И. Шапошников

bhv
www.bhv.ru
www.bhv.kiev.ua

Web-сайт своими руками

МАСТЕР

ПРАКТИЧЕСКОЕ РУКОВОДСТВО

Основы языка HTML

Возможности Microsoft FrontPage2000

Администрирование и безопасность сайтов

Создание электронного магазина средствами WebShop 2000

Игорь Шапошников

Web-сайт своими руками

Дюссельдорф ◆ Киев ◆ Москва ◆ Санкт-Петербург

УДК 681.3.06

Книга поможет пользователям создать собственную Web-страницу и свой Web-сайт. Рассматриваются основы языка HTML, процедура создания и обработки графических изображений, мультимедийные варианты наполнения сайтов. Кратко освещены вопросы размещения, администрирования и безопасности создаваемых Web-сайтов.

В книге содержится необходимая теоретическая информация и практические советы, которые помогут освоить специализированные прикладные программы Microsoft FrontPage 2000 и WebShop 2000, а также некоторые приложения для работы с графикой.

Для широкого круга пользователей

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зав. редакцией	*Наталья Таркова*
Редактор	*Елена Бабко*
Компьютерная верстка	*Натальи Караваевой*
Корректор	*Наталия Першакова*
Дизайн обложки	*Ангелины Лужиной*
Зав. производством	*Николай Тверских*

Шапошников И. В.

Web-сайт своими руками. — СПб.: БХВ — Санкт-Петербург, 2000. — 224 с.: ил.

ISBN 5-8206-0130-0

Лицензия ЛР № 065953 от 15.06.98. Подписано в печать 07.07.2000.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 18,06.
Тираж 4000 экз. Заказ
"БХВ — Санкт-Петербург", 198005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар, № 77.99.1.953.П.950.3.99
от 01.03.1999 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в Академической типографии "Наука" РАН.
199034, Санкт-Петербург, 9-я линия, 12.

Содержание

Благодарности

Автор никогда не является единоличным создателем книги. Всегда есть те, без чьего участия книга никогда не появилась бы на свет, несмотря на все усилия автора. Мне хотелось бы поблагодарить всех, чья поддержка была так необходима.

Прежде всего, это, конечно, главный редактор издательства "BHV — Санкт Петербург" Екатерина Кондукова и принимающий редактор Елена Бабко. Без вашей неоценимой помощи, видения перспектив и отточенного чувства языка я бы просто не справился.

Также нельзя не отметить всех работников издательства, которые принимали участие в работе над этой книгой. Спасибо вам.

И уж наверняка этой книги бы не было, если бы не постоянная неоценимая поддержка моей жены — Даниленко Ольги. Любимая, все, что я делаю, я делаю для тебя.

Спасибо всем моим друзьям в Сети. Мой почтовый ящик всегда открыт для вас.

Игорь Шапошников
shiva1@interdacom.ru

Введение

Другу и любимой — Даниленко Ольге

Когда человек получает доступ к Интернету, что он видит, выходя на безбрежные информационные просторы? Куда он движется по скоростным цифровым магистралям? В подавляющем большинстве случаев он видит Web-сайты и Web-страницы. И какое же вполне объяснимое желание начинает вскоре охватывать нашего путешественника? Правильно! Ему хочется сделать свою страничку. У него возникает желание создать свой сайт. Пожалуй, я не буду далек от истины, если скажу, что это желание посещало абсолютное большинство пользователей Интернета. Хорошо тем, кто, как говорится, родился с микрочипом в голове. Тем, кто знает все правила работы в Интернете и умеет программировать. А что же делать всем остальным? Неужели для того, чтобы сделать свое личное представительство во всемирной сети, необходимо сидеть и изучать все эти языки программирования, правила работы и протоколы? На самом деле нет, не надо.

Посмотрим внимательно на ситуацию в компьютерном мире. С каждым годом программы становятся все более дружественными простому пользователю. Создатели операционных систем и программных продуктов очень хорошо понимают, что компьютер является рабочим инструментом не только и не столько для программистов, сколько для обычных людей, для которых работа с компьютером должна быть проста. Те, кого обычно называют пользователями, и являются финансовым мотором для всей компьютерной индустрии. В конечном счете, весь этот многомиллиардный (в долларовом выражении, естественно) бизнес ориентирован именно на обычного человека, у которого достаточно забот и без того, чтобы заниматься программированием. Естественно, для правильной и эффективной работы на компьютере необходимо учиться. Но ведь и для того, чтобы научиться водить автомобиль тоже необходимо время. Мы не должны посвящать учебе слишком много времени. У нас и без этого довольно много дел.

Я должен сразу предупредить, что это будет не просто. Самую простую страничку сделать легко, но если мы захотим создать что-то серьезное, придется как следует потрудиться. Так как мы сразу ограничили себя тем, что программированием заниматься не будем, то некоторая часть возможностей и средств для создания содержимого наших будущих страниц и сайтов будет для нас недоступна. Но вы очень удивитесь, когда узнаете, как многого мы можем добиться, правильно используя те программные средства, которые мы уже знаем и которыми пользуемся в повседневной работе.

Необходимо осознавать, что Интернет — это совершенно особая среда существования информации, поэтому, естественно, существует и специальное программное обеспечение, "заточенное" именно под его нужды. Осваивая его, мы попутно рассмотрим механизм его работы, и после этого сможем корректировать полученный результат для наиболее полного и адекватного выражения нашего замысла. Это действительно необходимо, так как полностью автоматическое создание какого-либо продукта средствами программного обеспечения редко приносит желаемый результат. Вы помните, как выглядели ваши документы, когда в старых версиях Microsoft Word вы применяли к ним команду автоматического форматирования? Мне, например, это никогда не нравилось. Документ получался слишком невзрачным, и его оформление абсолютно не соответствовало моему замыслу. Прогресс, конечно, не стоит на месте, и результат работы программ все больше похож на то, чего хотел добиться пользователь, но полностью полагаться на них все-таки не стоит.

Сначала мы научимся создавать отдельные Web-странички, а затем начнем объединять их в сайты. Для этих целей мы будем применять программный комплекс Microsoft FrontPage 2000. Мы рассмотрим особенности правильного построения сайтов, проблемы, связанные с графикой, различные мультимедийные варианты наполнения сайта, попробуем научиться находить баланс между содержательной частью страницы и скоростью ее загрузки. Затем мы научимся делать промышленные сайты, связанные с теми данными, которые используются в повседневной работе. То есть мы рассмотрим набор процедур и решений, которые позволят привязать сайт к корпоративной базе данных и организовать правильную работу с ней в дистанционном режиме. Мы увидим, как организуется работа с клиентами при помощи сайта и как сотрудники организации могут выполнять свою работу, не находясь непосредственно в офисе. Мы узнаем, как организовать средства безопасности нашего сайта, на котором может находиться большое количество ценной информации. Нас ждет также процедура создания сайта для электронной коммерции, в которой мы используем программу Web-Shop 2000 от Boomerang Software версии 5.10. И я еще раз подчеркиваю, мы обойдемся без программирования. Все, что я предложу вам в этой книге, может сделать любой человек, который уже умеет работать на компьютере.

Вам не кажется, что вступление несколько затянулось? На мой взгляд, нам уже пора перейти к тому, что мы собираемся делать — творить Web-сайт своими руками.

Глава 1

От страницы к сайту

Как это все устроено

Что же происходит в тот момент, когда мы набираем адрес сайта в строке Address или Location нашего браузера? Как мы получаем содержимое страниц сайта в своем браузере? Как их готовят разработчики? В каком виде они хранятся? Ответы на все эти вопросы мы попробуем получить в этой главе.

Если вы пользуетесь Интернетом, то наверняка слышали аббревиатуру HTML, которая расшифровывается как HyperText Markup Language. Это язык, который применяется для создания Web-страниц. Как видно из его названия, он не является языком программирования. Это язык разметки документов, то есть определенный набор инструкций, которые вставляются в текст документа и указывают на то, каким образом необходимо отобразить тот или иной объект этого документа. Сами эти инструкции называются тегами. Нет нужды знать их все наизусть, но основные теги мы рассмотрим в первой главе книги. С помощью этих тегов производится форматирование текста, вставка рисунков, ссылок и активных элементов, оформляются мультимедийные части документов.

Самое главное слово в названии языка — Hypertext. То есть язык разработан специально для осуществления ассоциативных связей. К любому понятию и любому слову документа мы можем привязать ссылку на другой документ, который может разъяснять понятие или слово, толковать и расширять его. Сама по себе эта технология не являлась каким-либо значительным прорывом. В программном заявлении консорциума по WWW было сказано приблизительно следующее: "Гипертекст не делает чего-то такого, что нельзя было сделать без него. Он просто позволяет делать это быстрее". А потом из этой "непрорывной" технологии вырос Интернет, который стал, пожалуй, самым значительным явлением века. Его истинная ценность не в каких-либо технологиях или удобствах. Интернет является средством общения для множества людей. И чем больше людей подключается к этой всеобщей сети, тем выше становится ее ценностью. Боб Меткалф, один из

основателей фирмы 3Com, описывал это явление следующим образом: "Зачем был факс первому покупателю факсовой машинки? Кому он мог пересылать сообщения? А сегодня каждый покупатель факса покупает вместе с ним целую факсовую сеть. Следовательно, ценность (но не цена) каждого факса становится больше."

Итак, как мы уже знаем, сама Web-страничка хранится в виде текста, "разбавленного" тегами HTML. Когда мы соединяемся со своим провайдером и набираем адрес страницы или сайта, а точнее их URL (Universal Resource Location), то сервер провайдера переводит буквенный URL в цифровой адрес (IP-address), соединяется с сервером, имеющим этот адрес, и на его локальном диске находит файл, содержащий искомую страничку. После этого содержимое найденного файла пересылается на ваш компьютер как обычный текстовый файл с помощью протокола HTTP (HyperText Transfer Protocol), а браузер, запущенный на локальной машине, получает этот файл, обрабатывает его, находя в тексте теги HTML, формирует изображение странички и отображает результат. Щелчок кнопкой мыши на любой из гиперссылок этой страницы заставит весь процесс повторяться с самого начала.

Давным-давно, во время становления World Wide Web, странички и сайты делались при помощи обычных текстовых редакторов. Так можно делать и сейчас. Но не стоит. Писать HTML-код, потом загружать страницу в браузер, просматривать ее, находить ошибки, снова возвращаться в редактор, исправлять ошибки. И так раз за разом. Ужасно непроизводительно. Поэтому вскоре начали появляться специализированные HTML-редакторы и редакторы для проектирования Web-страниц в режиме WYSIWYG (What You See Is What You Get). Они позволили автоматизировать процесс создания информационных ресурсов Интернета. В первой главе мы рассмотрим один из таких редакторов — Microsoft FrontPage 2000.

Помимо страничек и сайтов, которые передаются при помощи протокола HTTP, есть и другие виды ресурсов, которые также интегрированы в Интернет. То есть, World Wide Web это не весь Интернет, как это часто ошибочно полагают, а лишь его часть. Есть еще службы для копирования файлов, которые используют протокол FTP (File Transfer Protocol) и, конечно, нельзя забывать про электронную почту, которая обычно работает при помощи протоколов POP3 и SMTP. Прелесть всей работы сейчас заключается в том, что нам не надо вникать в тонкости их реализации, так как они безболезненно интегрируются в возможности стандартных браузеров, а значит, и подключение подобных ресурсов к создаваемым страницам и сайтам не будет представлять больших трудностей.

Однако, перед тем как применять эти возможности, следует научиться создавать обычные Web-странички.

Знакомство с FrontPage

Слово FrontPage обычно используется для обозначения основной, заглавной страницы какого-либо сайта. Для нас это слово будет иметь еще один смысл. Программа, которой мы будем пользоваться для создания своих первых страниц, носит именно такое название — Microsoft FrontPage 2000. Полный номер версии — 4.0. FrontPage входит в состав Microsoft Office и по своему стилевому оформлению полностью соответствует его стандартам. Данная программа достаточно проста в работе, но несмотря на это позволяет делать очень и очень многое.

Стандартное окно FrontPage 2000 показано на рис. 1.1.

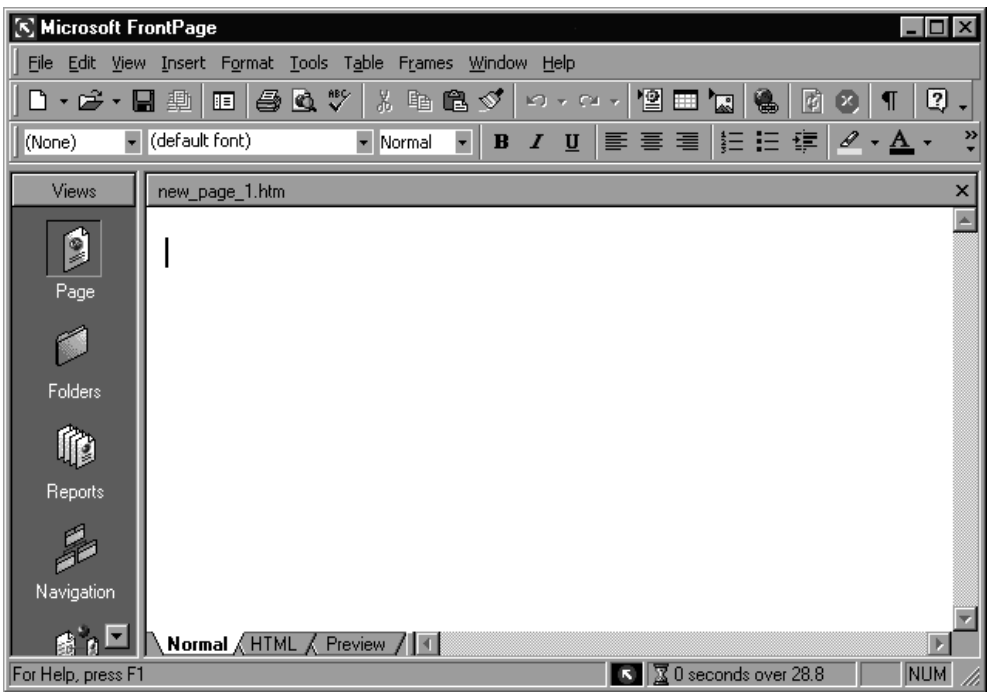

Рис. 1.1. Внешний вид приложения FrontPage 2000

Посмотрим, какими средствами для работы мы располагаем. Как и во всех продуктах Microsoft Office, здесь присутствуют меню и панели инструментов. Особых сложностей их освоение вызвать не должно. Основное рабочее поле разделено подвижной границей (сплиттером) на две неравные части. Левая часть с названием **Views** содержит кнопки, устанавливающие режим работы FrontPage. Правая часть содержит основную информацию, с которой приходится работать, находясь в указанном режиме. По умолчанию установлен режим **Page**, который предназначен для проектирования страницы. В этом режиме правая часть отображает ту страницу, которую создает пользователь. Как

видно на рисунке, правая часть рабочего пространства состоит из трех листов (вкладок), каждый из которых предназначен для различных режимов отображения проектируемой страницы. Лист **Normal** показывает страницу в режиме проектирования, **HTML** предназначен для просмотра HTML-кода страницы, а вкладка **Preview** показывает проектируемую страницу в том виде, в каком она будет отображаться браузером при загрузке ее удаленным пользователем. Может показаться, что страницы **Normal** и **Preview** показывают одно и то же, но на самом деле это не всегда так. Для простых страничек, содержащих только текст и графические изображения они действительно будут одинаковыми, но как только мы попробуем разместить на нашей странице видеофрагменты или какие-либо активные элементы, мы тут же заметим разницу между этими двумя режимами.

В самом низу окна **Microsoft FrontPage** находится строка статуса. На первый взгляд в ней нет ничего особенного. На самом деле строка статуса содержит очень интересную и необходимую для сайтостроителя информацию. В разделе с изображением песочных часов показывается время загрузки текущей страницы при определенной скорости связи. По умолчанию используется скорость 28,8 Кбод, однако всегда есть возможность изменить ее. Для этого достаточно щелкнуть мышью (причем, что интересно, любой кнопкой) и получить список различных скоростей. Помимо стандартных скоростей 14,4, 28,8 и 56,6, там можно найти возможность расчета скорости загрузки исходя из параметров ISDN, а также стандартов T1 и T3 [1].

Как видно, при первом знакомстве FrontPage не отпугивает пользователя своей сложностью. Конечно, при работе с ним будут определенные тонкости, но мы их обязательно рассмотрим. Парадигма же работы в FrontPage ничем не отличается от работы в обычном MS Word. Разместите текст и элементы оформления на странице так, как вы хотите, а для внесения изменений выделите объект и примените к нему необходимое действие.

FrontPage позволяет применять к тексту практически любое шрифтовое и стилевое оформление. Ограничения на формат текста накладывает не программа, а сама технология. К примеру, мы не можем жестко указывать шрифт текста. Связано это с тем, что никогда нельзя точно предсказать, какие шрифты установлены на машине удаленного пользователя. Более того, неизвестно, какую операционную систему он использует и какое разрешение экрана у него установлено. Поэтому вместо точного указания наименования и размера шрифта указывается семейство шрифтов и относительный размер. Более подробно о стилевом и шрифтовом оформлении страницы мы будем говорить в следующем разделе.

Для начала попробуем сделать самую простую страницу. Разместим на рабочем поле проектирования страницы любую строку текста. Например,

[1] Стандарты ISDN, T1 и T3 применяются для выделенных линий с различной пропускной способностью.

"Это моя первая Web-страница!". При этом, если мы посмотрим, что появится на вкладке **HTML**, то обнаружим там следующее:

```
<html>
<head>
<meta http-equiv="Content-Language" content="ru">
<meta http-equiv="Content-Type" content="text/html; charset=windows-
1251">
<meta name="GENERATOR" content="Microsoft FrontPage 4.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<title>New Page 1</title>
</head>
<body>
<p>Это моя первая Web-страница!</p>
</body>
</html>
```

На первый взгляд, этот текст может показаться достаточно непонятным. Действительно, нашу строку текста еще можно опознать, но вот что там делает вся остальная абракадабра? Нет, это не абракадабра, а те самые теги HTML, которые мы упоминали немного ранее. Давайте разберемся, для чего они предназначены.

Прежде всего, необходимо отметить, что большинство тегов, ответственных за правильное отображение страницы или за надлежащее оформление ее содержания являются парными. Они бывают объявляющими и закрывающими. То есть первый тег объявляет какое-либо оформление или раздел страницы, а закрывающий отменяет, заканчивает его действие. Каждый тег HTML заключен в угловые скобки, а закрывающие теги еще имеют в начале своего имени обратный слеш.

Так, например, каждый HTML-документ должен начинаться с тега `<html>`, а заканчиваться — `</html>`, что мы и видим в нашем примере. Сам документ обычно состоит из трех частей. Первая — заголовок, несущий служебную информацию, которая позволяет браузеру максимально адекватно отображать документ. Эта часть ограничивается тегами `<head>` и `</head>`. Внутри нее помещаются теги `<meta>`, содержащие в своем теле параметры. Например, тег

```
<meta http-equiv="Content-Language" content="ru">
```

содержит параметры `http-equiv` и `content`. Давайте подробнее рассмотрим параметры этого тега. Параметр `http-equiv` предназначен для обозначения переменной протокола HTTP, которой будет назначаться некоторое значение. Обычно HTML-редакторы сами заполняют значения этого параметра. В нашем примере эта переменная носит наименование `Content-Language` и предназначается для указания языка, на котором написано содержимое странички. Параметр `content` предназначается для задания значения этой переменной. Таким образом, первая строка блока заголовка указывает, что наша страничка содержит русскоязычное наполнение.

Параметр `content` может использоваться в паре с параметром URL, и тогда они будут определять дату и время повторной загрузки документа. Так, например, если мы хотим спустя пять секунд после загрузки удаленным пользователем нашей странички принудительно загрузить в его браузер HTML-документ с URL `http://www.myserver.ru/newpage.html`, то мы должны использовать следующую конструкцию:

```
<meta http-equiv="Refresh" content="5;
url = http://www.myserver.ru/newpage.html>
```

Параметр `name` применяется для задания наименования дополнительной информации, помогающей браузеру более адекватно отображать Web-страницу. Так, в теге `<meta name="GENERATOR" content="Microsoft FrontPage 4.0">` он применяется для того, чтобы указать имя параметра, чье значение объявляется в конструкции `content`. В этой строке указывается, что документ был сгенерирован программой FrontPage версии 4.0.

И последний параметр тега `<meta>` — `charset`. Он предназначен для определения кодовой страницы символов, которая использовалась при создании страницы. Так как FrontPage естественным образом ориентирован на операционную среду Microsoft Windows, то и используемая по умолчанию кодовая страница — windows-1251.

Второй раздел HTML-документа предназначен для задания названия странички. То есть, той строки, которая будет отображаться в заголовке окна браузера. Этот раздел ограничен тегами `<title>` и `</title>`. Как видно из нашего HTML-кода для нашей странички использован заголовок `New Page 1`. Не слишком оригинально, на мой взгляд, но мы всегда можем его изменить.

Последний, основной раздел документа, заключен между тегами `<body>` и `</body>` и используется для организации содержимого Web-страницы. Сама строка, как мы видим, обрамлена тегами `<p>` и `</p>`, которые предназначаются для выделения абзацев.

Вот так и состоялось наше первое знакомство с языком HTML.

После создания любой странички, ее необходимо сохранить, а потом просмотреть, чтобы убедиться в адекватном отображении ее браузером. Для сохранения используется команда меню **File/Save** и **File/Save As**, либо соответствующая кнопка на инструментальной панели. Обычно в начале работы FrontPage создает необходимую структуру каталогов в папке **My Webs**, которая в свою очередь находится в каталоге, зарезервированном для сохранения документов. В эту структуру входят каталоги **images**, предназначенный для хранения графических изображений, и **_private**, в котором размещаются данные, выкладываемые на создаваемый сайт. После сохранения страницы, основное рабочее поле делится сплиттером на две части, в одной из которых остается проектируемая страница, а вторая показывает структуру папок, предназначенных для сохранения сайта. Этот режим отображения включается и выключается при помощи команды меню **View/Folder List** или при

помощи кнопки **Folder List** на основной инструментальной панели. Сохранение страницы является необходимым условием для возможности ее просмотра как при помощи встроенного браузера на странице **Preview**, так и в основном браузере системы. В том случае, если в оформлении страницы были использованы какие-либо активные элементы, необходимо подготовить сайт к просмотру. Этот процесс называется "публикацией". Для этих целей используется команда меню **File/Publish Web** или соответствующая кнопка на инструментальной панели. Публикацию можно производить только после того, как были сохранены все страницы, которые входят в создаваемый сайт.

После сохранения странички необходимо увидеть, как она будет выглядеть в окне браузера удаленного пользователя. Для этого применяется команда меню **File/Preview in Browser** или соответствующая кнопка на основной инструментальной панели. Использование этой команды позволяет просмотреть созданную страницу при помощи браузера, являющегося для системы основным, используемым по умолчанию. Возникает законный вопрос, для чего это нужно, если уже есть страница **Preview** в основном рабочем окне. Дело в том, что разные браузеры могут по-разному отображать одну и ту же страницу. Связано это с тем, что во время "браузерных войн" между фирмами Microsoft и Netscape предлагались различные расширения и усовершенствования языка HTML. Естественно, что продвинутые возможности, которые предлагала одна фирма, не поддерживались второй. По наследству эти различия перешли и в последующие версии. Вторая причина заключена в том, что встроенный браузер FrontPage не всегда может адекватно работать с исполняемыми модулями CGI-приложений (CGI – Common Gateway Interface). Именно поэтому при заключительном тестировании созданной страницы настоятельно рекомендуется использовать просмотр при помощи основного браузера.

Ну что ж, мы рассмотрели основные принципы работы с программой Microsoft FrontPage 2000. Пора присмотреться к ней повнимательней, и понять, что мы можем сделать с ее помощью, а главное, как это сделать.

Оформление текста

Что главное в Web-странице? Не дизайн, не графика, и даже не видеовставки или активные элементы. Главное — это ее содержание, которое состоит, прежде всего, из текста. А вот оформление текста позволит наиболее эффективно донести его до потребителя, за чье внимание борется каждый сайт. Конечно, возможности текстового оформления, предоставляемые FrontPage, не так обширны, как, скажем, запас средств Microsoft Word, но нельзя сказать, что они недостаточны. Как мы уже говорили, ограничения на оформление текста налагает сама технология WWW, а из допустимых возможностей FrontPage предоставляет максимальные.

Панель инструментов для форматирования текста практически полностью повторяет такую же панель Microsoft Word. Но отличия все-таки есть. Начнем с установки шрифта. По умолчанию действует набор **default font**, который может быть изменен. Список шрифтов, которые можно применять, конечно же немного меньше, чем такой же список в других приложениях Microsoft Office, но выбор все-таки есть. Давайте для нашей строки установим какой-либо шрифт из группы Arial. При этом мы увидим, что изменился шрифт строки на вкладке **Normal**, а когда мы перейдем на вкладку **HTML**, то увидим следующую конструкцию:

```
<p><font face="Arial">Это моя первая Web-страница!</font></p>
```

После тега `<p>`, открывающего абзац, добавился тег `<font face="Arial">`. Теперь при отображении этого абзаца на компьютере удаленного пользователя система будет извещена, что необходимо использовать шрифт, максимально близкий по своему виду к шрифту Arial. То есть, замысел дизайнера Web-страницы пострадает не слишком сильно, даже если сам шрифт Arial не будет установлен в системе удаленного пользователя.

Помимо самого шрифта, мы можем выбрать и его начертание. То есть, сделать его полужирным, курсивом, подчеркнутым или выбрать любую комбинацию этих признаков. Для этого обычно используются три всем знакомые кнопки, находящиеся сразу после списка выбора размера шрифта. Итак, если мы для нашей строки установим полужирное начертание шрифта нажатием кнопки **Bold**, то при просмотре кода на вкладке **HTML** наша строка будет записана как `<p><b>Это моя первая Web-страница!</b></p>`. То есть, для установки полужирного атрибута текста используются теги `<b>` и `</b>`, которые устанавливаются внутри тегов, обозначающих абзац. Если мы нажмем кнопку **Italic** для получения курсива, то HTML-код будет выглядеть как `<p><i>Это моя первая Web-страница!</i></p>`. А для получения подчеркнутого текста мы должны нажать кнопку **Underline** и при этом строка реализуется с помощью конструкции `<p><u>Это моя первая Web-страница!</u></p>`. Как видим, для курсива используются теги `<i>` и `</i>`, а для подчеркивания — теги `<u>` и `</u>`.

Теперь рассмотрим процедуру выбора размера шрифта. По умолчанию в списке размеров шрифта стоит значение **Normal**. Так как никогда нельзя заранее сказать, какое разрешение монитора будет установлено на машине удаленного пользователя, и каков будет размер окна браузера, то и размеры шрифта будут лишь относительными. К тому же, привычные нам всем пункты, которые применяются в качестве единицы измерения размера шрифта, являются типографскими единицами, то есть ориентированы, прежде всего, на бумагу, а Web-страницы отображаются на мониторах, и поэтому размер шрифта будет зависеть от разрешения видеосистемы пользователя и размера окна браузера. Итак, мы можем использовать семь относительных размеров шрифта. Первый размер приблизительно соответствует восьми типографским пунктам, а седьмой — тридцати шести. Значение по умолчанию **Normal** соответствует третьему размеру, который составляет

приблизительно двенадцать пунктов. Таким образом, мы имеем диапазон размеров, достаточный, на мой взгляд, для оформления Web-страницы.

Две вышеописанные возможности объединяет в себе процедура стилевого оформления текста. В FrontPage 2000 список доступных стилей находится на панели форматирования текста в том же самом месте, где и его близнец в Microsoft Word. Но сама процедура стилевого оформления немного отличается. Word в качестве стиля использует совокупность признаков текста, таких как шрифт, его начертание и размер, межстрочный интервал, отступы абзаца и тому подобные параметры. FrontPage скован рамками технологии WWW, поэтому форматирование текста в нем опирается на теги, которые объявлены в стандартах языка HTML и поддерживаются браузерами. Правда, возможно, не всеми из них. Это, как мы помним, последствия браузерных войн. Итак, каждый образец форматирования текста объявляется конкретным тегом языка HTML. Рассмотрим примеры, а заодно узнаем, как выглядят эти теги для каждого конкретного случая.

Стиль **Normal** предполагает обычный текст, и он является единственным образцом стилевого оформления текста, который не устанавливает тегов перед текстом. Стиль **Formatted** предназначен для размещения предварительно отформатированного текста. В данном контексте это означает, что текст будет отображен моноширинным шрифтом, то есть таким шрифтом, в котором все символы имеют одинаковую ширину. Обычно, в качестве примера подобного шрифта приводится Courier. Если мы применим этот стиль к нашей строке, то на странице **HTML** мы увидим следующую конструкцию:

```
<pre>Это моя первая Web-страница!</pre>
```

Как видно, текст обрамляется тегами `<pre>` и `</pre>`. На самом деле, этот тег изначально предназначался для вставки текста, подготовленного в другом текстовом редакторе. Причем, неизвестно заранее, какая длина строки была установлена в этом текстовом редакторе, поэтому, чтобы не нарушать оформление текста, у тега `<pre>` был введен параметр `width`, который указывает длину строки. Таким образом, тег `<pre width=60>` указывает, что для любого размера окна браузера при просмотре этого текста необходимо установить длину строки в 60 символов. По умолчанию значение этого параметра равно 80.

Стиль **Address** используется, как легко догадаться, для оформления адресов. Он переводит строку, напечатанную обычным шрифтом, в отображение курсивом. Применение этого стиля приводит к следующему HTML-коду:

```
<address>
Это моя первая Web-страница!
</address>
```

Как мы видим, отображение адреса на Web-страницах производится при помощи тегов `<address>` и `</address>`. Все просто и легко.

Следующие 6 стилей с названиями от **Heading1** до **Heading6** предназначены для отображения заголовков различных уровней. Самый главный, естественно,

стиль **Heading1**, который и отображается наиболее массивным шрифтом с размером 6, который соответствует приблизительно 24 пунктам, а стиль **Heading6** отображается шрифтом с размером 1, то есть всего-навсего 8 пунктов. При применении стиля **Heading1** к нашей строке, мы увидим, что она будет обрамлена тегами `<h1>` и `</h1>`. На вкладке HTML это будет выглядеть как `<h1>Это моя первая Web-страница!</h1>`. Если мы используем стиль **Heading2**, то теги будут иметь вид `<h2>` и `</h2>`. Таким образом, мы видим, что номер заголовка записывается в теге после буквы h.

В том случае, если мы намерены изменить параметры стиля, и указать для него конкретный шрифт и размер, вместо его стандартных параметров, которые используются, когда в списках шрифта и размера стоят значения **default font** и **Normal** соответственно, это не так сложно сделать. Например, если мы к нашей строке применим стиль заголовка третьего уровня, а затем принудительно установим шрифт Arial и размер 3, то HTML-код для отображения нашей строки будет выглядет так:

```
<h3><font face="Arial" size="3">Это моя первая Web-страница! </font>
</h3>
```

Как видно, сразу после тега `<h3>` вставляется тег `<font>` с параметрами `face` и `size`, которые непосредственно задают шрифт для отображения строки. И в конце предложения стоит закрывающий тег `</font>`.

Следующие шесть стилей предназначены для оформления текста в виде нумерованных и маркированных списков. Собственно, нумерованный список там только один, а остальные пять стилей являются вариациями маркированного списка.

Для иллюстрации действия этих стилей создадим два абзаца со строками "Первый пункт" и "Второй пункт", а затем применим к этим двум абзацам стиль **Numbered List**. Как мы видим, в результате этого действия каждый абзац получил свой номер. То есть произошла стандартная операция нумерации абзацев, знакомая нам еще по Microsoft Word. Этого же результата можно добиться, нажав кнопку **Numbering** на панели инструментов **Formatting**. Рассмотрим теги, при помощи которых реализуется подобное форматирование текста. Если мы посмотрим на вкладку **HTML**, то увидим следующий текст:

```
<ol>
  <li>Первый пункт</li>
  <li>Второй пункт</li>
</ol>
```

Как мы видим, начало и конец нумерованного списка задаются с помощью тегов `<ol>` и `</ol>`, а обозначение каждого пункта производится тегами `<li>` и `</li>`. Все не так уж и сложно.

Хотелось бы также узнать, каким образом мы можем изменить вид нумерации и начальный номер. Для этого используется пункт меню **Format/Bullets and Numbering**. В том случае, если у нас на странице курсор находится

в строке, принадлежащей нумерованному списку, активизируется диалоговое окно **List Properties** на вкладке **Numbers** (рис. 1.2). По умолчанию выбран вариант с арабскими цифрами. Выбор варианта списка без нумерации приведет к полному снятию стиля **Numbered List** с нашего списка, и, соответственно, незамедлительное его превращение в самый обычный текст со стилем **Normal**. В случае выбора списка с нумерацией в виде римских цифр, отображаемых при помощи заглавных букв латинского алфавита, наш список будет открыт тегом `<ol type="I">` и закрыт тегом `</ol>`. Других изменений нет. Становится видно, что вид нумерации задается параметром `type` тега `<ol>`. Например, для создания списка, в котором нумерация осуществляется при помощи заглавных букв латинского алфавита (A, B, C и т. д.), используется параметр `type="A"`. Если это должны быть обычные строчные буквы, то параметр принимает вид `type="a"`. И, наконец, если нумерация списка осуществляется римскими цифрами, отображаемыми строчными латинскими буквами, параметр указывается как `type="i"`.

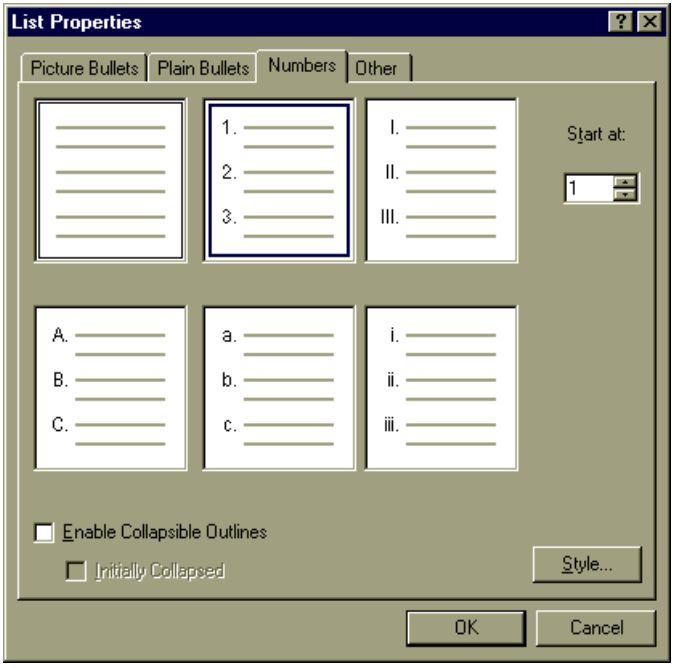

Рис. 1.2. Вкладка **Numbers** диалогового окна **List Properties**

Еще один параметр списка — это его стартовый номер, то есть число, с которого начинается нумерация. Оно устанавливается в поле **Start at** окна **List Properties**. При этом в тег `<ol>` добавляется параметр `start`. То есть, если мы хотим сделать список, нумерованный римскими цифрами из строчных букв, начинающийся с номера 5, то для этого будет использован тег `<ol type="i" start="5">`.

Теперь рассмотрим маркированные списки. Для их создания наиболее часто применяется стиль **Bulleted List** или кнопка **Bullets** на панели инструментов **Formatting**. При применении этого стиля к нашему тексту мы увидим, что для его реализации используется следующая конструкция HTML:

```
<ul>
  <li>Первый пункт</li>
  <li>Второй пункт</li>
</ul>
```

Как и в предыдущем примере, здесь есть теги, объявляющие сам список, то есть `<ul>` и `</ul>`, а пункты списка оформляются при помощи открывающего тега `<li>` и его закрывающего дополнения `</li>`. Так же, как и нумерованный, этот вариант списка может быть оформлен несколькими видами маркеров. Их вид выбирается в диалоговом окне **List Properties** на вкладке **Plain Bullets**, которое активируется при выборе команды меню **Format/Bullets and Numbering**. Но вариантов оформления там меньше, чем в нумерованном списке. Если не считать пустое оформление, остается всего три. Маркеры в виде точек устанавливаются по умолчанию. Мы можем изменить вид маркеров и оформить их в виде окружностей. Тогда тег, объявляющий начало маркированного списка, будет записан как `<ul type="circle">`. А если мы захотим увидеть маркеры в виде квадратов, то HTML-код, реализующий эту прихоть, будет выглядеть как `<ul type="square">`. Впрочем, для отображения маркеров мы можем использовать не только эти три зарезервированные символа, но и практически любое графическое изображение. Для этого все в том же диалоговом окне **List Properties** необходимо выбрать вкладку **Picture Bullets**, которая предназначена для установки внешнего вида маркеров. Проектировщику Web-страницы предоставляется на выбор две альтернативы (рис. 1.3). По умолчанию действует пункт **Use pictures from current theme**. Он указывает на то, что при оформлении страницы будут использоваться изображения маркеров, которые применяются в так называемой *теме* сайта, то есть наборе графических изображений и правил оформления текстов, которые будут употребляться для единообразного оформления всех страниц, входящих в состав сайта. Однако, если у создателя страницы есть свои соображения по поводу формы маркеров, то он должен выбрать **Specify picture**, и указать путь к графическому файлу, который он хочет использовать, при помощи кнопки **Browse**. При этом активизируется стандартное диалоговое окно (рис. 1.4).

Естественно, рекомендуется выбирать изображения из числа тех, которые находятся в пределах структуры папок, предназначенных для хранения файлов создаваемого сайта. Однако проектировщик Web-страницы может выбрать необходимое изображение тремя различными способами. Первый — наиболее привычный для всех нас выбор файла из локальной файловой системы. Это можно сделать либо простым указанием файла в структуре

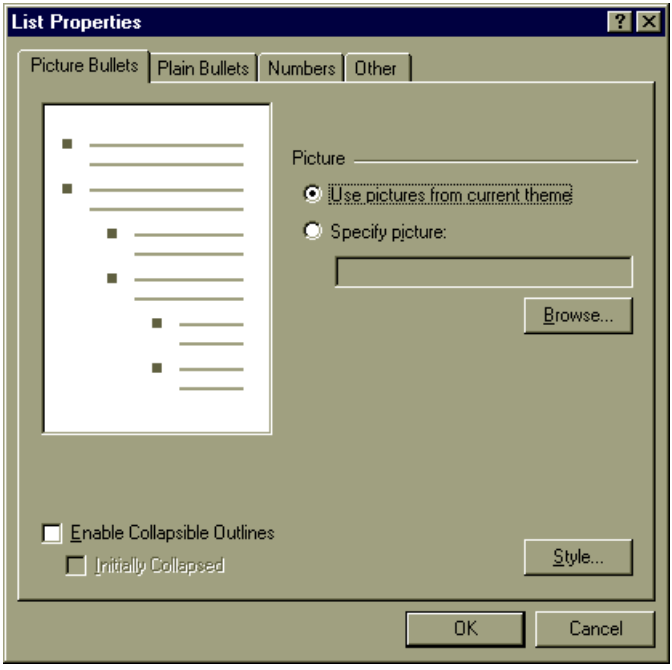

Рис. 1.3. Вкладка **Picture Bullets** диалогового окна **List Properties**

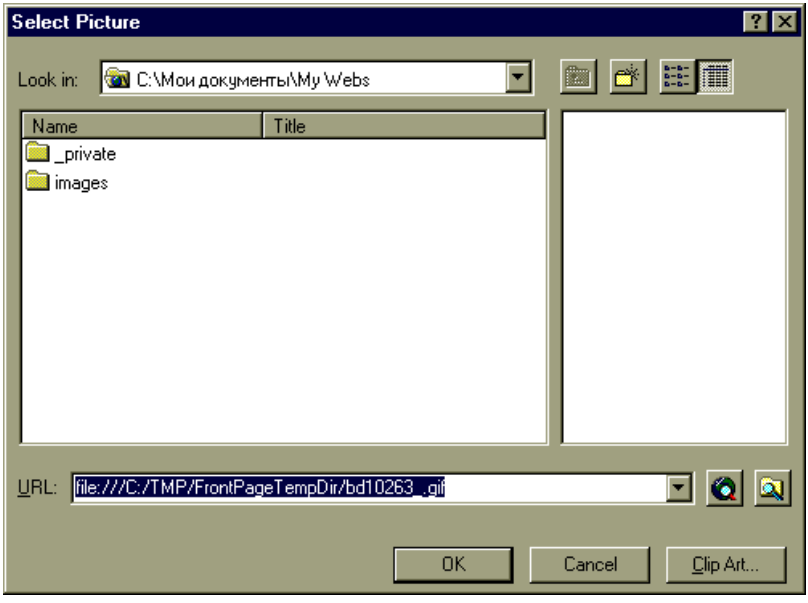

Рис. 1.4. Диалоговое окно **Select Picture**

папок сайта или, если файл находится за ее пределами, нажатием кнопки с изображением папки и лупы, которая позволит просмотреть любую папку

на локальном компьютере. Второй вариант связан с использованием изображения, которое находится на удаленном компьютере. Нажатие кнопки с изображением земного шара и лупы запускает браузер, установленный в системе, и пользователь может выбрать любой графический элемент, который он найдет в Интернете. При этом будет вставлен не сам этот элемент, а ссылка на него. Каждый раз при загрузке нашей страницы браузер будет вынужден обращаться по указанному адресу и скачивать оттуда изображение. Вопрос о целесообразности использования этого варианта целиком и полностью остается в ведении проектировщика. Третий вариант — выбор изображения из коллекции Clip Art, поставляемой вместе с Microsoft Office. Для этого необходимо нажать кнопку **Clip Art**. При этом файл с выбранным вами изображением будет перенесен во временную папку Microsoft FrontPage 2000. В этом случае тег, ответственный за объявление маркированного списка, получит дополнительный параметр и примет вид:

```
<ul imagesrc="file:///C:/TMP/FrontPageTempDir/bd10263_.gif">
```

Как мы видим, в параметре `imagesrc` все равно указывается URL файла с графическим изображением, но вместо протокола HTTP используется протокол file, который предназначен для работы с файловой системой локального компьютера.

Помимо обычного маркированного списка, FrontPage предоставляет также несколько других видов списков. Они не отличаются оформлением от обычного списка. Различие кроется в тегах, реализующих их. Стиль **Directory List** задается при помощи следующего HTML-кода:

```
<dir>
  <li>Первый пункт</li>
  <li>Второй пункт</li>
</dir>
```

Стиль **Menu List** открывается и закрывается при помощи тегов `<menu>` и `</menu>` соответственно. Элементы этого стиля отображаются абсолютно идентично элементам обычного маркированного списка. Наличие нескольких стилей с одинаковым отображением по умолчанию позволяет при помощи каскадных таблиц стилей (CSS) создавать различные варианты отображения элементов ненумерованных списков.

Для списка, который состоит из определений, предназначен стиль **Definition List**. Данным вариантом оформления часто пользуются в академических и обучающих материалах, когда необходимо в списке поместить несколько определений. В случае его применения меняются не только открывающий и закрывающий теги, но и теги, обрамляющие каждый отдельный пункт списка. Таким образом, получается следующий HTML-код:

```
<dl>
  <dd>Первый пункт</dd>
  <dd>Второй пункт</dd>
</dl>
```

При этом, каждый пункт приобретает стиль **Definition**. Его признаки, как мы видим, это теги `<dd>` и `</dd>`.

Задание шрифта, его начертания, размера и стилевого оформления текста может быть произведено при помощи диалогового окна **Font**, которое активизируется при выборе команды меню **Format/Font**.

