что и сделано в метках `<IMG>` соответствующих объектов. В противном случае они в течение 3 с находились бы на экране, после чего были бы скрыты и появились вновь с применением эффекта.

Выбор изображения из списка

Очередное изображение выбирается из списка, включающего более двух изображений в произвольном порядке, и появляется на фоне предыдущего с применением эффекта. Повторный выбор уже установленного изображения не должен приводить ни к каким изменениям.

Изображения предварительно должны быть приведены к одному размеру, например путем масштабирования и обрезки. После этого к ним можно применить одинаковое абсолютное позиционирование, в результате чего будет видимым только одно изображение (листинг 9.7).

Листинг 9.7. Пример выбора изображения в произвольном порядке с применением динамического эффекта

```
<html>
<head>
<title>Пример 2</title>
</head>
<script>
var z=0
function pr1(pic)
{
z++
if (pic.style.zIndex != z-1)
{
pic.style.visibility="hidden"
}
pic.filters[0].Apply()
pic.style.zIndex=z
pic.filters[0].Play()
pic.style.visibility="visible"
}
</script>
<body>
   <ul>
    <li><b><font size="5" onclick=pr1(pic1)>Рисунок
1</font></b></li>
    <li><b><font size="5" onclick=pr1(pic2)>Рисунок
2</font></b></li>
    <li><b><font size="5" onclick=pr1(pic3)>Рисунок
3</font></b></li>
    <li><b><font size="5" onclick=pr1(pic4)>Рисунок
4</font></b></li>
   </ul>
<img id=pic1 style="position: absolute; left: 180; top: 16;
      filter: progid:DXImageTransform.Microsoft.Blinds
(bands=8, direction=left, duration=3); visibility: visible"
```

```
      src="1.jpg">
<img id=pic2 style="position: absolute; left: 180; top: 16;
      filter: progid:DXImageTransform.Microsoft.Blinds
(bands=8, direction=left, duration=3); visibility: hidden"
      src="2.jpg">
<img id=pic3 style="position: absolute; left: 180; top: 16;
      filter: progid:DXImageTransform.Microsoft.Blinds
(bands=8, direction=left, duration=3); visibility: hidden"
      src="3.jpg">
<img id=pic4 style="position: absolute; left: 180; top: 16;
      filter: progid:DXImageTransform.Microsoft.Blinds
(bands=8, direction=left, duration=3); visibility: hidden"
      src="4.jpg">
</body>
</html>
```

После загрузки документа видимым будет изображение `pic1`, для которого задано `visibility: visible`, остальные изображения скрыты (`visibility: hidden`).

Перемещение выбранного изображения поверх остальных осуществляется путем изменения слоя, в котором оно располагается. Для этой цели используется свойство `Z-index` каскадной таблицы стилей. Объект, имеющий свойство `Z-index` больше остальных объектов, располагается поверх этих объектов. По умолчанию свойство `Z-index` объекта равно нулю. Переменная `z` — счетчик, предназначенный для увеличения свойства `Z-index` демонстрируемого объекта. При каждом вызове функции `pr1()` переменная `z` увеличивается на единицу. Условие `pic.style.zIndex != z-1` проверяет, является ли активизируемый объект новым или он стал видимым на предыдущем шаге. Появиться с применением эффекта может только скрытый объект, поэтому в случае выполнения условия объект скрывается.

Далее метод `Apply()` зафиксирует изображение. Свойству `Z-index` объекта присваивается новое значение и объект появляется с применением эффекта поверх остальных изображений. Если же условие не выполняется, то это означает, что осуществляется попытка активизировать уже видимый объект. Такой объект не будет скрыт и, следовательно, изменения изображения не произойдет.

Изображения демонстрируются одно за другим в заданной последовательности

Нажатие кнопки "Вперед" ведет к появлению следующего по порядку изображения, а кнопки "Назад" — предыдущего.

Подготовка и размещение изображений осуществляется так же, как и в предыдущем примере. Добавляются две кнопки и две функции: `prF()` — вызывается при нажатии на кнопку "Вперед" и `prR()` — вызывается при нажатии на кнопку "Назад" (листинг 9.8).

Листинг 9.8. Пример демонстрации изображений в заданной последовательности кнопками "Вперед" и "Назад"

```
<html>
<head>
<title>Пример 3</title>
</head>
<script>
var z=0
var kol=4 //общее число рисунков
var k=0
function prF()
{
if (k<kol-1)
{
k++
pr()
}
}
function prR()
{
if (k>0)
{
k--
pr()
```

```
}
}
function pr()
{
z++
document.images[k].style.visibility="hidden"
document.images[k].style.zIndex=z
document.images[k].filters[0].Apply()
document.images[k].filters[0].Play()
document.images[k].style.visibility="visible"
}
</script>
<body>
<img id=pic1 style="position: absolute; left: 180; top: 16;
      filter:
progid:DXImageTransform.Microsoft.RadialWipe(wipestyle=clock,
duration=3); visibility: visible"
      src="1.jpg">
<img id=pic2 style="position: absolute; left: 180; top: 16;
      filter:
progid:DXImageTransform.Microsoft.RadialWipe(wipestyle=clock,
duration=3); visibility: hidden"
      src="2.jpg">
<img id=pic3 style="position: absolute; left: 180; top: 16;
      filter:
progid:DXImageTransform.Microsoft.RadialWipe(wipestyle=clock,
duration=3); visibility: hidden"
      src="3.jpg">
<img id=pic4 style="position: absolute; left: 180; top: 16;
     filter:
progid:DXImageTransform.Microsoft.RadialWipe(wipestyle=clock,
duration=3); visibility: hidden"
      src="4.jpg">
  <span style="position: absolute; left: 63; top: 111">
       <input type="button" value="Вперед" name="B3" on-
click=prF()>
  </span>
  <span style="position: absolute; left: 68; top: 179">
```

```
      <input type="button" value="Назад" name="B3" on-
click=prR()>
  </span>
</body>
</html>
```