Для оформления текста также применяются различные выключки и отступы. *Выключкой* или *выравниванием* называется расположение текста относительно горизонтальной базовой линии страницы. Применяется четыре вида выравнивания текста: текст, прижатый к левому краю, отцентрированный текст, текст, прижатый к правому краю, и текст, растянутый на всю ширину страницы. В FrontPage 2000 на панели инструментов **Formatting** доступны кнопки **Align Left**, **Center** и **Align Right**, которые реализуют первые три варианта выравнивания текста. Текст, прижатый к левому краю, является стандартным вариантом расположения, но может быть задан и явно. Выравнивание задается при помощи параметра тега, объявляющего абзац `<p>`. Так, для явного объявления выключки текста по левому краю применяется тег `<p align="left">`. В случае отцентрированного текста используется конструкция `<p align="center">`. И, как нетрудно догадаться, прижатый к правому краю текст объявляется при помощи тега `<p align="right">`. Из приведенных примеров видно, что выключка текста задается применительно к целому абзацу, и действие тегов выравнивания текста прекращается стандартным тегом окончания абзаца `</p>`. Для равномерного растяжения текста по всей ширине страницы специальной кнопки нет. Чтобы его использовать, необходимо выполнить команду **Format/Paragraph**, которая активизирует диалоговое окно **Paragraph** (рис. 1.5). Это окно предназначено для задания параметров абзаца, а если говорить конкретнее, для задания отступов и интервалов. Выравнивание текста указывается в списке **Alignment**. Для того чтобы равномерно растянуть текст абзаца на всю длину строки, необходимо выбрать пункт **Justify**. При этом, тег объявления абзаца примет вид `<p align="justify">`. А выбор элемента **Default** в списке **Alignment**, полностью убирает параметр `<p>`, отвечающий за назначение выравнивания из тела тега `<p>`.

Следующий блок органов управления **Indentation** диалогового окна **Paragraph** предназначен для указания отступов текста от краев страницы. Поле ввода **Before text** позволяет указывать отступ абзаца от левого края страницы, а поле **After text** — от правого. Поле ввода **Indent first line** указывает отступ красной строки, то есть, первой строки каждого абзаца. Если мы укажем отступ перед текстом как три единицы, после него — четыре, а для красной строки — пять единиц, то тег объявления абзаца примет вид `<p style="text-indent: 5; margin-left: 3; margin-right: 4">`. То есть, тег `<p>` получает параметр `style`, значением которого будет одна длинная строка, в которой сначала указывается отступ красной строки, а затем отступы всего абзаца слева и справа. По умолчанию в качестве единиц измерения используются миллиметры.

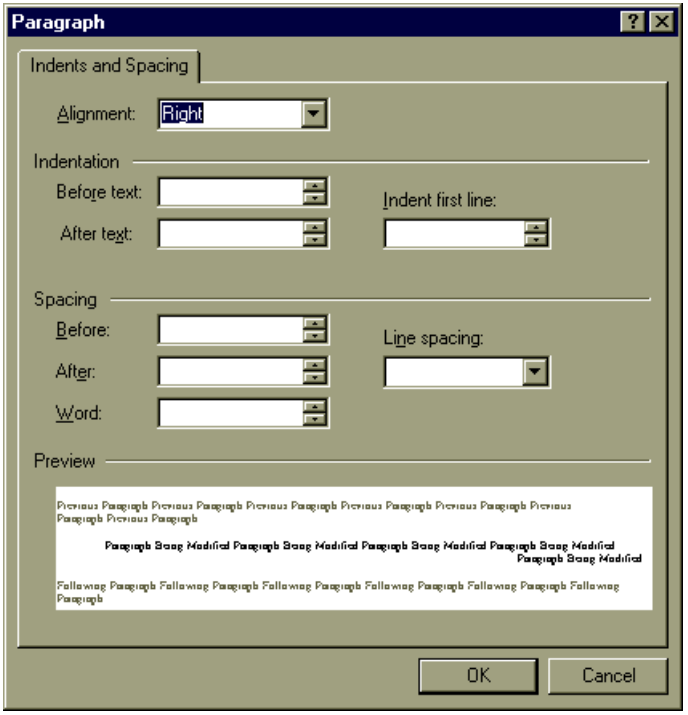

Рис. 1.5. Диалоговое окно **Paragraph**

Отступы текста от края страницы могут быть изменены при помощи кнопок **Decrease Indent** и **Increase Indent**, которые позволяют уменьшать и, соответственно, увеличивать отступ текущей строки. Увеличение отступа текста осуществляется при помощи отдельных тегов. Для уменьшения отступа (который до этого был увеличен) достаточно удалить эти теги. Если для нашей строки отступ не задан явным образом и мы воспользуемся кнопкой **Increase Indent** для его установки, то при просмотре HTML-кода для отображения нашей строки, мы увидим следующую конструкцию:

```
<blockquote>
  <p>Это моя первая Web-страница</p>
</blockquote>
```

При двойном нажатии этой кнопки добавится еще одна пара тегов `<blockquote>` и т. д.

Есть еще один способ позиционирования текста — при помощи клавиши табуляции. Так как для Web-страниц не создано специальных табуляторов, то для отработки табуляции применяется совокупность неразрывных пробелов. При единичном нажатии клавиши табуляции в начале строки, строка немного сдвигается вправо, а HTML-код для нашего абзаца принимает вид `<p>    Это моя первая Web-страница</p>`. Как видно, после тега, объявляющего начало абзаца, вставляется три символьные после-

довательности, которые обозначают неразрывные пробелы. Подобные последовательности применяются для отображения всех нестандартных символов и тех символов, которые зарезервированы для языка HTML, то есть угловых скобок, амперсантов и двойных кавычек.

Последний блок органов управления диалогового окна для установки атрибутов абзаца носит название **Spacing** и предназначен для установки различных интервалов. Поле ввода **Before** позволяет указать интервал перед началом абзаца, а поле **After**, естественно, интервал после него. Необходимо отметить, что если эти параметры применяются к нескольким абзацам, то расстояние между ними будет складываться из отступа **After** одного абзаца и отступа **Before** последующего абзаца. Поле ввода **Word** предназначается для указания величины пробела между отдельными словами, а список **Line spacing** позволяет указывать межстрочный интервал. Для него используется три значения: **Single**, который обозначает обычный отступ, **1.5 lines** — для установки полуторного отступа, и **Double** — для двойного.

Если мы установим значения в этом блоке и применим их к текущему абзацу, нажав кнопку **OK**, то реализация этих условий будет выполнена при помощи тега

```
<p style="word-spacing: 4; line-height: 150%; margin-top: 5; margin-bottom: 5">
```

где `word-spacing` указывает расстояние между словами, `line-height` — межстрочный интервал, а `margin-top` и `margin-bottom` — отступы перед абзацем и после него соответственно.

В этой главе нам осталось рассмотреть только возможности цветового оформления текста. Цвет может быть задан для фона текста и для самого шрифта. Цвет фона задается при помощи кнопки **Highlight Color** с изображением маркера. Применение этой возможности изменяет фон вводимого или заранее выделенного текста, и возникает ощущение, что по нему действительно провели цветным маркером. Если мы для нашей строки зададим желтый фон, то HTML-код для ее реализации будет выглядеть как

```
<p><span style="background-color: #FFFF00">Это моя первая Web-страница!</span></p>
```

Как видно, цвет задается в теге `<span>` и определяется при помощи сочетания шести шестнадцатеричных цифр. Это сочетание задает RGB-код цвета. Первые две цифры показывают насыщенность красного цвета, вторые две — насыщенность зеленого, и, наконец, последние две — насыщенность синего.

Для того чтобы определить цвет самого шрифта, используется кнопка **Font Color**. Если мы назначим для нашей строки светло-зеленый цвет (Lime), то наша строка будет реализована с помощью конструкции

```
<p><font color="#00FF00">Это моя первая Web-страница!</font></p>
```

В этом случае, применяется уже знакомый нам тег `<font>` с параметром `color`, а сам цвет все так же задается шестизначным кодом.

Очень часто отдельные разделы документа отделяются друг от друга при помощи горизонтальных линий. Для вставки горизонтальной линии применяется команда меню **Insert/Horizontal Line**. Линия будет вставлена в то место, где расположен текстовый курсор. Применяемый для этого HTML-тег выглядит как `<hr>`. Вставленная линия является объектом, следовательно, у нее есть свои свойства, которые можно устанавливать и редактировать. Для этого необходимо выделить вставленную горизонтальную линию и выполнить команду **Format/Properties** или команду **Horizontal Line Properties** контекстного меню, вызываемого щелчком правой кнопки мыши. При этом активизируется одноименное диалоговое окно для задания свойств горизонтальной линии. В поле **Width** мы можем указать ширину линии, а затем одним из переключателей указать единицу измерения. Длина линии может указываться как в пикселах, так и в процентах ширины окна просмотра браузера удаленного пользователя. В последнем случае, при изменении пользователем размеров окна просмотра, ширина горизонтальной линии также будет пересчитана. В поле **Height** указывается толщина линии в пикселах. По умолчанию она составляет два пиксела. С помощью группы радиокнопок **Alignment** мы можем указать, как будет выравниваться линия в окне просмотра. Переключатель **Left** прижимает ее к левому краю, **Center** позволяет отцентрировать ее, а радиокнопка **Right** прижимает нашу линию к правому краю окна просмотра. Если установлен переключатель **Solid line (no shading)**, то линия отображается плоской, без трехмерного выделения тенями. Для указания цвета линии используется список цветов **Color**. После установки значений свойств линии, отличных от вариантов по умолчанию, мы увидим, что тег, объявляющий линию, изменился. Он приобрел вид:

```
<hr size="3" width="80%" color="#00FFFF" align="left" noshade>
```

В параметре `size` указывается толщина линии в пикселах, параметр `width` задает длину линии, `color`, как нетрудно догадаться, указывает ее цвет, `align` — выравнивание, а наличие параметра `noshade` убирает трехмерные тени.

Ссылки и графика

Что главное в гипертексте? Главное в нем то, что он — "гипер" и позволяет реализовывать ссылки на другие документы, которые могут помочь при работе с основным текстом. То есть, то, что называется гиперссылками. Это сердцевина и основа, альфа и омега World Wide Web. Гиперссылки разделяются на два типа — внешние и внутренние. Внешние ссылки позволяют удаленному пользователю переходить к другим HTML-документам, а внутренние служат для быстрого передвижения внутри одного документа. Простейшим примером организации внутренних ссылок является оглавление содержимого Web-страницы, в котором каждая строка является ссылкой на начало новой части документа.

Чаще всего ссылки создаются в виде текста, указывающего адрес, куда произойдет переход. Для того чтобы их было можно отличить от основного текста,

браузер выделяет их цветом. Причем цветовое оформление обычной ссылки и ссылки, к которой пользователь уже обращался, как правило, различается.

Бывает, что ссылки оформляются в виде графического изображения. То есть, для активизации ссылки и начала процесса перехода, необходимо щелкнуть мышкой на рисунке. Именно поэтому вопросы создания графики и ссылок мы поместили в одной главе. Сначала мы рассмотрим вопросы создания гиперссылок, а потом перейдем к использованию графических изображений при оформлении Web-страниц.

Итак, приступаем к созданию гиперссылок. Для вставки гиперссылки в текст Web-страницы необходимо выполнить команду меню **Insert/Hyperlink** или нажать кнопку **Hyperlink** с изображением земного шара и звена цепи, которая находится на основной инструментальной панели. При этом активируется диалоговое окно **Create Hyperlink** (рис. 1.6).

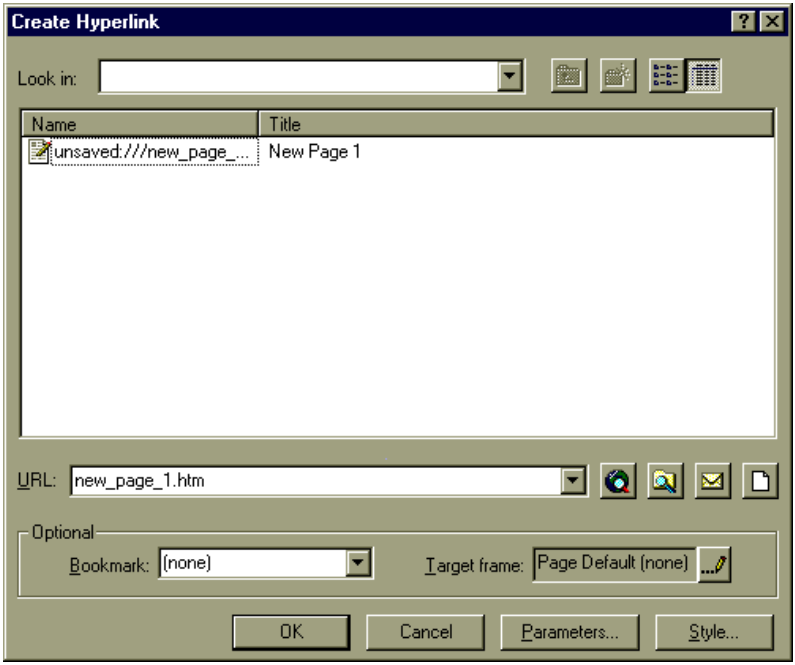

Рис. 1.6. Диалоговое окно **Create Hyperlink**

В том случае, если при создании гиперссылки в макете Web-страницы не было выделенного текста, гиперссылка будет вставлена в виде обычного URL. Если же был выделен текст, то ссылка связывается с этим выделенным текстом и URL не отображается на экране.

Для того чтобы указать URL гиперссылки, то есть адрес точки перехода, его надо написать в поле **URL**. В том случае, если ссылка должна указывать на документ, входящий в состав сайта, его можно выбрать из основного списка

диалогового окна. Для поиска адреса можно воспользоваться дополнительными кнопками, расположенными правее строки ввода URL. Кнопка с изображением земного шара и лупы запускает браузер, установленный в системе по умолчанию. При этом проектировщик страницы может найти необходимый документ обычными средствами Интернета. Обычно при закрытии браузера URL последней просмотренной страницы автоматически подставляется в поле ввода URL. В том случае, если необходимый ресурс находится на локальном компьютере, следует нажать следующую кнопку с изображением папки и лупы, которая активизирует стандартное диалоговое окно для открытия файла. Третья кнопка с изображением конверта предназначена для создания гиперссылки, которая позволяет отсылать электронное письмо. При нажатии на эту кнопку появляется дополнительное диалоговое окно, в единственное поле ввода которого необходимо ввести адрес, по которому письмо будет отправлено. Как мы уже знаем, World Wide Web по большей части основан на протоколе HTTP, а электронная почта реализуется при помощи протоколов POP3 и SMTP. Следовательно, Web-страницы и электронная почта являются разными вещами. Но в том-то и прелесть нынешнего Интернета, что все различные протоколы интегрированы в единое целое. Но как я и обещал, нам не надо вдаваться в тонкости. Мы можем просто творить. Браузеры в чистом виде не приспособлены к передаче электронной почты, поэтому, как только пользователь щелкает мышью на ссылке с адресом электронной почты, запускается та почтовая программа, которая установлена в его системе. Текст письма он набирает самостоятельно, а адрес подставляется тот, который указан в ссылке.

И, наконец, последняя кнопка с изображением чистого листа. Она используется в том случае, когда страницы, на которую устанавливается ссылка, еще нет. Нажатие этой кнопки добавляет к проектируемому сайту чистую страницу, а в текущей странице размещается ссылка на нее.

Выпадающий список **Bookmark** из группы элементов управления **Optional** предназначен для создания внутренних ссылок, которые в FrontPage 2000 реализуются при помощи закладок. Каждая закладка вставляется в необходимом месте текста при помощи команды **Insert/Bookmark**. Так же, как и в Microsoft Word каждая закладка может иметь свое имя. Но нам это не важно. В HTML-документах каждая закладка является маркером, к которому можно привязать гиперссылку. Таким образом, мы видим, что перед тем, как создавать внутренние ссылки, необходимо создать набор закладок в тексте, на которые мы потом получим возможность ссылаться.

И последний рассматриваемый орган управления — поле **Target frame**. Это поле не предназначено для прямого ввода. Чтобы установить в него значение, используется привязанная к этому полю кнопка. Поле **Target frame** задает имя окна, в которое будет загружаться документ. По умолчанию используется параметр **Page Default**, который отрабатывает стандартный вариант отображения последовательности страниц браузером. Параметр **Same Frame** указывает на то, что документ должен быть загружен в то же самое

окно, где расположена ссылка. Параметр **Whole Page** указывает на то, что загружаемый документ займет все окно браузера. Параметр **New Window**, как нетрудно догадаться, принудительно открывает для документа новое окно. И, наконец, параметр **Parent Frame** указывает на то, что документ будет загружен в окно, которое является родительским, по отношению к текущему.

Теперь давайте посмотрим, как механизм ссылок реализуется в HTML. Для этого на пустой странице разместим две строки. Одна будет являться закладкой, а другая будет представлять собой ссылку. Сначала попробуем создать локальную гиперссылку. Для этого, как уже говорилось ранее, мы используем список закладок **Bookmark** диалогового окна **Create Hyperlink**. В том случае, если установка закладки на первую строку прошла гладко, мы увидим ее имя в списке закладок. Достаточно будет щелкнуть мышью на названии закладки и в строке **URL** появится ее адрес. Нажмем кнопку **OK**, и перейдем на страницу **HTML**, чтобы посмотреть HTML-код нашего творения. Он будет выглядеть приблизительно следующим образом:

```
<p><a name="Место для закладки">Место для закладки</a></p>
<p><a href="#Место для закладки">Ссылка на закладку</a></p>
```

Итак, как мы видим, помимо тегов объявления абзаца, в каждой строке появился тег `<a>` с различными параметрами. Именно он применяется для реализации гиперссылок. В первой строке мы объявили закладку с именем "Место для закладки". Именно поэтому в теге `<a>` появился параметр `name`, который создает маркер для внутренних ссылок. После текста закладки выставлен закрывающий тег `</a>`. Во второй строке установлена сама ссылка. В этом случае используется параметр `href`, в качестве значения которого указывается URL необходимого документа. Так как ссылка в данном случае внутренняя, то перед самим URL ставится знак решетки, а сам URL будет просто именем маркера. Причем цветовое оформление гиперссылки будет применено ко всему тексту, который находится между тегом `<a>` и его закрывающей парой `</a>`.

Теперь попробуем создать внешнюю гиперссылку. Для этого выделим вторую строку и нажмем кнопку **Hyperlink** или выберем в контекстном меню, вызываемом правой кнопкой мыши, команду **Hyperlink Properties**. Результатом этих действий будет активизация диалогового окна **Edit Hyperlink**, которое является близнецом уже знакомого нам диалогового окна **Create Hyperlink**. Введем в поле **URL**, скажем, адрес наиболее известного книжного магазина Сети — http://www.amazon.com. А также укажем окно для загрузки основной страницы этого сайта. Выберем в списке **Target Frame** значение **Whole Page**, т. е. эта страница займет все окно просмотра браузера. После нажатия на кнопку **OK** наша гиперссылка будет изменена. Теперь при просмотре HTML-кода мы увидим следующее:

```
<a href="http://www.amazon.com" target="_top">Ссылка на закладку </a>
```

Как мы видим, в параметре `href` теперь указан обычный адрес с указанием протокола. Параметр `target` содержит значение, которое указывает тип окна для загружаемого документа.

Вот так создаются обычные гиперссылки. Перейдем теперь к использованию графики в оформлении Web-страниц.

FrontPage 2000 позволяет использовать три вида графики: стандартные графические файлы, картинки из коллекции ClipArt и видеофрагменты. Да-да, видеофрагмент тоже считается картинкой, и команда его вставки находится в одной группе с другими командами использования графики. Заниматься видеофрагментами мы будем позже, а сейчас сосредоточимся на обычных изображениях.

Вставка графического файла осуществляется при помощи команды меню **Insert/Picture/From File** или соответствующей кнопки на основной инструментальной панели. При этом активизируется диалоговое окно **Picture** (рис. 1.7).

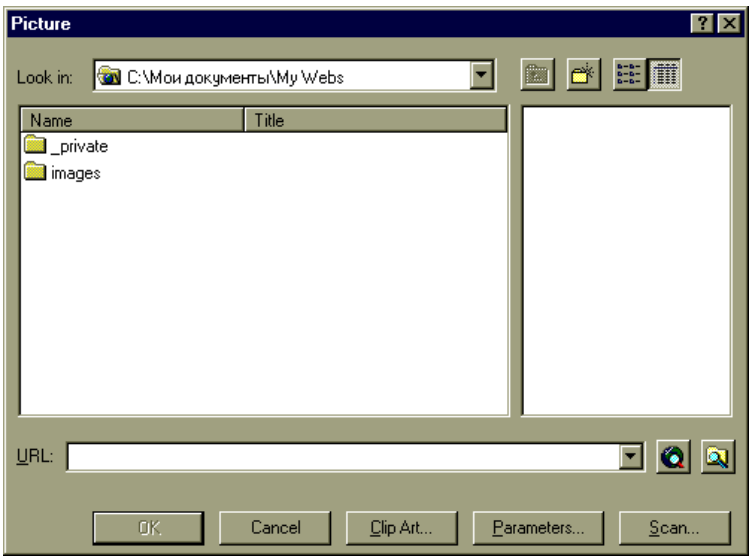

Рис. 1.7. Диалоговое окно **Picture**

Как и при создании гиперссылки, диалоговое окно предназначено для указания источника изображения. Для этого используется не имя файла, а его URL. Конечно, было бы неплохо, если бы файл с изображением находился на той же машине, где установлен Web-сайт, но никаких ограничений относительно его местоположения не существует.

Итак, по умолчанию считается, что файл с графическим изображением находится в структуре каталогов создаваемого сайта, которая показана в основном списке диалогового окна **Picture**. Однако, местонахождение файла будет записано в поле ввода **URL**. Справа от этого поля находятся уже известные нам две кнопки, которые позволяют отыскивать требуемое изображение в необъятных просторах Интернета и в глубинах дискового пространства локальной машины. Помимо этого есть и другие кнопки, которые рас-

положены в самом низу диалогового окна. Кнопки **OK** и **Cancel** не должны вызвать особых затруднений, действия, выполняемые с их помощью, понимаются интуитивно. Кнопка **Clip Art** запускает диалоговое окно, показывающее коллекцию рисунков, поставляемых в составе Microsoft Office, а кнопка **Scan** предназначена для получения графического изображения с цифровой камеры, сканера и тому подобных устройств. Для всех них действует единое правило — они должны поддерживать TWAIN-интерфейс. Итак, при нажатии на кнопку **Scan** появляется дополнительное диалоговое окно. Для начала надо выбрать периферийное устройство, с которого будет получаться изображение, то есть его источник. Естественно, для этого используется кнопка **Source**. Пользователь получает список всех подходящих устройств. После того, как периферийное устройство выбрано, необходимо нажать кнопку **Acquire**, и изображение попадет на проектируемую страничку.

Теперь рассмотрим на примере процесс вставки графики, и HTML-представление этого действия. Для этого выберем какой-нибудь графический файл на своем компьютере и разместим его на создаваемой странице. Изображение сразу проявится на предоставленном ему месте на листе **Normal**. При этом если мы щелкнем мышью на этом рисунке, и тем самым выделим его, то FrontPage 2000 визуализирует еще одну инструментальную панель, которая предназначена для работы с изображениями.

Но сейчас нас интересует то, какие теги применяются в HTML для вставки изображений. Если перейти на вкладку **HTML**, то можно увидеть конструкцию

```
<img border="0" src="file:///C:/WIN98/Hlplogo.gif" width="97" height="40">
```

Как мы видим, любое графическое изображение может быть вставлено при помощи тега `<img>` с несколькими параметрами. Основной и самый главный параметр — `src`, значение которого указывает местонахождение искомого графического файла. В нашем случае это `file:///C:/WIN98/Hlplogo.gif`. Первая часть значения указывает на то, что будет использоваться стандартный протокол доступа к файлам на локальном компьютере. После него указано полное имя файла. В том случае, если необходимый файл находился бы на другом Web-сервере, в адресе вместо префикса `file`, был бы указан префикс `http`.

Параметр `border` предназначен для указания толщины рамки вокруг рисунка. В нашем случае ему присвоено нулевое значение, а значит, рамки как таковой просто не будет. Параметры `width` и `height` указывают ширину и высоту рисунка соответственно.

На самом деле, параметров у тега `<img>` гораздо больше. Но все они появляются лишь в ответ на наши изменения установок рисунка. Тот набор параметров, который мы рассмотрели, при работе с FrontPage 2000 обычно устанавливается по умолчанию.

Давайте рассмотрим те возможности работы с графическими изображениями, которые нам предоставляет FrontPage 2000. Большая часть из них отрабатывается при помощи кнопок на инструментальной панели **Pictures**, ко-

торая визуализируется каждый раз при выделении какого-либо рисунка на странице. Впрочем, добиться появления этой инструментальной панели на экране можно, выполнив команду меню **View/Toolbars**, а затем поставив галочку напротив наименования этой панели. Однако, помимо этих кнопок, каждая из которых совершает определенное действие, есть еще и простые свойства рисунка, которые можно редактировать в диалоговом окне **Picture Properties** (рис. 1.8), которое активизируется одноименной командой из контекстного меню, вызываемого правым щелчком мыши на самом изображении. Основная часть элементов управления окна интуитивно понятна.

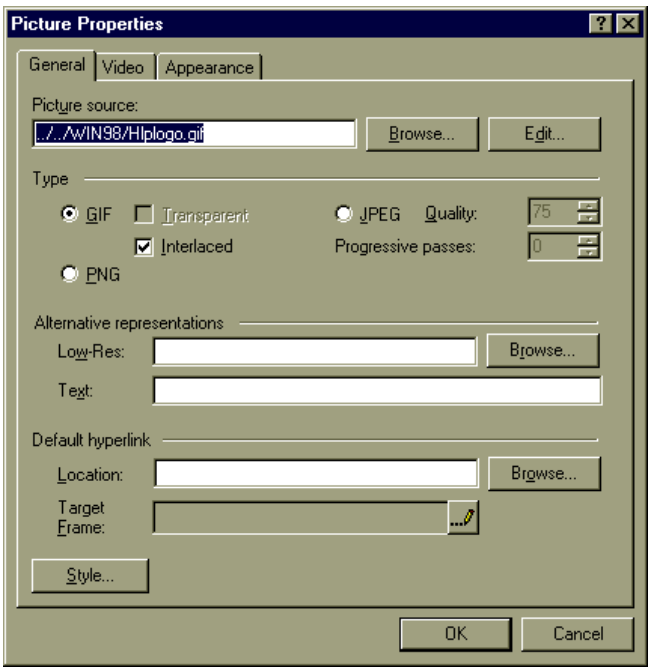

Рис. 1.8. Вкладка **General** диалогового окна **Picture Properties**

Так, в строке **Picture source** указывается местонахождение графического файла, рядом кнопка для просмотра файловой системы **Browse** и кнопка **Edit** для запуска графического редактора, встроенного в FrontPage. Под ними находится группа из трех зависимых переключателей (радиокнопок), которые показывают тип графического файла. Как видно, в WWW поддерживается три графических формата: GIF, JPEG и PNG. Помимо этого у файлов GIF и JPEG есть свои свойства, которые включаются и выключаются посредством флажков. Так, например, если выбран GIF-файл, то можно указать свойство **Interlaced**, которое указывает на то, что этот рисунок поддерживает чересстрочное изображение. Подобный рисунок будет проявляться на экране браузера постепенно, по мере загрузки увеличивая свою четкость. Если быть абсолютно точным, то сначала загружаются строки

изображения с номером, кратным восьми, затем — четырем, потом — двум, а затем уже нечетные строки. Свойство **Transparent** предназначено для "прозрачных" GIF-файлов, то есть таких рисунков, в которых один из цветов установлен как прозрачный. Через детали рисунка с таким цветом будет просвечивать то, что находится под рисунком, на слой ниже. Подобные рисунки довольно часто применяются в дизайне Web-страниц, и умелое их применение весьма эффективно. Для файлов JPEG, сохраняющих изображение в сжатом виде с некоторыми потерями информации, можно задать параметры сжатия и, соответственно, потери качества изображения. Ну, а файлы формата PNG вставляются в Web-страницы как есть, без каких-либо дополнений и опций.

Под этим блоком расположена достаточно интересная группа полей ввода под общим названием **Alternative representations**. Здесь задаются способы альтернативного представления рисунка на тот случай, если по каким-либо причинам он не может быть правильно отображен в браузере удаленного пользователя. В поле **Low-Res** помещается адрес иконки, которая будет подменять основное изображение. Второе поле с названием **Text** позволяет ввести текстовую строку, которая будет отображаться вместо рисунка в тех случаях, когда отображение искомого графического файла невозможно. Раньше это диктовалось тем, что некоторые браузеры просто не были приспособлены для отображения рисунков. Да-да, было и такое! Теперь же этот текст используется, когда в браузере удаленного пользователя выставлен запрет на отображение графики. Давайте заполним эти два поля и посмотрим, как это отразиться на HTML-представлении рисунка. После ввода этих значений и подтверждения изменений мы можем перейти на вкладку **HTML** и обнаружить, что к нашему тегу добавились параметры `lowsrc` и `alt`. Посредством простейшего анализа HTML-текста можно догадаться, что значение параметра `lowsrc` содержит путь к графическому файлу подмены основного изображения, а значение параметра `alt` представляет собой текстовую строку, печатаемую на месте нашего рисунка, когда браузер удаленного пользователя не может его отобразить.

В нижней части диалогового окна расположена группа элементов управления, в которой можно поставить гиперссылку в соответствие нашему рисунку. Эта группа носит название **Default hyperlink**. В поле **Location** вводится URL места назначения, а в поле **Target Frame** — значение, указывающее, в каком окне отображать полученный документ. Рядом с полем для ввода URL находится кнопка **Browse**, которая активизирует уже знакомое нам диалоговое окно создания гиперссылки. Впрочем, того же самого эффекта можно добиться, просто выполнив команду **Hyperlink** контекстного меню рисунка. Давайте попробуем привязать к рисунку какую-либо гиперссылку, а затем посмотрим, как это реализуется с помощью HTML-кода. Если после выполнения всех действий мы перейдем на вкладку **HTML**, то мы увидим конструкцию, подобную нижеследующей:

```
<a href="2.htm"><img border="0" src="images/Hlplogo.gif" width="97"
height="40"></a>
```

Как мы видим, тег, размещающий изображение из графического файла Hlplogo.gif, находится в обрамлении тегов, объявляющих гиперссылку на файл с именем 2.html. Помимо этого способа создания графических ссылок, есть еще технология, позволяющая привязать несаколько гиперссылок к одному изображению. В таком случае рисунок разбивается на несколько частей, для каждой из которых задается своя ссылка. Эта технология носит название сегментированной графики. Ее мы рассмотрим, когда дойдем до инструментов, позволяющих реализовать эту технологию.

У рассматриваемого нами диалогового окна **Picture Properties** есть еще одна вкладка, задающая некоторые свойства изображений. Она носит название **Appearance** (рис. 1.9). На этой вкладке мы можем указать относительное расположение графического изображения и текста, находящихся рядом.

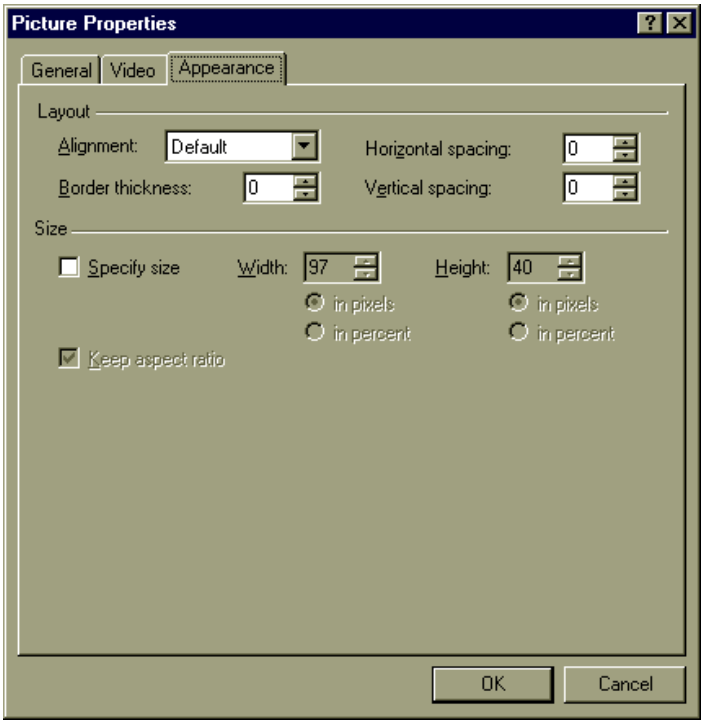

Рис. 1.9. Вкладка **Appearance** диалогового окна **Picture Properties**

То есть, выравнивание их относительно друг друга. Способ выравнивания указывается в выпадающем списке с именем **Alignment**. Если мы выберем значение **Top**, то текст, расположенный сразу после тега, объявляющего вставку рисунка, без промежуточного объявления нового абзаца, будет выравнен по верхней границе изображения. Значение **Bottom** выравнивает текст по нижней границе изображения. Если мы выберем значение **Middle**, то первая строка текста будет размещена по центру изображения. Текст может раз-

мещаться как слева от изображения, так и справа от него. Задается это значениями **Left** и **Right**, соответственно. Есть и более точные методы выравнивания текста. Так, параметр **Texttop** осуществляет выравнивание текста относительно самых высоких символов в строке. Если указано выравнивание **Absmiddle**, то выравнивается середина первой строки текста относительно середины рисунка. А значение **Baseline** позволяет выравнивать текст по нижнему краю рисунка относительно базовой линии первой строки. *Базовой линией* называется воображаемая линия, на которой расположены символы текста. Значение **Absbottom** предназначено для выравнивания текста по нижнему краю рисунка относительно нижней границы первой строки текста. Необходимо осознавать, что базовая линия не всегда совпадает с нижней границей строки. Нижняя граница располагается на уровне, где кончаются так называемые "хвостики" отдельных символов, которые спускаются ниже базовой линии.

В поле **Border thickness** вводится желаемая толщина рамки вокруг рисунка. В полях **Horizontal spacing** и **Vertical spacing** указываются расстояния в пикселах по горизонтали и вертикали, отделяющие графическое изображение от текста.

Для изменения размеров изображения используется группа элементов управления **Size**. По умолчанию, в полях **Width** и **Height** указываются истинные размеры рисунка. Поставив флажок **Specify size**, мы получаем возможность прямого редактирования содержимого этих полей. Причем, размеры окна могут указываться как в пикселах, так и в процентах от размера окна просмотра браузера.

Теперь, когда мы узнали, какие свойства мы можем установить для нашего изображения, разберемся в деталях реализации при помощи HTML-кода. Как устанавливаются размеры изображения, мы уже знаем. Это параметры `height` и `width`. А для того, чтобы увидеть отработку других свойств, заполним все предлагаемые нам поля, нажмем кнопку **OK**, подтверждая изменения, и перейдем на вкладку **HTML**. Там мы увидим код, похожий на

```
<img border="1" src="images/Hlplogo.gif" width="97" height="40"
align="left" hspace="2" vspace="2">
```

Как мы можем увидеть, толщина границы задается при помощи параметра `border`, для выравнивания зарезервирован параметр `align`, а установка отступа по горизонтали и вертикали производится при помощи параметров `hspace` и `vspace` соответственно.

Теперь перейдем к заключительной части главы, в которой мы рассмотрим средства для работы с графическими изображениями, предлагаемые FrontPage 2000. Как мы уже говорили, каждое действие привязано к отдельной кнопке из общего комплекта, находящегося на инструментальной панели **Pictures**. Начнем с левого края и пройдем до конца панели, не пропуская ни одну кнопку.

Самая первая кнопка доступна всегда, вне зависимости от наличия выделенного рисунка. Это и понятно, ведь кнопка **Insert Picture From File** предназначается для вставки рисунка. То есть она дублирует кнопку основной инструментальной панели. Следующая кнопка **Text** при нажатии вставляет в центр рисунка прямоугольный блок для вставки текста. Блок поддается перемещению и изменению размеров. Шрифт вставляемого текста выбирается при помощи элементов управления инструментальной панели форматирования. По умолчанию используется шрифт Times New Roman третьего размера с выключкой по центру. Текстовый блок является обычным текстовым объектом, поэтому текст внутри него можно редактировать. Вставка подобного текстового дополнения реализуется при помощи активного элемента, для понимания работы которого необходим небольшой экскурс в программирование, а мы договорились этого не делать. Таким образом, если нам потребуется создать подобный текстовый блок в рисунке, то мы воспользуемся стандартным инструментом FrontPage, а не будем пробовать написать его на HTML вручную. Кнопка **Auto Thumbnail** позволяет создавать так называемые "ноготки" рисунков. Ноготками (thumbnails) называют уменьшенные копии графических изображений, этакие ярлычки. Обычно они применяются в том случае, если основное изображение достаточно велико, а скорость его загрузки в браузере удаленного пользователя оставляет желать лучшего. Обычно на каждый ноготок вешается ссылка, по которой будет отображаться основное изображение.

Следующая триада кнопок посвящена точному позиционированию рисунка на странице и его расположению относительно текста. Что касается точного позиционирования, необходимо отметить, что сам по себе HTML не имеет подобных механизмов. Причину этого мы уже обсуждали. Однако это можно сделать средствами FrontPage. Для выполнения позиционирования рисунка достаточно нажать кнопку **Position Absolutely**. После этого, как только курсор мыши попадает на наше графическое изображение, он принимает стандартную форму курсора для перемещения объекта, то есть крестика с четырьмя наконечниками. Теперь можно нажать левую клавишу мыши, и, не отпуская ее, перенести рисунок точно в то место, которое мы для него предназначили. Необходимо отметить, что эта кнопка является переключателем. Пока она находится в нажатом состоянии, данный объект будет точно позиционирован на странице. При отключении этого режима он вернется на свое исходное место. Если при нажатой кнопке мы заглянем на вкладку **HTML**, то увидим, что тег, отображающий наш рисунок, заключен в теги `<span>` и `</span>`. Обычно он выглядит как `<span style="position: absolute; left: 74; top: 32">`. Как мы видим, параметр `style` задает тип позиционирования (в нашем случае по абсолютным координатам страницы) и координаты верхнего левого угла рисунка. Хотелось бы обратить внимание, что этот тег может не поддерживаться всеми браузерами, поэтому рекомендуется применять его с некоторой осторожностью. Если говорить честно, то FrontPage максимальным образом приспособлен именно к Internet Explorer, что, вполне объяснимо, но накладывает определенные ограничения

на его использование. Некоторые особо продвинутые возможности могут не распознаваться другими браузерами, и создаваемая страница не будет выглядеть адекватно замыслу дизайнера. Следующие две кнопки **Bring Forward** и **Send Backward** позволяют установить положение рисунка поверх или позади текста соответственно. При их использовании включается режим абсолютного позиционирования, и в теге `<span>` у параметра `style` появляется новая константа с именем `z-index`. Именно она указывает положение картинки в слоях страницы. Чем больше это значение, тем ближе к пользователю находится этот объект, перекрывая все остальные объекты, находящиеся под ним, у которых значение параметра `z-index` меньше. Таким образом фон нашей страницы имеет наименьшее значение параметра `z-order`, а те объекты, которые мы видим полностью, имеют максимальное значение этого параметра.

Далее расположены четыре кнопки, позволяющие вращать рисунок вправо и влево и отражать его относительно вертикальной и горизонтальной осей. При этом нажатие этих кнопок вносит изменения именно в рисунок, а не в HTML-код, отображающий его. Кнопки **Rotate Left** и **Rotate Right** вращают рисунок влево и вправо на девяносто градусов. Кнопки **Flip Horizontal** и **Flip Vertical** отражают рисунок относительно его главных осей.

Следующая группа кнопок предназначена для регулировки параметров яркости и контраста рисунка. Кнопки **More Contrast** и **Less Contrast** повышают и понижают степень контрастности, следующие две кнопки **More Brightness** и **Less Brightness** то же самое делают с яркостью. Все эти изменения делаются непосредственно с рисунком и сохраняются во временном файле. После сохранения страницы все подобные изменения будут записаны в основной графический файл.

Дальше следует блок инструментов, который применяется также для обработки всего рисунка. Так, кнопка **Crop** предназначена для изменения размеров рисунка, а точнее, для уменьшения его путем обрезания границ. При нажатой кнопке **Crop** курсор мыши меняет свою форму, и мы можем обрезать картинку так, как мы этого хотим. Повторное нажатие этой кнопки отключает этот режим. Кнопка **Set Transparent Color** применяется для того, чтобы один из цветов рисунка сделать прозрачным. Применять этот инструмент лучше всего к рисункам, хранимым в GIF-формате. После нажатия на эту кнопку, курсор принимает форму специализированного указателя. Если мы в это время щелкнем мышью на рисунке, то FrontPage определит цвет того пиксела, на который указывал курсор, и все пикселы рисунка этого цвета будут перекрашены в так называемый "прозрачный" цвет. Специфику применения прозрачного цвета мы рассмотрим в следующей главе. Следующая кнопка с именем **Black and White**, как нетрудно догадаться, переводит рисунок в черно-белый формат. Или, если быть более точным, отображает его при помощи градаций серого цвета. Кнопка **Wash Out** как бы "выстирывает" рисунок, заставляя все его цвета поблекнуть и стать приглушенными. Подобные рисунки очень хорошо работают в качестве подложек на страницах. Кстати, в рабочих панелях русскоязычного Microsoft Office

подобная кнопка так и называется — "подложка". Кнопка **Bevel** добавляет к нашему рисунку маленькую угловую рамку, как бы приподнимая его над общим уровнем страницы. Эффект, на мой взгляд, несколько сомнительный. А вот последняя кнопка **Resample** чрезвычайно полезна. Дело в том, что при изменении общих размеров рисунка при помощи перетаскивания мышью, пропорции рисунка могут несколько искажаться. Так вот, эта кнопка позволяет очень точно восстановить пропорции рисунка, нарушенные при изменении его границ. Я настоятельно рекомендую пользоваться этим инструментом каждый раз, когда размер изображения был изменен.

Следующий блок кнопок предназначен для реализации технологии сегментированной графики. Первая кнопка **Select** чаще всего находится в нажатом состоянии. Она переводит FrontPage в обычный режим выбора элементов. Чаще всего эту кнопку приходится нажимать для того, чтобы отменить ошибочное нажатие других кнопок этой группы. Следующие три кнопки предназначены для создания на рисунке так называемых "горячих областей" (hotspots) — участков рисунка к которым будут привязаны гиперссылки. Кнопка **Rectangular Hotspot** создает горячую область прямоугольной формы. При нажатии на эту кнопку, курсор мыши, находясь на территории рисунка, приобретает форму карандаша. Нажатие левой кнопки мыши фиксирует один угол прямоугольника. Затем, не отпуская нажатую кнопку, необходимо перемещать мышь, одновременно растягивая горячую область до необходимых размеров. Для фиксации размера достаточно отпустить нажатую кнопку. После определения размера горячей области активизируется уже знакомое нам диалоговое окно для создания гиперссылки. После этого горячая область становится стандартным объектом со своим свойствами. Так же, как и все остальные объекты, горячие области могут перемещаться проектировщиком, их границы могут быть изменены, сами они могут быть удалены. Таким же образом действуют и две следующие кнопки — **Circular Hotspot** и **Polygonal Hotspot**. Они создают горячие области в форме круга и многоугольника. А последняя кнопка в этой группе с названием **Highlight Hotspots** предназначена для того, чтобы проектировщик Web-страницы мог увидеть все горячие области, размещенные им на рисунке. При нажатии на кнопку само изображение рисунка исчезает, а остаются лишь его конуры и границы всех горячих областей. Таким образом, мы получаем очень удобный режим работы, в котором мы можем посмотреть, где размещены горячие области, не перекрываются ли они, и т. д.

Давайте утановим на наш рисунок три горячие области различных форм и рассмотрим реализацию технологии "сегментированной графики" с точки зрения HTML. Одна область будет прямоугольной, вторая — круглой, а границы третьей будут очерчены многоугольником. При просмотре HTML-кода, мы обнаружим следующий набор тегов:

```
<map name="FPMap0">
<area href="2.htm" shape="rect" coords="6, 4, 46, 20">
<area href="3.htm" shape="circle" coords="54, 44, 18">
```

```
<area href="4.htm" shape="polygon" coords="100, 17, 134, 27, 128,
53, 98, 53, 82, 29"></map>
<img     border="0"     src="images/Hlplogo.gif"     usemap="#FPMap0"
width="178" height="73">
```

Теперь внимательно разбиремся с кодом. Отметим, что этот блок состоит из двух частей. Первая часть, заключенная между тегами `<map>` и `</map>` задает карту сегментов гиперссылок тех самых горячих областей. Вторая часть содержит тег, реализующий вставку изображения и привязку карты к нему. У объявляющего тега первой части есть параметр `name`, который задает имя карты. Затем это имя используется во второй части конструкции. Между тегами `<map>` и `</map>` располагаются теги `<area>`, каждый из которых объявляет одну горячую область. Параметр `href`, как нетрудно догадаться, задает адрес гиперссылки. Параметр `shape` предназначен для определения формы горячей области, а параметр `coords` задает ее координаты в рисунке относительно верхнего левого угла. Из кода видно, что у этого параметра есть три значения. Значение `rect` говорит о том, что сегмент гиперссылки имеет форму прямоугольника, а в качестве значения параметра `coords` записывается строка, состоящая из четырех чисел, разделенных запятыми. Это координаты верхнего левого и нижнего правого угла прямоугольника. В том случае, если горячая область имеет вид окружности, значение параметра `shape` будет `"circle"`. Впрочем, браузер сможет правильно понять и значение `"circ"`. При этом параметром `coords` передается три числа, задающие координаты центра окружности и ее радиус. И, наконец, значение `"polygon"` или просто `"poly"` задает сегмент гиперссылки в виде многоугольника. В этом случае параметр `coords` задает пары значений координат вершин многоугольника.

С первой частью, определяющей саму карту гиперссылок, мы разобрались. Вторая намного проще. Это обычный тег вставки изображения, только теперь там есть еще один параметр — `usemap`. В качестве значения этому параметру передается имя подключаемой карты гиперссылок.

В конце этой главы, когда мы уже знаем, как создавать гиперссылки, вставлять рисунки и объединять гиперссылки с рисунками самыми различными способами, нам осталось рассмотреть лишь одну-единственную кнопку, расположенную в самом конце инструментальной панели **Pictures**. Она носит название **Restore**, и позволяет сразу отменять все изменения рисунка, которые произошли с момента последнего сохранения. Конечно, для этой же цели можно воспользоваться стандартными средствами отмены действий, но это не всегда удобно. Например, когда объем изменений был слишком велик, или после работы с изображением мы успели сделать что-то еще, не относящееся к самой картинке. Именно в таких случаях стоит обращаться к этой последней кнопке.

Мультимедийные возможности

Под мультимедийными возможностями мы будем понимать размещение на Web-страницах аудио- и видеоклипов. К сожалению, простых способов вставки звука в страницу таким образом, чтобы удаленный пользователь мог самостоятельно решать, когда ему запустить воспроизведение аудиофрагмента, не существует. Чаще всего используется фоновая музыка, которая начинает воспроизводиться в момент загрузки страницы. Это удовольствие считается свойством страницы, поэтому установку фонового аудиоклипа необходимо осуществлять при помощи команды меню **File/Properties**. При этом активизируется диалоговое окно **Page Properties**. Интересующие нас элементы управления размещаются на вкладке **General** (рис. 1.10), в группе **Background sound**.

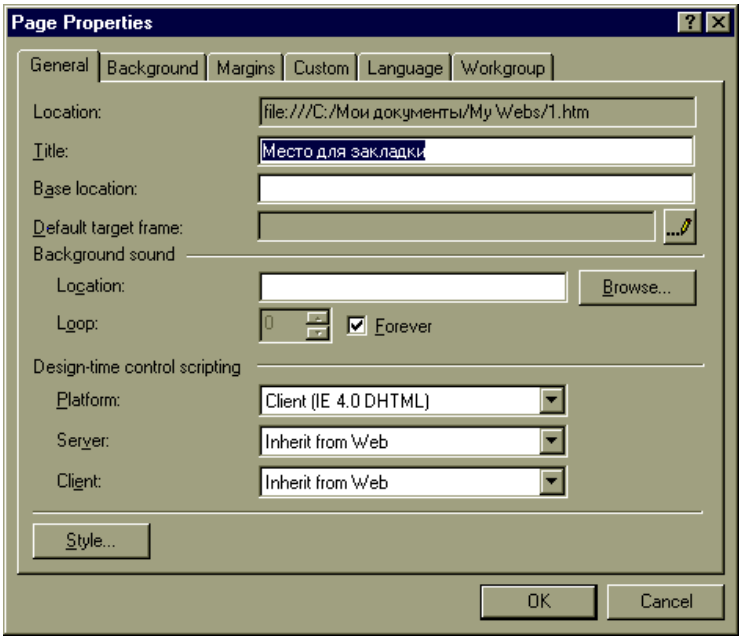

Рис. 1.10. Вкладка **General** диалогового окна **Page Properties**

В поле **Location** вводится местонахождение музыкального файла. Если трудно ввести местонахождение файла по памяти, можно воспользоваться кнопкой **Browse**, которая предоставляет стандартный диалог для поиска файла. Помимо этого, у подключаемого видеофрагмента могут быть и еще некоторые свойства. Мы можем определить, сколько раз он будет проигрываться при загрузке страницы или вообще зациклить его воспроизведение, и тогда этот фрагмент будет начинаться снова и снова. Для указания количества повторов имеется поле ввода **Loop**, а для того чтобы поставить клип на постоянное воспроизведение,

достаточно поставить флажок **Forever**. Если мы укажем все необходимые данные, и нажмем кнопку **OK** для подтверждения изменений, то, перейдя на вкладку **HTML**, мы увидим, что перед тегом `</head>`, закрывающим область заголовка, вставлен тег `<bgsound src="Canyon.mid" loop="-1">`, объявляющий фоновый аудиофрагмент. В нем присутствует уже знакомый нам параметр `src`, в качестве значения которого передается местонахождение подключаемого звукового файла и параметр `loop`, указывающий количество повторов аудиоклипа. Если его значение будет равно −1, то фрагмент будет воспроизводиться постоянно, в ином случае браузер распознает число, представленное в качестве значения этого параметра, и проигрывает клип положенное число раз. Как видно, все не так уж и сложно.

Вставка видеофрагмента осуществляется при помощи команды меню **Insert/Picture/Video**. При этом активизируется стандартный диалог поиска и вставки файла с объектом. FrontPage поддерживает файлы форматов Video for Windows (avi), Windows Media (asf), Real Video (ram и ra). Вставка видеофрагмента осуществляется при помощи тега `<img dynsrc="DelpCool.avi">`. То есть вставка видеоклипа является вариантом вставки обычного изображения, но есть и некоторые отличия. Так, если мы попробуем установить свойства для видеоклипа при помощи команды контекстного меню **Picture Properties** или при помощи команды **Format/Properties** в тот момент, когда вставленный видеоклип выделен, то активизируется диалоговое окно **Picture Properties**, но при этом на первый план выходит вкладка **Video** (рис. 1.11). В поле ввода **Video source** указывается месторасположение файла, содержащего видеофрагмент. Естественно, эта информация поддается редактированию как путем прямого ввода, так и при помощи кнопки **Browse**. Ниже поля ввода находится флажок **Show controls in Browser**, включение которого визуализирует элементы управления воспроизведением при просмотре страницы удаленным пользователем. То есть под видеоклипом будут находиться кнопки запуска/паузы и индикатор положения. Количество повторов видеоклипа устанавливается в поле с названием **Loop**. Пауза между циклами воспроизведения устанавливается в поле **Loop delay**. Время паузы измеряется в миллисекундах. Ну, а для того, чтобы клип воспроизводился постоянно, необходимо установить флажок **Forever**. И что самое интересное, мы можем указать, когда должен начинать воспроизводиться видеоклип. В группе **Start** находятся два флажка, с помощью которых можно устанавливать правила воспроизведения. Так, если установлен флажок **On file open** (по умолчанию), то воспроизведение начинается сразу после загрузки страницы в браузер удаленного пользователя. Если же выбран вариант **On mouse over**, то запуск видеоклипа происходит в тот момент, когда к нему подводится курсор мыши. Причем, необходимо отметить, что это независимые переключатели, то есть мы можем выбрать любую их комбинацию. Правда, в том случае, если ни один из переключателей не выбран, имеет смысл вставить элементы управления видеоклипом, так как иначе не будет возможности запустить его.

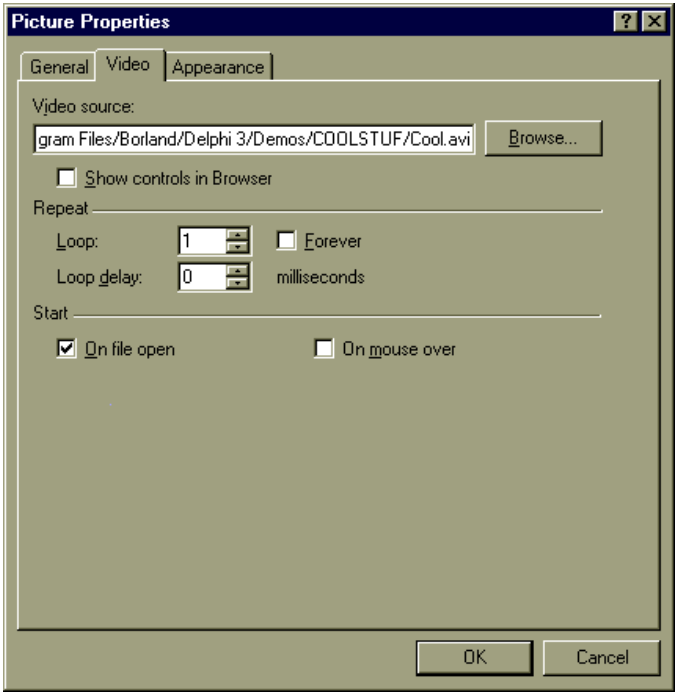

Рис. 1.11. Вкладка **Video** диалогового окна **Picture Properties**

Теперь давайте узнаем, как эти свойства реализуются при помощи HTML. Если мы установим эти свойства и посмотрим на вкладку **HTML**, то увидим приблизительно такой HTML-код:

```
<img border="0" dynsrc="images/Cool.avi" start="fileopen,mouseover"
controls width="123" height="133" loop="2" loopdelay="5">
```

Попробуем разобраться со всеми этими параметрами. Параметр `border` показывает толщину рамки, ограничивающую вставленный клип. Параметр `start` предназначен для указания события, после которого начнется воспроизведение видеофрагмента. В нашем случае указаны сразу два варианта: после загрузки файла и в момент попадания курсора мыши на то место, где находится рабочее пространство видеоклипа. Параметр `controls` указывает на то, что под видеофрагментом будут расположены элементы управления им. Параметры `width` и `height`, как обычно, указывают ширину и высоту области воспроизведения клипа. Параметр `loop` устанавливает количество повторений фрагмента, а `loopdelay` — паузу между отдельными циклами воспроизведения. Необходимо отметить, что если мы хотим добиться постоянного воспроизведения видеоклипа, параметр `loop` принимает значение `"infinite"`.

Исходя из того, что видеоклип считается одним из вариантов графического изображения, к нему можно присоединить гиперссылку, и реализация ее не

будет отличаться от создания гиперссылки на обычном рисунке. Помимо этого можно указать выравнивание окна видеоклипа и текста, расположенного рядом с ним, при помощи элементов управления, расположенных на вкладке **Appearance** окна **Picture Properties**. Параметры, указывающие тип выравнивания, ничем не отличаются от параметров для выравнивания обычных графических изображений.

Таблицы

Таблицы являются одним из самых важных инструментов при построении Web-страниц. Помимо размещения информации в табличной форме, они активно применяются для правильного позиционирования текста и графики. При этом необходимые блоки текста и графики размещаются в ячейках таблицы, а ее границы делаются невидимыми. Основная часть инструментов для работы с таблицами сосредоточена на инструментальной панели **Tables**, которая визуализируется при помощи команды меню **View/Toolbars**. Большая часть средств, размещенная на этой инструментальной панели, дублируется командами меню **Table**. Для создания таблицы на Web-странице необходимо выполнить команду меню **Table/Insert Table**. При этом активизируется диалоговое окно **Insert Table** (рис. 1.12). Для создания

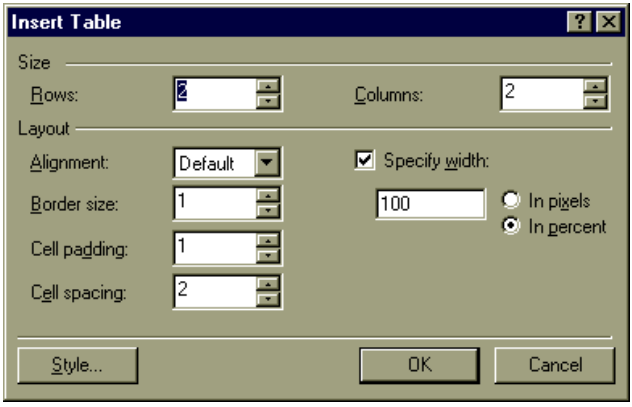

Рис. 1.12. Диалоговое окно **Insert Table**

таблицы нам необходимо установить в ней количество строк и столбцов при помощи полей редактирования **Rows** и **Columns** соответственно. Остальные значения не обязательны. Выпадающий список **Alignment** предназначен для установки выравнивания таблицы на странице. Поле **Border size** позволяет установить толщину линий, ограничивающих каждую ячейку. По умолчанию она составляет один пиксель. Если мы установим нулевое значение толщины, то границы будут не видны, а табличная структура останется. Поле **Cell padding** позволяет установить расстояние между краями ячейки таблицы и ее содержимым. Эта величина не может быть установлена для каж-

дой ячейки отдельно. Скорее, это свойство всей таблицы. Значение в поле **Cell spacing** указывает расстояние между рамкой и ячейками таблицы. Флажок **Specify width** позволяет использовать возможность принудительного указания ширины таблицы. Причем, ее ширина может указываться как в пикселах, так и в процентах от ширины окна просмотра браузера удаленного пользователя. Если мы установим в диалоговом окне **Insert Table** некоторые значения, отличные от установленных по умолчанию, и нажмем на кнопку **OK**, то полученный результат будет выглядеть примерно следующим образом (рис. 1.13).

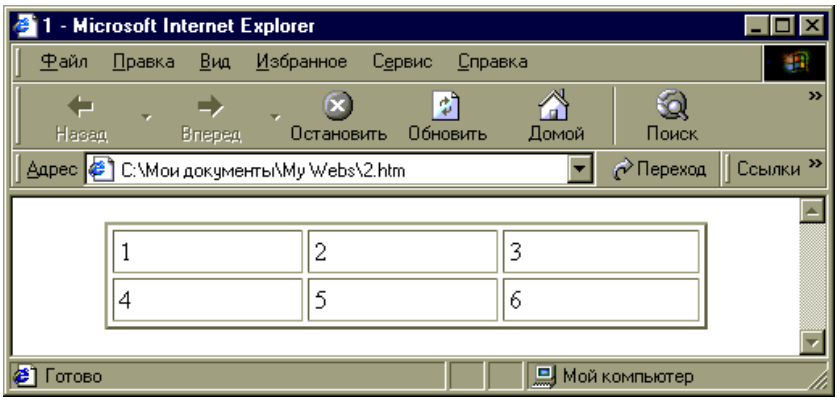

Рис. 1.13. Внешний вид Web-страницы с встроенной таблицей

Подобная таблица реализуется при помощи следующего HTML-кода:

```
<div align="center">
  <center>
  <table border="2" width="80%" cellpadding="3" cellspacing="3">
    <tr>
      <td width="33%">1</td>
      <td width="33%">2</td>
      <td width="34%">3</td>
    </tr>
    <tr>
      <td width="33%">4</td>
      <td width="33%">5</td>
      <td width="34%">6</td>
    </tr>
  </table>
  </center>
</div>
```

Парные теги `<div>` и `<center>` предназначены для указания выравнивания таблицы относительно Web-страницы. В том случае, если в списке **Alignment** при создании таблицы было бы выбрано значение **Default**, эти теги вообще не появились бы в коде HTML. Сама таблица объявляется тегом `<table>`, а закрывается его парой `</table>`. Внутри этого блока поочередно объявля-

ется каждая строка при помощи тегов `<tr>` и `</tr>`. И уже внутри каждой строки объявляются ячейки, составляющие столбцы, при помощи тегов `<td>` и `</td>`. У каждого объекта, будь то таблица целиком, строка или ячейка, есть свои свойства, а значит и параметры для их указания, входящие в состав тегов. Мы воспользуемся нашей стандартной схемой работы, когда мы сначала узнаем, как объявить эти свойства, пользуясь возможностями FrontPage 2000, а затем рассмотрим, как эти свойства реализуются при помощи HTML.

Закончим сначала рассмотрение той таблицы, которую мы только что создали. В теге `<table>` имеется параметр `border`, указывающий толщину рамки в пикселах. Следующий параметр `width` говорит о том, что наша таблица будет занимать 80% ширины окна просмотра браузера. Параметр `cellpadding`, соответствующий одноименному полю ввода диалогового окна **Insert Table**, задает расстояние в пикселах между краями ячейки и ее содержимым. Значение параметра `cellspacing` указывает расстояние между внешней рамкой таблицы и ее ячейками. Так как все столбцы и строки объявляются внутри табличного HTML-блока, нет необходимости указывать их количество в теге `<table>`, однако необходимо знать параметры, позволяющие это делать. Для указания количества столбцов используется параметр `cols`.

Теперь начнем устанавливать свойства таблицы. Для этого можно воспользоваться командой меню **Table/Properties/Table** или **Format/Properties**.Такого же результата можно добиться, используя команду контекстного меню **Table Properties**. Для изменения и установки свойств таблицы предназначено диалоговое окно **Table Properties** (рис. 1.14).

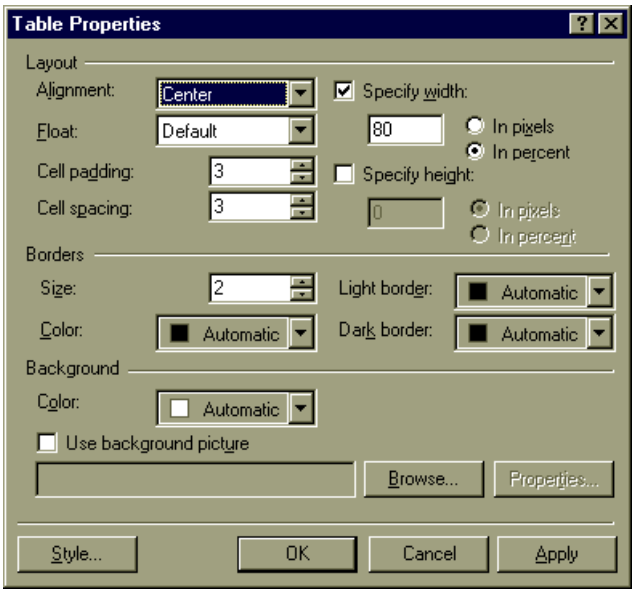

Рис. 1.14. Диалоговое окно **Table Properties**

Список **Alignment** и переключатель **Specify width** нам уже знакомы, и рассматривать их второй раз мы не будем. Список **Float** предназначен для позиционирования таблицы относительно текста, расположенного на странице, но не входящего в таблицу. Если мы выберем значение **Left**, то таблица будет прижата к левому краю, а текст будет расположен справа. Для этого в тег `<table>` вставляется параметр `align`. В принципе, этот параметр предназначен для указания выравнивания самой таблицы. В нашем случае, для нее было указано выравнивание по центру, и осуществлялось оно тегами `<center>` и `</center>`. Но так как параметр `align` находится в самом теге `<table>`, он отменяет центрирование всего блока HTML-кода, содержащего в себе искомую таблицу.

Группа элементов **Specify height** позволяет указать высоту строк таблицы. Этот выбор применяется ко всем строкам сразу. В HTML для указания высоты строк применяется параметр `height`, входящий в тег `<table>`.

Блок **Borders** содержит элементы управления, с помощью которых задаются свойства внешней границы таблицы. Поле **Size**, как мы уже знаем, позволяет указывать толщину внешней рамки. Остальные три списка этой группы ответственны за цветовое оформление рамки. Если для всей границы планируется использовать только один цвет, то используется список **Color**. В HTML это свойство задается параметром `bordercolor`. Когда мы, к примеру, хотим сделать границу розового цвета, используется параметр `bordercolor="#FF00FF"`. Однако у нас есть возможность несколько более изощренного цветового оформления границы. Для достижения трехмерного изображения таблицы рамка для них делается двух цветов. Левая и верхняя границы делаются более светлыми, а правая и нижняя — более темными. Для подобного оформления в нашем диалоговом окне используют два списка: **Light border** и **Dark border**, в которых указываются светлый и темный цвета рамки, соответственно. Для задания этих цветов в HTML, используются параметры `bordercolorlight` и `bordercolordark`. При этом необходимо учитывать, что если какой-либо из этих цветов, или оба сразу не указаны явно, то для их замены используется основной цвет границы, выбранный в списке **Color**, и заданный параметром `bordercolor`. Но если они указаны, то основной цвет вообще не будет использоваться.

Теперь перейдем к оформлению фона таблицы. Для этого используется группа элементов управления **Background**. В списке **Color** можно установить цвет фона таблицы. В HTML этот цвет устанавливается при помощи параметра `bgcolor`. Так, если мы захотим установить для таблицы фон светло-зеленого цвета, который FrontPage называет lime, то этот параметр тега `<table>` примет вид `bgcolor="#00FF00"`. В том случае, если вместо однотонного фона, мы хотим установить какой-либо рисунок, необходимо поставить флажок **Use background picture**, и в поле ввода указать местонахождение графического файла. Для этого можно воспользоваться кнопкой **Browse**. В HTML вставка графического файла в качестве фона производится при помощи параметра `background`, значением которого является местонахождение файла.

На этом перечень свойств таблицы, которые предлагаются нам FrontPage 2000, заканчивается, но в действительности их немного больше и существуют другие параметры тега `<table>`, которые мы не рассмотрели. Но в нашу задачу входит не рассмотрение языка HTML, как такового, а изучение средств проектирования Web-страниц. Поэтому, пока перейдем к свойствам отдельных ячеек, а если в ходе повествования нам встретятся новые возможности, то тогда мы их и рассмотрим.

Свойства как отдельной ячейки, так и группы, редактируются при помощи команды меню **Table/Properties/Cell** или при помощи команды контекстного меню **Cell Properties**. При этом активизируется диалоговое окно **Cell Properties** (рис. 1.15).

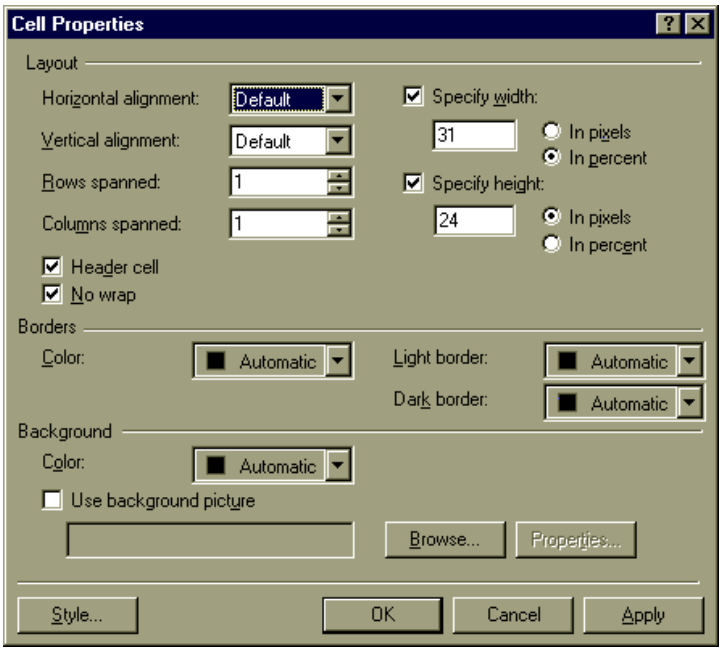

Рис. 1.15. Диалоговое окно **Cell Properties**

Как мы уже знаем, ячейки таблицы объявляются тегами `<td>`. Следовательно, для указания свойств ячеек будут использоваться параметры именно этого тега. Блок **Layout** содержит элементы, управляющие внешним видом расположения ячейки и отображения ее содержимого. Выпадающие списки **Horizontal alignment** и **Vertical alignment** позволяют выравнивать содержимое ячейки (это не обязательно должен быть текст) по горизонтали и вертикали соответственно. Так, если вместо значений по умолчанию (**Default**) мы укажем горизонтальное выравнивание содержимого по ширине ячейки (**Justify**), а по вертикали содержимое ячейки мы прижмем к ее нижнему краю (**Bottom**), то при просмотре HTML-кода, реализующего эту ячейку, мы

увидим тег `<td valign="bottom" align="justify">`. Как мы видим, ячейка объявляется при помощи тега `<td>`, а уже в этот тег вписываются параметры `valign` и `align`, указывающие вертикальное и горизонтальное выравнивание содержимого ячейки. Варианты горизонтального выравнивания мы знаем и так. Это прижатие к левому и правому краю, центрирование и растяжка по ширине. Для вертикального выравнивания применяются варианты смещения содержимого вверх (значение `top` параметра `valign`), выравнивание по центру ячейки (`middle`), по нижней границе ячейки (`bottom`) и выравнивание по базовой линии текста (`baseline`). Помимо выравнивания содержимого ячейки мы можем принудительно установить ее размеры. Для этого необходимо выставить флажки **Specify width** и **Specify height**, которые ответственны за указание размеров ячейки по горизонтали и по вертикали соответственно. Останется только указать размеры в полях ввода и единицы измерения. В качестве единиц измерения традиционно предлагаются пикселы и проценты. Если размеры ячейки установлены принудительно, тег, объявляющий ячейку, принимает вид `<td width="31%" height="30">`. Из этого примера видно, что ширина ячейки устанавливается при помощи параметра `width`, и составляет тридцать один процент от общей ширины таблицы. Высота ячейки задается значением параметра `height`, которое в данном случае равно тридцати пикселям.

Теперь перейдем к полям ввода **Rows spanned** и **Columns spanned**. С помощью этих значений можно указать, что одна ячейка будет объединять несколько соседних. Подобные ячейки часто применяются для заголовков нескольких колонок или строк сразу. Значение в поле **Rows spanned** показывает, сколько соседних ячеек по вертикали будет объединять данная ячейка. Если левую верхнюю ячейку растянуть по вертикали, поставив двойку в поле **Rows spanned**, то полученная таблица будет выглядеть следующим образом (рис. 1.16).

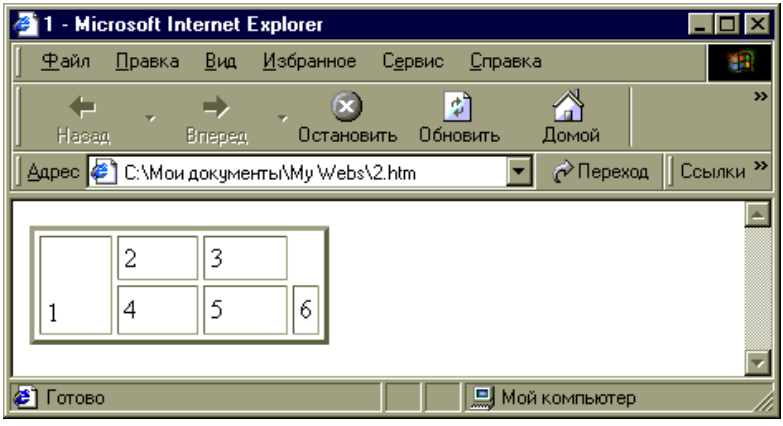

Рис. 1.16. Внешний вид таблицы с объединенными ячейками

Такая таблица будет реализована с помощью следующего HTML-кода:

```
<table border="3" width="43%" cellpadding="3" cellspacing="3" bor-
dercolor="#FF00FF" height="72" align="left" bordercolor-
light="#C0C0C0" bordercolordark="#808080">
    <tr>
      <td width="31%" height="24" valign="bottom" align="justify"
rowspan="2">1</td>
      <td width="34%" height="24">2</td>
      <td width="35%" height="24">3</td>
    </tr>
    <tr>
      <td width="31%" height="30">4</td>
      <td width="34%" height="30">5</td>
      <td width="35%" height="30">6</td>
    </tr>
  </table>
```

Как видим, первая ячейка заняла весь крайний левый столбец, что было обеспечено за счет параметра `rowspan="2"`. При этом ячейки нижней строки смещены вправо. Так как размер таблицы ограничен параметром `width="43%"` тега `<table>`, то последняя ячейка нижней строки заняла очень мало места, несмотря на то, что ей отводилось тридцать пять процентов от ширины всей таблицы, что видно из тега, ее объявляющего `<td width="35%" height="30">6</td>`. Из этого следует, что те конструкции языка HTML, которые браузер не может правильно выполнить или интерпретировать, просто игнорируются.

В случае, если мы хотим растянуть ячейку по горизонтали, следует выставить необходимое значение в поле ввода **Columns spanned**. При этом у тега `<td>` появится параметр `colspan`.

В блоке **Layout** нам осталось рассмотреть всего два флажка. Флажок **Header cell** позволяет из обычной ячейки сделать ячейку заголовка. При этом она будет реализована при помощи тегов `<th>` и `</th>`, которые несколько изменят отображение содержимого ячейки. Для ячеек заголовка можно устанавливать те же свойства, что и для обычных ячеек и реализовываться они будут при помощи идентичных параметров.

Поставить флажок **No wrap** нужно в тех случаях, когда мы хотим запретить перенос содержимого ячейки на другую строку, если оно не помещается в окне просмотра. То есть, если текст ячейки не виден полностью в окне просмотра браузера удаленного пользователя, он обычно переносится на другую строку, тем самым немного увеличивая высоту таблицы. Это представляется достаточно разумным решением, но в некоторых случаях перенос содержимого является недопустимым и тогда используется переключатель **No wrap**. При этом в тег, объявляющий ячейку, добавляется параметр `nowrap`.

Блок **Borders** содержит три выпадающих списка, устанавливающих цвета границ конкретной ячейки, или нескольких ячеек сразу, входящих в текущее выделение. Список **Color** позволяет указывать цвет всех границ ячейки. В том случае, когда

необходимо создать псевдотрехмерную рамку за счет выделения ее двумя цветами, используются списки **Light border** и **Dark border**. Принципы взаимодействия этих трех выпадающих списков полностью аналогичны случаю установки цветов границы для всей таблицы в целом. То есть, если какой-либо из цветов темного и светлого обрамления не установлен, вместо него используется общий цвет рамки, указанный в списке **Color**. Реализация этих свойств происходит при помощи уже знакомых нам параметров. Так, ячейка, для которой установлены все три цвета, реализуется при помощи конструкции `<td bordercolor="#008000" bordercolorlight="#00FFFF" bordercolordark="#FF0000">`. В данном теге при помощи параметра `bordercolor` установлен зеленый (green) цвет рамки, при помощи параметров `bordercolorlight` и `bordercolordark` установлены голубой (aqua) цвет для светлой границы и красный (red) цвет для темной.

В рассматриваемом нами диалоговом окне остался только раздел для установки фона ячейки — **Background**. Цвет фона ячейки устанавливается при помощи выпадающего списка с наименованием **Color**. Реализуется подобная возможность при помощи параметра `bgcolor` тегов `<td>` или `<th>`. Если же необходимо в качестве фона использовать графическое изображение, следует выставить флажок **Use background picture** и в поле ввода ввести местонахождение графического файла. В этом случае к тегу `<td>` добавляется параметр `background`, в качестве значения которого устанавливается URL искомого файла.

Еще одним объектом таблицы является ее заголовок. Впрочем, он может быть и подписью, но это не меняет его сущности, так как реализация в языке HTML от этого не изменяется. Для добавления к таблице заголовка необходимо выполнить команду меню **Table/Insert/Caption**. После этого над таблицей возникает дополнительное пространство с текстовым курсором, где можно ввести наименование таблицы. Реализуется этот заголовок при помощи тегов `<caption >` и `</caption>`, между которыми будет находиться тег заголовка. Он может находиться как в верхней, так и в нижней части таблицы. Размещение заголовка реализуется при помощи параметра `valign`. В том случае, если мы желаем поместить подпись таблицы снизу, этот параметр принимает значение `bottom`. Для расположения заголовка сверху, по умолчанию, используется значение `top`. Впрочем, вручную набирать этот параметр в коде страницы не придется. Мы всегда можем использовать контекстное меню заголовка, где есть команда **Caption Properties**. Эта команда вызывает одноименное диалоговое окно (рис. 1.17).

Как видно, единственное свойство заголовка, которое мы можем принудительно устанавливать, это его положение относительно таблицы. Регулируется оно двумя радиокнопками с названиями **Top of table** и **Bottom of table**. В том случае, если мы хотим указать выравнивание названия таблицы по горизонтали, необходимо выделить его и поступить как с обычным текстом, то есть использовать кнопки выключки. При этом, весь заголовок заключается в теги объявления обычного абзаца `<p>` и `</p>` и к ним применяются

параметры выравнивания текста. Впрочем, у тега <caption> тоже есть тег выравнивания align, но различные браузеры интерпретируют его по-разному, поэтому его прямое применение нежелательно.

Рис. 1.17. Диалоговое окно **Caption Properties**

Теперь перейдем к тем инструментам работы с таблицами, которые не затрагивают непосредственно саму реализацию таблицы на языке HTML. Мы начнем со способов создания таблиц. Один способ мы уже рассмотрели ранее. Это использование команды **Insert/Table**. Однако помимо прямой вставки, таблицу можно нарисовать. Для этого используется команда меню **Table/Draw Table** или одноименная кнопка панели инструментов **Tables**. После выполнения этой команды курсор принимает форму карандаша, и им можно просто нарисовать таблицу. В том случае, если необходимо удалить ошибочно проведенную линию, следует нажать на кнопку **Eraser**, расположенную на панели инструментов для работы с таблицами. Курсор принимает форму ластика, и им можно стереть все, кроме внешних границ таблицы.

Если текст, который необходимо преобразовать в таблицу, уже набран, то используется команда меню **Table/Convert/Text to Table**. При этом активируется диалоговое окно **Convert Text To Table**, в котором необходимо указать способ разделения текстовых блоков, для того, чтобы FrontPage мог правильно разбить текстовые блоки на ячейки. На выбор предлагаются разделение по абзацам (**Paragraphs**), символами табуляции (**Tabs**) и запятыми (**Commas**). В том случае, если весь текстовый блок необходимо поместить в одну ячейку, следует выбрать вариант **None**. И, конечно, у нас есть возможность указать свой собственный символ-разделитель. Для этого необходимо выбрать альтернативу **Other**, и в поле ввода поместить необходимый символ. Впрочем, FrontPage может произвести и обратную операцию, то есть преобразовать таблицу в текст. Для этого используется команда меню **Table/Convert/Table To Text**. При этом содержимое каждой ячейки записывается в отдельном абзаце.

Теперь посмотрим, как изменить количество столбцов или строк таблицы. Удаление этих элементов производится достаточно просто. Необходимо просто выделить часть таблицы и нажать кнопку **Delete Cells** на инструментальной панели, или выполнить команду меню **Table/Delete Cells**. Выделенный блок исчезнет. Необходимо отметить, что если просто нажать клавишу <Delete> на клавиатуре, то удалится только содержимое этих ячеек. Для вставки дополнительных столбцов или строк используются кнопки **Insert Columns** и **Insert Rows**, или соответствующие команды меню.

Несколько ранее мы рассматривали случай создания ячеек, занимающих пространство, отведенное под несколько ячеек сразу. Для этих целей применяется также команда меню **Table/Merge Cells** или одноименная кнопка на панели инструментов. Команда меню и кнопка панели инструментов становятся доступными в том случае, если в таблице выделено несколько смежных ячеек. Обратная операция по разбиению одной ячейки на несколько других, производится при помощи команды меню **Table/Split Cells** или соответствующей кнопки. Для уточнения информации о том, как необходимо разбить ячейку, на экран выводится диалоговое окно **Split Cells**. Пользователь имеет возможность при помощи двух радиокнопок выбрать порядок разбиения. Разбиение по горизонтальной границе указывается альтернативой **Split into rows**, то есть "разбить на строки". Для разбиения ячейки на столбцы используется альтернатива **Split into columns**. В поле ввода необходимо ввести количество новых ячеек, на которое будет разбита исходная.

На панели инструментов **Tables** сразу после уже рассмотренных нами кнопок **Merge Cells** и **Split Cells**, находится группа кнопок, которые позволяют принудительно устанавливать вертикальное выравнивание содержимого ячеек. Как мы помним, такое выравнивание задается при помощи параметров `valign`. Эти кнопки носят названия **Align Top**, **Center Vertically** и **Align Bottom** и позволяют выравнивать содержимое по верхнему краю, уравновешивать в центре ячейки и прижимать к нижнему краю соответственно. К сожалению, аналогов этих кнопок среди команд меню **Table** нет, но, как мы помним, операцию вертикального выравнивания можно производить при помощи диалогового окна задания свойств ячеек **Cell Properties**.

Теперь на очереди инструменты, автоматически задающие размеры ячеек. В том случае, если нам необходимо сделать несколько ячеек одинаковой высоты, стоит воспользоваться кнопкой **Distribute Rows Evenly**. Естественно, все выделенные ячейки должны составлять совокупность смежных строк. Для подобной операции, которая позволяет выровнять ширину столбцов, используется кнопка **Distribute Columns Evenly**. Для этих кнопок существуют соответствующие команды меню. Необходимо учесть, что операции могут производиться только для ячеек, которые не имеют в своих HTML-тегах параметров `rowspan` и `colspan`. Есть и еще один вариант — FrontPage может сжать ячейки таким образом, что они будут максимально плотно облегать свое содержимое. Для этого используется кнопка **AutoFit**, или одноименная команда меню.

На нашей панели инструментов осталась всего одна кнопка, которая носит название **Fill Color**, и позволяет указывать цвет фона ячейки или всего выделения. Подобную возможность мы уже рассматривали при обзоре свойств ячеек.

На этом мы можем закончить обзор приемов работы с таблицами. Идем дальше.

Фреймы

Все мы сталкивались с такими приложениями, которые позволяют работать сразу с несколькими дочерними окнами. В мире World Wide Web подобная возможность реализуется как совокупность нескольких документов в пространстве одного просмотрового окна. Существует основное правило — одно окно просмотра на один документ. Не больше и не меньше. Однако всегда есть возможность обойти практически любое правило. Для этого и были придуманы страницы с фреймовой структурой. Фрейм — это место на странице, где показывается другой HTML-документ. Сам фрейм является как бы обособленным окном просмотра, содержащим свои полосы прокрутки. Границы фреймов по умолчанию могут быть изменены удаленным пользователем в процессе просмотра Web-страницы. То есть, фреймы разделены подвижной границей, сплиттером. При этом страница, на которой располагаются фреймы, сама не несет никакого содержимого. По сути, она является просто площадкой для расположения других Web-страниц. Естественно, нет смысла вставлять в исходный документ всего один фрейм. Обычно их не менее двух, но редко на странице размещается более четырех фреймов сразу, так как окна просмотра каждого документа, вставленного во фрейм, становятся достаточно маленькими, и работать удаленному пользователю с ними неудобно. Если у проектировщика страницы появилась необходимость поместить более четырех фреймов на странице, чаще всего это значит, что структура сайта спроектирована неграмотно.

Обычно в виде фреймов вставляются HTML-странички, содержащие панели навигации по сайту, а также документы с единообразно оформленными элементами. Очень часто в одном фрейме располагается форма поиска информации на сайте, а в другом фрейме выводится найденная информация. Иногда два фрейма используются для реализации уточняющего поиска. Впрочем, все примеры правильного использования фреймов можно с легкостью найти в Интернете. Как отличить правильное построение фреймовой структуры страницы от некачественного. Очень просто. Если со страницей работать удобно, внимание не рассеивается и принципы получения информации интуитивно понятны, следовательно, структура выстроена правильно. В ином случае — разработчики ошиблись. К сожалению, не всегда свое творение можно оценить объективно. Разработчик чаще всего отлично представляет себе, где лежит тот или иной документ, входящий в состав сайта, помнит все связи и структуры и от этого несколько страдает навигация по сайту и представление информации, так как ему не приходится задумываться об оптимальном поиске. Удаленный пользователь может иметь совершенно другое мнение, которое часто оказывается намного объективнее.

К счастью, FrontPage 2000 содержит несколько готовых фреймовых структур, которые давно опробованы и признаны эффективными. Идеология FrontPage 2000 не позволяет вставить фрейм в уже существующую страницу, даже если она пуста. Страницу с фреймовой структурой необходимо соз-

дать. Для этого применяется команда меню **File/New/Page**, которая активизирует диалоговое окно **New**, предназначенное для создания сайтов и страниц с использованием заранее приготовленных шаблонов и мастеров. Нас будет интересовать вкладка с названием **Frames Pages** (рис. 1.18). На ней расположены десять шаблонов для создания Web-страниц с фреймовой структурой. Рассмотрим эти шаблоны в порядке их следования.

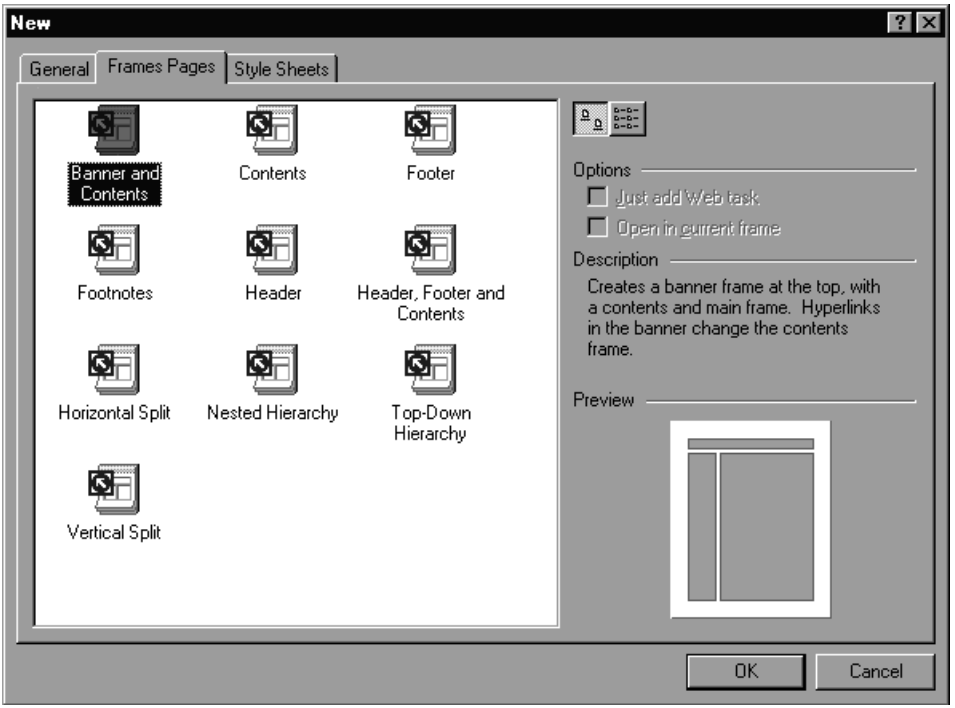

Рис. 1.18. Вкладка **Frames Pages** диалогового окна **New**

Диалоговое окно выбора шаблона достаточно удобно, ведь каждый шаблон снабжен краткой аннотацией, а в окне предварительного просмотра видна структура создаваемой страницы. Итак, самый первый шаблон с наименованием **Banner and Contents** создает страницу с тремя фреймами. В верхней части страницы расположен так называемый баннер, который обычно содержит общее наименование сайта и панель навигации. Правая часть отдана под содержание данной части сайта. Причем содержание может меняться в зависимости от того, какую часть сайта выбрал для просмотра удаленный пользователь, используя для этого навигационную панель верхнего фрейма. А основную часть окна занимает главный фрейм, в котором и отображается основной HTML-документ.