В программе используются три переменные. Назначение переменной `z`, как и в предыдущем примере, — увеличение свойства `Z-index` демонстрируемого объекта. Переменная `kol` — количество используемых рисунков. При желании изменить количество рисунков достаточно изменить значение этой переменной. Переменная `k` — счетчик, определяющий порядковый номер активного рисунка. Назначение функции `prF()` — увеличение на единицу значения переменной `k` при каждом вызове функции до тех пор, пока не будет достигнуто максимального значения `kol-1` (нумерация рисунков ведется от 0). В этом случае значением выражения `k<kol-1` станет `false` и изменение переменной `k` прекратится. На практике это означает, что повторные нажатия кнопки "Вперед" после появления последнего рисунка, не будут приводить ни к каким изменениям. В случае изменения переменной `k` вызывается на выполнение функция `pr()`. Функция `prR()` предназначена для уменьшения на единицу значения переменной `k` и вызова функции `pr()` после нажатия кнопки "Назад" до тех пор, пока не будет показан рисунок с номером 0, т. е. значение условия `k>0` не станет `false`. Назначение функции `pr()` аналогично назначению функции `pr1()` из предыдущего примера, за исключением того, что в ней отсутствует проверка на новизну объекта, так как она может быть вызвана только при изменении номера рисунка. В отличие от предыдущего примера, переход к новому объекту осуществляется путем выбора рисунка с заданным порядковым номером `k` из семейства рисунков `document.images[k]`.

Демонстрация изображений в заданной последовательности с использованием одной кнопки

Как и в предыдущем случае, изображения демонстрируются в определенной последовательности, но для этого используется только одна кнопка, на которой надпись "Вперед" меняется на

надпись "Назад" при достижении последнего изображения и вновь восстанавливается надпись "Вперед" при завершении перехода от последнего изображения к первому.

В отличие от предыдущих примеров, замена изображения осуществляется при любом нажатии кнопки. Появилась новая переменная `shag`, которая принимает значения: 1 — осуществляется переход в направлении увеличения порядкового номера рисунка, −1 — уменьшение порядкового номера рисунка. При достижении максимального номера переменная `shag` изменяет значение на −1, а кнопка приобретает надпись "Назад". При достижении минимального номера — переменная `shag` становится равной 1, а на кнопке появляется надпись "Вперед". Для изменения надписи на кнопке используется свойство `value` объекта `KN1`. Текст документа с программой приведен в листинге 9.9.

Листинг 9.9. Пример демонстрации изображений в заданной последовательности с использованием одной кнопки

```
<HTML>
<HEAD>
<TITLE>Пример4</TITLE>
<SCRIPT>
var z=0
var kol=4 //общее число рисунков
var k=0
var shag=1
function prD()
{
k+=shag
pr()
if (k==0)
{
KN1.value="Вперед"
shag=1
}
if (k==kol-1)
```

```
{
KN1.value="Назад"
shag=-1
}
}
function pr()
{
z++
document.images[k].style.visibility="hidden"
document.images[k].style.zIndex=z
document.images[k].filters[0].Apply()
document.images[k].filters[0].Play()
document.images[k].style.visibility="visible"
}
</SCRIPT>
</HEAD>
<BODY>
<img id=pic1 style="position: absolute; left: 180; top: 16;
      filter:
progid:DXImageTransform.Microsoft.RadialWipe(wipestyle=clock,
duration=3); visibility: visible"
      src="1.jpg">
<img id=pic2 style="position: absolute; left: 180; top: 16;
      filter:
progid:DXImageTransform.Microsoft.RadialWipe(wipestyle=clock,
duration=3); visibility: hidden"
      src="2.jpg">
<img id=pic3 style="position: absolute; left: 180; top: 16;
      filter:
progid:DXImageTransform.Microsoft.RadialWipe(wipestyle=clock,
duration=3); visibility: hidden"
      src="3.jpg">
<img id=pic4 style="position: absolute; left: 180; top: 16;
     filter:
progid:DXImageTransform.Microsoft.RadialWipe(wipestyle=clock,
duration=3); visibility: hidden"
      src="4.jpg">
```

```
<SPAN style="position: absolute; left: 63; top: 111">
<INPUT name=KN1 onclick=prD() type=button value=Вперед>
</BODY>
</HTML>
```

Функция `prD()` предназначена для увеличения или уменьшения порядкового номера рисунка на 1 в зависимости от значения переменной `shag` и вызывается при нажатии кнопки. Кроме того, в ней осуществляются проверки достижения первого и последнего порядковых номеров рисунков с целью изменения переменной `shag` и надписи на кнопке, а также вызов функции `pr()`. Функция `pr()` аналогична используемой в предыдущем примере.

Предметный указатель

О

П

Р

С

Т

Ф

Ц