Шаблон **Contents** очень похож на предыдущий, но не содержит баннера. То есть на создаваемой странице будут размещены два фрейма. В левой части

страницы узкий фрейм предназначен для оглавления сайта, правая часть отдана под основной фрейм, который показывает выбранную пользователем страницу.

Следующий шаблон, **Footer**, фрейм с содержанием сайта размещает внизу основной страницы.

Шаблон **Footnotes** практически полностью повторяет предыдущий, но нижний фрейм с содержанием занимает несколько больше места.

Шаблон **Header** тоже относится к группе страниц с двумя фреймами, один из которых отводится под навигационную панель. В этом шаблоне верх страницы занимает узкий фрейм, в котором обычно находится панель навигации. Применяется этот шаблон, как и шаблон **Footer**, в том случае, когда нет нужды делать большое оглавление сайта, а можно обойтись обычной навигационной панелью.

Все рассмотренные нами случаи объединяет еще один шаблон с названием **Header, Footer and Contents**, который создает страницу с четырьмя фреймами. То есть помимо основного фрейма просмотра документа, создается заголовок, подвал страницы и в левой ее части — фрейм для оглавления.

Следующие два шаблона предназначены для создания страниц с двумя фреймами, содержащими независимые друг от друга документы. Шаблон **Horizontal Split** делит страницу на две части по горизонтали, а **Vertical Split** — по вертикали.

Последние два шаблона помогают создавать страницы с иерархической структурой фреймов. В одном фрейме находится оглавление или панель навигации, второй фрейм показывает документ, с помощью которого можно более точно выбрать необходимую информацию, а в основном фрейме можно просматривать искомый документ. Единственное различие между этими двумя шаблонами — порядок расположения фреймов. Так, в шаблоне **Nested Hierarchy** в левой части страницы располагается фрейм, отводимый под оглавление сайта, в правом верхнем углу расположен фрейм, в котором уточняется информация, а оставшаяся часть отведена под основное окно просмотра полученного документа. Шаблон **Top-Down Hierarchy** создает вертикальную иерархическую страницу. В верхнем колонтитуле страницы обычно размещается панель навигации, немного ниже находится уточняющий фрейм, а под ним — основной.

После выбора необходимого шаблона достаточно нажать кнопку **OK** и новая страница с фреймовой структурой почти готова. В каждом фрейме есть кнопка **Set Initial Page**, которая позволяет при помощи стандартного диалогового окна для создания гиперссылки, привязать к фрейму уже существующую страницу в качестве стартовой. Если такой страницы еще нет, ее можно создать при помощи кнопки **New Page**.

При создании Web-страниц с фреймовой структурой, к трем обычным режимам работы добавляется еще две вкладки основного рабочего окна.

Вкладка **Normal**, как обычно, показывает проектируемую страницу. Про вкладку **No Frames** необходимо рассказать подробнее. Дело в том, что далеко не все браузеры изначально поддерживали фреймовую технологию. Поэтому у каждой страницы с фреймами должен быть блок HTML-кода, в котором находится блок с сообщением, что эта страница содержит фреймы, которые данный браузер не может обработать. Так вот, на вкладке **No Frames** записывается текст, который отображается в этом случае. По умолчанию там находится строка "This page uses frames, but your browser doesn't support them". Однако нам ничто не мешает заменить ее русским текстом. Вкладка **HTML** разбивается так же, как и страница, и в каждом отдельном блоке находится HTML-код документа, который ассоциирован с этим фреймом. А на вкладке **Frames Page HTML** находится HTML-код самой страницы, содержащей фреймы. Вкладка **Preview** не меняет своего предназначения, и используется для предварительного просмотра.

Перейдем на вкладку **Frames Page HTML**, и посмотрим, при помощи какого кода реализуется фреймовая технология:

```
<frameset rows="*,*">
  <frame name="top" src="1.htm">
  <frame name="bottom" src="3.htm">
  <noframes>
  <body>
  <p>Эта страница содержит фреймы, но ваш
  браузер не поддерживает их.</p>
  </body>
  </noframes>
</frameset>
```

Итак, как мы видим, вся структура страницы заключается между тегами `<frameset>` и `</frameset>`. У тега `<frameset>` в данном случае есть параметр `rows`, которому приписано значение `"*,*"`. Этот параметр задает высоту фреймов. На этой странице находятся два горизонтальных независимых фрейма, поэтому в параметре `rows` указывается высота каждого фрейма. Высота может указываться как в процентах от высоты окна просмотра, так и в пикселах, но в нашем случае использовались символы звездочки. Этот символ используется в том случае, когда фрейм должен занять все доступное ему пространство. Так как в значении параметра находятся две звездочки, то изначально фреймы занимают одинаковое пространство. А если бы мы указали, скажем, `rows="20%,*"`, то первый фрейм занял бы пространство, по высоте равное двадцати процентам от высоты окна просмотра браузера, а второй фрейм — все оставшееся. В том случае, если бы мы разместили два фрейма, разделенные вертикальной страницей, то вместо параметра `rows` использовался бы параметр `cols`.

После объявления структуры фрейма поставлены два тега, реализующие отдельные фреймы. Как видно, типичная конструкция выглядит как `<frame name="top" src="1.htm">`. Тег `<frame>` содержит параметр `name`, задающий

имя фрейма, и параметр `src`, в котором указывается URL подключаемого документа. Так как имена фреймов указываются, то у создателя Web-страниц появляется возможность выбирать фрейм, в который будет загружен документ.

После объявления тегов располагается секция кода, которая используется в том случае, если браузер удаленного пользователя не поддерживает фреймы. Как видно, эта часть ограничивается тегами `<noframes>` и `</noframes>`. Внутри этих тегов объявляется тело HTML-документа тегом `<body>`, и внутри него устанавливается строка извещения. Все достаточно легко.

Теперь перейдем к свойствам отдельно взятого фрейма. Как известно, прежде чем редактировать свойства какого-либо объекта, его необходимо выделить. FrontPage 2000 считает активным именно тот фрейм, в котором на данный момент находится текстовый курсор. При этом граница активного фрейма дополнительно выделяется синим цветом. Для установки свойств фрейма используется диалоговое окно **Frame Properties** (рис. 1.19), которое активизируется одноименной командой контекстного меню или командой основного меню **Frame/Frame Properties**.

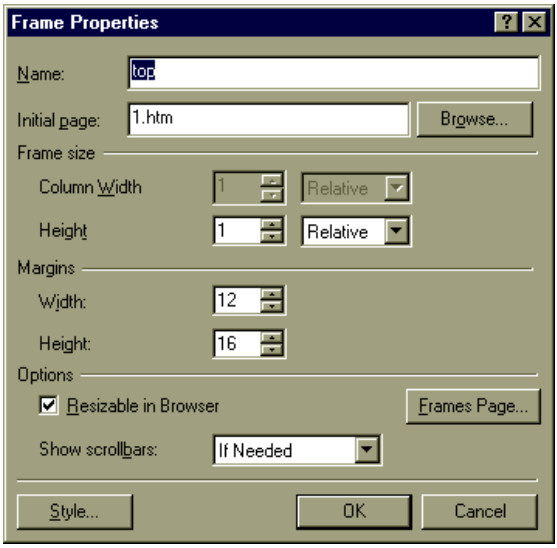

Рис. 1.19. Диалоговое окно **Frame Properties**

Как нетрудно заметить, поле ввода **Name** позволяет изменить предварительно установленное имя фрейма, а поле **Initial page** — указать Web-страницу, показываемую изначально в этом фрейме. Помочь в этом может находящаяся рядом кнопка **Browse**. Дальше все не так элементарно, а, значит, немного интереснее.

Элементы управления, размещенные в блоке **Frame size**, используются для указания размеров фрейма. В том случае, если фрейм должен занимать всю

длину страницы по вертикали или горизонтали, соответствующее поле ввода блокируется, так как изменение соответствующего размера фрейма просто запрещено. Итак, в полях **Column Width** и **Height** указываются ширина и высота фрейма соответственно. Но самое интересное не в этом. Дело в том, что рядом с каждым полем расположен выпадающий список, в котором указывается единица измерения. Как обычно, здесь есть проценты от размера окна просмотра и пикселы. Соответственно выбранной единице измерения будет изменяться значение параметров `rows` и `cols` в теге `<frameset>`. Если выбраны пикселы, то будет указано обычное число, если проценты, то добавится значок процента. Но в выпадающем списке есть и еще один вариант — альтернатива **Relative**, которая выбирается в случае указания пропорций. Она используется по умолчанию. Возьмем, например, уже рассматривавшийся нами вариант страницы с двумя фреймами, размещенными один над другим, и занимающими всю длину страницы по ширине. Эта страница создается по шаблону **Horizontal Split**. Изначально оба фрейма имеют одинаковую высоту. Если мы попробуем изменить свойства верхнего фрейма, то мы увидим, что значение в поле **Column Width** недоступно для редактирования, а в поле **Height** выставлена единица. Теперь, если мы установим относительный размер по высоте для этого фрейма, равный двум, то он займет две трети высоты окна просмотра, а нижний фрейм, соответственно, одну треть. А если мы посмотрим на страницу **Frames Page HTML**, то увидим, что тег `<frameset>` приобрел вид `<frameset rows="2*,*">`. Теперь мы знаем, как указывать относительные размеры фреймов, используя средства HTML.

В группе **Margins** находятся два поля ввода — **Width** и **Height**, которые позволяют указывать величину отступа содержимого фрейма от его границ. Величина отступа указывается в пикселах. Если в поле **Width** мы установим значение ширины отступа, равное пятидесяти пикселам, а в поле **Height** установим высоту отступа, равную двадцати пикселам, то в теге `<frame>`, объявляющем соответствующий фрейм, мы обнаружим добавленный блок параметров `marginwidth="50" marginheight="20"`. Как видно, ширина отступа указывается с помощью параметра `marginwidth`, а высота — в `marginheight`.

Последняя группа элементов управления носит название **Options**. Флажок **Resizable in Browser** служит для установки режима отображения фрейма, при изменении размеров фрейма удаленным пользователем во время просмотра Web-страницы. В случае, если этот флажок выставлен, удаленный пользователь имеет возможность передвигать границы фреймов. Если же данная возможность разработчиком принудительно отключена, то границы замораживаются, и пользователь не может изменять размеры фреймов. По умолчанию размеры фреймов могут изменяться. Если границы заморожены, в теге, объявляющем данный фрейм, появляется параметр `noresize`. В этом же блоке элементов управления находится выпадающий список **Show scrollbars**. Он предназначен для установки режима отображения полос прокрутки содержимого фрейма. По умолчанию действует режим **If Needed**, то есть полосы прокрутки появляются только тогда, когда в них возникает необходимость. Например, пользователь

уменьшил высоту фрейма, и содержимое, ранее умещавшееся по вертикали целиком, теперь отображается не полностью. В этом случае появляется вертикальная полоса прокрутки. Также в списке находится значение **Always**, которое указывает на то, что полосы прокрутки наличествуют во фрейме постоянно, вне зависимости от их необходимости, и значение **Never**, которое прямо запрещает отображение полос прокрутки для этого фрейма. В HTML это регулируется при помощи параметра `scrolling` тега `<frame>`. У этого параметра может быть три значения: `auto`, `yes` и `no`, которые указывают на режимы появления по необходимости, постоянного наличия и отсутствия полос прокрутки содержимого фрейма соответственно.

В рассматриваемом нами диалоговом окне осталась всего одна кнопка, которой мы еще не уделили внимания. Она носит название **Frames Page**, и предназначена для установки свойств не отдельного фрейма, а всей страницы с фреймовой структурой. При нажатии на эту кнопку активизируется диалоговое окно **Page Properties** на вкладке **Frames**. На всем пространстве этого, не самого маленького диалогового окна, находится всего два элемента управления. Первый — поле ввода **Frame Spacing**, в котором указывается ширина сплиттера, разделяющего фреймы. Обратите внимание, что это свойство принадлежит не отдельному фрейму, или их группе, а всей странице сразу. То есть, ширина границы между фреймами будет везде одинаковой. В поле ввода эта ширина границы указывается, естественно, в пикселах. В HTML ширина сплиттеров, разделяющих фреймы, регулируется параметром `framespacing`, который в свою очередь принадлежит тегу `<frameset>`. А второй и последний орган управления, расположенный на этой вкладке, является обычным флажком с именем **Show Borders**. То есть, если он будет установлен, то границы между фреймами будут отображаться. Если его убрать, то при просмотре Web-страницы границ, как таковых, видно не будет. Если при этом отображаются полосы прокрутки, то получается достаточно забавный образец дизайна, который, тем не менее, считается неудачным. Поэтому не стоит скрывать границы фреймов без особых на то причин. Режим отображения границ используется в HTML по умолчанию, значит, он считается наиболее приемлемым. В случае, если отображение границ принудительно отключено, FrontPage 2000 добавляет к тегу `<frameset>` строку параметров `border="0" frameborder="0"`. Параметр `border` указывает толщину рамки фрейма в пикселах, а параметр `frameborder` используется для указания режима отображения трехмерной рамки фрейма. В том случае, если задано нулевое значение параметра `border`, рамка не создается. Комбинация этих двух параметров действительно убирает границы между фреймами. Но на параметр `framespacing`, который задает в пикселах отступ содержимого фрейма от его границы, эта связка не действует. Пусть границу и не видно, но она есть, и ширина расстояния между фреймами будет именно такая, как указано в этом параметре. На этом мы заканчиваем обзор свойств фреймов и страниц, содержащих эти структурные единицы, и переходим к рассмотрению команд меню для работы с фреймами.

В пункте меню **Frames** есть всего лишь две команды, которые реально производят какие-либо действия с выбранным фреймом. Так, команда **Split Frame** позволяет разбить выбранный фрейм на несколько других. Эта команда активизирует одноименное диалоговое окно, которое по своей структуре сильно напоминает такое же, позволяющее разбивать одну ячейку таблицы на несколько новых. В нем необходимо уточнить, как будет разбиваться фрейм, по горизонтали или по вертикали. Если планируется разбить его на две колонки, проведя вертикальную границу, необходимо выбрать радиокнопку **Split into Columns**. Для разбиения фрейма на два горизонтальных окна, используется альтернатива **Split into Rows**. После указания вида разбиения, один из новообразованных фреймов наследует страницу, отображаемую в его фрейме-прародителе, а для другого фрейма загружаемую в начале просмотра документа страницу необходимо задать.

Для удаления выделенного фрейма используется команда меню **Delete Frame**. Если этот фрейм не является единственным на создаваемой Web-странице, то он будет немедленно удален.

Следующий блок команд пункта меню **Frames** предназначен для работы с HTML-документами, расположенными в этих фреймах. Так, команда **Open Page in New Window** создает новое окно, занимающее всю рабочую область FrontPage 2000, и помещает в него HTML-документ, который размещался в текущем фрейме на момент выполнения команды меню. Это чрезвычайно удобно для привычной и комфортной работы по проектированию Web-странички. Команды **Save Page** и **Save Page As** предназначены для обычного сохранения и сохранения с измененным именем того HTML-документа, который находится в выделенном фрейме. Обратите внимание, сохраняется не вся страница с фреймовой структурой, а отдельный документ из фрейма.

Последнюю команду с именем **Frame Properties** мы уже рассматривали в этом разделе.

На этом обзор возможностей FrontPage 2000 для работы с фреймами заканчивается. Их, может быть, не так уж и много, но они, пожалуй, полностью исчерпывают нужды фреймовой технологии. Самое главное — это создать правильную фреймовую структуру страницы, и активно использовать имена фреймов в гиперссылках, для того, чтобы управлять правильным отображением документов.

Активные элементы

Под активными элементами мы будем подразумевать те объекты, которые чаще всего невозможно реализовать средствами HTML, но, тем не менее, весьма широко применяемые в оформлении Web-страниц. Это могут быть кнопки, которые меняют свой цвет, когда курсор мыши попадает на них, блоки текста, которые, прежде чем попасть на положенное место, демонст-

рируют мультипликационные элементы, интерактивные блоки данных, такие, как таблицы с автоматическим пересчетом данных. Для создания активных элементов необходимо использовать программирование, а, значит, детали их реализации выходят за оговоренные ранее рамки этой книги. Конечно, мы рассмотрим классификацию этих объектов, узнаем, в чем их сходства и различия, но самостоятельно сделать свой активный элемент, опираясь на эту книгу, невозможно. Если же необходимо использовать какой-либо активный элемент, который не входит в коллекцию FrontPage, его можно отыскать в Интернете. Причем, как мы знаем, специфика Интернета такова, что в одном месте эти элементы будет предложено купить, в другом их предложат в аренду, а где-нибудь еще совершенно законным образом предоставят в бесплатное пользование. Эта ситуация возможна практически всегда. Главное — провести полный поиск и методично пройти по всем найденным ссылкам. Но прежде чем запускать свой браузер для этого поиска, следует ознакомиться с тем списком активных элементов, который предлагает нам FrontPage 2000.

Для начала мы рассмотрим эффекты из группы DHTML (Dynamic HTML). Они применяются для анимации текстовой строки. Для выбора какого-либо эффекта лучше всего использовать инструментальную панель **DHTML Effects**, которая активизируется при помощи стандартной команды **View/Toolbars** или **Format/Dynamic HTML Effects**. Итак, мы выделяем какую-либо строку, и переходим к инструментальной панели. При этом на ней становится доступным для использования выпадающий список **On**, позволяющий выбрать событие, при наступлении которого будет проявляться выбранный эффект. Следует отметить, что далеко не все эффекты могут быть использованы в каждом случае. В качестве инициирующих событий нам предлагаются значения **Click**, то есть щелчок мышью на данной строке, **Double click** — двойной щелчок, **Mouse over** — появление мышиного курсора над строкой, и **Page load**, то есть при загрузке страницы браузером удаленного пользователя. Именно последнее событие позволяет использовать каждый эффект DHTML, который предусмотрен в FrontPage.

Выбор конкретного эффекта производится в выпадающем списке **Apply**, который становится доступен после того, как пользователь установил событие, инициирующее запуск анимации. Первый эффект с наименованием **Drop down** выбрасывает все слова, входящие в строку поочередно из-за верхнего края окна просмотра браузера или из-за верхней границы фрейма, в котором размещен этот HTML-документ.

В том случае, если проектировщик Web-страницы выбирает эффект **Elastic**, то активизируется третий и последний выпадающий список инструментальной панели **DHTML Effects**. В нем можно устанавливать отдельные параметры выбранных эффектов. **Elastic** заставляет всю строку выскакивать из-за границы окна просмотра или фрейма, лететь к своему месту, постепенно замедляя скорость движения, проскакивать по инерции мимо него, и плавно двигаться в обратном направлении к предустановленной позиции. Что ж,

на то он и **Elastic**. В третьем выпадающем списке мы можем выбрать ту границу, из-за которой строка будет появляться. Значение **From right** заставит нашу строку двигаться по горизонтали, появляясь из-за правой границы окна, а значение **From bottom** выведет строку из-за нижнего края окна, и заставит ее двигаться по вертикали.

Если применить эффект **Fly in**, то строка будет следовать к своей позиции, вылетая из указанного места. Место старта строки и особенности эффекта указываются в последнем выпадающем списке инструментальной панели **DHTML Effects**. Для эффекта **Fly in** третий список содержит действительно много значений. И когда я говорю "много", я имею в виду действительно много. Значения **From right**, **From bottom**, **From left** и **From top** указывают в качестве места старта полета строки правую, нижнюю, левую и верхнюю границы окна просмотра соответственно. Значения **From bottom-left**, **From bottom-right**, **From top-right** и **From top-left** позволяют использовать уже не границы окна, а его углы. Левый нижний угол, правый нижний, правый верхний и левый верхний угол соответственно. Значение **Along corner** позволяет "скрестить" эффект **Fly in** с уже рассматривавшимся нами эффектом **Elastic**. При старте эффекта, строка сначала поднимается из-за нижней границы окна просмотра по вертикали до нужного места в стиле эффекта **Elastic**, со всеми его особенностями, то есть торможение разогнавшейся строки, маленький промах, плавный откат на нужное место, и лишь потом по горизонтали направляется в необходимую позицию. Чрезвычайно интересный и нестандартный вариант. И последние два варианта, то есть **From top-right by word** и **From bottom-right by word** выбрасывают не строку, а ее слова по очереди из правого верхнего и правого нижнего угла окна просмотра соответственно.

Эффект **Hop** также разбивает строку на отдельные слова и с ними уже производит затейливую комбинацию. Сначала из-за правого края окна по дуге, выгнутой вверх, выбрасывается первое слово. В конце своего дугообразного полета это слово оказывается на изначально предусмотренном для него месте. Потом уже из-за этого слова выбрасывается следующее, и так далее Очень занимательно. Достаточно похож на него и сосед по списку — эффект **Spiral**. Этот эффект, правда, оперирует целой строкой, как и большинство других эффектов DHTML. При его применении строка вылетает из правого верхнего угла браузера, и по спирали приходит на свое место.

Эффект **Wave** разбивает строку на слова, и каждое слово по дуге прилетает на свое место. Это было бы очень похоже на эффект **Hop**, если бы не одно обстоятельство. Поочередно используются дуги, выгнутые вверх и вниз. Поэтому и образуется волнообразный эффект.

Эффект **Wipe** очень мало похож на остальные эффекты. Он не заставляет текст совершать какие-либо перемещения. Он просто плавно "проявляет" его. Если в установках этого эффекта выбрано значение **Left to right**, то сначала появится первый символ, за ним второй, третий, и так далее до самого конца строки. Причем появление новых символов является непрерывным, а не

дискретным. Эффект проявляется очень плавно, создавая реалистическое ощущение, будто строка была закрыта полосой бумаги цвета фона, а потом эту полоску медленно убирают, и из под нее появляются буквы. Значение **Top to bottom** "проявляет" строку по вертикали сверху вниз, а значение **From middle** показывает строку, начиная от ее середины к краям по горизонтали.

И последний из эффектов, применяемых при загрузке страницы, носит название **Zoom**. Уже по названию можно предположить механизм его действия. Скорее всего, предположение окажется верным. Этот эффект постепенно увеличивает строку, начиная с самого маленького размера символов, а затем плавно доводя ее до необходимого размера. Впрочем, эффект может быть и обратным. Изначально может быть показана очень большая строка, а потом ее размер уменьшится до изначально заданного. Варианты работы этого эффекта указываются в третьем списке. Параметр **In** производит увеличение строки, а параметр **Out** — ее уменьшение.

Теперь перейдем к эффектам, которые могут воспроизводиться при одинарном или двойном щелчке мыши на строке текста. Первый из них носит наименование **Fly out**. Он действует подобно эффекту **Fly in**, но в другую сторону, то есть теперь строка не прилетает на свое место, а наоборот, улетает с экрана. Достаточно произвести щелчок мышью, одинарный или двойной, в зависимости от выбранного события и строка, по которой щелкнули, исчезнет из окна просмотра браузера. В свойствах эффекта можно указать направление полета строки. Все эти параметры уже знакомы нам по эффекту **Fly in**.

Второй эффект, который может использоваться также для работы с событием **Mouse over**, носит название **Formatting**. Кстати, это единственный эффект, связанный с этим событием. Он позволяет изменить внешний вид строки. В нем применяется изменение шрифта и/или рамки, ограничивающей эту строку. Для выбора нового шрифта, с помощью которого будет отображаться строка при наступлении заданного события, в третьем списке выбирается альтернатива **Choose font**, которая активизирует стандартное диалоговое окно. Для установки границы строки используется значение **Choose border**, активизирующее диалоговое окно **Borders and Shading** (рис. 1.20). Как легко заметить из названия, окно предназначается для установки рамок и теней.

Вкладка **Borders**, которую мы видим на рисунке, практически ничем не ограничивает фантазию дизайнера. В блоке **Setting** находится три образца возможных типов рамок. Вариант **None**, используемый по умолчанию, не предусматривает наличия каких-либо рамок вообще. Тип **Box** позволяет устанавливать единую рамку для всей строки. Тип ограничивающей линии можно выбрать из списка **Style**. Рядом с этим списком находится блок предварительного просмотра создаваемой страницы. Как видно, рядом с образцом находятся четыре кнопки, каждая из которых управляет отображением границы одной из сторон блока. В принципе, находясь в режиме **Box**, можно принудительно отключить

отображение одной или нескольких границ, но при этом мы автоматически перейдем в режим **Custom**, который позволяет устанавливать тип ограничивающей линии для каждой из сторон блока. Для этого необходимо выбрать стиль линии и нажать на кнопку, отвечающую за необходимую границу. Цвет линий выбирается по правилам указания стилей, а сам выбор производится при помощи выпадающего списка **Color**. Под этим списком находится еще одно поле ввода **Width**, которое позволяет указывать ширину одной или нескольких ограничивающих линий. То есть это диалоговое окно предоставляет проектировщику полный контроль над внешним видом будущей рамки. Остается посоветовать только не увлекаться разнообразием, а стараться соблюдать единый стиль, принятый для оформления страницы или сайта. Последний блок элементов управления на этой вкладке с названием **Padding** содержит четыре поля ввода, в которых можно указать отступы от границ до содержимого блока в пикселах.

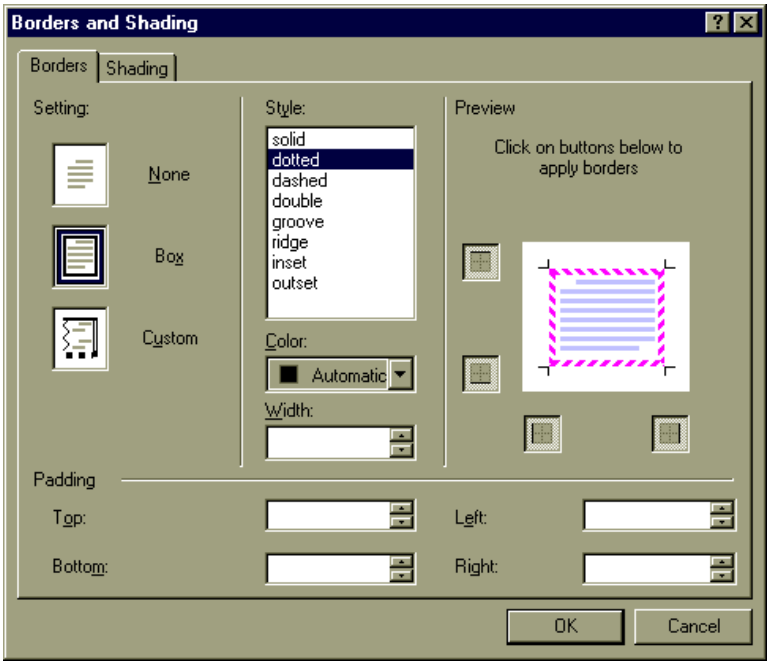

Рис. 1.20. Вкладка **Borders** диалогового окна **Borders and Shading**

Теперь перейдем к вкладке **Shading** (рис. 1.21). Она предназначена для установки цвета тени рамки и задания фона блока. Список **Background color** предназначен для установки цвета фона строки. Список **Foreground color** позволяет указывать дополнительный цвет рамки, который применяется в случае трехмерного ее выделения.

С помощью блока **Patterns** создается заполнение фона строки. В поле **Background picture** вводится наименование графического файла, изображение из

которого будет применяться в качестве заполнителя фона. Остальные поля ввода предназначены для установки координат, правил заполнения фона рисунком и выравнивания этого рисунка. На этом список эффектов DHNML заканчивается, и нам остается рассмотреть две последние кнопки панели **DHTML Effects**. Кнопка с наименованием **Remove Effect** становится доступной для пользователя в тот момент, когда текстовый курсор появляется в строке, которая выводится с применением того или иного эффекта DHTML. В этом случае нажатие кнопки **Remove Effect** позволяет снять установленный эффект. Иногда, правда, возникает другая проблема — отыскать все строки, к которым были применены динамические эффекты. В этом нам может помочь кнопка с именем **Highlight Dynamic HTML Effects**. Если она нажата, то все строки, которые при просмотре документа удаленным пользователем будут отображены с применением какого-либо анимационного эффекта, будут подсвечены голубым цветом. Если кнопку отжать — подсветка исчезнет. На этом обзор технологии DHTML заканчивается, и мы переходим к другим активным элементам, поддерживаемым FrontPage 2000.

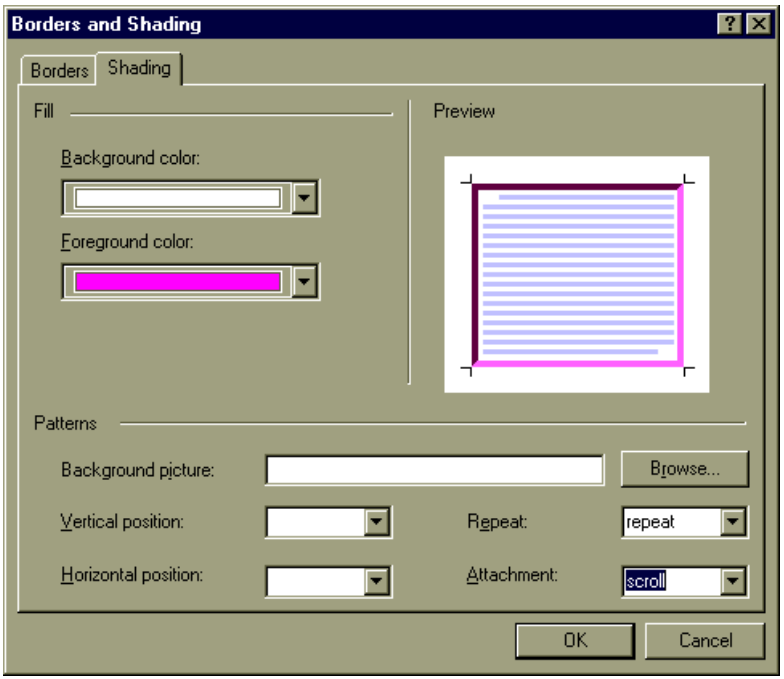

Рис. 1.21. Вкладка **Shading** диалогового окна **Borders and Shading**

Вставка активного элемента осуществляется при помощи команды **Insert/Component**, после выполнения которой, FrontPage предлагает на выбор список компонентов. Первый из них носит название **Office Spreadsheet**. Он представляет собой подобие листа электронной таблицы Microsoft Excel.

Этот активный элемент реализован при помощи технологии ActiveX. Эта технология, на мой взгляд, является одной из наиболее впечатляющих разработок фирмы Microsoft. При всем неоднозначном отношении к ней, необходимо отметить, что без технологии ActiveX разные компьютеры не могли бы так легко общаться друг с другом, как сейчас. Давайте посмотрим на историю появления стандарта ActiveX, и попробуем уяснить, что же он из себя представляет.

С появлением Windows, которая претендовала на новую парадигму единой среды для приложений, встал вопрос о связи этих приложений между собой. Разнородные данные необходимо было связывать друг с другом, и импортировать из одних программ в другие. Пользователям было необходимо, например, вставлять таблицы из Excel в текст документа, набранного в Word. Причем, чтобы функциональность таблиц не терялась. Это ведь единая система, правда? Честно скажем, задача была не из простых, но решение нашлось. Была использована концепция объектов. Отныне таблицы, тексты, диаграммы и прочие блоки данных могли сохраняться в виде объектов, которые несли в себе не только данные и правила их форматирования и отображения, но еще и правила работы с этими данными. На основе этих принципов вырос общеизвестный стандарт OLE (Object Linking and Embedding), то есть стандарт внедрения и связывания объектов. Чужеродный объект мог внедряться в данные другого приложения, а для редактирования использовалась его связь с приложением-родителем, которое могло добавлять свои методы работы, меню и инструментальные панели в принимающее приложение. Естественно, изначально этой возможностью в полной мере пользовались только программы, созданные самой Microsoft. Это и неудивительно, ведь первая реализация этого стандарта была очень запутанной и в критических ситуациях часто давала сбои. Поэтому вскоре была предложена версия стандарта OLE 2.0. Именно в этой версии была в полной мере использована компонентная модель объекта — COM (Component Object Model). Модель COM имела гораздо больше возможностей, чем внедрение и связывание. По существу, COM является перечнем функций, которые может осуществлять создаваемый объект. Уже из этой модели были созданы активные элементы ActiveX. Для их работы уже не нужно было наличие на машине той программы, которая изначально породила данные. Так, если мы используем элемент ActiveX, который реализует работу с электронной таблицей, мы можем не заботиться о наличии у удаленного пользователя программы Microsoft Excel. ActiveX может сам отображать, форматировать и обрабатывать свои данные. Причем, все современные браузеры поддерживают эту технологию. Работа с ней происходит следующим образом. Каждый элемент ActiveX имеет свой номер версии и идентификационный уникальный номер класса, которые указываются в тексте HTML-кода. При загрузке страницы, содержащей элемент ActiveX, система удаленного пользователя просматривает список зарегистрированных в ней объектов ActiveX. В том случае, если элемент с данным идентификационным номером уже есть в системе, то сам элемент не загружается. Грузится только

содержимое страницы. А потом к ней подключается уже имеющийся в системе элемент. В том случае, если элемент с данным идентификационным номером не прописан в системе, то есть страница с этим элементом загружается впервые, или версия элемента ActiveX, установленного на странице, больше номера версии элемента, прописанного в системе, элемент загружается полностью и прописывается в реестре. Система продумана достаточно хорошо.

Но вернемся к нашему листу электронной таблицы. Этот элемент, как и два последующих, является одним из так называемых Microsoft Office Web Components, наряду с Office PivotTable и Office Chart. Эти элементы являются стандартными блоками Microsoft Office 2000, поэтому при его инсталляции их необходимо явно установить на жесткий диск. В ином случае, не удастся подготовить к публикации сайт, содержащий эти элементы. Самое интересное состоит в том, что в отличие от нелокализованного FrontPage, эти элементы полностью русифицированы и справка для них, естественно, поставляется на русском языке.

Элемент **Office Spreadsheet** помимо стандартного поля электронной таблицы, содержит еще и панель инструментов. Первая кнопка с изображением эмблемы Microsoft Office просто выводит окно **About** с информацией об этом элементе и держателе его лицензии. Следом за ней расположена кнопка отмены последнего действия. Потом следует триада стандартных кнопок для работы с буфером обмена, которые позволяют вырезать выделение, копировать его в буфер обмена и вставлять содержимое буфера в выбранное место таблицы. Тут же, рядом расположена кнопка, предназначенная для автосуммирования. Следом за ней расположены три кнопки, отвечающие за сортировку и фильтрацию содержимого ячеек. Первые две — это сортировка в прямом и обратном порядке, третья включает средство автофильтра. Кнопка с изображением эмблемы Excel и карандаша позволяет экспортировать выделенные ячейки листа в Microsoft Excel. Очень удобный для удаленного пользователя инструмент, позволяющий перенести все данные и формулы к себе на лист, и в спокойной обстановке разобраться с полученной информацией. Следующая кнопка предназначена для вызова окна, предназначенного для установки свойств листа электронной таблицы (рис. 1.22). Здесь нас ожидает еще один приятный сюрприз. Web-элементы Office локализованы, а, значит, и диалоговое окно будет содержать русскоязычную информацию. Как видно из рисунка, мы можем в полном объеме управлять шрифтовым оформлением содержимого ячеек, указывать их формат, сливать в одну и разбивать на несколько, указывать тип границы, управлять выравниванием содержимого относительно границ ячеек, указывать требуемый формат чисел и выбирать те части активного элемента, которые будут показаны. В это число могут входить заголовок, панель инструментов, наименования строк и столбцов, а также сетка, разделяющая ячейки.

Лист **Office Spreadsheet** позволяет вводить числовые и текстовые значения в ячейку, а также формульные выражения. К сожалению, помимо автосум-

мирования, никаких других средств, облегчающих ввод формул, не предусмотрено. Казалось бы, и эту проблему можно решить. Ввести все данные в Microsoft Excel, пользуясь всеми его прелестями, а затем перенести их в пространство Office Spreadsheet. Но не тут-то было. Копирование данных в табличной форме, находившихся в буфере обмена, чаще всего удаляет со страницы активный элемент **Office Spreadsheet**, и вместо него создает обычную HTML-таблицу с цифрами, взятыми из таблицы Excel. Поэтому ввод формул приходится осуществлять вручную.

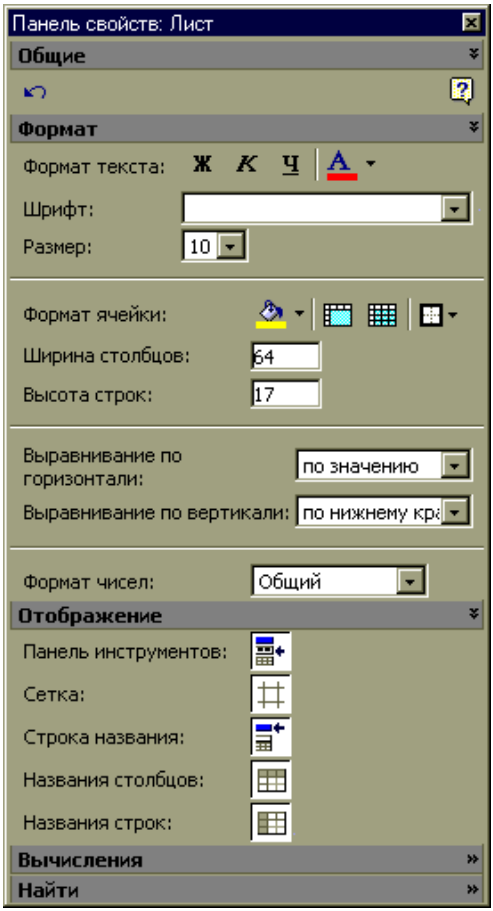

Рис. 1.22. Диалоговое окно для установки свойств встроенного активного элемента электронной таблицы

На самом деле, владельцу комплекта Microsoft Office 2000 не стоит злоупотреблять использованием активных элементов **Office Spreadsheet**, **Office PivotTable** и **Office Chart**. Второй элемент представляет собой сводную таблицу со средствами анализа данных, а третий — обычную диаграмму. Дело в том, что все программы из семейства Microsoft Office 2000 позволяют со-

хранять свои данные в HTML-формате, а значит, и размещать их на наших Web-страницах будет достаточно просто. Эту возможность мы рассмотрим в третьей части. А пока от Microsoft Office Web Components мы перейдем к рассмотрению остальных активных элементов.

При помощи команды меню **Insert/Component/Banner Ad Manager** или кнопки на основной панели инструментов **Insert Component** можно вставить плакат (баннер), содержащий ссылку на какой-либо Web-ресурс. При исполнении этой команды активизируется диалоговое окно **Banner Ad Manager Properties** (рис. 1.23). Его также можно вызвать для изменения свойств уже созданного баннера-ссылки посредством команды контекстного меню или пункта меню **Format/Properties**.

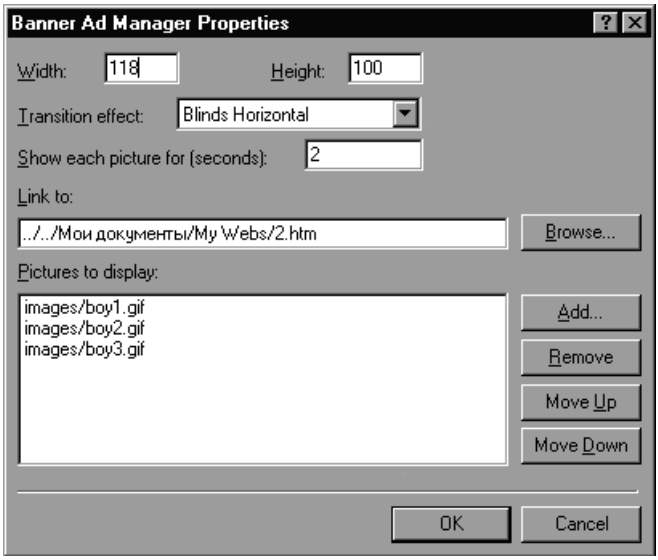

Рис. 1.23. Диалоговое окно **Banner Ad Manager Properties**

В поле ввода **Link to** указывается URL того документа, на который устанавливается ссылка. В списке **Pictures to display** находятся имена графических файлов, которые будут отображаться на баннере. Добавление и удаление файлов производится при помощи кнопок **Add** и **Remove**. А изменение порядка следования рисунков производится при помощи кнопок **Move Up** и **Move Down**. Эти изображения могут просто сменять друг друга, причем каждое будет отображаться то количество секунд, которое установлено в поле ввода **Show each picture for (seconds)**. Эти изображения могут либо просто заменять друг друга, либо для этого будет использоваться какой-либо эффект, указанный в выпадающем списке **Transition effect**. Высота и ширина баннера указываются в полях **Height** и **Width** соответственно.

Этот элемент реализован в виде Java-скрипта, то есть сценария, написанного с использованием команд на языке Java. Язык Java изначально был заду-

ман как средство создания межплатформенных приложений. То есть, написав приложение один раз, его можно без перекомпиляции использовать на различных программных и аппаратных платформах. Естественно, эта идеология как нельзя лучше подошла для нужд Интернета, так как никогда заранее неизвестно, какой платформой оснащен удаленный пользователь. Главное, чтобы его браузер поддерживал язык сценариев Java. Также, в качестве альтернативы или дополнения могут использоваться скрипты, написанные на языке Visual Basic. Но нам не придется заниматься их написанием. Мы воспользуемся теми сценариями, которые уже есть в коллекции FrontPage 2000.

Java-скрипты создаются в виде обычных текстовых файлов. Браузер читает соответствующий файл, и сам его интерпретирует, выполняя необходимые действия.

Следующий активный элемент, предлагаемый проектировщику, носит наименование **Hit Counter**. Он, пожалуй, наиболее часто используется и на русскоязычных страницах именуется счетчиком посещений. При выборе соответствующей команды меню активизируется диалоговое окно **Hit Counter Properties** (рис. 1.24).

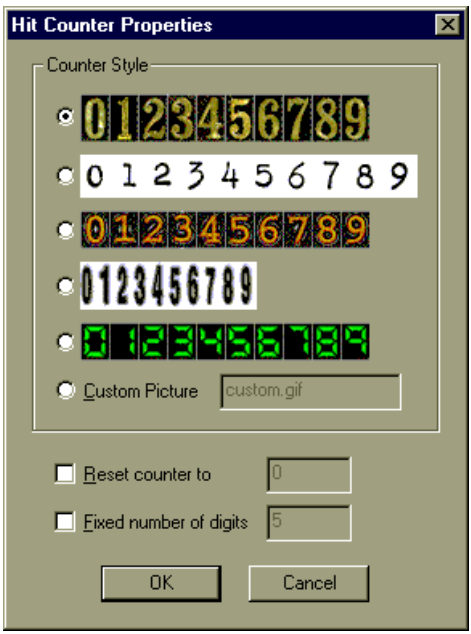

Рис. 1.24. Диалоговое окно **Hit Counter Properties**

В группе **Counter Style** можно выбрать одно из пяти различных начертаний цифр, использующихся в счетчике, либо выбрать альтернативу **Custom Picture** и в поле ввода указать имя графического файла в формате GIF, содержащего рисунок с другим начертанием цифр. Если необходимо указать стартовое значение счетчика, которое отличается от нуля, надо включить позицию **Reset counter to** и в соответствующем поле ввода указать выбран-

ное стартовое значение. По умолчанию для счетчика посещений используется пять цифр. Но если хочется изменить это количество, например, увеличить (гигантомания?), то необходимо включить альтернативу **Fixed number of digits** и указать требуемое количество цифр в соответствующем поле ввода.

Следуя по блоку меню по порядку, мы встречаем кнопку с наименованием **Hover Button**. Эта кнопка используется для красивого оформления гиперссылки, которая навешена на эту кнопку. "Красивость" заключается в том, что при появлении на кнопке курсора мыши, ее вид изменяется в зависимости от выбранного эффекта. Говорят, что люди намного чаще и охотнее нажимают на подобные кнопки. Для редактирования и установки ее свойств используется диалоговое окно **Hover Button Properties** (рис. 1.25).

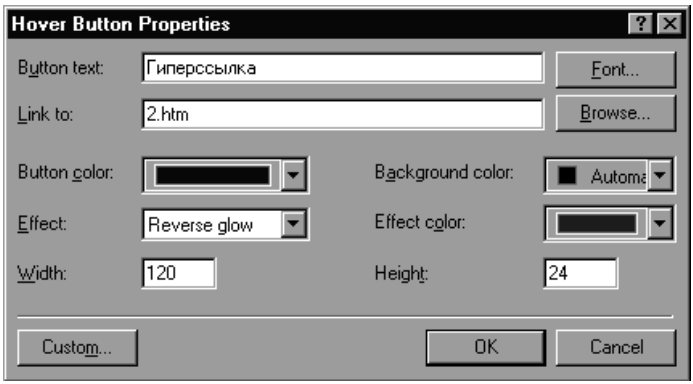

Рис. 1.25. Диалоговое окно **Hover Button Properties**

Текст, располагаемый на кнопке, вводится в поле **Button Text**. Для установки шрифтового оформления этого текста используется кнопка **Font**. Установка гиперссылки производится при помощи кнопки **Browse**, URL вписывается в поле **Link to**. В выпадающем списке **Button color** устанавливается основной цвет кнопки, а в списке **Background color** — цвет фона. Правда, его практически не будет видно, если только цвет кнопки или цвет, порождаемый эффектом, не будут прозрачными. Порождаемый эффектом цвет можно выбрать в списке **Effect color**. Но самое интересное в этом окне — список **Effect**, в котором можно выбрать то преобразование, которое произойдет с кнопкой при попадании на нее курсора мыши. Эффекты **Color fill** и **Color average** заменяют основной цвет кнопки дополнительным, выбранным в списке **Effect color**. Эффекты **Glow**, **Reverse glow** и **Light glow** создают градиентную заливку кнопки из этих двух цветов. А последние два эффекта с наименованиями **Bevel out** и **Bevel in** вообще не используют дополнительного цвета. Они имитируют трехмерность кнопки, немного повышая или понижая надпись на ней при появлении курсора мыши. Поля **Width** и **Height**, как обычно, предназначаются для установки ширины и высоты кнопки. Но это еще не все. На кнопке может быть установлен фоновый рису-

нок, меняющийся при попадании курсора на кнопку, а также к ней могут быть привязаны звуковые файлы, проигрывающиеся в момент появления курсора и нажатия. Для установки этих свойств необходимо нажать на кнопку **Custom**. При этом на экране появится одноименное диалоговое окно. В группе полей ввода **Play sound** устанавливаются звуковые файлы. В поле **On click** указывается имя звукового файла, который будет воспроизводиться каждый раз при нажатии на кнопку, а в поле **On hover** — файл, используемый при появлении курсора мыши на кнопке. Блок **Custom** предназначен для установки графических изображений, используемых в оформлении кнопки. В поле **Button** вводится имя файла, содержащего изображение, которое будет постоянно находиться на кнопке. А в поле **On hover** записывается имя графического файла, который будет появляться на кнопке при попадании туда курсора. Для облегчения ввода имен файлов, рядом с каждым полем находится кнопка **Browse**.

Этот активный элемент представляет собой Java-апплет. Апплеты — это мини-приложения, написанные на Java, и заранее откомпилированные. В этом и заключается их отличие от сценариев на Java. Как мы помним, скрипты содержат обычные текстовые команды, поэтому объем файла может быть достаточно велик, а скорость выполнения сценариев обычно невелика, так как их приходится интерпретировать, то есть выполнять построчно, читая и выполняя команду за командой. Апплеты представляют собой следующий шаг в развитии технологии Java. Разработчик после написания апплета компилирует его. Но в данном случае компиляция не будет обычным переводом текста программы в инструкции для операционной системы или процессора, так как это подорвало бы идею многоплатформенности. Компиляция производится в собственный байт-код Java, который не может быть напрямую обработан каким-либо процессором или операционной системой. Этот байт-код намного компактнее, а значит, для его загрузки требуется меньше времени, да и скорость его выполнения выше, чем для сценария. Для его обработки требуется наличие так называемой виртуальной машины Java. Эта виртуальная машина пишется для каждой отдельной операционной системы, которая, как известно, обычно привязана к конкретному типу процессора. Эта виртуальная машина в настоящее время поставляется практически с каждым современным браузером, а в некоторые операционные системы встроена уже во время их разработки. Чем еще привлекательна технология Java, так это своей безопасностью. Все Java-компоненты работают только в пространстве браузера. Они не могут напрямую воздействовать на систему удаленного пользователя как, например, элементы ActiveX, которые потенциально могут быть вредны, так как они могут получить доступ ко всей файловой системе компьютера удаленного пользователя. В связи с этим не рекомендуется использовать элементы ActiveX, полученные из источника, которому пользователь не вполне доверяет, или в доброжелательности которого он не уверен. Именно поэтому в большинстве совре-

менных браузеров выстроена целая политика безопасности, основанная на репутации создателя сайта, разделении зон безопасности Интернета и использовании цифровой подписи активных элементов. Наличие этой подписи позволяет точно идентифицировать создателя элемента и определить степень его благонадежности, исходя из его репутации.

А вот следующий активный элемент **Marquee**, который создает бегущую строку в HTML-документе, реализуется только средствами HTML. Выполнение команды меню **Insert/Component/Marquee** активизирует диалоговое окно **Marquee Properties** (рис. 1.26).

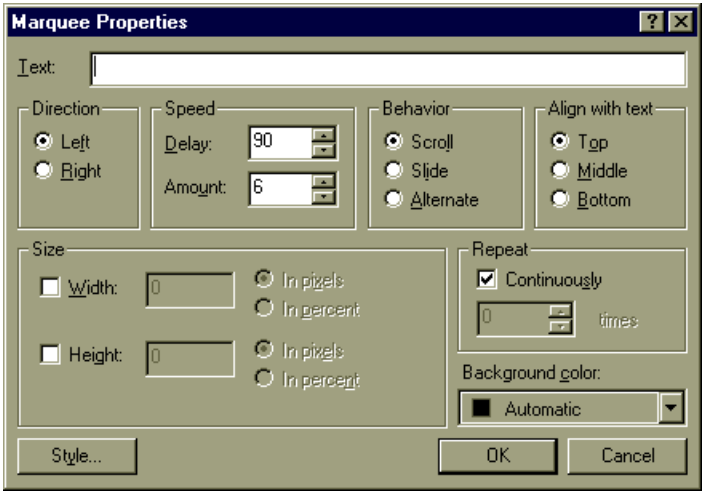

Рис. 1.26. Диалоговое окно **Marquee Properties**

Если в поле **Text** мы введем значение "Бегущая строка", то именно этот текст будет оформлен в виде активного элемента. При просмотре созданной страницы мы увидим, как этот текст будет выезжать из-за правой границы окна просмотра браузера и плавно двигаться по направлению к левой границе. При переходе на страницу **HTML** мы обнаружим там конструкцию `<marquee>Бегущая строка</marquee>`. Отсюда видно, что бегущая строка объявляется тегом `<marquee>`. Однако, как видно из рисунка, у бегущей строки есть достаточно много свойств, которые поддаются редактированию и установке. Так как этот элемент реализуется средствами HTML, то и свойства будут задаваться параметрами тега `<marquee>`. Рассмотрим эту ситуацию. Направление движения строки задается группой радиокнопок **Direction**. Установленная по умолчанию директива **Left** задает привычный всем порядок движения справа налево. Если его необходимо заменить, хотя это и не приветствуется с точки зрения удобочитаемости, необходимо выбрать альтернативу **Right**. При этом у тега `<marquee>` появляется параметр `direction="right"`. Скорость перемещения бегущей строки задается в бло-

ке **Speed**. Поле **Delay** указывает задержку перед началом движения строки в миллисекундах. Значение по умолчанию равно 90. В поле **Amount** задается скорость перемещения строки. Это значение равно количеству пикселов, на которые смещается бегущая строка за один шаг. Если мы зададим значения в этих полях, отличные от установленных по умолчанию, то, заглянув на вкладку **HTML**, увидим появление блока `scrollamount="8" scrolldelay="160"`. Как видно, скорость перемещения устанавливается параметром `scrollamount`, а задержка — `scrolldelay`. Группа переключателей **Behavior** предназначена для указания типа перемещения строки, и реализуется оно при помощи соответствующего параметра `behavior`. Значения его совпадают с наименованиями переключателей.

Бегущая строка является стандартным объектом HTML с изменяемыми размерами, и, следовательно, для нее можно указать выравнивание на странице. Делается это при помощи группы флажков **Align with text**. Реализация в HTML происходит при помощи уже знакомого нам параметра `align`. Размеры блока, отводимого под бегущую строку, могут быть изменены как путем простого изменения размера при помощи граничных маркеров объекта, когда он выделен и считается текущим, так и при помощи указания конкретных значений в полях **Width** и **Height**. Для получения возможности прямого указания размеров необходимо сначала выставить соответствующие флажки. Традиционно размеры могут задаваться в пикселах и процентах от величины окна просмотра браузера. В HTML-реализации бегущей строки для этих целей задействованы одноименные параметры `width` и `height`.

По умолчанию, бегущая строка движется постоянно, пока она находится в окне просмотра. Но можно задать и конкретное количество повторов. Для этого в блоке **Repeat** необходимо снять флажок в переключателе **Continuously**, который обеспечивает непрерывное повторение бегущей строки, и в поле ввода **times** указать необходимое количество повторов. В том случае, если мы укажем три повтора, то в теге `<marquee>` появится параметр `loop="3"`. И последний рассматриваемый нами элемент управления — выпадающий список **Background color**, задающий цвет фона. В HTML ему соответствует уже знакомый нам параметр `bgcolor`.

Следующий активный элемент носит имя **Confirmation Field**. Однако он работает только в паре с формами HTML-документов, которые мы будем рассматривать в следующей главе. Формы представляют собой набор элементов управления, в которых можно вводить некоторые значения или выбирать их из списка. Очень часто их используют для регистрации, когда в поля ввода необходимо ввести имя, фамилию или пароль. Для обработки форм обычно применяют специальные программы, которые носят название CGI-приложений. Их отличие от обычных программ заключается в том, что они специально приспособлены для получения данных из форм и передачи своих данных в браузер удаленного пользователя. Для этого применяется стандарт CGI (Common Gateway Interface). Так, например, если мы хотим узнать

имя удаленного пользователя, а потом отобразить страницу, в которой будет уже указано это имя ("Дорогой Петр! Мы очень рады, что Вы посетили наш сайт"), необходимо, чтобы соответствующее CGI-приложение получило это имя и создало новую страницу. Но мы условились, что мы не занимаемся программированием. Для того, чтобы выйти из этой ситуации FrontPage 2000 и предлагает нам элемент **Confirmation Field**. Этот активный элемент позволяет показывать значение, введенное в определенное поле удаленным пользователем. Для реализации вышеописанной ситуации мы должны создать одну страницу с полем ввода имени пользователя и кнопкой, к которой привязывается гиперссылка, указывающая на страницу, которая будет создана с учетом данных, введенных посетителем сайта. А уже на этой странице, в том месте, где бы мы хотели увидеть имя пользователя, вставляется активный элемент **Confirmation Field**, указывающий на наименование того поля, в которое удаленный пользователь ввел свое имя. Напоминаю, что полностью эту ситуацию мы разберем в следующем разделе, когда будем рассматривать создание форм.

Следующий активный элемент носит наименование **Include Page**. Он предназначен для того, чтобы на одной Web-странице показывать содержимое другой страницы без подключения ее через гиперссылку. Как сказано в справке FrontPage, эта возможность применяется в том случае, если вам необходимо часто помещать на нескольких страницах одну и ту же информацию, например, уведомление об авторском праве. Тогда, если вы поменяете статус страниц, изменив это уведомление, вам уже не придется исправлять эту информацию на каждой странице, содержащей объекты авторского права. Достаточно будет изменить одну-единственную страничку, и изменения коснуться всех документов, где присутствует это уведомление. Удобно? Безусловно.

Итак, для вставки содержимого другой страницы необходимо выполнить команду меню **Insert/Component/Included Page**, и в появившемся диалоговом окне указать URL подключаемой страницы. Проще не бывает. Кстати, эта возможность становится доступна только после сохранения страницы. Если страница только создается, и сохранение еще не производилось, данный пункт меню будет недоступен.

Следующий элемент носит наименование **Scheduled Picture**. Он предназначен для размещения на странице графического изображения, которое будет отображаться только в определенное время. Например, если мы заведуем электронным магазином и для стимулирования спроса в момент его сезонного падения собираемся делать скидку, то на необходимых страницах можно разместить этот элемент, для которого время воспроизведения будет совмещено с периодом действия скидок. Вставка изображения и задание параметров производится в диалоговом окне **Scheduled Picture Properties** (рис. 1.27). В поле **During the scheduled time** указывается графический файл, который содержит искомое изображение. Стоит еще раз обратить внимание на то, что в качестве полного имени может использоваться URL. Файлу со-

всем необязательно находиться именно на вашей машине, но о целесообразности использования картинок, скачиваемых непосредственно из Интернета, мы уже говорили. В поле **Before and after the scheduled time (optional)** указывается имя файла, содержащего изображение, которое будет отображаться до наступления выбранного срока и после его прохождения. Как видно из названия поля редактирования, этот параметр является необязательным. Начало и конец периода отображения искомой картинки указываются в группах полей **Starting** и **Ending** соответственно.

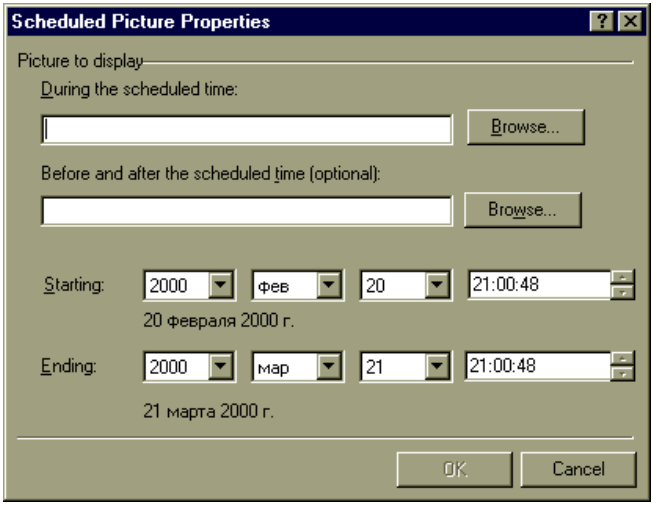

Рис. 1.27. Диалоговое окно **Schedule Picture Properties**

Активный элемент, который мы будем сейчас рассматривать, является гибридом двух предыдущих. Это внедряемая страница, отображаемая по расписанию — **Scheduled Include Page**. Ее диалоговое окно очень похоже на диалоговое окно рисунка, отображаемого по расписанию, но вместо графических файлов необходимо указывать URL подключаемой страницы, и той страницы, которая будет отображаться на ее месте в неурочное время.

Следующий элемент — **Substitution** — обычно используется в нижних колонтитулах страниц. Он позволяет вставлять заранее зарезервированные значения. То, что в Microsoft Word называют полями. В FrontPage можно вставлять имя автора, краткое содержание страницы, имя того, кто последний изменял ее содержимое и URL страницы. Для указания конкретного поля используется выпадающий список окна **Substitution Properties**. В нем находятся значения **Author**, **Description**, **Modified By** и **Page URL**.

Пять вышеперечисленных активных элементов похожи друг на друга, да и реализуются практически одинаково, поэтому в меню они были помещены в один блок. Следующий блок содержит три элемента, призванные помочь пользователю ориентироваться в ассоциативных связях сайта.

FrontPage 2000 позволяет приписывать каждой странице определенную категорию, то есть, разбивать их по темам: бизнес, путешествия, планирование и т. д. Список категорий, в принципе, поддается редактированию — они могут добавляться и удаляться. Для изменения списка категорий или определения какой-либо категории для создаваемой Web-страницы используется диалоговое окно **Page Properties**, а точнее, его вкладка **Workgroup**. Весь сайт, естественно, может (и должен) содержать страницы нескольких категорий. А активный элемент **Categories** позволяет в списке всех категорий, используемых при работе с сайтом, пометить некоторые. После этого FrontPage создает гиперссылки на все страницы данных категорий, входящие в проектируемый сайт. Так как страниц каждой категории может быть несколько, они могут быть отсортированы как по имени документа (а не файла, это разные вещи), так и по дате последнего их изменения. В качестве дополнительных элементов могут быть показаны дата последнего изменения страницы и комментарии, созданные для нее. Чаще всего активный элемент **Categories** размещается внизу страницы для реализации ассоциативных связей. Хотя можно попытаться на его основе создать панель навигации или тематическое содержание сайта.

В том случае, если посетителю сайта необходимо провести свой собственный поиск, а не пользоваться тематическими ссылками, проектировщик сайта может воспользоваться формой для проведения полнотекстового поиска во всех страницах, входящих в состав сайта. Для этого используется активный элемент **Search Form**. Его вставка производится при помощи команды меню **Insert/Component/Search Form**. При выполнении этой команды активируется диалоговое окно **Search Form Properties**. На первой вкладке с именем, которое полностью совпадает с наименованием окна, проектировщик сайта может задать сопроводительный текст для поля ввода, где посетитель будет указывать условие поиска. Для этого используется поле **Label for Input**, с установленным по умолчанию значением "Search for:". Естественно, если сайт рассчитан на русскоязычную аудиторию, сопроводительный текст стоит поменять. Длина поля ввода условия поиска измеряется в символах и указывается в поле **Width in characters**. В форме поиска помимо редактируемой строки условия, располагаются кнопки для запуска процесса поиска и очистки поля ввода. Надписи на кнопках задаются в полях **Label for "Start Search" button** и **Label for "Clear" button** соответственно.

На странице **Search Results** (рис. 1.28) размещены элементы управления, регулирующие внешний вид и функциональность страницы, на которой будут показаны результаты поиска. Данный активный элемент отображает список всех файлов (то есть Web-страниц, входящих в состав сайта), в которых встречается слово или словосочетание, заданное пользователем.

Заголовок для этого списка может быть задан в поле **Word list to search**, а формат самого списка и его информативность устанавливаются при помощи группы флажков **Display options**. Флажок **Display score (closeness of match)** позволяет рядом с именем страницы показывать уровень ее реле-

вантности к запросу. **Display file date** позволяет управлять отображением даты последнего изменения файла. Очень полезная опция для оценки новизны разыскиваемой информации. В том случае, если этот флажок установлен, создателю страницы становятся доступны выпадающие списки **Date format** и **Time format**, предназначенные для установки формата отображения даты и времени последнего изменения файла. И последний флажок **Display file size (in K bytes)** показывает, в дополнение ко всему, размер файла в Кбайт. После задания всех необходимых свойств достаточно нажать кнопку **OK** и форма поиска будет помещена на страницу. Для ее тестирования необходимо не только сохранить файл, но и произвести публикацию всего сайта, так как этот активный элемент динамически создает страницу с результатами поиска, а для этого необходима именно публикация сайта.

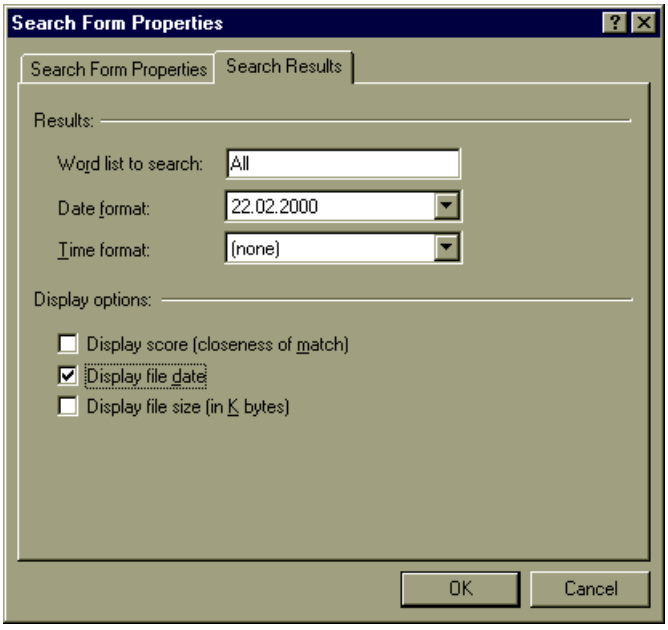

Рис. 1.28. Вкладка **Search Results** диалогового окна **Search Form Properties**

Последний активный элемент, входящий в коллекцию FrontPage 2000, является одним из самых важных атрибутов сайта, — это его оглавление. Данный элемент носит наименование **Table of Contents**. Оглавление можно начать с любой страницы и в него будут входить все страницы, которые в иерархии сайта являются подчиненными ей. URL стартовой страницы задается в поле **Page URL for starting point of table**. Теперь вспомним, по какому принципу создается оглавление в Microsoft Word. Эта программа сканирует документ в поисках текста, оформленного одним из стилей, предназначенных для заголовков. Так как FrontPage 2000 входит в состав семейства Microsoft Office, то для него действуют такие же правила. В Web-

страницах все заголовки выделяются при помощи шрифта. Как мы помним, зарезервированных размеров для шрифтов у FrontPage насчитывается ровно семь. Размер шрифта, который будет использоваться для заголовков, задается в поле **Heading font size**. Остальные элементы управления представляют собой флажки, управляющие правилами создания оглавления. Так, установка флажка **Show each page only once** указывает на то, что каждая Web-страница, входящая в состав оглавления, будет показываться только один раз, несмотря на количество входящих ссылок. Флажок **Show pages with no incoming hyperlinks** позволяет отображать те страницы, гиперссылок на которые со стартовой страницы оглавления не существует. FrontPage 2000 называет их сиротскими страницами (orphan pages). А установка последнего флажка **Recompute table of contents when any other page is edited** заставляет оглавление пересчитываться каждый раз, когда одна из страниц сайта изменяется.

На этом список компонентов FrontPage 2000 исчерпывается. Но это далеко не полный список тех возможностей, которые может захотеть использовать проектировщик Web-страницы. Мы уже говорили, что многие активные элементы можно найти в Интернете. Но вот как их включить в нашу страницу? Для этого предназначена команда меню **Insert/Advanced**, позволяющая вставлять найденные Java-апплеты (**Java Applet**), подключаемые модули расширения (**Plug-In**), элементы ActiveX (**ActiveX Control**). Более того, если проектировщик не доверяет механизму FrontPage 2000 по переводу создаваемой страницы в теги HTML, он может напрямую написать часть страницы в тегах HTML. Для этого применяется команда меню **Insert/Advanced/HTML**. Этот вариант используется в случае, если результат работы FrontPage 2000 не вполне устраивает проектировщика, и он знает, как написать HTML-код, чтобы он адекватнее отображал его замысел.

Теперь, когда мы знаем, как правильно использовать коллекцию активных элементов FrontPage и элементы, найденные в Интернете, мы сможем достойно оформить наш сайт, добавив ему интерактивности и динамичности. Впрочем, полную интерактивность вместе с обратной связью позволяют создавать формы, вставляемые в наши Web-страницы. С одной из таких форм мы уже сталкивались, когда создавали форму для полнотекстового поиска. В следующем разделе мы подробно рассмотрим их возможности и недостатки.

Формы

Очень часто владельцу Web-сайта необходимо, чтобы посетитель ввел какую-либо информацию, которую потом можно было бы обработать. Чаще всего это применяется при регистрации посетителя, при получении от него восторженного отзыва в гостевую книгу, проведении различного рода викторин и анкетирований. Конечно, могут быть и другие, более серьезные применения форм, но их обработка достаточно сложна, а, значит, недоступна нам. Типичная форма представляет собой набор полей редактирова-

ния текста, выпадающих списков, радиокнопок и флажков. Также ко всей этой совокупности элементов управления обычно добавляется две кнопки. Одна из них обновляет введенные пользователем значения, приводя их к установленным по умолчанию, а вторая завершает процесс ввода и отсылает данные на сайт для обработки. Сами формы создаются средствами HTML, а вот для обработки применяются CGI-приложения. Это приложение обычно привязывается к кнопке, отсылающей результаты на сервер. Обычно эту кнопку называют "Submit". Процедура работы формы не так уж сложна. Браузер посетителя сайта получает HTML-код страницы с формой и отображает ее. После заполнения всех полей удаленный пользователь нажимает кнопку "Submit". При нажатии на эту кнопку браузер формирует специальную строку, в которой записаны все значения, указанные пользователем. Такая строка пересылается на сайт с использованием протокола HTTP. На сайте располагается то самое CGI-приложение (или его аналог ISAPI-расширение), которое призвано осуществлять прием данных и их обработку. Эта программа всегда находится на сервере и никогда не загружается на машину удаленного пользователя. Он общается с ней по протоколу HTTP, как для приема данных, так и для посылки ответа. Итак, CGI-приложение получает строку с включенными в нее значениями, расшифровывает ее и обрабатывает данные. После этого CGI-приложение само формирует новую страницу (обычно с учетом данных удаленного пользователя), и пересылает ее HTML-код браузеру посетителя сайта с использованием все того же протокола HTTP.

Понятно, что без определенного опыта программирования мы не сможем самостоятельно написать CGI-приложение или ISAPI-расширение. Вместо этого мы воспользуемся средствами, предлагаемыми FrontPage 2000, а если их будет недостаточно, в просторах Интернета всегда можно отыскать несколько подходящих программ, которые нужно будет только правильно установить на сервер. Процедура эта очень проста, но чтобы сейчас не отвлекаться от рассмотрения форм, мы отложим ее подробное рассмотрение до третьей части нашей книги. А сейчас перейдем к созданию самих форм при помощи FrontPage 2000.

Для вставки минимальной формы используется команда **Insert/Form/Form**. При выполнении этой команды на странице появляется форма без какого-либо элемента управления для ввода данных, но с двумя кнопками. Кнопка **Reset**, как мы уже говорили, предназначается для установки во всех средствах работы с данными начальных значений, которые были установлены при разработке, а кнопка **Submit** используется для пересылки данных серверу. В том виде, как она получилась, сама форма сделать ничего не может. Данные-то вводить некуда. Поэтому этот вариант может рассматриваться только как заготовка. Место, отведенное для формы, всегда можно увеличить, и уже потом добавлять в нее необходимые элементы управления.

Теперь перейдем на вкладку **HTML** рабочего пространства FrontPage 2000 и посмотрим, как реализована эта маленькая форма. Мы увидим там следующий блок:

```
<form method="POST" action="--WEBBOT-SELF--">
  <!--webbot bot="SaveResults" U-File="_private/form_results.txt"
  S-Format="TEXT/CSV" S-Label-Fields="TRUE" -->
  <p><input type="submit" value="Submit" name="B1"><input
type="reset" value="Reset" name="B2"></p>
</form>
```

Как видно, блок, содержащий в себе объявление формы, ограничивается тегами `<form>` и `</form>`. У открывающего тега `<form>` есть несколько параметров. Один из них с именем `method` позволяет указывать вариант передачи данных CGI-приложению. Мы не будем вдаваться в подробности этих методов, отметим только, что у этого параметра есть только два значения: `"POST"` и `"GET"`. Чаще всего используется первый. Второй параметр, который мы видим, с именем `action`, предназначен для указания имени программы, которая будет обрабатывать введенные данные. Эта программа запускается при нажатии на кнопку **Submit**. Следующие две строки вышеприведенного HTML-кода как раз задают свойства этой программы. Детальное их рассмотрение не входит в наши планы, однако стоит сказать, что все введенные значения эта программа будет сохранять в текстовом файле с именем form_results.txt, расположенном в каталоге **_private**. Как мы помним, этот каталог входит в стандартную структуру каталогов создаваемого сайта.

Теперь перейдем к той конструкции, которая находится внутри обрамляющих тегов абзаца `<p>` и `</p>`. Эти теги объявляют две наши кнопки. Как нетрудно заметить, каждая из них реализуется при помощи тега `<input>`. Он применяется для создания практически всех элементов управления, которые применяются в формах. Конкретный вид используемого элемента управления обозначается параметром `type`. Как видно, кнопка, предназначенная для отправки данных, задается при помощи значения `"submit"`. Для создания второй кнопки, возвращающей к начальным значения в полях ввода данных, используется значение `"reset"`. Надписи на кнопках задаются значениями параметра `value`. В нашем случае они совпадают с типами кнопок. Но ведь нам никто не мешает сделать свои, предназначенные для русскоязычной аудитории, названия кнопок. Впрочем, производить это посредством изменения HTML-кода совсем не обязательно. Немного позже мы рассмотрим более цивилизованные способы изменения свойств, как кнопок, так и остальных элементов ввода данных. И, наконец, последний параметр тега `<input>` предназначен для задания имени элемента управления, которое полностью идентифицирует его. Этот параметр с именем `name` является обязательным.

Теперь перейдем к обещанному цивилизованному способу изменения свойств кнопок. Для того, чтобы активизировать диалоговое окно **Push Button Properties**, необходимо выделить нужную кнопку одиночным щелчком мыши, а затем в контекстном меню выбрать альтернативу **Form Field Prop-**

erties. В этом диалоговом окне можно указать идентифицирующее имя кнопки, надпись на ней и ее тип. Тип задается группой радиокнопок **Button type**, в которой помимо уже известных нам типов "Submit" и "Reset", наличествует еще и тип "Normal". Он предназначен для обычных кнопок, которые активизируют привязанное к ним CGI-приложение. Возникает естественный вопрос, а как присоединить это приложение или, если брать общую постановку вопроса, как можно регулировать работу формы. Для этого необходимо установить определенные свойства целой формы. Эти действия производятся при помощи диалогового окна **Form Properties** (рис. 1.29), активизируемого при помощи одноименной команды контекстного меню. Главное для нас — установить порядок передачи данных, введенных в форму удаленным пользователем. Для этого мы можем воспользоваться группой радиокнопок **Where to store results**. В том случае, если нам нужно сохранить эти данные в каком-либо текстовом файле или переслать их по электронной почте, необходимо выбрать **Send to** и, в соответствующих полях, ввести имя принимающего файла, либо электронный адрес для пересылки данных. Альтернатива **Send to database** применяется в том случае, если проектировщик Web-страницы умудрился подключить к ней свою базу данных. Ну, а третья радиокнопка позволяет передавать данные для обработки сторонним приложениям, либо стандартным процедурам регистрации пользователя. Эти варианты чаще всего используются в том случае, если сайт создается на основе поставляемых образцов, в этом случае какой-либо особой настройки обычно не требуется.

Блок **Form properties** содержит поля ввода, в которых можно указывать имя самой формы и фрейм, в котором она должна будет отображаться.

В том случае, если мы решили сохранять введенные посетителями данные в своем файле, или пересылать их на какой-либо электронный адрес, есть возможность более тщательно указывать правила обработки результатов.

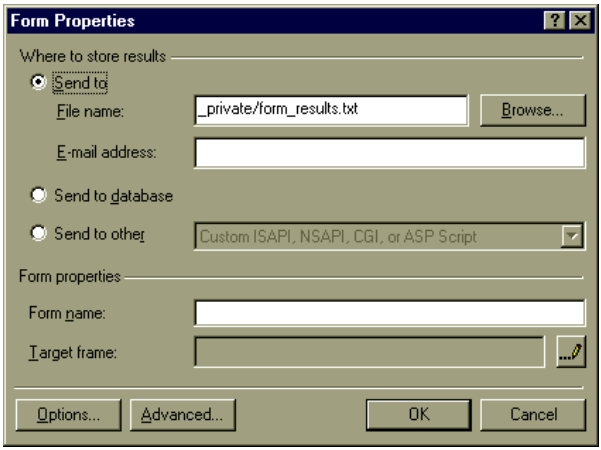

Рис. 1.29. Диалоговое окно **Form Properties**

Для этого следует нажать кнопку **Options**, которая активизирует дополнительное диалоговое окно **Options for Saving Results of Form**. Это окно содержит четыре вкладки. Вкладка **File Results** предназначена для установки свойств файлов, содержащих введенные данные. Да-да, именно файлов, так как помимо основного сохраняющего файла можно задать и второй, дублирующий. Для обоих задается имя в полях **File name**. Выпадающие списки **File format** позволяют указывать формат записи данных. В этих списках есть все стандартные варианты файлов с разделителями, которые понимают современные системы управления базами данных, а при помощи флажков можно указать, что помимо самих данных в эти файлы включаются имена полей, и задать порядок сохранения данных. Так, если выставить флажок **Latest results at end**, то более поздние результаты будут записываться в конец принимающего файла.

Вкладка **E-mail Results** содержит органы управления, предназначенные для регулировки процесса отсылки введенных данных по какому-либо электронному адресу. В поле **E-mail address to receive results** вводится этот самый адрес. Список **E-mail format** предназначен для указания формата посылаемого электронного письма. Уже знакомый нам флажок **Include field names** позволяет указывать детальность отсылаемых данных. Если флажок не установлен, то данные идут последовательным потоком в том формате, в каком они были введены. Если же флажок установлен, то к каждому переданному значению добавляется имя поля, в котором это значение было введено удаленным пользователем.

Для обработки электронных писем, которые хлынут на указанный адрес, FrontPage 2000 позволяет задавать строки, которые будут вставляться в заголовок письма в качестве темы (Subject) и отправителя (From). Для этого применяются поля ввода **Subject line** и **Reply-to line** соответственно.

Следующая вкладка с именем **Confirmation Page** предназначена для установки URL страницы, которая будет показана в качестве подтверждения приема данных. Как мы помним, формирование подобной страницы обычно является завершающим этапом работы CGI-приложения или ISAPI-расширения. Итак, в том случае, если пользователь ввел все необходимые данные и успешно отправил их на сервер, в его браузере отображается подтверждающая страница. Ее URL вводится в поле **URL of confirmation page**. Но ведь бывают случаи, когда результат ввода данных может считаться неудовлетворительным. Например, когда пользователь пытается зарегистрироваться под уже существующим именем. В таких случаях обычно показывается страница, содержащая соответствующее сообщение. Естественно, эта возможность не является обязательной (как, впрочем, и вообще указание подтверждающих страниц). Тем не менее, такая возможность предусмотрена и URL этой страницы может быть введен в поле **URL of validation failure page**.

На последней вкладке с именем **Saved Fields** мы можем установить список полей, значения из которых мы хотели бы получить. Здесь же можно вы-

брать форматы отображения даты и времени. В группе флажков **Additional information to save** можно указать дополнительную информацию, которую мы бы хотели получать вместе с каждым ответом. В качестве этих дополнительных сведений могут быть получены имя удаленного компьютера, имя пользователя, зарегистрированного на этом компьютере, и тип используемого для просмотра страницы браузера. Для этого достаточно установить флажки **Remote computer name**, **Username** и **Browser type**. На этом мы заканчиваем обзор свойств формы и переходим к тем элементам управления, которые мы можем использовать в этой самой форме.

Для размещения в форме однострочного поля ввода используется команда меню **Insert/Form/One-Line Text Box**. Это поле будет создано на том месте, где расположен текстовый курсор. При этом в HTML-коде, реализующем саму форму, появится тег `<input type="text" name="T1" size="20">`. Сам тег нам уже известен, но у него в параметре `type` установлено новое значение `"text"`, которое позволяет создавать однострочные поля для ввода текста. При этом помимо уже знакомого параметра `name`, в теге появился еще один параметр `size`, предназначенный для установки длины создаваемого поля в символах. Как видно, все достаточно просто. Причем, если мы хотим изменить длину поля, совсем необязательно исправлять соответствующее значение прямо в HTML-коде. Достаточно выделить поле на странице **Normal**, щелкнув по нему кнопкой мыши, и, потянув за ограничивающие маркеры, изменить его размер. Впрочем, такого же результата можно добиться, активизировав диалоговое окно, предназначенное для изменения свойств поля. Для этого, как обычно, используется команда **Form Field Properties** из контекстного меню. В появившемся диалоговом окне можно установить имя поля ввода, применяемое для его идентификации. Функция установки имени будет присутствовать во всех диалоговых окнах, устанавливающих свойства любого элемента управления формы. В поле **Initial Value** устанавливается значение, используемое по умолчанию. Если мы укажем его явно, то в теге `<input>` появится дополнительный параметр `value="Умолчание"`. Ширину нашего элемента управления в символах можно установить в поле **Width in character**. Также, каждому элементу управления формы может быть приписан порядковый номер. Делается это для того, чтобы между ними можно было перемещаться не только при помощи мыши, но и с помощью клавиши табуляции. Нумерация обычно начинается с нуля и порядковые номера приписываются каждому элементу управления. Для этого используется поле **Tab order** соответствующего диалогового окна. При указании порядкового номера в HTML-код, реализующий элемент управления, добавляется параметр `tabindex`.

Однострочные поля для ввода текста бывают двух видов. Первый вариант — обычную, стандартную реализацию строки ввода — мы уже рассмотрели. Но существует информация, не предназначенная для всеобщего обозрения, при вводе которой вместо вводимых символов будут отображаться звездочки. Для создания подобного варианта поля текстового ввода необходимо в диа-

логовом окне **Text Box Properties** выбрать в группе **Password Field** радиокнопку **Yes**. После нажатия кнопки **OK**, в соответствующем поле на вкладке **Normal** сразу будут отображены звездочки, заменяющие предустановленное значение. А для реализации при помощи HTML-кода в параметре `type` тега `<input>` указывается значение `"password"`.

Для полей текстового ввода могут быть установлены правила соответствия значения. То есть всегда можно потребовать от пользователя, чтобы вводимое значение соответствовало бы некоторым ограничениям. Например, если мы хотим получить возраст, то можно принять только числовое значение в определенном интервале (например, от пяти до восьмидесяти лет). Для установки подобных ограничений используется кнопка **Validate** диалогового окна **Text Box Properties** или команда **Form Field validation** из контекстного меню объекта. Общий вид появившегося диалогового окна, предназначенного для установки ограничений на вводимые значения, показан на рис. 1.30.

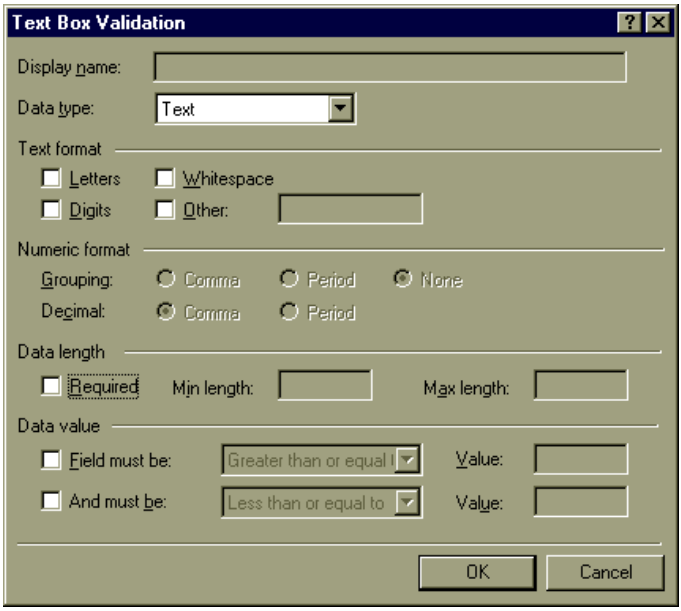

Рис. 1.30. Диалоговое окно **Text Box Validation**

В этом окне самым главным элементом управления является выпадающий список **Data type**, в котором устанавливается тип получаемого значения. Если мы не собираемся устанавливать какие-либо ограничения на тип получаемых данных, то выбирается пункт **No constraints**. Для текстового типа зарезервировано значение **Text**. Для обычного числового типа — **Number**, а если необходимо получить целочисленные данные, используется пункт **Integer**. В зависимости от того, какой тип получаемых данных установлен, активизируется тот или иной блок элементов управления диалогового окна.

Так, для текстовых данных предназначен блок **Text format** с группой флажков. Каждый из этих флажков предназначен для разрешения или запрета ввода определенных символов. Флажок **Letters** отвечает за ввод букв, **Digits** — за цифры. Если в состав вводимых данных могут входить пробелы, необходимо установить флажок **Whitespace**. Если помимо вышеперечисленных могут вводиться и другие символы, то необходимо установить флажок **Other** и указать их в соответствующем поле ввода.

Для числовых данных зарезервирован блок **Numeric format**. Причем формат целочисленных значений регулируется всего одной группой радиокнопок **Grouping**. Этот элемент управления позволяет регулировать порядок разбиения числа на триады. В том случае, если подобное разбиение не требуется, выбирается **None**. Если планируется разделять триады точками или запятыми, используются переключатели **Period** и **Comma** соответственно. Если в проектируемое поле планируется ввод численных значений, которые могут представлять собой десятичные дроби (выбран тип **Number**), то можно указать символ, который будет применяться в качестве десятичного разделителя. Для этого используется группа радиокнопок с названием **Decimal**. В качестве десятичного разделителя, также как и для триад, может быть использована либо точка, либо запятая. Естественно, не стоит в качестве разделителя и для триад и для десятичного числа использовать один и тот же символ.

Вполне вероятно, что в нашей форме не все поля ввода будут равнозначны. Одна информация нам будет более важна ("Укажите, пожалуйста, Ваше полное имя и номер Вашей кредитной карты"), а другая — менее ("Как Вы назвали свою собаку?"). Поэтому некоторые поля могут быть объявлены обязательными для ввода. Более того, длина вводимого значения тоже может регламентироваться. Для этого используются элементы управления, сосредоточенные в блоке **Data length** нашего диалогового окна. В том случае, если создаваемое поле ввода является обязательным для заполнения, то устанавливается флажок **Required**. Минимальную и максимальную длину вводимого значения в символах можно (и нужно) указать в полях **Min length** и **Max length** соответственно.

Бывает так, что посетитель сайта пытается нажать на кнопку отсылки информации, не введя обязательные данные. В этом случае бывает полезно отобразить напоминание. Для каждого обязательного поля ввода текст напоминания одинаков и задан заранее. Меняется только наименование поля. Это наименование вставляется в текст предупреждения. Указать его можно в строке **Display name**.

Однако самую главную возможность диалогового окна, применяемого для установки ограничений для вводимых удаленным пользователем данных, мы еще не рассмотрели. Как мы уже говорили, на вводимые данные можно накладывать определенные ограничения. Так вот, для указания отношений сравнимости (больше, меньше, равно, не равно и так далее) используется блок **Data value**. Всего можно наложить два ограничения, связанные логической операцией "И". Сама процедура установки ограничений достаточно

проста. Для первого ограничения активизируется переключатель **Field must be**, затем в соответствующем списке выбирается необходимое условие, а в поле **Value** выставляется сравнительное значение. Для второго ограничения выбирается переключатель **And must be**, а потом операция повторяется. Рассмотрим эту процедуру на примере.

Нам необходимо получить возраст посетителя сайта. Для отбраковки заведомо неверных ответов мы устанавливаем возможный возраст от десяти до восьмидесяти лет. То есть наше получаемое значение должно быть больше или равно десяти, и меньше или равно восьмидесяти. В нашем блоке **Data Value** мы устанавливаем оба флажка. Затем, в первой строке, выбираем операцию сравнения **Greater than or equal to**, а в поле **Value** выставляем десятку. Во второй строке мы выбираем операцию **Less than or equal to** и устанавливаем значение восемьдесят. Как видим, все достаточно просто.

Итак, с обычным однострочным полем текстового ввода мы разобрались. Переходим к следующему элементу ввода данных, каковым является блок для ввода нескольких строк текста, снабженный полосами прокрутки. Для его вставки применяется команда меню **Insert/Form/Scrolling Text Box**. По умолчанию эта команда вставляет область из двух строк по двадцать символов, но ее размеры вместе с остальными немногочисленными свойствами можно изменить при помощи соответствующего диалогового окна. Если после установки всех необходимых свойств мы заглянем на вкладку **HTML**, то обнаружим следующую конструкцию:

```
<textarea rows="3" name="S1" cols="40" tabindex="0">Текст по умолчанию</textarea>
```

Как видно, многострочное поле ввода реализуется не при помощи тега `<input>`. Это один из немногих элементов управления, для которого зарезервирован собственный тег `<textarea>`. Его параметры нам уже знакомы. В параметре `name`, как обычно, задается уникальное идентифицирующее имя органа управления. Параметры `rows` и `cols` предназначены для указания количества отображаемых строк и столбцов соответственно. Естественно, это не означает, что вводимый текст будет ограничиваться подобным же образом. У поля есть полосы прокрутки, которые будут использоваться по мере необходимости. Параметр `tabindex` как всегда применяется для указания порядкового номера формы, который используется при навигации по форме. Текст, отображаемый в поле по умолчанию, располагается между объявляющим тегом `<textarea>` и его закрывающим близнецом `</textarea>`.

Следующий элемент управления, предназначенный для использования в формах, вставляется при помощи команды меню **Insert/Form/Check Box**. При этом в форме появляется независимый переключатель в виде флажка, причем, без привязанного к нему текста и его придется вводить вручную. Свойств у таких переключателей не так уж и много. Как нам уже известно, у каждого элемента управления есть свое наименование и порядковый номер, используемый при навигации по форме при помощи клавиши табуля-

ции. Также можно указать стартовое состояние переключателя, то есть будет или нет в нем установлен флажок при первом отображении формы. Это свойство, как и все остальные, задается в диалоговом окне установки свойств переключателя **Check Box Properties**. Для установки начального состояния независимого переключателя используется группа радиокнопок **Initial state**. Естественно, кнопка **Checked** заставляет отображаться переключатель уже с установленным флажком. Радиокнопка **Not checked** оставляет его пустым. Поле **Value** предназначено для настройки взаимодействия переключателя с программой, обрабатывающей данные. Если в случае текстовых полей, обрабатывающей программе передавался введенный в эти поля текст, то здесь ситуация несколько другая. На сервер передается значение, указанное в поле **Value**, то есть, если посетитель сайта при работе с формой установил флажок в этом переключателе, то он отсылает соответствующее текстовое значение, связанное с ним. Если переключатель не был использован, то ничего и не пересылается. В случае если мы сами создаем обрабатывающее приложение, содержимое этого поля необходимо привести в соответствие с правилами приема данных нашей программой, но если планируется использовать стандартные решения FrontPage 2000, то настоятельно рекомендуется не изменять значение поля **Value**. Оно будет применяться в обрабатывающей программе и в активных элементах, использующих информацию из форм, таких, как **Confirmation Field**.

Итак, если мы явно установим свойства независимого переключателя, и заглянем на вкладку **HTML**, то мы увидим там следующий код:

```
<input type="checkbox" name="C1" value="ON" checked tabindex="1">
```

Здесь видно, что, как и подавляющее число всех наших элементов управления, независимый переключатель реализуется при помощи тега `<input>`. Значение параметра `type`, задающее тип создаваемого элемента равно `"checkbox"`. Параметры `name` и `tabindex` нам давно знакомы, и рассматривать их нет смысла. Параметр `checked` используется, если при первом отображении переключателя он должен быть помечен флажком. Напомню, что по умолчанию все независимые переключатели отображаются пустыми. И, наконец, параметр `value` в качестве своего значения содержит текстовую строку, которая будет передаваться обрабатывающей программе, в том случае, если на момент отсылки данных на сервер флажок будет установлен.

После рассмотрения независимых переключателей будет логичным рассмотрение радиокнопок. Эта возможность также предусмотрена FrontPage 2000. Для вставки радиокнопки используется команда меню **Insert/Form/Radio Button**. Эта команда, как и предыдущая, создает только сам элемент управления, а текст, ассоциированный с ним, необходимо вводить отдельно. Так как создавать одну радиокнопку несколько бессмысленно, то разместим в форме группу из двух радиокнопок, а затем посмотрим, как они реализованы посредством HTML-кода. На вкладке **HTML** мы увидим следующий блок кода:

```
<p><input type="radio" value="V1" name="R1" checked>Радиокнопка 1</p>
<p><input type="radio" name="R1" value="V2">Радиокнопка 2</p>
```

Все конструкции, реализующие отдельные радиокнопки, помещены здесь между тегами объявления абзаца. Как видно, значение параметра `type`, задающего конкретный тип создаваемого элемента управления формы, является текстовой строкой `"radio"`. Но если мы внимательно присмотримся к параметрам `name` в обоих тегах, то обнаружим, что значения в них совпадают. Дело в том, что отдельная радиокнопка не может считаться самостоятельным элементом управления, поэтому в параметре `name` задается имя группы радиокнопок. Если у нескольких радиокнопок, размещенных в одной форме, причем, совсем необязательно они будут рядом друг с другом, значения этого параметра совпадают, то они и вести себя будут соответственно, то есть, при активизации одной кнопки выбранная перед этим кнопка утратит свой статус, и выделение с нее будет снято. Параметр `value` в радиокнопках служит той же цели, что и одноименный параметр в HTML-реализации независимых переключателей. То есть при отсылке данных принимающей программе на сервер, рядом с именем группы будет указана строка, являющаяся значением параметра `value` выбранной удаленным пользователем радиокнопки. При первом отображении формы, в группе радиокнопок одна может быть выбранной заранее, для чего в ее тег добавляется параметр `checked`. Он, естественно, не нуждается в присвоении какого-либо значения. Однако наличие одной активизированной радиокнопки в группе не является обязательным, поэтому вполне возможна ситуация, когда ни один тег радиокнопок не имеет этого параметра.

Редактирование свойств каждой отдельно взятой радиокнопки производится при помощи диалогового окна **Radio Button Properties**. Как мы помним, подобные диалоговые окна для каждого элемента управления, размещаемого в формах, активизируются командой контекстного меню **Form Field Properties**. В данном диалоговом окне в поле **Group name** вводится имя группы, к которой принадлежит создаваемая радиокнопка. В поле **Value** вносится идентифицирующее имя радиокнопки, а группа **Initial state** позволяет указать ее начальное состояние. Конечно, как только в этой группе для какой-либо радиокнопки мы указываем начальное состояние **Selected**, всем остальным радиокнопкам этой группы приписывается начальное состояние **Not selected**.

И последний элемент ввода информации, используемый в формах — выпадающий список, который создается при помощи команды **Insert/Form/Drop-Down Menu**. Изначально вставляется маленький и пустой список, затем значения для выбора и все остальные свойства списка мы можем задать в диалоговом окне **Drop-Down Menu Properties** (см. рис. 1.31).

Самой главной частью этого диалогового окна, очевидно, является основной список, в котором размещаются все элементы списка и добавление нового элемента в который производится нажатием кнопки **Add**. Эта кнопка

визуализирует дополнительное диалоговое окно, в котором можно указать все свойства нового элемента. В строке **Choice** вводится текст который будет являться значением по умолчанию, передаваемым на сервер. Но для русскоязычных списков по вполне понятным причинам это несколько неудобно, поэтому следует установить флажок **Specify value** и в соответствующее поле ввести необходимое текстовое значение. Как видно (рис. 1.31), для элемента списка "Первый пункт" соответствующее значение представляет собой текстовую строку "first". Как и все подобные значения, эти строки могут использоваться не только принимающим приложением, но еще и в подтверждающих активных элементах, таких как **Confirmation Field**. При помощи группы радиокнопок **Initial state** можно установить начальное состояние этого пункта, независимо от того, будет он выбран или нет. Изменение свойств уже существующего элемента списка производится при помощи кнопки **Modify**, а для удаления какого-либо пункта используется кнопка **Remove**. Изменение порядка элементов производится кнопками **Move Up** и **Move Down**. Высота списка задается в поле **Height**. Правда, если мы укажем высоту списка более одной строки, то он будет больше напоминать прокручиваемый, а не выпадающий список, но суть его от этого не изменится. В том случае, если этот список спроектирован для множественного выбора, то есть посетитель сайта может выбрать несколько элементов списка сразу, следует отметить радиокнопку **Yes** в группе **Allow multiple selections**. После установки всех значений перейдем на вкладку **HTML** и посмотрим, как реализован это список. Мы увидим следующий код:

```
<select size="2" name="D1" multiple>
    <option value="first">Первый пункт</option>
    <option selected value="second">Второй пункт</option>
    <option value="third">Третий пункт</option>
</select>
```

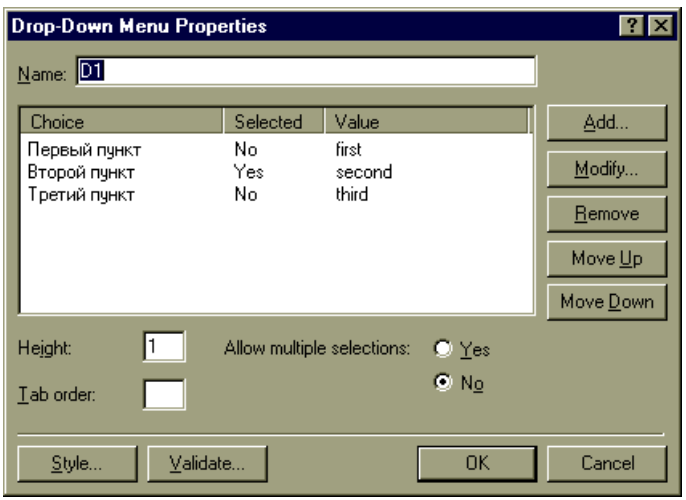

Рис. 1.31. Диалоговое окно **Drop-Down Menu Properties**

Здесь мы встречаем еще один (и последний) элемент ввода данных, для которого зарезервирован собственный тег. Теперь это тег `<select>`. Между ним и его закрывающим близнецом `</select>` располагаются теги `<option>`, реализующие отдельные элементы выпадающего списка. Сначала рассмотрим параметры тега `<select>`. Высота списка в строках задается параметром `size`, а так как в диалоговом окне задания свойств списка мы установили возможность множественного выбора, то здесь появился параметр `multiple`.

Теги элементов списка достаточно просты для анализа. Прежде всего, необходимо отметить, что текст этих элементов находится в обрамлении тегов `<option>` и `</option>`. Параметр `value` содержит значение, идентифицирующее данный элемент списка, а если этот элемент установлен по умолчанию, то добавляется еще и параметр `selected`.

На этом список элементов управления, предназначенных для ввода данных, заканчивается. Мы уже сталкивались с двумя типами кнопок — **Reset** и **Send**. Теперь нам осталось рассмотреть обычную кнопку. Для ее вставки используется команда меню **Insert/Form/Push Button**. Изменения ее свойств осуществляется в диалоговом окне **Push Button Properties**. В поле **Value/Label** можно указать надпись, помещаемую на кнопке, а в группе радиокнопок **Button type** — выбрать тип создаваемой кнопки. Из предлагаемых трех типов два (кнопки **Send** и **Reset**) мы уже знаем и рассматривать их не будем. Нас интересует тип **Normal**. С его помощью можно создать кнопку без привязки к какому-либо действию. Подобные кнопки обычно не используются простыми пользователями, так как для них нужно написать и присоединить собственный сценарий действий. Но все-таки посмотрим, как такая кнопка реализуется посредством HTML-кода. Для ее объявления используется тег:

```
<input type="button" value="Кнопка" name="B3">
```

Единственное отличие от HTML-реализации других известных нам кнопок — в параметре `type`, задающем тип кнопки. В нашем случае используется значение `"button"`. В параметре `value`, как всегда, задается надпись на кнопке.

Для посылки данных из формы на сервер не обязательно использовать кнопку типа **Submit**, а, например, обычное графическое изображение, которое будет играть роль этой кнопки. С этой целью часто используют картинку с красивой надписью, которая гласит "Отправить!" или что-нибудь в таком же роде. Для вставки подобного элемента управления используется команда меню **Insert/Form/Picture**. При этом появляется обычное окно вставки графического изображения. Если после размещения на форме этой картинки мы посмотрим код HTML, то обнаружим там следующий тег:

```
<input border="0" src="images/Hlplogo.gif" name="I1" width="97"
height="40" type="image">
```

Здесь в качестве значения параметра `type` указано значение `"image"`. Все остальные параметры кроме параметра `name` полностью повторяют параметры тега, объявляющего обычное графическое изображение.

При работе с формой можно заметить, что для выбора какой-либо радиокнопки или независимого переключателя необходимо щелкнуть мышью именно на самой радиокнопке или квадратике переключателя. А это очень неудобно. А в обычных приложениях Windows для этой цели достаточно щелкнуть мышью на строке, которая связана с этим элементом управления. То же самое можно сделать и в наших формах. Для этого необходимо создать нужный элемент, набрать рядом с ним текстовую строку, которая будет привязана к нему, затем выделить строку вместе с самим элементом управления и выполнить команду меню **Insert/Form/Label**. После этого надпись окажется привязанной к элементу и желаемый эффект будет достигнут. В HTML это можно сделать при помощи следующего кода:

```
<input type="checkbox" name="C1" value="ON" id="fp1"><label
for="fp1">  Метка</label>
```

Из текста этого блока видно, как объявлен независимый переключатель, но помимо известных нам параметров появился параметр `id`, введенный специально для взаимодействия с тегом `<label>`, который объявляет привязанную текстовую строку. В теге `<label>` находится параметр `for`, значение которого должно совпадать со значением параметра `id` из тега `<input>` для того, чтобы связать переключатель и соответствующий ему текст между собой. На этом мы заканчиваем обзор возможностей FrontPage 2000, которые применяются для создания форм. Более того, мы закончили рассмотрение всех основных возможностей создания Web-страниц. Теперь нам предстоит перейти на более высокий уровень. Сайты должны создаваться в едином стиле, а достигнуть этого можно не только прямым указанием всех необходимых параметров. Это даже не поощряется, ведь у нас есть достаточно мощное средство единого стилевого оформления. Но об этом мы поговорим уже в следующем разделе.

Главное — это стиль...

Мы уже говорили, что при задании атрибутов текста для Web-страниц нельзя точно указывать наименование шрифта и его размер в пунктах, так как никогда не известно заранее, какие шрифты установлены на машине посетителя сайта. Однако можно указать тип применяемого шрифта, и на машине удаленного пользователя будет найден шрифт, максимально близкий к указанному. Как мы помним, параметры шрифтового оформления указывались непосредственно в тегах, объявляющих тот или иной абзац. Это первый вариант оформления текста, но он не может считаться эффективным. Представьте себе, что после завершения работы, во время приемки сайта заказчик пожелает изменить оформление всех заголовков первого уровня.

Это значит что необходимо будет в каждой Web-странице найти все заголовки первого уровня и изменить их оформление.

Но ведь FrontPage 2000 содержит библиотеку стилей. Когда мы говорили об оформлении текста, то мы пользовались его стандартной коллекцией стилей. Определения этих стилей действуют в пределах одного документа. Использование подобных локальных решений позволяет эффективно автоматизировать работу. Теперь, вместо того, чтобы во всем документе отыскивать текстовые элементы и изменять их атрибуты, достаточно один раз отредактировать соответствующий стиль, и вся Web-страница приобретет требуемый вид. Это второй вариант стилевого оформления сайта.

Но есть и третий путь. Для сайта может быть разработано единое стилевое оформление, которое записывается в общий файл. Этот объединенный файл потом использует каждый HTML-документ, входящий в состав сайта. Подобные файлы называются каскадными таблицами стилей (CSS — Cascading Style Sheets). Мы рассмотрим оба последних варианта работы со стилями.

Итак, предположим, что у нас уже есть спроектированная Web-страница, и нам необходимо исправить оформление заголовков первого уровня. Для этого мы должны отредактировать уже существующий стиль **Heading 1**. Сначала необходимо выполнить команду **Format/Style**. При этом будет активизировано диалоговое окно **Style** (рис. 1.32).

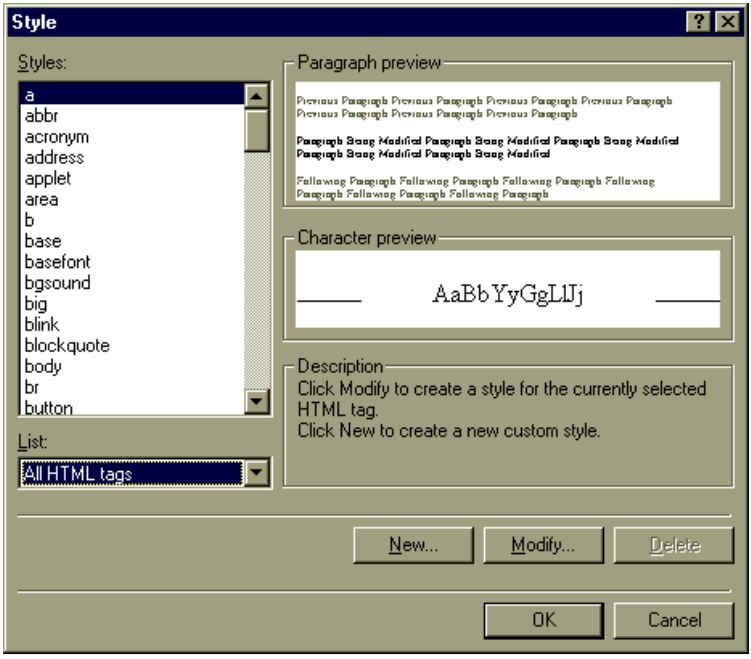

Рис. 1.32. Диалоговое окно **Style**

В списке **Styles** показаны все теги HTML, которые распознаются и используются FrontPage 2000. В этом списке нам необходимо выбрать тот тег, который ответственен за создание заголовка первого уровня. Это, естественно, `h1`. Теперь мы можем его изменить. Для этого мы используем кнопку **Modify**. Она визуализирует очередное диалоговое окно **Modify Style**. Само по себе это окно не очень информативно. В поле **Name (selector)** отображается тег, стилевое оформление которого мы собираемся модифицировать, в блоке **Preview** весьма схематично показывается внешний вид нового текстового оформления. Нам в этом окошке потребуется кнопка **Format**. Будучи нажатой, она показывает меню из пяти пунктов, каждый из которых является частью внешнего оформления текстового блока. Пункт **Font** позволяет изменять шрифт, приписанный к этому стилю. Альтернатива **Paragraph** показывает диалоговое окно, которое позволяет управлять отступами и выравниванием. Пункт **Border** ответственен за указание внешнего вида границ. Пункт **Numbering** применяется в том случае, если необходимо использовать какие-либо списки, а, следовательно, требуется выставить параметры их отображения. И, наконец, альтернатива меню **Position** позволяет указывать правила точного позиционирования текста. Все эти действия производятся при помощи стандартных диалоговых окон, и работа с ними не представляет каких-либо затруднений. После изменения параметров какой-либо части текстового оформления результат должен показываться в окошке предварительного просмотра **Preview**. Но очень часто получается так, что эти изменения достаточно малы, и внешний вид демонстрируемого текста отличается от предыдущего варианта не слишком сильно. Здесь, конечно же, спасает только просмотр созданной страницы при помощи браузера. После завершения всей работы достаточно нажать кнопки **OK** во всех диалоговых окнах и посмотреть на результаты своей работы. Если мы точно знали, что делали, они не должны нас сильно разочаровать. Все заголовки первого уровня должны изменить свой вид. Теперь посмотрим, какой ценой это было достигнуто, то есть, что изменилось в HTML-представлении нашей страницы. Если мы перейдем на соответствующую страницу и посмотрим код, то увидим, что появился новый блок кода:

```
<style>
<!--
h1            { font-family: Arial; text-transform: uppercase; font-
weight: bold }
-->
</style>
```

Итак, все локальные определения стилей для данного документа размещаются в теле документа, сразу после тега `<body>`. Блок определения стилей ограничен тегами `<style>` и `</style>`. В самом блоке FrontPage 2000 помещает идентификаторы стилей, в нашем случае это `h1`, а в фигурных скобках описывает их оформление. В вышеприведенном коде указано, что для отображения заголовков первого уровня будет использоваться шрифт из семейства Arial (`font-family: Arial`), текст будет отображаться только заглавными

буквами (`text-transform: uppercase`) полужирного начертания (`font-weight: bold`).

Таким способом мы можем изменить любой стиль, используемый при работе с FrontPage 2000. Вполне вероятно, что этого будет недостаточно. Что ж, мы можем создавать и применять свои стили. В диалоговом окне (см. рис. 1.32) мы видим выпадающий список **List**, который управляет отображаемым списком стилей. По умолчанию используется режим **All HTML tags**, который показывает все теги HTML. Но в списке **List** есть еще значение **User-defined styles**, которое визуализирует список всех стилей, созданных или модифицированных пользователем. После того, как мы изменили стиль `h1`, он должен отобразиться в этом списке. Если мы нажмем кнопку **New**, то сможем создать свой собственный стиль. Для создания стиля используется диалоговое окно **New Style**, которое является полным функциональным аналогом диалогового окна **Modify Style**, которое мы рассматривали несколько ранее. Прежде чем приступать к заданию параметров текстового оформления стиля при помощи кнопки **Format**, стоит указать имя нового стиля, под которым он будет использоваться в коллекции стилей FrontPage 2000. Для этого, как всегда, используется поле ввода **Name (selector)**. Если мы правильно выполним все действия, то потом в списке стилей мы сможем увидеть свой, только что определенный стиль. Стили, создаваемые пользователем, в терминологии FrontPage называются классами, и родительским классом для них является стиль Normal. Поэтому если мы создали стиль **ms1**, то его полное имя будет выглядеть как **Normal.ms1**. Часто имя родительского класса опускается. Поэтому, если мы будем рассматривать HTML-реализацию определения стиля, то его имя будет ".ms1". Впрочем, это легко видеть из нижеприведенного кода.

```
<style>
<!--
.ms1          { text-align: Right; line-height: 200%; font-family:
Century Gothic;
                font-style: italic }
-->
</style>
```

Здесь, в качестве стиля определен текст, прижатый к правому краю страницы, отображаемый шрифтом Century Gothic с курсивным начертанием. Также используется двойной межстрочный интервал. Если же мы обратим внимание на текст, которому приписан данный стиль, то увидим следующую конструкцию:

```
<p class="ms1">Эпиграф</p>
```

Здесь у тега `<p>` появляется параметр `class`, в качестве значения которого используется имя определенного выше класса. Все достаточно стройно и логично.

Применение созданных стилей не вызывает каких-либо проблем. Для того, чтобы какому-либо блоку текста приписать какой-либо стиль, его нужно,

как всегда, выделить и выбрать необходимый стиль из выпадающего списка **Style** на панели форматирования.

Как мы помним, при создании встроенного стиля происходит расширение стиля **Normal**, но у нас всегда есть возможность расширить и остальные стили, которые заданы изначально. Так, например, если мы будем делать ненумерованный список, а потом вызовем команду **Format** для одного из его пунктов, в появившемся диалоговом окне будет находиться уже знакомая кнопка **Style**, при нажатии на которую отобразится диалоговое окно, позволяющее напрямую задавать тот стиль, который мы ранее объявили в этом документе. Если мы припишем элементу списка наш стиль **ms1** подобным образом, то он будет отображен при помощи стиля **Bulleted list.ms1**.

Мы рассмотрели создание и использование стилей, встроенных в отдельный HTML-документ. Необходимо напомнить, что стили, определенные в одном документе, не могут использоваться в оформлении других Web-страниц. Другие документы их просто не видят. Естественно, подобное решение не очень хорошо подходит для оформления всего сайта. Нам бы хотелось определить необходимые стили сразу для всех страниц, входящих в состав сайта. Для этих целей идеально подходят файлы CSS (Cascading Style Sheets) — каскадные таблицы стилей. В этих файлах, которые представляют собой неотображаемые документы, содержатся объявления всех стилей, применяемых для оформления Web-страниц. Останется только явно подключить эти таблицы к HTML-файлам и стили будут доступны для использования. FrontPage 2000 создает подобные файлы в виде специальных страниц. Для создания нового файла со стилевой таблицей необходимо выполнить команду меню **File/New/Page** и в появившемся диалоговом окне выбрать вкладку **Style Sheets**. На ней мы увидим список готовых стилевых таблиц. Для начала рассмотрим создание собственной таблицы стилей, а уже после этого проведем обзор заранее приготовленных для нас решений.

Для создания своей собственной каскадной таблицы стилей в диалоговом окне **New** необходимо выбрать образец **Normal Style Sheet**. При этом создается пустая страница с именем по умолчанию **new_page_x.css**, где в качестве **x** указывается номер создаваемой страницы. При создании страницы активизируется инструментальная панель **Style** с одной-единственной одноименной кнопкой при нажатии которой появляется уже знакомое нам диалоговое окно **Style**, используемое для создания новых и переопределения уже существующих стилей. После внесения необходимых изменений на вкладке HTML появляется список всех измененных и созданных стилей вместе с их определениями. Выглядит это следующим образом:

```
big          { font-size: 18pt }
.ms2         { font-family: Matisse ITC; font-size: 12pt; font-
weight: bold }
h1           { font-family: Arial; text-transform: uppercase; font-
weight: bold }
h3           { font-family: Times New Roman; color: #0000FF }
```

Здесь указано, что был переопределен способ отображения укрупненного текста (тег `big`). Теперь он будет отображаться шрифтом с размером 18 пунктов. Определен свой собственный стиль `ms2`, которому приписан шрифт Matisse ITC, размер 12 пунктов, полужирное начертание. Также были переопределены стили для отображения заголовков первого (`h1`) и второго (`h2`) уровней. Для первого был указан шрифт семейства Arial в полужирном начертании, причем все буквы текста с этим стилем автоматически будут преобразованы в заглавные (`text-transform: uppercase`). Для заголовков второго уровня зарезервирован шрифт Times New Roman, текст отображается синим цветом (`color: #0000FF`). Остальные параметры этого стиля не изменены. Если мы сохраним эту страницу, и просмотрим ее обычным текстовым редактором, то увидим, что вышеприведенный текст и является ее кодом.

Теперь попробуем подключить созданную стилевую таблицу к какой-либо Web-странице. Для этого необходимо в обычном режиме работы выполнить команду меню **Format/Style Sheet Links**. При этом активизируется диалоговое окно **Link Style Sheet** (рис. 1.33). В нем мы можем указать, будет подключаться стилевая таблица ко всем страницам сайта (альтернатива **All pages**) или к ограниченному их количеству (радиокнопка **Selected page(s)**). Имена файлов подключаемых внешних стилевых таблиц (а подключать мы можем сразу несколько таблиц) отображаются в основном списке **URL**. Естественно, мы указываем не просто имена, а местонахождение этих файлов, поэтому у нас есть возможность подключать таблицы прямо из Интернета. Но делать этого не стоит, если вы заботитесь об удобстве работы с сайтом, а лучше все подключаемые ресурсы держать непосредственно на сервере.

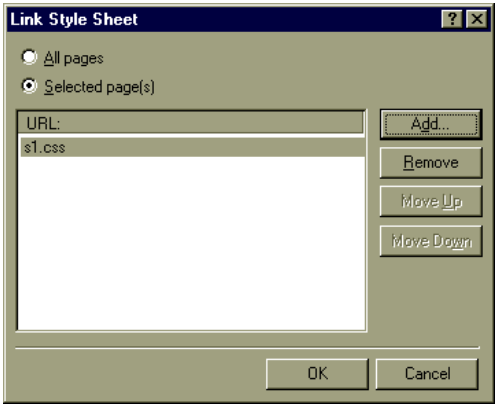

Рис. 1.33. Диалоговое окно **Link Style Sheet**

Для добавления файла в список доступных стилевых таблиц нужно нажать кнопку **Add**, которая активизирует стандартный диалог для подключения гиперссылки. Для удаления файла из этого списка используется кнопка **Remove**. Для изменения порядка расположения файлов применяются кнопки

Move Up и **Move Down**. Это удобно применять в том случае, если в нескольких стилевых таблицах есть переопределение одного и того же стиля. В этом случае в документе будет использовано то определение, которое расположено последним.

После подключения внешней стилевой таблицы в коллекции предопределенных стилей появятся и все стили, созданные нами. Пользуйтесь на здоровье.

Для подключения внешних стилевых таблиц используется HTML-конструкция:

```
<link rel="stylesheet" type="text/css" href="s1.css">
```

Здесь мы видим основной тег `<link>` и несколько параметров. Параметр `rel` указывает на содержание подключаемого ресурса. В нашем случае ему приписано значение `"stylesheet"`, идентифицирующее подключаемый ресурс как таблицу стилей. Для указания типа файла, в котором эта таблица стилей содержится, использован параметр `type` со значением `"text/css"`. И, наконец, самый главный параметр — `href`. В качестве его значения записывается URL файла с подключаемой таблицей.

Мы рассмотрели возможности создания собственных внешних каскадных таблиц стилей. Теперь пришла очередь стилевых таблиц, которые входят в коллекцию FrontPage 2000. Для использования какой-либо, заранее приготовленной добрыми разработчиками, каскадной таблицы необходимо, как и прежде, выполнить команду меню **File/New/Page** и перейти на вкладку **Style Sheets**. Все элементы списка за исключением уже рассмотренного нами **Normal Style Sheet** являются готовыми таблицами. Обзор будем производить в алфавитном порядке.

Таблица **Arcs** предназначена для оформления страниц в желто-коричневом цвете. Основной текст записывается шрифтом Verdana коричневого цвета. Используется следующая конструкция:

```
body
{
       font-family: Verdana, Arial, Helvetica;
       background-color: rgb(255,255,204);
       color: rgb(102,102,51);
}
```

Здесь видно, что при отсутствии на компьютере посетителя сайта шрифта Verdana необходимо использовать шрифт семейства Arial. А если нет даже его — шрифт Helvetica. Цвет шрифта и фона задается прямым указанием его RGB-кода. Также в этой таблице явно задаются стили всех заголовков. Для них зарезервированы шрифты семейства Times. Также для каждого уровня заголовка явно указан цвет шрифта. А в самом начале стилевой таблицы объявляется стиль для тега `<a>`. Как мы помним, этот тег применяется для создания гиперссылок. У всех гиперссылок есть три состояния: обычное отображение, пройденная гиперссылка и активная. Обычное отображение задается

классом `a:link`, для пройденной заготовлен класс `a:visited`, а активная гиперссылка отображается при помощи класса `a:active`. В нашей стилевой таблице не налагается каких-либо условий на выбор шрифта для отображения гиперссылок. В определениях этих классов задан только цвет шрифта.

Следующей в коллекции расположена стилевая таблица **Bars**, которая содержит определения все тех же элементов страниц. Основной текст при использовании этой таблицы отображается шрифтом Arial, для заголовков, как и прежде, приготовлен шрифт Times, а фон страниц приобретет нежный оливковый цвет. Web-страницы, оформленные при помощи каскадной стилевой таблицы **Bars**, на мой взгляд, являются достаточно сбалансированными.

Стилевая таблица **Blocks** для основного текста и для заголовков применяет шрифт Bookman Old Style. Но гиперссылки оформляются при помощи красного цвета, а сам фон страниц приобретает светло-серебряный цвет. Сочетание достаточно агрессивное. Но нельзя не признать, что вкус присутствует.

Таблица **Blueprint** на желтом фоне отображает текст и заголовки при помощи шрифта Century Gothic. Для гиперссылок предусмотрен темно-красный, практически пурпурный цвет. Как выглядит красное на желтом, я думаю, каждый может себе представить.

Шаблон **Capsules** использует светло-зеленый фон и красно-оранжевые гиперссылки. С подобным вариантом оформления страниц часто приходится стакиваться на сайтах, чьи владельцы заинтересованы в том, чтобы посетители активно пользовались гиперссылками. Подобное оформление очень резко выделяет ссылки, и, как утверждают некоторые исследования, интенсивность использования гиперссылок в подобных контрастных страницах несколько выше средней. Для отображения текста и заголовков в этой стилевой таблице указан шрифт семейства Arial.

Стилевая таблица **Downtown** для обычного текста использует шрифт Garamond желтого цвета, для заголовков припасен шрифт Verdana, фон обещает быть синим, а гиперссылки оранжевыми. Опять контраст.

Следующая по очереди (или по алфавиту) — каскадная стилевая таблица **Expedition**. Очень сдержанный образец оформления. На персиковом фоне отображается содержимое страницы шрифтом Book Antiqua.

Одна из наиболее агрессивных схем оформления носит название **Highway**. На черном фоне отображается белый текст, набранный шрифтом Verdana. Гиперссылки — оранжевые. Без комментариев.

Но еще сильнее выражает эту идею оформления стилевая таблица **Neon**. Текст отображается ядовито-зеленым цветом на черном фоне. Шрифт, как и в предыдущей схеме, — Verdana. Что тут можно сказать? Неон, он и есть неон.

Стилевая таблица **Poetic** должна, видимо, применяться для оформления страниц в несколько лирическом стиле. Все очень скромно и достойно. Пурпурный текст, набранный шрифтом Book Antiqua на белом фоне. Для заголовков используется все тот же шрифт.

Схема **Street** используется для неформального оформления страниц. Подобный эффект достигается использованием шрифта Verdana для основного текста и Comic Sans MS для заголовков.

И последняя стилевая таблица **Sweets** для текста использует лаконичный Arial, а для заголовков его несколько более претенциозную версию — Arial Rounded MT Bold. Все это великолепие отображается на нежном желтом фоне страницы.

Необходимо отметить, что все эти схемы напрямую рассчитаны на удаленного пользователя, работающего на платформе Wintel (Windows + Intel), так как большинство из применяемых шрифтов наиболее распространены именно там.

На этом обзор предустановленных каскадных стилевых таблиц заканчивается. Но в этой главе мы должны рассмотреть еще один вариант единообразного оформления страниц. Он основан на применении так называемых "тем оформления". С этим понятием сталкивался любой пользователь пакета MS Plus. В нем содержались комплекты настроек для внешнего вида Windows, включавшие в себя фон рабочего стола, стандартные иконки, обычные и анимированные курсоры, звуковые схемы, хранители экрана и т. д. Все эти компоненты были подобраны таким образом, чтобы создавать стилевое единство. Подобная концепция была перенесена и в семейство MS Office. Так, например, в известном всем MS Word есть свои темы оформления для текстовых документов. Естественно, что FrontPage 2000 как член семейства MS Office перенял эту возможность. Для оформления текущей страницы, или всех Web-страниц, входящих в состав проектируемого сайта, с применением какой-либо темы, необходимо выполнить команду меню **Format/Theme**. При этом активизируется диалоговое окно **Themes** (рис. 1.34).

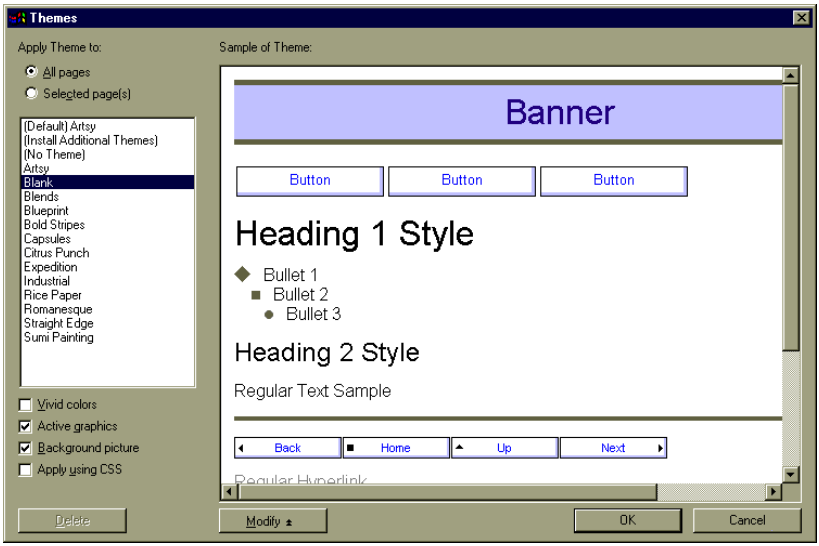

Рис. 1.34. Диалоговое окно **Themes**

Прежде всего, необходимо указать границы применения темы оформления при помощи группы радиокнопок **Apply Theme to**. Она может быть использована только в пределах одной или нескольких выбранных страниц (альтернатива **Selected page(s)**). В том случае, если планируется данное оформление применить ко всему сайту, а это достаточно логичное решение, необходимо использовать радиокнопку **All pages**. После указания границ распространения можно из основного списка выбрать саму тему. При перемещении по списку в окне предварительного просмотра **Preview** показывается образец текущей темы. На рисунке виден образец темы **Blank**.

Темы оформления Web-страниц позволяют определить внешний вид стандартного заголовка страницы, называемого также баннером, кнопок с привязанными гиперссылками, фоновый рисунок, вид обычного текста, заголовков первого и второго уровней, списков, обычных горизонтальных линий, кнопок панели навигации и, естественно, гиперссылок всех трех видов (стандарт, активная, пройденная).

Любую тему оформления пользователь может изменить по своему вкусу. Для этого используется кнопка **Modify**. При нажатии на нее в окне появляются пять дополнительных кнопок. Сама тема всегда состоит из цветовой схемы, набора применяемых рисунков и предустановленного шрифтового оформления. Все эти наборы объектов изменяются при помощи специальных диалоговых окон, которые активизируются кнопками **Colors**, **Graphics** и **Text** соответственно. После изменения необходимых параметров схему можно сохранить при помощи кнопки **Save**, или записать ее под другим именем при помощи кнопки **Save As**. Естественно, после сохранения под другим именем образуется новая тема, которая также будет потом предлагаться в общем списке, так что у пользователя всегда есть возможность создавать свои собственные темы оформления. Искренне рекомендую воспользоваться этой возможностью, так как эксклюзивный дизайн всегда ценится выше.

После выбора необходимой темы оформления, в HTML-коде создаваемой Web-страницы сразу после раздела заголовка появляется тег `<meta name="Microsoft Theme" content="blank 1111, default">`.

Вид этого тега сразу должен навести нас на некоторые размышления. С тегами `<meta>` мы уже встречались. Мы помним, что они обрабатываются браузером в самом начале работы и несут в себе информацию, которая должна помочь правильному отображению загруженной Web-страницы. В этом теге обычно присутствуют параметры `name` и `content`. Первый указывает категорию оформления, а второй — конкретное значение из этой категории. Как мы видим, в нашем случае оба этих параметра присутствуют. Но значение параметра `name` равно `"Microsoft Theme"`. То есть, если на машине удаленного пользователя есть Microsoft Office с установленными темами, и он использует Internet Explorer, то оформление Web-страницы гарантированно будет таким, как мы и задумывали. Но если посетитель нашего сайта "сидит" на Макинтоше, то вероятность адекватного отображения

страницы резко падает. Радует только то, что те теги, которые браузер не может опознать, просто игнорируются. Следовательно, в этом случае, пострадает только оформление, а "начинка" страницы останется на месте. Упомянутая особенность является частью достаточно основательной проблемы. Каждый браузер поддерживает свое, отличающееся от других, подмножество тегов HTML. Поэтому, если мы собираемся создавать сайт, который гарантированно будет правильно и адекватно отображаться на всех типах систем, следует пользоваться наиболее распространенными тегами и системно-независимыми средствами обобщенного оформления. Лучше всего, использовать каскадные стилевые таблицы. Действительно, темы сильно облегчают и ускоряют процесс единообразного оформления сайта, но они не оригинальны. Вы же не хотите, чтобы посетители сайта увидели оформление, которое очень часто встречается в Интернете. Если хочется действительно оригинального и стильного оформления, придется достаточно много поработать.

Основная идея этой главы, наверное, заключается в том, что единое оформление для сайта — это правильно и хорошо, но оно должно разрабатываться самостоятельно. Не стоит использовать всем знакомые образцы дизайна.

Режимы работы

Создание сайта это не только проектирование Web-страниц. При создании сайта всегда необходимо держать в голове его внутреннюю структуру, схему внешних и локальных гиперссылок, поддерживать правильную схему навигации, планировать свою работу и осуществлять контроль. Все эти функции поддерживаются FrontPage 2000. Выбор того или иного режима работы осуществляется при помощи кнопок, собранных в окне **Views**. При запуске FrontPage по умолчанию используется режим проектирования Web-страницы. То есть кнопка-переключатель **Page** находится в нажатом состоянии. С этим режимом мы уже знакомы.

Следующий режим предназначен для отображения структуры сайта. Как мы помним, хороший сайт создается на основе структуры папок. В одну из них помещаются графические файлы, в другую активные элементы, некоторые папки используются для хранения Web-страниц, освещающих одну тему и т. д. Нажатие на кнопку **Folders** переводит FrontPage 2000 в режим отображения этой структуры (рис. 1.35). Больше всего это похоже на стандартный проводник Windows. В левой части окна с названием **Folder List** показывается древовидная структура папок создаваемого сайта. Правая часть служит для отображения более детальной информации, то есть помимо папок показываются еще и файлы. Здесь необходимо отметить, что показ древовидной структуры сайта может быть включен и в режиме **Page**. Для этих целей служит кнопка-переключатель **Folder List** на основной панели инструментов или команда меню **View\Folder List**. Естественно, в режиме **Folders** отображение дерева сайта является обязательным, поэтому вышеуказанные кнопка и команда меню становятся недоступными.

Данный режим работы FrontPage 2000 является, по большей части, просто информационным. Его основная задача — дать максимально полную информацию о файловой структуре создаваемого сайта. Поэтому основная часть рабочего пространства выстроена в виде таблицы. Это достаточно хорошо видно на рисунке. В колонке **Name** показывается имя файла или папки. Колонка **Title** предназначена для отображения заголовка страницы или наименования остальных элементов, хранящихся в отдельных файлах в составе сайта. Колонки **Size** и **Type** показывают соответственно размер файла и его расширение. Дата и время последнего изменения какого-либо файла показываются в колонке **Modified Date**. Так как все семейство программ Microsoft Office, а значит и FrontPage 2000 в его составе, рассчитано на работу в группе, когда над одним и тем же проектом работает несколько человек, то возникает необходимость указывать, кто именно изменял тот или иной файл последним. Имя этого человека отображается в колонке **Modified By**. И, наконец, последняя колонка **Comments** предназначена для размещения комментариев относительно какого-либо файла или какой-либо папки. Для добавления комментариев к одному из объектов необходимо в контекстном меню выбрать команду **Properties**, а в появившемся диалоговом окне — вкладку **Summary**. На ней и будет размещено поле для ввода комментариев.

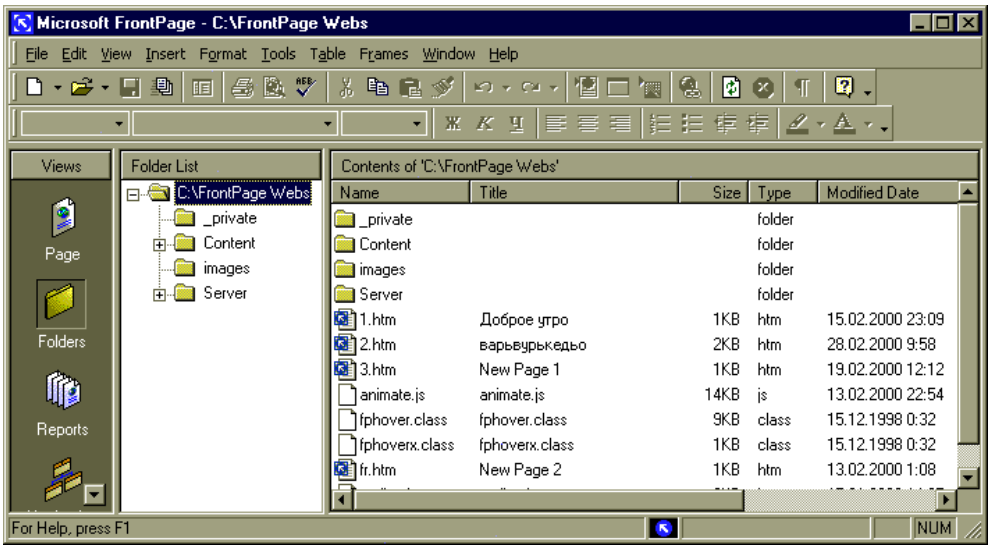

Рис. 1.35. Режим работы **Folders** приложения FrontPage 2000

Следующий режим работы — **Reports**. Он позволяет получить практически всю необходимую информацию о нашем сайте. При выборе этого режима работы автоматически активизируется панель инструментов **Reporting**. По умолчанию показывается статистика всего сайта, то, что называется **Site Summary** (рис. 1.36). Каждая строка таблицы представляет собой сводный

отчет по какой-либо категории статистики. В том случае, если необходимо получить полный отчет, требуется выбрать соответствующую категорию в выпадающем списке **Report** инструментальной панели **Reporting** или произвести двойной щелчок на соответствующей строке отчета. Также может быть использована команда меню **View/Reports**.

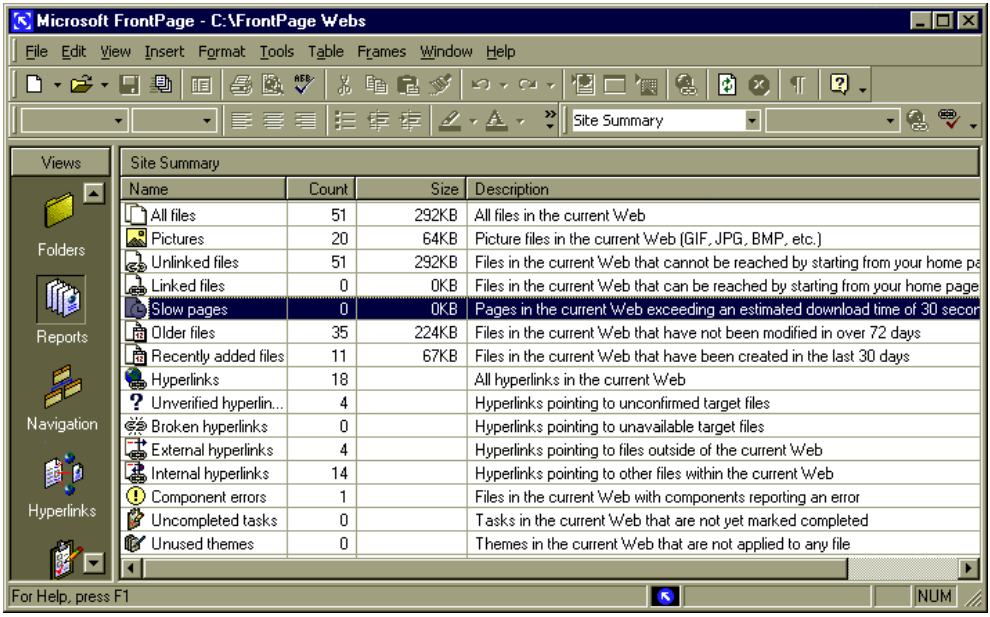

Рис. 1.36. Режим работы **Reports** приложения FrontPage 2000

Итак, пройдемся по всему полному отчету, чтобы узнать, что показывается в каждой строке. Строка **All files** показывает количество и общий размер всех файлов, входящих в наш сайт. По сути, в этой строке показывается объем всего сайта. В строке **Pictures** отображается количество и общий объем графических файлов, использованных при оформлении сайта. Количество файлов, не связанных напрямую с основной страницей сайта, показывается в строке **Unlinked files**. Соответственно, информация о файлах, связанных с этой основной страницей, отображается в строке **Linked files**. Строка **Slow pages** показывает количество HTML-файлов, для загрузки которых требуется более тридцати секунд. То есть, так называемые "медленные страницы". Конечно, временной критерий в данном случае кажется немного завышенным. Так, по данным последних исследований компании Zona Research, максимальное время, в течение которого пользователь обычно готов дожидаться ответа сервера, составляет всего восемь секунд. Поэтому всегда стоит держать этот критерий в голове при создании Web-страниц. Естественно, при получении статистики хотелось бы иметь возможность для изменения временного критерия при определении медленных страниц. Для этого в выпадающем списке **Report** инструментальной панели **Reporting** необходимо

выбрать значение **Slow pages**, а в списке **Report Setting** выбрать необходимое значение временного интервала.

В строке **Older files** показывается количество файлов, которые не изменялись в течение определенного количества дней. По умолчанию, этот временной промежуток равен семидесяти двум дням, но, как и в случае с определением медленных страниц, мы можем его изменить. Для этого в выпадающем списке **Report** выбирается соответствующее значение, в данном случае это **Older files**, и в выпадающем списке **Report Settings** можно указать необходимый срок. При этом вся информация о подобных файлах будет показана детально: с именами, типами, сроками последнего изменения и т. д.

Следующая строка весьма похожа на предыдущую. Но в строке с названием **Recently added files** показывается общее количество файлов, которые были добавлены к сайту в течение последних нескольких дней. По умолчанию используется тридцатидневный срок. Изменение его производится в режиме детального просмотра этой части статистики, которая доступна при использовании все того же выпадающего списка **Report** и соответствующего одноименного значения.

Следующие четыре строки предназначены для отображения информации о гиперссылках, размещенных в теле Web-страниц нашего сайта. Каждую гиперссылку можно и нужно проверить. Для этих целей применяется кнопка **Verify Hyperlinks** на инструментальной панели **Reporting** или команда меню **Tolls/Recalculate Hyperlinks**. Количество непроверенных на данный момент гиперссылок отображается в строке **Unverified hyperlinks**. Информация обо всех неработающих гиперссылках собирается в строке **Broken hyperlinks**. Как мы помним, гиперссылки разделяются на внешние и внутренние. Те ссылки, которые ведут удаленного пользователя на другие страницы этого же сайта, считаются внутренними. Информация о них отображается в строке **Internal hyperlinks**, а о внешних — в строке **External hyperlinks**.

Мы уже знаем, что при организации функционального сайта трудно обойтись без активных элементов. Естественно, их необходимо тестировать. Некоторые ошибки в размещении и настройке активных элементов FrontPage 2000 может находить сам. Информация о таких ошибках суммируется в строке **Component error**. Естественно, для того, чтобы узнать, в каких именно элементах допущены ошибки, необходимо затребовать более детальную информацию.

Несколько позже в этом разделе мы будем рассматривать средства планирования и организации работы в FrontPage 2000. Эти средства строятся на применении задач. Например, если мы планируем через некоторое время создать страницу в сайте, то для напоминания об этом создается задача. После ее выполнения мы присваиваем ей статус законченной. FrontPage позволяет увидеть количество незавершенных задач. Сводная информация о них помещается в строку **Uncompleted tasks**.

И, наконец, последняя строка из этой всеобщей статистики относится к оформлению сайта. Да-да, к тому самому, которое мы разбирали в предыдущем разделе. В его последней части мы рассматривали использование тем при оформлении сайта. Как известно, для сайта можно указать одну или несколько тем. Так вот, последняя строка с именем **Unused themes** показывает количество тем, которые были выбраны для использования в сайте, но так и не были применены ни к одной из страниц.

Из режимов просмотра статистики стоит упомянуть, пожалуй, еще режим **Categories**, который можно установить только из выпадающего списка **Report**. Он показывает, какие категории приписаны тем или иным объектам, входящим в состав файла. Механизм использования категорий мы рассматривали немного выше, в разделе, посвященном созданию активных элементов. Также стоит воспользоваться режимом **Publish Status**. Он позволяет узнать, какие элементы сайта успешно прошли процедуру опубликования, а какие — нет.

Следующий режим работы FrontPage 2000 позволяет разобраться во внутренней навигации сайта. Рабочее поле FrontPage (рис. 1.37) в этом режиме опять разбивается на два окна, в одном из которых отображается дерево файла. Впрочем, в этом режиме его можно скрывать и показывать по желанию, так как соответствующая кнопка и команда меню доступны. Данный режим чрезвычайно полезен для получения наглядной информации о внутренней структуре и логике сайта.

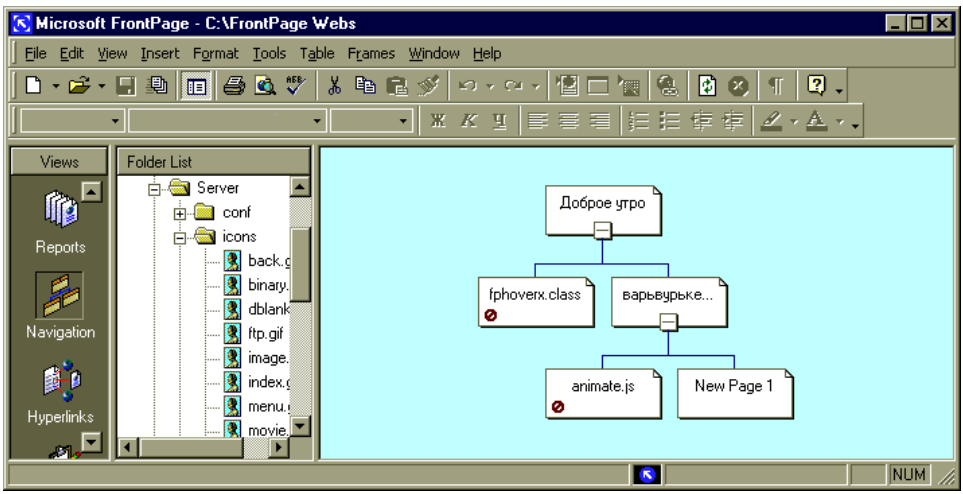

Рис. 1.37. Режим работы **Navigation** приложения FrontPage 2000

Естественно, вся навигация по сайту происходит только при помощи гиперссылок. Для того чтобы увидеть структуру внешних и внутренних связей какой-либо страницы, необходимо использовать режим **Hyperlinks** (рис. 1.38). Процедура работы в этом режиме достаточно проста. В окне

Folder List, содержащем дерево сайта, выбирается нужная Web-страница. Ее входящие и исходящие ссылки схематично показываются в основном окне. При этом сами гиперссылки отображаются синими стрелками, а связи, не являющиеся ссылками, например, с внедренными активными объектами, — коричневыми линиями. При этом те страницы, которые также содержат гиперссылки, отображаются со встроенным значком плюса. На рисунке видно, что именно таким способом показан файл с именем 2.htm. Щелчок на значке этого файла позволяет отображать гиперссылки и для него. При этом знак плюса заменяется на знак минуса. Подобным образом можно в рабочем поле развернуть всю схему внешних и локальных гиперссылок сайта. Естественно, при создании хорошего, объемного сайта эта схема будет достаточно велика. Для расположения той или иной страницы в центре рабочего поля, что более удобно при рассмотрении схемы ее гиперссылок, можно кроме полос прокрутки воспользоваться командой контекстного меню **Move to Center**. Эта команда, естественно, действует для всех страниц, кроме той, которая была выбрана для начала отображения этого режима. Подобная стартовая страница показывается при помощи несколько увеличенного значка. На рисунке этим значком отображена страница 1.htm. Команда **Move to Center** не только перемещает выбранную страницу в центр рабочего поля, но и делает ее основной для данного режима. Двойной щелчок по значку какой-либо Web-страницы, переводит FrontPage в режим проектирования страницы **Page** и позволяет редактировать ее содержимое.

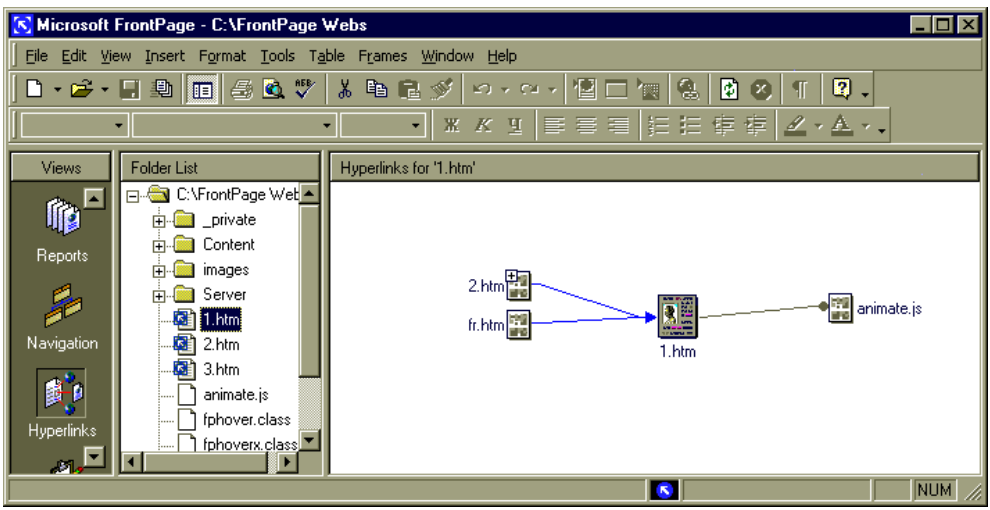

Рис. 1.38. Режим работы **Hyperlinks** приложения FrontPage 2000

Последний режим работы, предусмотренный для FrontPage 2000, предназначен для организации общего цикла работы. Режим **Tasks** позволяет контролировать степень выполнения запланированных заданий. Как мы уже говорили, планирование любых действий в FrontPage 2000 осуществляется

посредством назначения задач. Практически любое действие можно отложить на будущее, создав для него задачу. Добавление задачи производится либо при помощи прямого выполнения команды меню **Edit/Task/Add Task**, либо при помощи соответствующей кнопки, которая присутствует во многих диалоговых окнах. При этом визуализируется диалоговое окно **New Task** (рис. 1.39). В поле **Task name** записывается наименование задачи, которое будет ее идентифицировать. Группа радиокнопок **Priority** позволяет назначить приоритет задачи. Приоритетов, то есть степеней важности и срочности, всего три. По умолчанию выставляется средний приоритет (**Medium**). Назначенную задачу надо кому-то выполнять, и для указания человека, которому придется это делать, служит выпадающий список **Assigned to**. В нем помещаются имена всех членов рабочей группы, в которой прописан данный компьютер. Для описания самого задания, то есть того объема работы, который необходимо выполнить, служит поле текстового ввода **Description**. После ввода всех необходимых данных задачу необходимо записать. После того, как нашему сайту будет приписана хотя бы одна задача, использование режима **Tasks** станет оправданным. Задачи могут быть привязаны как к каким-либо Web-страницам, так и быть независимыми. Если задача создается в момент редактирования какой-либо страницы, то в строке **Associated with** записывается имя html-файла, содержащего данную Web-страницу. Для привязанных таким образом задач действует команда меню **Edit/Task/Start**. Данная команда становится доступной для выбора пользователем в том случае, если действует режим работы **Tasks**, и выделена задача, связанная с какой-либо страницей. Выполнение этой команды влечет за собой переход в режим редактирования Web-страницы **Page** и загрузку в рабочее пространство соответствующего HTML-файла.

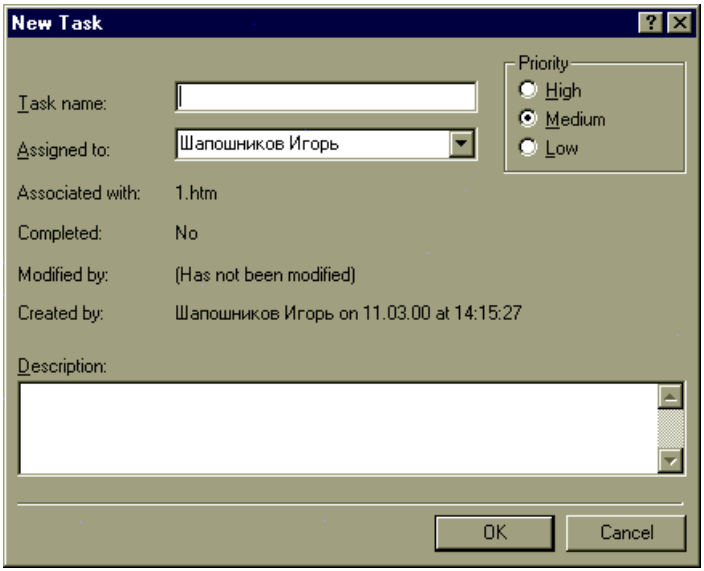

Рис. 1.39. Диалоговое окно **New Task**

Как мы уже говорили, управление задачами производится в режиме **Tasks** (рис. 1.40). Нас будет интересовать общий ход работы над сайтом, который легко контролировать по степени выполнения задач. Каждая задача имеет определенный статус, который показан в колонке **Status**. Обычно используется три состояния: работа над задачей еще даже не начиналась (**Not Started**), начата, но еще не закончена (**Not Completed**), или работа завершена (**Completed**). Первый статус снимается после выполнения команды **Start Task** контекстного меню или уже знакомой нам команды **Edit/Task/Start**. Соответственно, после того, как задача выполнена, ей необходимо присвоить соответствующий статус. Делается это при помощи команды **Edit/Task/Mark as Completed** или одноименной команды контекстного меню.

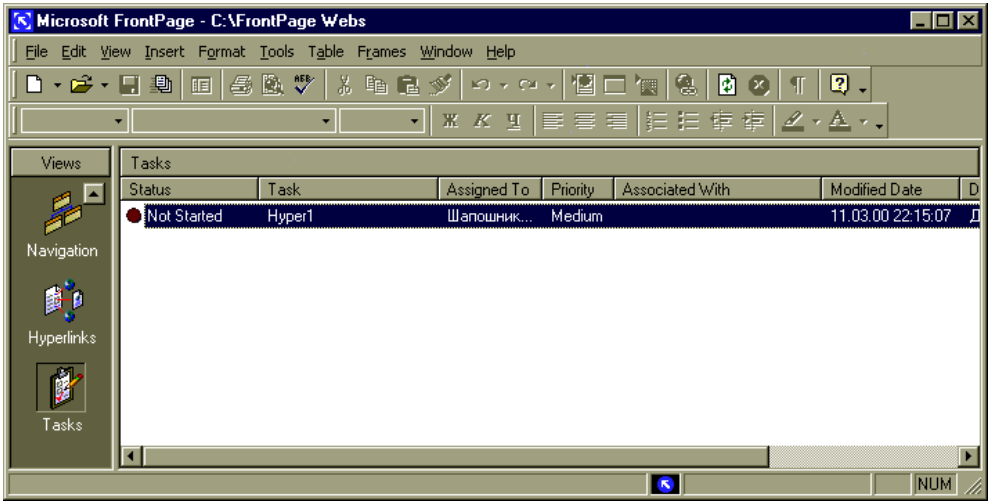

Рис. 1.40. Режим работы **Tasks** приложения FrontPage 2000

Естественно, для полноценной работы с механизмом задач нужно не только иметь возможность изменять их статус, необходимо еще и получать информацию о них. Вся возможная информация о назначенных задачах отображается в основной таблице рабочего режима **Tasks**. О колонке **Status** мы уже говорили. Колонка **Task** содержит идентифицирующее короткое имя задачи. Имя человека, ответственного за ее выполнение, помещается в колонке **Assigned To**, а ее приоритет в колонке **Priority**. В том случае, если задача привязана к какой-либо Web-странице, имя соответствующего HTML-файла пишется в колонке **Associated With**. Дата и время последнего изменения состояния задачи указываются в колонке **Modified Date**. Ну, а полная расшифровка задания прописывается в колонке **Description**.

Данные о любой задаче могут быть изменены посредством прямого ввода, минуя стандартные механизмы изменения статуса. Для этого используется команда контекстного меню **Edit Task**. Эта команда активизирует диалоговое окно **Task Details**, которое является функциональным двойником окна

New Task (см. рис. 1.39). При этом те параметры задачи, которые могут быть изменены, помещаются в соответствующих элементах управления. Хотя, конечно, может возникнуть ситуация, когда проще уничтожить задачу, а потом создать новую. Для уничтожения задачи используется команда контекстного меню **Delete**.

На этом обзор режимов работы FrontPage 2000 можно закончить и перейти к примерам создания Web-страниц и целых сайтов.

Готовые страницы

Качество сайта и его посещаемость, то есть те параметры, которые мы всегда стараемся улучшить, напрямую зависят от правильной структуры сайта и страниц, входящих в его состав. Каждый из нас, поработав некоторое время в Интернете, может составить свое мнение о структуре того или иного сайта. Очень часто хорошее наполнение страдает из-за неправильного оформления. Для того чтобы найти правильный баланс между жаждой оригинальности самовыражения и здоровой консервативностью необходим большой опыт работы и множество специальных знаний и навыков. Что же делать, если опыт работы Web-мастером еще откровенно невелик и на стене не висит диплом дизайнера? Одним из выходов может быть использование заранее заготовленных образцов Web-страниц. В нашем случае это будут обычные шаблоны FrontPage 2000. Доступ к их коллекции происходит при выполнении команды **File/New/Page**. В появившемся диалоговом окне **New** на вкладке **General** помимо обычного шаблона **Normal Page**, который создает пустую страницу без какого-либо наполнения, доступны еще двадцать четыре шаблона и один мастер. Эта группа шаблонов демонстрирует все наиболее удачные варианты компоновки содержимого страниц и основные правила оформления текста. Пройдем последовательно по всем шаблонам, а потом уже рассмотрим единственный мастер.

Итак, первый шаблон с наименованием **Bibliography** не демонстрирует особых изысков верстки. Он формирует весьма консервативного вида страницу, которая обычно применяется для списка использованной литературы, так называемой библиографии. На созданной странице располагается три строки, которые содержат шаблон строки, содержащей информацию об одном из использованных источников. Причем фамилии авторов оформлены в виде гиперссылок. При создании подобных страниц, пожалуй, не стоит изменять правила стилевого оформления элементов текста, так как изначально заданные параметры производят очень хорошее впечатление выдержанного стиля.

Следующий шаблон носит наименование **Confirmation Form**. Он позволяет быстро создавать страницы, предназначенные для подтверждения ввода данных в форму пользователем. Подобный механизм работы мы обсуждали в главах о формах и встраиваемых активных элементах. Мы уже говорили о том, что в подобных страницах-подтверждениях обычно отображается

стандартный текст, уведомляющий о благополучном приеме данных, в который вставляется имя, под которым зарегистрировался посетитель сайта. Наш шаблон использует целый набор полей подтверждения. Напоминаю, что если есть необходимость использовать подобный механизм ответа, то наименования активных элементов подтверждающих полей должны совпадать с уникальными идентифицирующими наименованиями полей ввода данных в исходной форме. В создаваемой странице используется подтверждающее поле с именем посетителя сайта **Username**, тип полученного сообщения и его тема (**MessageType** и **Subject**). Также использованы введенные координаты посетителя: его адрес электронной почты (**UserEmail**), номер телефона (**UserTel**) и факс (**UserFAX**). Остальной текст имеет смысл поменять на русскоязычное сообщение.

Для налаживания обратной связи между компанией и ее клиентом, при использовании стандартного Web-сайта компании, можно предложить использовать шаблон **Feedback Form**, который позволяет создать страницу, содержащую форму для ввода данных. По умолчанию в форме располагается группа радиокнопок, которые характеризуют тип отсылаемого сообщения. Предмет разговора выбирается из выпадающего списка. Если требуемого значения в нем нет, посетитель сайта может ввести его в поле ввода, находящееся рядом со списком. Само сообщение вводится в многострочное поле ввода. В самом низу формы расположены поля, в которые вводятся координаты посетителя. Стоит отметить, что эта форма идеально стыкуется со страницей подтверждения, которую мы только что рассматривали. Радиокнопки идентифицируются именем **MessageType**, поле для ввода имени удаленного пользователя — **Username**, и т. д. По сути, создателю страницы потребуется только заменить все сообщения и надписи, и получить готовую функциональную страницу.

Следующий пример представляет собой реализацию списка наиболее часто задаваемых вопросов (FAQ — Frequently Asked Questions). Тоже очень часто используемый механизм работы сайта. Если придется делать сайт, достаточно серьезно завязанный на общении с посетителями, не поленитесь, создайте подобную страничку, иначе придется отвечать на град однотипных вопросов. Стандартный вариант создания хорошей гипертекстовой странички со списком вопросов и ответов приведен в шаблоне с характерным именем **Frequently Asked Questions**. В верхней части страницы помещается список вопросов, каждый из которых выполнен в виде локальной гиперссылки, которая привязана к ответу на соответствующий вопрос. Таким образом, для добавления нового вопроса и ответа в наш список, в необходимое место сначала вставляется текст вопроса и ответа, оформляются в виде закладки, а потом в списке вопросов добавляется еще одна строка, которая реализуется в виде локальной гиперссылки. В качестве места назначения данной гиперссылки указывается только что созданная закладка. Ну, и в самом конце страницы размещаются реквизиты администратора, поддерживающего страницу, данные об организации, предупреждение об авторских правах, дата последнего обновления списка (это важно!).

Очень часто на различных сайтах можно встретить так называемые гостевые книги. Это на самом деле обычный гипертекстовый файл, в который дописываются отзывы посетителей. Процедура добавления своего мнения к общей массе достаточно проста. Для этого используется простейшая форма с многострочным полем текстового ввода и парой кнопок. Одна для очистки поля, вторая для отсылки текста. Подобная страница в FrontPage 2000 реализуется при помощи шаблона **Guest Book**. На ней после вышеупомянутой формы располагается активный элемент, отображающий html-файл, в который записаны все отзывы. Естественно, после добавления своего мнения необходимо перезагрузить страницу в браузер, чтобы увидеть обновленный файл.

Следующий шаблон представляет собой не реализацию какого-либо механизма сайта, а один из образцов грамотной верстки Web-страницы. Он носит название **Narrow, Left-aligned Body**. Этот шаблон создает страницу, содержимое которой укладывается в таблицу с невидимыми границами. Как мы помним, это практически единственный способ добиться адекватного отображения верстки. У этой страницы левое поле пустое, под основной текст выделен наиболее обширный блок, который, тем не менее, выглядит смещенным влево, а правая часть отведена для размещения иллюстраций.

Весьма похож на предыдущий, шаблон, который носит название **Narrow, Right-aligned Body**. Но здесь иллюстрации занимают левую часть, а текст смещен вправо. Если быть точным, и обратить внимание на количественные характеристики то, исходя из HTML-кода этой страницы, можно увидеть, что под иллюстрации отводится пятьдесят процентов ширины окна просмотра браузера. Для текста зарезервировано сорок процентов, а оставшиеся десять процентов остаются для правого поля. Естественно, эту ситуацию можно изменить простым изменением размеров ячеек в режиме проектирования страницы, или прямым изменением HTML-кода в следующей конструкции:

```
<td align="right" valign="top" width="50%"></td>
    <td valign="top" width="40%"></td>
    <td width="10%"></td>
```

Естественно, все последующие подобные блоки также подлежат изменению.

Шаблон **One-column Body** создает страницу, в которой текст располагается по центру одной колонкой. Реализуется подобная верстка также при помощи таблиц. Под правое и левое поле отведено по двадцать пять процентов ширины окна просмотра браузера. В центральной части страницы располагается сам текст. Для его отображения используется выключка по центру.

Продолжает эту идею оформления страницы шаблон **One-column Body with Contents and Sidebar**. В странице, создаваемой этим шаблоном, текст, как и в предыдущем случае, располагается в одну колонку, но в данном случае она занимает пятьдесят четыре процента от ширины окна просмотра. Вместо достаточно объемных полей, использованных в предыдущем шаблоне, теперь слева располагается содержание страницы, а справа — иллюстрации.

Эти колонки по умолчанию занимают шестнадцать и восемнадцать процентов соответственно. Все остальное место отдано под колонки таблицы, разделяющие заполненные столбцы. К сожалению, в данном случае содержание представляет собой обычный текст. Автор же настоятельно рекомендует делать оглавление в виде локальных гиперссылок.

Если же иллюстраций не настолько много, чтобы отводить им весьма ощутимую часть свободного пространства, можно разместить их в самом тексте, а освободившееся пространство использовать более эффективно. Подобный образец верстки страниц приведен в шаблоне **One-column Body with Contents on Left**. Под оглавление отводится двадцать процентов, под основной текст — восемьдесят.

Если нет никакого желания размещать оглавление слева, следует воспользоваться шаблоном **One-column Body with Contents on Right**. Он полностью повторяет по структуре предыдущий шаблон, но оглавление перенесено на правый край.

Шаблон **One-column Body with Staggered Sidebar** с левой стороны страницы резервирует место для иллюстраций и подписей. Они располагаются блоками в шахматном порядке.

Весьма похожий вариант верстки Web-страницы реализуется при помощи шаблона **One-column Body with Two Sidebars**. В этом случае правое поле немного увеличено. Как правило, оно применяется для размещения иллюстраций. Левое поле разбито на две колонки, в которых, обычно в шахматном порядке, помещаются гиперссылки на ресурсы с дополнительной информацией.

Эти две части можно разместить не по разные стороны от основного текста, а рядом друг с другом. Для создания подобного порядка расположения материала используется шаблон **One-column Body with Two-column Sidebar**. Здесь дополнительное поле разбивается на две колонки, в одной из которых обычно размещаются иллюстрации, а во второй дополнительные гиперссылки или подписи к рисункам.

Следующий (по алфавиту) шаблон представляет собой не пример верстки, а реализацию механизма полнотекстового поиска во всех документах, входящих в состав сайта. Для этого, естественно, используется обычная форма, которая весьма похожа на активный элемент полнотекстового поиска. Этот активный элемент мы уже рассматривали ранее в соответствующей главе. Шаблон носит название **Search Form**. Он создает страницу, на которой расположена сама форма для осуществления полнотекстового поиска, примеры задания критериев поиска (как всегда на английском языке) и привычные данные об администраторе страницы, авторских правах и прочие атрибуты, которые должны находиться в конце страницы. Самое хорошее в этой странице то, что воспроизводимый механизм позволяет указывать язык поиска и использовать логические операторы AND, OR и NOT, которые реализуют "логическое И", "логическое ИЛИ" и отрицание. Причем без какой-либо дополнительной настройки.

Каждый уважающий себя сайт должен иметь страницу с оглавлением. Необходимо заметить, что оглавление и панель навигации, какой бы изощренной структуры она не была, не всегда совпадают. Обычно для файла, содержащего код Web-страницы с оглавлением, задают имя index.htm. Подобная страница реализуется при помощи шаблона **Table of Contents**. На этой странице в самом начале располагается гиперссылка на основную, домашнюю страницу сайта, а затем в иерархическом порядке ссылки на все остальные страницы.

Теперь, следуя порядку расположения шаблонов в диалоге создания новой страницы, мы снова переходим к шаблонам, которые показывают правильное и грамотное оформление. Шаблон **Three-column Body** позволяет отображать текст в трех колонках. Естественно, реализуется все это при помощи все тех же таблиц. Однако подобное расположение текста будет не слишком хорошо выглядеть на мониторах с маленьким разрешением, так как окно просмотра браузера будет не слишком велико и колонки могут оказаться излишне узкими, а, значит, и текст потеряет свою читабельность.

Несколько более эффективным в этом смысле является разбиение текста на две колонки. Оно легко реализуется при помощи шаблона **Two-column Body**. В добавок к основному тексту на странице может быть размещено оглавление данного документа и поле с гиперссылками на дополнительные источники или иллюстрации. Этот вариант воплощен в шаблоне **Two-column Body with Contents and Sidebar**. Естественно, его стоит применять для достаточно объемных документов, разбитых на несколько глав, заголовки которых представляют собой закладки, так как иначе создание оглавления теряет смысл. В том случае, если нет необходимости использовать ссылки на дополнительные источники, вынесенные из тела документа, стоит использовать шаблон **Two-column Body with Contents on Left**. В этом варианте текст все так же разбит на две колонки, но внешние гиперссылки и рисунки размещаются все в тех же колонках, а левая часть страницы отдана под оглавление.

Иногда текст размещают в двух колонках блоками, расположенными в шахматном порядке. В этом случае на странице остается достаточно много места, но мы получаем некоторый выигрыш в выразительности и акцентируем внимание посетителя сайта на отдельных блоках. Подобный вариант верстки производится при помощи шаблона **Two-column Staggered Body**. Иногда к тексту, сверстанному подобным образом, могут добавляться содержание (традиционно слева) и колонка внешних гиперссылок и дополнительной информации (традиционно справа). Для создания страниц такого типа применяется шаблон **Two-column Staggered Body with Contents and Sidebar**.

Наряду с гостевой книгой, полнотекстовым поиском, обратной связью и прочими атрибутами грамотных сайтов используется механизм регистрации посетителей. Все-таки одна из основных задач большинства сайтов — тем или иным способом зарабатывать деньги. Для этой цели желательно иметь как можно больше информации о посетителях сайта. Многие сайты разбиваются на две части: открытую и закрытую. Открытая часть доступна всем и каждому, попадание же к закрытой части может быть осуществлено

только после регистрации пользователя. Страница с формой регистрации создается при помощи шаблона **User Registration**. Необходимо отметить, что для работоспособности этой формы, страницу с ней необходимо размещать в корневом каталоге сайта. Введенные данные сохраняются в текстовом файле regdb.txt, который находится в каталоге **_private**. Для регистрации используются четыре поля. Посетитель должен ввести свое имя, пароль, под которым он будет входить в закрытую область, подтвердить пароль и добавить свой адрес электронной почты. Естественно, набор полей ввода можно изменять.

Из подобных страниц потом собирается сайт. Не так уж важно, какого он будет размера и объема. Гораздо важнее правильно и аккуратно подключить все ресурсы, гиперссылки, свежим взглядом просмотреть его структуру, чтобы она не была излишне запутанной, оценить среднее время загрузки каждой страницы и только потом выкладывать его на заранее отведенное место.

Финальный аккорд

После того, как были созданы все необходимые страницы, проверены все локальные и внешние гиперссылки, вставлены все необходимые графические изображения, необходимо произвести некоторые завершающие действия для того, чтобы сайт был готов к помещению на машину провайдера.

Прежде всего, конечно, необходимо проверить орфографию текста, помещенного на каждой странице. Идеальным вариантом, конечно, было бы нанять для этой цели профессионального корректора, но в том случае, если бюджет для создаваемого сайта весьма ограничен, его придется заменить комбинацией из собственного знания правил правописания русского языка и средства автоматической проверки орфографии — "спеллчекера". Несмотря на то, что этот инструмент достаточно слаб по меркам грамотного человека, все-таки настоятельно рекомендуется его применять в работе, так как он позволяет выловить все очевидные ошибки достаточно быстро и надежно, экономя тем самым ощутимое количество рабочего времени. Для того чтобы процедура проверки орфографии была проведена корректно, необходимо установить язык содержимого страницы. Иначе в качестве стандартного набора правил проверки будут использоваться правила и словарь американского варианта английского языка. Установка языка документа производится на вкладке **Language** диалогового окна **Page Properties**, которое активизируется при использовании команды меню **File/Properties**. В блоке элементов управления **Page language** есть выпадающий список **Mark current document as:**, в котором для русскоязычных страниц необходимо выбрать вариант **Russian**. После этого достаточно выполнить команду меню **Tools/Spelling** или нажать одноименную кнопку на стандартной инструментальной панели и процесс проверки орфографии с использованием правил и словаря русского языка будет запущен.

Если есть система проверки правописания, то должна присутствовать и возможность поиска синонимов для нужного слова. Эта возможность ак-

тивизируется командой меню **Tools/Thesaurus** или комбинацией клавиш <Shift>+<F7>. Эту комбинацию стоит запомнить, так как команда подбора синонима для слова, в котором находится текстовый курсор, не помещена в объектное меню, а перемещение мыши к верхнему краю окна для использования меню занимает определенное время. При частом использовании подбора синонимов это время постепенно накапливается. Ведь недаром люди, которые вынуждены набирать достаточно объемные текстовые документы в Microsoft Word, часто пользуются сочетаниями клавиш, предпочитая их командам меню и инструментальным кнопкам. Это реальная экономия времени, которое всегда очень ценно.

После того, как вся орфография и пунктуация были выверены, самое время добавить некоторые дополнительные свойства для нашего сайта. Как известно, общее количество сайтов в Интернете очень велико. Действительно, *очень* велико. Если же хочется, чтобы его посещали не только те люди, которым вы сами дали адрес сайта, необходимо дать возможность найти его обычному обитателю Интернета. Как известно, обычный поиск производится при помощи поисковых машин. Каждая такая машина имеет свою собственную базу данных сайтов, в которой помимо его URL указывается тематическая направленность. Для этого используется механизм ключевых слов. У каждого сайта есть некая скрытая для обычного посетителя часть, которая содержит эти самые ключевые слова. Именно по этим словам и производится поиск сайта поисковыми системами. Реализуется механизм ключевых слов при помощи тега `<meta>`. Как известно, у каждой поисковой машины есть кнопочка с наименованием **Add URL** или аналогичным ему. Используя ее, вводится URL собственного сайта. После этого, обычно в течение двух-четырех суток поисковая машина посещает сайт и индексирует его. Под индексацией понимается поиск и идентификация тех самых ключевых слов поиска. После этого определяется, в какую категорию сайтов будет помещен наш адрес.

Как мы помним, у тега `<meta>` есть два параметра: `name` и `content`. Параметру `name` приписывается значение `"keywords"`, а параметру `content` в качестве значения приписывается текстовая строка, содержащая искомые ключевые слова поиска.

У параметра `name` может быть еще параметр `"description"`. В этом случае в параметре `content` указывается содержание данной страницы. Краткая аннотация, можно сказать. Многие поисковые машины для индексации используют именно этот параметр.

Эти теги должны указываться в коде основной страницы сайта, которая чаще всего сохраняется в HTML-файле с именем default.htm. Естественно, FrontPage 2000 может взять на себя добавление этих тегов в тело кода, избавив нас от ручного ввода. Для этого используется вкладка **Custom** уже известного нам диалогового окна **Page Properties** (рис. 1.41). Как видно, там присутствует два списка. Один содержит системные переменные, которые используются браузером посетителя сайта для более правильного отображения страницы (**System variables**), а второй — список переменных, определяемых

пользователем (**User variables**). Нас будет интересовать как раз второй список. Мы нажимаем кнопку **Add**, после чего в появившемся диалоговом окне **User Meta Variable** вводим необходимые данные. Например, если мы хотим указать, что сайт посвящен собакам, то в поле **Name** вводим значение keywords, а в поле **Value** слово dogs. Нажимаем на кнопку **OK** и в HTML-коде нашей страницы обнаруживаем строку `<meta name="keywords" content="dogs">`.

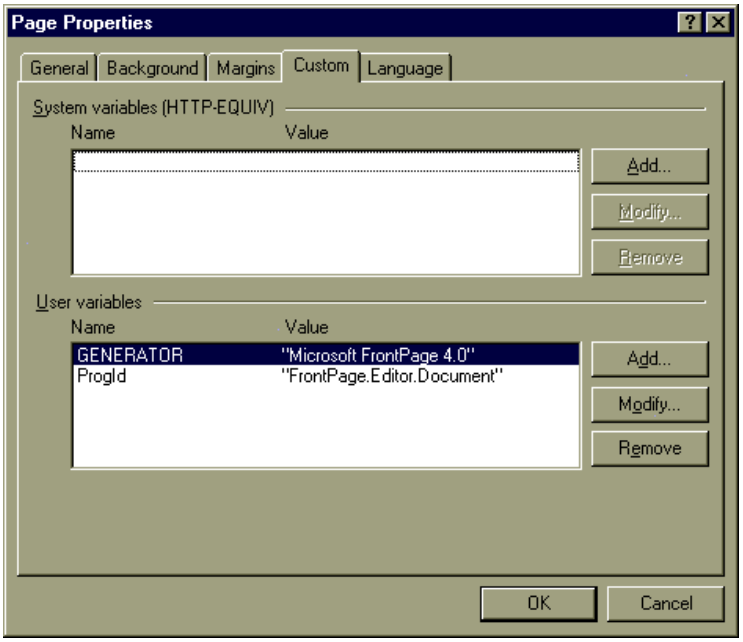

Рис. 1.41. Вкладка **Custom** диалогового окна **Page Properties**

Конечно, всегда присутствует соблазн добавить в подобное объявление ключевые слова, которые, может быть, не отражают тематическую направленность сайта, но очень часто запрашиваются посетителями Интернета. Должен сразу предупредить, что этот фокус может не получиться. Дело в том, что многие поисковые машины после получения ключевых слов проводят анализ текста страницы. И в том случае, если заявленная тема не упоминается в тексте, индексация по ней не производится.

И в самом конце, когда вся деятельность по производству собственно сайта закончена, необходимо его опубликовать. О технологии публикации мы уже говорили. Она помещает все файлы на отведенное им место в файловой структуре локальной машины или по указанному URL и приводит в готовность все активные элементы. Очень часто при использовании в оформлении страниц активных элементов адекватное предварительное отображение этих страниц невозможно без предварительной публикации. Эта процедура производится при помощи команды меню **File/Publish Web**. Перед выполне-

нием этой команды необходимо сохранить все страницы, которые были модифицированы со времени своего последнего сохранения. После этого активизируется диалоговое окно **Publish Web** (рис. 1.42). В основном поле ввода указывается URL публикуемого сайта. В том случае, если сайт размещается в файловой структуре локальной машины разработчика, у указываемого URL префикс с http изменяется на file. Это видно на рисунке. В том случае, если у разработчика нет собственного провайдера, который позволяет разместить сайт, используется кнопка **WPP's**, которая позволяет найти стороннего провайдера.

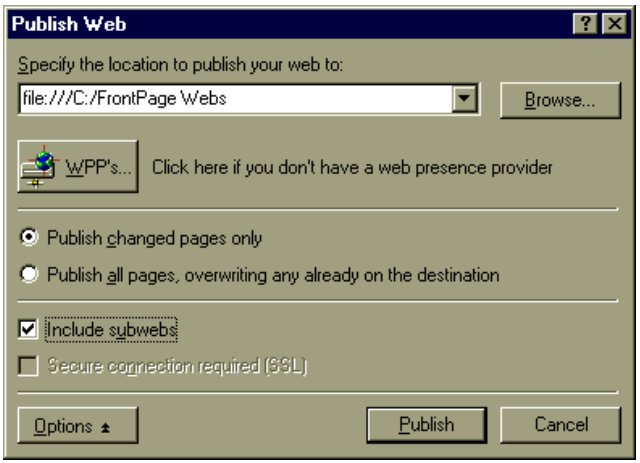

Рис. 1.42. Диалоговое окно **Publish Web**

Это диалоговое окно также позволяет указывать некоторые правила публикации. Элементы управления, задающие их, доступны по кнопке **Options**. На рис. 1.42 мы как раз можем видеть расширенный вариант диалогового окна. В группе радиокнопок можно указать порядок публикации страниц. Если выбрана альтернатива **Publish changed pages only**, то публикации подвергаются только те страницы, которые были модифицированы со времени последней процедуры публикации. Радиокнопка **Publish all pages, overwriting any already on the destination** указывает, что публикации будут подвергнуты абсолютно все страницы, входящие в сайт, а существующие их аналоги в месте расположения сайта будут перезаписаны.

На этом мы заканчиваем рассмотрение HTML-редактора FrontPage 2000 и переходим к рассмотрению проблем, связанных с графикой для Web-страниц, и их решений.

Глава 2

Графика для Web

Какие бывают картинки

Подготовка и размещение графических изображений в HTML-документах является нетривиальной задачей. Всегда необходимо поддерживать баланс между красочностью изображения и объемом графического файла, в котором оно храниться. Понятно, что если в качестве фона использовано достаточно объемное изображение с высоким разрешением, посетитель сайта скорее всего его не увидит. В том случае, если необходимо разместить на сайте объемное изображение, лучше установить на него отдельную ссылку. Тогда тот, кому оно действительно нужно, сможет получить изображение, а все остальные не будут долго ждать загрузки этой картинки в браузер.

В том случае, если графические изображения играют решающую роль в оформлении сайта, без консультации профессионального дизайнера не обойтись. Но очень многое зависит и от технических деталей. Рассмотрением их мы и займемся в этой главе.

Чаще всего для использования в HTML-документах применяются файлы форматов GIF и JPEG. В первой части мы упоминали еще формат PNG, но он пока еще не очень сильно распространен. Исторически сложилось так, что формат GIF появился на свет первым. В файлах этого формата рисунок хранится в сжатом виде. Это необходимо, так как в нашем случае, чем меньше размер графического файла, тем быстрее загружается изображение. А так как используется сжатие, в относительно небольших файлах могут записываться достаточно большие рисунки.

Еще одним из преимуществ данного формата, является возможность один цвет сделать прозрачным. Мы уже упоминали эту возможность в главе "Ссылки и графика". В файлах этого формата могут быть сохранены анимированные изображения, то есть маленькие мультфильмы. Именно этот формат позволяет создавать файлы с чересстрочным (interlaced) изображением.

О чересстрочном изображении стоит упомянуть особо. В Интернете достаточно часто можно встретить изображения, которые в начале процесса

загрузки выглядят очень нечетко, а по мере продолжения загрузки страницы изображение все больше и больше детализируется. Удобно это тем, что можно заранее оценить содержание картинки, и в том случае, если она интересна, дождаться ее полной загрузки. Если же она не нужна, достаточно пролистать страницу дальше, чтобы рисунок ушел из окна просмотра браузера, или перейти по гиперссылке и дальнейшая загрузка изображения не произойдет. То есть данный режим является достаточно экономичным, что в условиях работы в Интернете является весьма важным фактором. Видимо, пропускная способность каналов растет медленнее, чем запросы и ожидания пользователей, а, значит, от оптимизации скорости загрузки страниц нам еще долго не придется отказываться.

Чересстрочные изображения хранятся в GIF-файле особым образом. В начале файла записываются строки с номером, кратным восьми, затем — четырем, и т. д. Таким образом, подобное изображение проявляется в окне просмотра браузера постепенно, увеличивая свою четкость по мере загрузки новых порций данных.

Так как формат GIF применяет сжатие изображений без потери качества, он удобен для хранения рисунков с четкими контурами, таких, как различные схемы, копии экрана и т. д.

В том случае, если используется изображение с плавными переходами цвета и отсутствием мелких четких деталей, имеет смысл воспользоваться графическими файлами формата JPEG. Этот формат основан на сжатии изображения с потерями качества. Сжатие происходит как раз за счет сглаживания цветовых и контурных границ. Подобный формат очень хорошо подходит для рисунков и фотографий с плавными переходами цвета и значений яркости.

Обычно при создании файла формата JPEG можно указать допустимый процент потери качества. Вполне вероятно, придется сделать несколько попыток, прежде чем будет найден баланс между приемлемым качеством изображения и компактностью загружаемого файла. Оптимальный размер графического файла, подключаемого к HTML-документам, составляет несколько десятков Кбайт.

Формат JPEG позволяет также хранить изображения с высоким цветовым разрешением. В этих изображениях может использоваться до шестнадцати миллионов цветов. Формат GIF по умолчанию сохраняет всего двести пятьдесят шесть цветов.

Весьма часто на Web-страницах можно встретить изображения, которые меняются при щелчке на них, или просто при прохождении над ними курсора мыши. Подобные рисунки чаще всего выполняются в виде Java-скриптов. Обычно для создания подобных активных элементов необходимо достаточно хорошо знать язык Java, но в данном конкретном случае это не нужно, так как программный продукт, который мы будем рассматривать, может генерировать их самостоятельно. В этой главе мы обязательно уделим внимание этой нетривиальной возможности.

В качестве основного инструмента мы будем использовать Adobe ImageStyler 1.0. Это достаточно профессиональный продукт, созданный компанией, которая долго и плодотворно занимается графическими приложениями. Он позволяет полученное изображение сохранять в любом из трех форматов, используемых в HTML-документах. С помощью него мы попробуем создавать рисунки в виде Java-скриптов.

Для создания анимированных GIF-файлов мы будем использовать Animagic GIF Animator. Этот продукт является условно бесплатным (shareware). Официальный сайт продукта расположен по адресу **www.rtlsoft.com/animagic**.

Первый взгляд

При первом запуске Adobe ImageStyler 1.0 пользователь, не работавший ранее с графическими продуктами фирмы Adobe, может несколько удивиться. Помимо основного рабочего окна для размещения рисунка выводится еще семь (!) дополнительных окон. Правда при ближайшем рассмотрении одно из этих окон оказывается инструментальной панелью, зато на остальных шести можно заметить многостраничные вкладки. Приготовьтесь. Возможностей для работы с изображениями действительно много. Но если разбираться с ними планомерно, по очереди, то рано или поздно все это удивлять перестанет.

Как известно, компания Adobe достаточно давно и небезуспешно занимается производством программных продуктов, предназначенных для работы с графикой. Название Adobe Photoshop, кажется, известно каждому, кто хоть чуть-чуть интересовался графическими редакторами. На основе отработанных за долгие годы технологий работы с графикой создавался и сам ImageStyler, но он изначально был предназначен для создания изображений для Интернета. Эти изображения не могут быть объемными, следовательно, и многие "тяжелые" средства обработки не понадобятся, но вся идеология представления изображения и работы с ним сохранена. В том числе и использование так называемых слоев (layers). Любые повороты изображения, использование текстур и цветовых заливок, большая коллекция малых форм позволяют зачислить Adobe ImageStyler в разряд наиболее часто используемых программ разработчика Web-сайтов.

Графика для Web разделяется на иллюстративную и вспомогательную. В качестве иллюстративной части обычно используются фотографии и насыщенные, достаточно объемные рисунки. Вспомогательная графика это различные фоновые рисунки, графические кнопки, маленькие изображения, обозначающие тот или иной раздел текста (так называемые отбивки), маркеры для ненумерованных списков, горизонтальные линии, выполненные в графике, и прочие мелкие элементы оформления. То есть то, что создает общее впечатление о сайте. Как говорится, мелочи не играют решающей роли, они решают все. Поэтому к созданию подобных элементов оформления стоит подходить очень ответственно.

Сам ImageStyler не слишком хорошо приспособлен для работы с иллюстративной графикой. Для этого гораздо лучше подойдет его старший брат Adobe Photoshop, у которого есть множество инструментов для рисования и самой изощренной обработки изображения. Зато ImageStyler практически идеально приспособлен для работы с мелкими изображениями. Ими мы и будем заниматься.

Обычно используется один из двух вариантов создания требуемого рисунка. Первый — выбрать наиболее подходящий рисунок из какой-либо коллекции графики для Web, а затем обработать его соответствующим образом. Второй — воспользовавшись штатными средствами соответствующего графического пакета, создать подобный рисунок самостоятельно. Оба способа достаточно просты.

К глубочайшему сожалению, первый вариант нам практически недоступен. ImageStyler при команде открытия файла **File/Open** читает только свои собственные файлы с расширением ist. Единственная возможность использования внешнего изображения — импортирование его с внешнего источника, поддерживающего формат TWAIN. Это обычно сканеры и различные цифровые фото- и видеокамеры. С одной стороны, это достаточно серьезное неудобство, но на самом деле, если изображение уже существует, то для чего нужен ImageStyler? Как мы уже говорили, средства обработки изображений в продуктах, находящихся в более высокой весовой категории, намного обширнее и функциональнее.

ImageStyler 1.0 практически полностью ориентирован на замкнутый цикл создания графических изображений. У него существует достаточно обширная библиотека графических примитивов, весьма серьезный набор средств их обработки, и практически полный набор инструментов для формирования конечных файлов.

Несмотря на то, что ImageStyler обладает достаточно большим количеством средств обработки изображения, его нельзя назвать сложной программой. У него нет сильно разветвленного меню, мощных фильтров и прочих атрибутов полновесных графических приложений.

Для начала нам предстоит узнать, как создавать рисунки. Прежде всего, необходимо задать размер создаваемого графического изображения. По умолчанию ширина рисунка устанавливается в пятьсот пятьдесят пикселов, а его высота — в пятьсот пикселов. Изменить его размер можно при помощи команды меню **Edit/Canvas Size**. Эта команда активизирует одноименное диалоговое окно, которое содержит два поля ввода. Именно в них и вводятся необходимые значения.

После этого, на появившемся белом листе можно размещать рисунок. Средства для самостоятельного рисования несколько скудны. Не найти на инструментальной панели ни карандаша, ни кисти, ни распылителя. Но, впрочем, оно и к лучшему. Положа руку на сердце, часто ли кому-нибудь удавалось при помощи мыши нарисовать что-либо хорошее? Для этого

необходимо обладать твердой рукой, глазом охотника и умением художника. Все эти качества очень редко встречаются вместе. Поэтому большинству пользователей придется пользоваться библиотекой примитивов, которая поставляется вместе с ImageStyler.

Средства для рисования основных геометрических фигур находятся на стандартной и единственной инструментальной панели. Кнопка **Rectangle Tool** размещает в выбранном месте прямоугольник. Кнопка **Rounded-rectangle Tool** позволяет создавать все тот же прямоугольник, но со скругленными углами. Для создания эллипсов используется кнопка **Ellipse Tool**. В том случае, если требуется создать многоугольник, следует использовать кнопку **Polygon Tool**.

Размещение выбранного объекта на рисунке весьма стандартно. Первое нажатие левой кнопки мыши устанавливает место расположения левого верхнего угла фигуры, затем, не отпуская нажатой кнопки и перемещая мышь, область можно растянуть до требуемых размеров. Отпускание кнопки фиксирует расположение объекта.

Внешний вид всех объектов может быть изменен путем редактирования их свойств. Для этого используется одно из основных диалоговых окон, которое всегда находится на экране. Рассмотрению свойств всех графических объектов мы посвятим отдельный раздел.

Цвет, которым будет нарисована та или иная фигура, показан на инструментальной панели. В ее нижней части расположены два квадрата. Верхний показывает цвет, которым будет производиться рисование, а квадрат, находящийся на заднем плане, указывает цвет фона. По умолчанию фон — белый, а в качестве основного цвета рисования используется черный. Изменение этих цветов происходит в диалоговом окне **Color**. У этого окна помимо основного рабочего поля для выбора цвета есть элемент управления, состоящий из все тех же квадратов, обозначающих основной цвет и цвет фона.

Перед выбором необходимо указать, какой, собственно, цвет нам нужен. Если планируется установить основной цвет рисования, необходимо произвести единичный щелчок мышью на соответствующем квадрате. Если же будет выбираться цвет фона рисунка, сначала стоит щелкнуть мышью на квадрате, находящемся на заднем плане.

Выбор цвета производится на основном поле окна **Color**. Для удобства дизайнера приготовлено пять представлений цветовых гамм. Конкретный выбор представления цветового поля производится при помощи кнопки со стрелкой, направленной вправо, которая находится в верхнем правом углу окна. Модели носят следующие названия: **Saturation View**, **Value View**, **Hue View**, **HSB View** и **RGB View**.

Модель **Saturation View** основана на выборе насыщенности цвета. То есть, сначала необходимо выбрать основной цвет на спектральном ползунке, находящемся справа от основного поля выбора, а, затем, на основном поле выбрать необходимый цвет, исходя из его насыщенности.

Представление **Value View** весьма похоже на предыдущую модель, но основное поле упорядочено уже по яркости цвета.

Модель **Hue View** несколько больше ориентирована на представление в виде цветового круга. Данная модель опирается на спектральный ползунок, на котором выбирается основной цвет. Очень хорошо подходит для точного выбора требуемого оттенка. Кстати, именно эта цветовая модель показывается в окне **Color** по умолчанию.

Следующие две модели достаточно сильно отличаются по степени представления. Так, в модели **HSB View** используются три ползунка. Первый ползунок **H** позволяет указывать основной цвет, основываясь на модели цветового круга. Соответственно, значения положения ползунка варьируются от нуля до трехсот шестидесяти, и показывают градус отклонения по цветовому кругу. Второй ползунок **S** позволяет указывать насыщенность цвета в процентах, а третий ползунок **B** — яркость. Впрочем, если нет желания заниматься подбором необходимых числовых коэффициентов, можно выбрать приблизительный цвет на цветовой шкале, расположенной в нижней части окна, а потом уже, при помощи ползунков найти необходимый оттенок.

Последняя цветовая модель — **RGB View** по сравнению с остальными очень проста и понятна. Она предназначена для выбора цвета путем установки процентного соотношения красного, зеленого и синего цветов.

Во всех моделях присутствует независимый переключатель, рядом с которым находится изображение трехмерного куба. Переключатель носит имя **Web Safe Colors**. При установке флажка в этом переключателе цветовая гамма любой модели искусственно обедняется до двухсот пятидесяти шести цветов. Такой цветовой набор часто используется браузерами для отображения графических элементов с глубиной цвета восемь бит. Эта цветовая палитра является стандартным подмножеством системных цветовых палитр Windows и Mac OS. Так как сильная цветовая насыщенность для изображений, использующихся на Web-страницах, нежелательна, имеет смысл активно пользоваться этим ограничением. Здесь необходимо отметить, что в объемных эффектах и градиентных переходах цвета используется цветовая палитра без таких жестких ограничений. Впрочем, для достижения необходимого размера выходного графического файла в ImageStyler есть свои отдельные механизмы.

Для размещения текстового объекта внутри рисунка есть соответствующий инструмент **Type Tool**, привязанный к одной из кнопок панели инструментов. При нажатии на эту кнопку, курсор мыши принимает форму пунктирного квадрата с изображением текстового курсора внутри него. Щелчок левой кнопкой мыши фиксирует положение текстового объекта на канве рисунка и визуализирует диалоговое окно **Type Tool**, предназначенное для первичной установки свойств текстового объекта (рис. 2.1). Сам текст вписывается в основное многострочное поле ввода. Шрифт, которым его необходимо отобразить, выбирается в выпадающем списке **Font**. Рядом с этим списком находится второй, в котором выбирается конкретное начертание

шрифта. Размер шрифта в пунктах указывается в поле **Size**. В том случае, если текстовый объект содержит несколько строк, можно указывать межстрочный интервал в поле **Leading**. Буквы могут быть как сплошными, так и в виде линий, ограничивающих их контур. Если планируется использование именно такого "проволочного" вида букв, необходимо выставить флажок в независимом переключателе **Outline** и, в соответствующем поле ввода, указать толщину образующих линий. Точно так же, как мы устанавливали интервал между строками текста, можно установить интервал между отдельными символами. Для этого используется поле ввода **Tracking**. Ориентация текста выбирается при помощи выпадающего списка **Alignment**. На выбор предоставляется три значения: **Horizontal** — текст располагается по горизонтали; **Vertical – Left to Right** — текст располагается по вертикали, строки идут по порядку слева направо; **Vertical – Right to Left** — текст отображается по вертикали, строки идут справа налево. Рядом с этим списком располагаются три кнопки для установки выключки текста. В том случае, если используется горизонтальное расположение текста, кнопки позволяют прижимать текст к левому или правому краям текстовой области или размещать его по центру. Если используется один из вариантов вертикального расположения текста, то вместо прижатия к правому и левому краю задействуется выравнивание по нижнему или верхнему краям текстового блока.

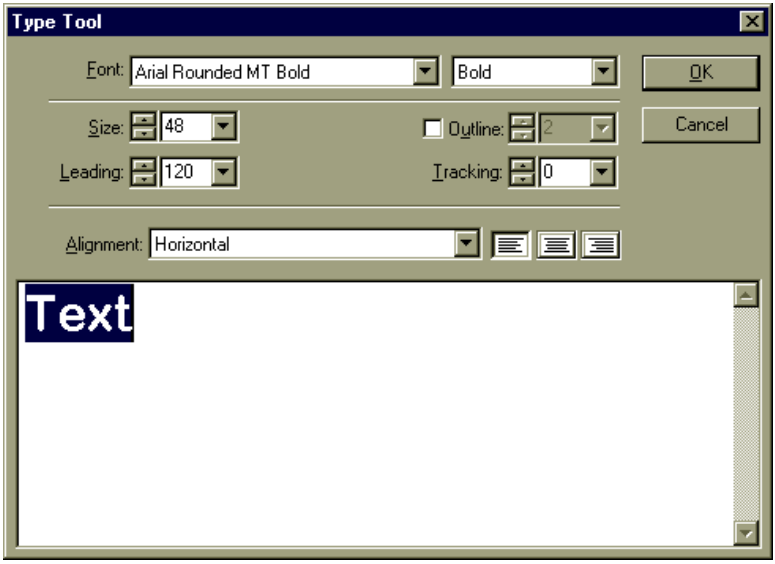

Рис. 2.1. Диалоговое окно **Type Tool**

Если необходимо уменьшить размер объекта без изменения масштаба самого изображения, стоит воспользоваться инструментом с названием **Crop Tool**. При нажатии на одноименную кнопку, находящуюся на инструментальной панели, курсор меняет свой вид. Теперь, если подвести его к одной

из границ какого-либо объекта и нажать левую кнопку мыши, произойдет захват границы. Теперь можно сдвигать ее, уменьшая размер объекта. При достижении задуманного эффекта нажатая кнопка мыши отпускается и граница фиксируется. Для того чтобы выйти из режима действия этого инструмента, необходимо нажать на кнопку **Selection Tool**, которая обычно используется для прерывания действия любого режима стандартных средств воздействия на объекты.

В том случае, когда нужно развернуть графический объект относительно его центра, используется средство **Rotate Tool**. При нажатии на соответствующую кнопку включается режим поворота объекта. Как известно, любой объект, не зависимо от его формы, может быть вписан в параллелограмм. В момент активности средства вращения объектов, при подведении курсора мыши к одному из угловых граничных маркеров выделенного объекта, он принимает форму окружности со стрелкой. После этого достаточно нажать левую кнопку мыши и, не отпуская ее, производить вращение объекта.

При создании графического объекта его границы имеют вид прямоугольника. Изменение внутренних углов превращает его в параллелограмм, а сам объект при этом немного меняет форму, вписываясь в измененные границы. То есть к каждой точке графического объекта, а значит, и к нему самому в целом, применяется некое конформное преобразование. Для того чтобы не забивать себе голову формами, можно просто представить, что одна из границ объекта смещается, не разрывая контакта с другими границами. При этом из прямоугольника получается параллелограмм. Подобное преобразование осуществляется при помощи средства **Skew Tool**. Нажатие на одноименную кнопку, находящуюся на инструментальной панели, переводит ImageStyler в соответствующий режим. При подведении курсора мыши к маркерам выделенного объекта, которые находятся в центре отображаемых границ, он, то есть курсор, меняет свой вид. Теперь, если произвести захват маркера нажатием на левую кнопку мыши, можно перемещать границу. При этом противоположная граница смещается в другую сторону относительно центра объекта, сохраняя при этом параллельность линий. Таким образом, у нас появляется возможность "перекосить" изображение так, как нам нужно.

Как и в любом другом графическом редакторе, в ImageStyler существует возможность заливки какой-либо части изображения одним цветом. Средство называется **Paint Bucket Tool**. При нажатии на активизирующую кнопку курсор мыши принимает вид наклоненного ковшика с краской. Курсор переводится к необходимому участку рисунка и заливка осуществляется нажатием левой кнопки мыши. Стандартный алгоритм заливки достаточно прост. При активизации инструмента, в качестве закрашиваемого цвета берется цвет той точки, над которой находился курсор мыши в момент активизации. После этого происходит последовательное закрашивание всех соседних точек такого же цвета, как и стартовая. При этом для закрашивания используется основной цвет, установленный пользователем. Напомню, что этим цветом окрашен передний цветовой индикатор из пары квадратов, на-

ходящихся на инструментальной панели. Таким образом, если стартовая точка заливки находится в области, которая полностью ограничена со всех сторон линиями другого цвета, внутри этого региона будут перекрашены все точки, чей цвет совпадает с цветом стартовой точки. Если же граница разомкнута, то заливка вырывается из ограничения и распространяется на весь рисунок. Если этот эффект не был запланирован, но все-таки произошел, не стоит забывать, что в меню **Edit** есть пункт **Undo**, который позволяет отменить опрометчиво совершенное действие.

Иногда возникает потребность окрасить объект в цвет, который уже применялся для окраски другой детали изображения, но который не является текущим. Например, у нас есть нежно-голубая стрелочка и мы хотим добавить к ней какой-либо завиток такого же цвета, но рабочий цвет уже изменен. Если мы не помним точных RGB-значений этого цвета, то, кажется, о точной установке цвета можно и не мечтать, так как общее количество доступных цветов превышает шестнадцать миллионов, и точно попасть в необходимый оттенок просто невозможно. Конечно, если использовать палитру **Web-Safe Colors**, то шансы на правильную установку цвета резко увеличиваются, но все равно, решение неизящно. Для подобных целей зарезервирована стандартная процедура. Соответствующий инструмент носит название **Eyedropper Tool** и привязан к кнопке с изображением пипетки, расположенной, как всегда, на основной инструментальной панели. Именно таким образом моделируется всем понятная метафора. При нажатии на кнопку курсор принимает вид пипетки. Остается только поднести ее к точке, чей цвет мы будем устанавливать в качестве текущего, и нажать левую кнопку мыши. Тут же основной цвет меняется. При этом, если у нас был выделен какой-либо объект, то в нем все точки, которые были окрашены в цвет, являющийся основным на момент активизации инструмента, будут перекрашены в новый цвет. То есть происходит не только смена основного цвета, но и применение этой операции к выделенному графическому объекту.

В том случае, если весь рисунок не умещается в окне, на помощь приходит средство **Hand Tool**. При нажатии на соответствующую кнопку, курсор мыши принимает форму человеческой ладони и позволяет "хвататься" за рисунок и перемещать его внутри рабочего окна. Подобный инструмент подменяет собой полосы прокрутки.

В случае, если нужно увеличить масштаб изображения, используется кнопка **Zoom Tool**. Механизм, привязанный к ней, увеличивает масштаб рисунка с шагом в сто процентов. Максимальный масштаб составляет восемьсот процентов. Также можно использовать команды меню для выполнения этих действий. Команда **View/Zoom In** позволяет увеличивать масштаб, **View/Zoom Out** — уменьшать масштаб все с тем же шагом в сто процентов, а команда **View/Actual Size** восстанавливает обычный масштаб изображения.

На этом мы заканчиваем обзор стандартных инструментов ImageStyler и переходим к рассмотрению различных типов графических объектов и примитивов вместе с их свойствами.

Графические объекты и их свойства

Для начала мы упомянем те свойства, которые принадлежат текстовым объектам. О том, как создать текстовый объект, можно прочитать в предыдущем разделе. Как мы помним, при создании текстового элемента активизируется диалоговое окно, которое позволяет задавать все необходимые свойства. Изменение их после создания самого объекта может производиться двумя способами. Можно просто подвести курсор к изображению одной из букв так, чтобы он принял форму треугольной стрелки вместо штатного перекрестия, и произвести двойной щелчок. При этом активизируется диалоговое окно **Type Tool**. Также изменить предустановленные значения свойств можно в стандартном окне **Properties**. Набор свойств, которые поддаются изменению, в обоих окнах совпадает. Это, естественно, шрифт, его размер и начертание, расстояние между символами строки и расстояние между самими строками, способ начертания букв — контурные или заполненные, толщину образующего контура, тип и вид выравнивания. Все эти свойства могут изменяться и соответствующим образом менять внешний вид текста. Несмотря на их кажущуюся малочисленность, с их помощью можно сделать практически все. Остальные изменения текстового объекта производятся уже не при помощи свойств, а при помощи штатных инструментов и механизмов ImageStyler.

При создании одного из геометрических объектов, которые мы также рассматривали в предыдущей главе, соответствующим образом меняется и вид стандартного окна **Properties**. Как мы помним, ImageStyler поддерживает четыре вида геометрических объектов: прямоугольник, прямоугольник со скругленными углами, эллипс и многоугольник. Так вот, после создания одного из них, мы можем заменить его на другой при помощи выпадающего списка в верхней части стандартного окна **Properties**. При этом новый геометрический объект будет вписан в те же границы, что и его предшественник, не нарушая, таким образом, общих пропорций всего изображения.

Для каждого геометрического примитива можно установить его внешний вид при помощи группы радиокнопок, находящихся в стандартном рабочем окне **Properties**. Если планируется создавать заполненный объект, то следует нажать кнопку **Fill**, если же необходимо показать только контур, следует выбрать альтернативу **Outline**. При этом ширина обрамляющих линий регулируется ползунком и полем ввода **Width**. В том случае, если создается прямоугольник с закругленными углами — **Rounded Rectangle**, появляется еще одна связка ползунка с полем ввода — **Radius**, при помощи которой можно указать радиус закругления уголков прямоугольника. А если в качестве геометрического примитива используется многоугольник — **Polygon**, то при помощи элемента управления **Sides** устанавливается количество сторон в создаваемом многоугольнике. Минимальное количество сторон, естественно, три, максимальное — десять.

Для остальных графических объектов, которые в ImageStyler называются формами (Shapes) в качестве свойства указывается только использование

так называемого альфа-канала изображения, который основан на информации о яркости. Проиллюстрировать концепцию альфа-канала можно на следующем примере. Если у нас есть фотопортрет человека на белом фоне прямоугольной формы, то при помещении его на эллиптическую подложку, скажем, фиолетового цвета, мы получим в фиолетовом эллипсе белый прямоугольник с портретом. Если же мы укажем, что при работе с фотопортретом необходимо использовать альфа-канал, то белый фон пропадет и портрет будет правильно вписан в фиолетовый эллипс. Отсюда следует, что при использовании многоцветных изображений в комбинации с "неродной" подложкой настоятельно рекомендуется использование альфа-каналов.

Перед тем как задавать свойства для подобных объектов, их необходимо расположить на канве рисунка. Для этого используется стандартное окно **Shapes**.

В том случае, если рабочее пространство приложения заполнено множеством объектов, и это окно не видно, можно выполнить команду меню **Window/Shapes** и это окно будет визуализировано. В окне **Shapes** находится так называемая палитра форм, которая содержит множество маленьких рисунков, достаточно хорошо подходящих для оформления Web-страниц. Вид самой палитры можно установить при помощи маленькой кнопки со стрелкой, находящейся в верхнем правом углу окна. По умолчанию используется представление **Swatches View**. В этом режиме каждое изображение размещено в собственной ячейке маленькой таблицы. В режиме **Preview View** в основном пространстве этого окна расположен список имен всех изображений, входящих в палитру форм ImageStyler, а немного левее находится поле для предварительного просмотра. Режим отображения **Name View** является неким разумным компромиссом между двумя предыдущими. В этом случае в общем списке отображается одновременно и имя рисунка и его уменьшенный вид, то есть, то, что обычно называется "ноготок" (Thumbnail).

После выбора необходимой формы достаточно нажать на кнопку **Place** и данный объект будет помещен в центр рисунка. Если же в центре рисунка ему делать нечего, можно установить курсор мыши на необходимую форму, нажать левую кнопку, и, не отпуская ее, перетащить объект на приготовленное для него место. То есть задействовать метод drug-and-drop. В том случае, если в уже существующем графическом объекте необходимо заменить форму, то нужно при выделенном объекте перейти на вкладку **Shapes** и, после указания необходимого рисунка, нажать кнопку **Replace**.

Также в этом окне находится кнопка **Matte**. Ее действие довольно интересно. Дело в том, что она тоже заменяет в графическом объекте одну форму на другую, но при этом отображаются только те участки рисунка, в которых изображения обеих форм совпадают. То есть происходит как бы наложение форм, а все непересекающиеся участки уничтожаются.

Палитра форм ImageStyler не задана строго и может видоизменяться. Для добавления в палитру нового изображения необходимо нажать кнопку **New**

с изображением листочка с загнутым уголком. При этом активизируется стандартное диалоговое окно открытия графического файла. Именно таким образом и можно заранее заготовленные изображения импортировать в ImageStyler. Если же из палитры необходимо удалить какую-либо форму, то используется кнопка **Delete** с изображением мусорной корзины.

Необходимо отметить, что расположение конкретного объекта на канве рисунка и задание его размеров может задаваться не только при помощи мыши, но и более точно, прямым указанием численных значений. Для этого используется стандартное окно **Transform**. Там, в полях **X** и **Y** указываются координата по оси абсцисс самой левой точки прямоугольника (в общем случае — параллелограмма, ограничивающего выделенный графический объект) и координата по оси ординат самой верхней точки области границы. Для задания высоты и ширины этого объекта в том же самом окне используются поля **H** и **W** соответственно.

Также необходимо осознавать, что на одном рисунке может находиться несколько графических объектов и геометрических примитивов. Они, естественно, могут перекрывать друг друга. Как всегда, объекты, созданные последними, находятся на самом верху иерархии, полностью повторяя поведение обычных окон в системе Microsoft Windows. Но у нас есть возможность управлять перекрыванием объектов. Это осуществляется при помощи команд, собранных в пункте меню **Object/Arrange**. Команда **Bring Forward** перемещает выделенный объект на один план вперед. Команда **Send Backward** перемещает объект на один план назад. Если же необходимо переместить объект на самый передний план или, наоборот, спрятать его на самый задний план, стоит использовать команды соответственно **Bring to Front** и **Send to Back**.

После установки всех свойств объектов можно приступать к работе над самими графическими объектами, приводя их в соответствие с замыслом дизайнера. Делается это при помощи самых различных эффектов и штатных инструментов ImageStyler.

Эффекты

Самые простые эффекты, которые обычно применяются к всевозможным графическим объектам, — это различные искажения пропорций. Помимо равномерных изменений размеров, не затрагивающих угловые коэффициенты изображения, оставляющих углы между границами рисунка прямыми, существуют еще изменения, которые ограничивающий прямоугольник превращают в параллелограмм, сдвигая одну границу относительно второй, параллельной ей. Это может быть сделано при помощи элементов управления, расположенных в окне **Transform**. Для этого предназначены два поля, у которых нет буквенных обозначений. Они идентифицируются при помощи вполне понятных пиктограмм. Так, верхнее поле отвечает за сдвиг границ по горизонтали, а нижнее обеспечивает сдвиг по вертикали. Симметрично сдвигаются две параллельные границы, относительно центра объекта.

Также существует возможность повернуть графический объект на любой угол с точностью до градуса. Для этого используется либо поле ввода с пиктограммой, обозначающей угол, либо элемент управления, представляющий собой круг с проведенным радиусом. Можно "ухватиться" курсором мыши за этот радиус и, не отпуская нажатой левой кнопки мыши, повернуть его на необходимый угол. Вместе с ним на такой же угол повернется и выделенный графический объект.

В том случае, если необходимо развернуть графический объект относительно одной из его главных осей, придется использовать команды меню, так как эти средства работы недоступны из стандартных рабочих окон. Так, для разворота объекта относительно его вертикальной оси используется команда **Object/Transform/Flip Horizontal**. Если же необходимо разворачивать объект относительно горизонтальной оси, то применяется команда **Object/Transform/Flip Vertical**.

В ImageStyler заготовлена достаточно объемная библиотека так называемых стилей, которые применяются для оформления различных графических объектов. Вся эта библиотека показана в стандартном окне **Styles**. При применении какого-либо стиля к выделенному объекту нужно выбрать необходимый стиль и нажать на кнопку **Apply**. В этой библиотеке находятся стили, позволяющие создавать тени от объекта, придавать ему трехмерность, использовать ограничивающие видимые рамки, оформлять в заданной цветовой гамме и т. д. Всего не перечислить. Надо просто смотреть и пробовать.

Рассмотрим механизм наложения текстур на рисунок. В качестве текстуры обычно выступают некие небольшие изображения регулярной формы, при стыковке которых не будут видны места стыков. Такое изображение тиражируется и накладывается на рисунок. Для выбора и наложения текстуры применяется стандартное окно **Textures**. В этом окне располагаются образцы текстур, входящих в библиотеку текстур ImageStyler. После выбора текстуры, как обычно, нажимается кнопка **Apply**, после чего выбранный рисунок накладывается на выделенный графический объект.

Библиотека текстур, как и библиотека форм, не задана раз и навсегда. Она может дополняться. Для добавления текстуры в библиотеку применяется кнопка **New Texture** с изображением листка с загнутым углом. При ее нажатии визуализируется стандартное диалоговое окно открытия файла. ImageStyler декларирует адекватное восприятие всех (!) графических форматов. Единственное, о чем необходимо позаботиться при добавлении нового рисунка текстуры, так это о его определенной регулярности. Если ее не будет, то вместо гладкого наложения текстуры будут явственно видны стыки при тиражировании матрицы текстуры.

При оформлении графических объектов всегда трудно удержаться от применения эффектов градиентных цветовых заливок. Для того чтобы получить доступ к элементам управления, регулирующим применение градиентных эффектов, необходимо активизировать стандартное рабочее окно **Opacity**.

Если его не видно на рабочем поле, всегда можно выполнить команду меню **Windows/Opacity**. В этом окне находится ползунок, связанный с полем ввода **Opacity**. Этот элемент управления отвечает за "прозрачность" изображения. Это не столько насыщенность цвета, сколько насыщенность цветом. При минимальном значении этого параметра объект полностью сливается с фоном. К сожалению, полиграфия книги не позволяет нам адекватно отобразить разницу изображения объекта при изменении параметра **Opacity**, а рассказать на словах о переходах цвета и оттенков просто невозможно. Ниже этого ползунка находится группа элементов управления, при помощи которых можно задавать градиентные переходы значения насыщенности цвета. Обратите внимание, что здесь градиентные эффекты накладываются не на цвет, а именно на насыщенность цвета. Для установки цветовых градиентных схем применяется окно **Gradient**. Те параметры, которые указываются в нем, влияют именно на цвет формы. Прежде, чем мы приступим к рассмотрению градиентных эффектов, следует обратить внимание на одиночный переключатель **Relative**. Он применяется для привязки градиентного эффекта к ориентации самого графического объекта. Так, если мы указали, например, что применяется стандартный линейный градиентный переход цвета, когда у нижней границы цвет более темный, а у верхней границы он светлеет, то при развороте по вертикали объекта сам градиентный эффект по умолчанию не развернется. Если же мы хотим, чтобы при подобных разворотах градиентный переход разворачивался бы вместе с изображением, необходимо установить флажок **Relative**.

Теперь рассмотрим сами градиентные эффекты. Выбор той или иной градиентной цветовой схемы осуществляется в выпадающем списке и его эффект тут же показывается на демонстрационном элементе в виде цветового круга с проведенным радиусом. Значение **No Gradient** указывает, что к данному объекту не будет применяться какой-либо градиентный эффект.

Значение **Linear** активизирует стандартную линейную градиентную схему. Стартовый цвет по умолчанию устанавливается на левом крае объекта, а при переходе к правому краю цвет темнеет. В нижней части окна **Gradient** находится цветовая шкала с двумя ползунками, которые устанавливают стартовый и конечный цвета градиентной цветовой схемы. Если есть желание установить градиентный переход не строго по горизонтали, а под каким-либо углом, то можно на круговом индикаторе "зацепить" мышью радиус и повернуть его в необходимую позицию. При этом, соответственно, изменится и направление градиентного перехода. Если точность указания направления градиентного перехода при помощи мыши недостаточна, стоит использовать поле ввода, которое обозначено пиктограммой угла. В нем вводится угол направления в градусах относительно стандартного горизонтального положения.

Эффект **Burst** устанавливает горизонтальную линию исходного цвета в центре изображения и от нее осуществляет переход к темному цвету на верхней и нижней границе соответственно. Таким образом, центр объекта остается светлым, а границы темнеют.

Эффект **Double Burst** делает две светлые линии, вертикальную и горизонтальную, пересекающиеся в центре под прямым углом.

Эффект **Radial** основным цветом отображает только центральную точку объекта, а от нее равномерно по всем направлениям изменяет цвет. Очевидно, что в случае применения этой схемы изменение угла направления градиентного перехода не принесет какого-либо видимого эффекта, так как изменение цвета происходит по окружности.

Для стандартных графических форм предусмотрены эффекты, придающие им иллюзию трехмерности. Активизация этих эффектов производится при помощи элементов управления, расположенных в рабочем окне **3D**. На выбор предлагается четыре эффекта. Эффект **Cutout** как бы вырезает объект из фона и вдавливает его. Между фоном и рисунком образуются тени, которые и создают эффект трехмерности. Эффект **Emboss** наоборот, придает выпуклость объекту, причем производится это достаточно мягко, приподнимая объект над общим фоном рисунка. Эффект **Bevel** создает иллюзию, что с поверхности графического объекта была снята фаска. А самым экзотичным, пожалуй, является эффект **Ripple**. Он на всем объекте прорисовывает концентрические волны. Создается полная иллюзия, что объект состоит из жидкости и в его центр упала капля.

В окне **3D** расположены и другие элементы управления, задающие параметры применения этих эффектов. Так, ползунком **Depth** регулируется глубина применения эффекта в пикселах. Значение может находиться в промежутке от нуля до сорока. Ползунок **Softness** задает так называемое смягчение эффекта, немного размывая контуры объекта. В редакторе Photoshop этот эффект носит название *gaussian blur*. Ползунок **Lighting** позволяет устанавливать яркость зоны применения эффекта. Возможные значения находятся в промежутке от нуля до двухсот. Единицы измерения — проценты. Нормальное значение по умолчанию равно ста процентам. Угол направления главной оси эффекта задается при помощи уже знакомого нам круга с проведенным радиусом.

Все псевдотрехмерные эффекты основаны на добавлении к рисунку темных и светлых частей. При помощи выпадающего списка **Light** мы можем управлять их отображением. Значение **Normal** отображает и темную, и светлую часть эффекта, значение **Light Only** указывает на то, что отображаться будут только осветленные участки, значение **Dark Only** отдает предпочтение темным участкам объекта.

Теперь, когда мы знаем, как можно видоизменять отдельные объекты, перейдем к манипуляциям с двумя и более объектами.

Комбинирование изображений

Если на одном рисунке находится несколько графических объектов, мы получаем доступ к командам меню, которые позволяют манипулировать с ними, объединять их и указывать их расположение относительно друг друга.

Для начала разберемся с объединением двух и более объектов. Предположим, что на канве рисунка находится несколько объектов. Для того чтобы выполнить какое-либо действие с двумя объектами сразу, необходимо их выделить. Как выделить один объект, мы уже знаем. Достаточно щелкнуть на нем мышью. А для выделения еще одного объекта, не снимая выделения с первого, необходимо нажать клавишу <Shift> и, не отпуская ее, щелкнуть мышью на другом объекте. Впрочем, если требуется выделить все объекты, присутствующие на рисунке, можно воспользоваться командой меню **Edit/Select All**. После этого команды меню, ответственные за работу с несколькими объектами, становятся доступны. Рассмотрим возможности комбинирования объектов. Команда **Object/Combine/Unit** объединяет несколько графических объектов в один. При этом у нового объекта создается новая граница, включающая в себя все выделенные объекты. Новому объекту присваивается цвет, соответствующий тому объекту, который находился на заднем плане.

При использовании команды **Object/Combine/Unit with Color** новый объект не приобретает один общий цвет. Все объекты, входящие в него, сохраняют свое цветовое оформление.

Следующий режим объединения работает при выполнении команды **Object/Combine/Minus Front**. В этом случае из объекта, находящегося на заднем плане, вырезается та часть, которая перекрывается другими объектами, входящими в объединение, и которые расположены на передних планах. Режим **Intersect** доступен по команде **Object/Combine/Intersect** и показывает только пересекающиеся части объектов, входящих в состав текущего объединения. При этом изображению присваивается цвет объекта, находящегося на заднем плане. Самый последний вариант объединения доступен по команде **Object/Combine/Exclude**. Он является полной противоположностью только что рассмотренному режиму **Intersect**. В этом режиме отображаются только те части, которые не совпадают у объектов.

Теперь перейдем к способам выравнивания объектов относительно друг друга. Для этого применяется группа команд меню **Object/Align**. Команды этой группы становятся доступны для использования только тогда, когда выделено несколько графических объектов сразу. Команда **Object/Align/Left** совмещает левые границы объектов. Команды **Right**, **Top** и **Bottom** совмещают соответственно правые, верхние и нижние границы объектов, входящих в текущее выделение. Команда **Object/Align/Horizontal Centers** перемещает все выделенные объекты таким образом, что их центральные вертикальные оси совпадают. Обратите внимание на то, что при выполнении команды **Horizontal Centers** происходит перемещение объектов по горизонтали, но со-

вмещаются вертикальные оси, а по команде **Object/Align/Vertical Centers** совмещаются центральные горизонтальные оси объектов. Перемещение при этом происходит по вертикали.

В том случае, если в текущее выделение входит более двух графических объектов, можно устанавливать равномерное расположение их по вертикали или по горизонтали, то есть создавать равные промежутки между ними. Для равномерного распределения объектов по вертикали без изменения их горизонтальных позиций используется команда меню **Object/Distribute/Vertical**. При этом координаты верхнего и нижнего объектов, входящих в текущее выделение, не меняются. Если необходимо произвести такое же равномерное распределение, но по горизонтали, следует использовать команду меню **Object/Distribute/Horizontal**.

Из нескольких объектов всегда можно сделать один. Для этого необходимо выделить объединяемые объекты и сгруппировать их командой **Object/Group**. В том случае, если подобный комбинированный объект необходимо разбить на первоначальные составляющие, применяется команда **Object/Ungroup**.

Графические слои

Концепция слоев пришла в ImageStyler из его старшего и более уважаемого собрата — Adobe Photoshop. Смысл ее заключается в том, что изображение в целом представляется как комбинация из нескольких других. Каждое из таких "элементарных" изображений называется слоем. В качестве иллюстрации можно предложить следующий пример. Если у нас есть неоднородный фон, на нем расположена какая-либо форма, а на саму форму наложена текстура. И ко всему этому вместе еще применен какой-либо эффект трехмерности. В этом случае рекомендуется фон, саму форму и наложенную текстуру помещать в отдельные слои. Этот вариант работы сильно упрощает кардинальные изменения рисунка. А в нашем конкретном случае, при работе с ImageStyler, использование слоев позволяет создавать рисунки, изменяющиеся в зависимости от действий пользователя и сохранять их в виде Java-скриптов.

Для работы со слоями предназначено стандартное рабочее окно **Object Layers**. Оно содержит список всех слоев, использующихся при работе с рисунком, вместе с их схематичным изображением, что позволяет достаточно хорошо ориентироваться в них. И, конечно же, присутствуют все необходимые команды меню для работы со слоями. В этой главе мы обязательно рассмотрим их.

По умолчанию для рисунка создается один слой, в который помещается выделенный объект. Если на канве рисунка расположено несколько отдельных графических объектов, то при переносе выделения с одного из них на другой, меняется содержимое отображаемого по умолчанию слоя. Всего может быть не более пяти слоев, которые независимы друг от друга. Действия с одним из них ни коим образом не затрагивают другие слои. Именно эта особенность и является, пожалуй, источником удобств их применения.

Итак, для начала рассмотрим команды создания и удаления слоев. Для создания нового слоя можно воспользоваться кнопкой **New Layer** с изображением одиночного листочка с загнутым уголком, которая находится в стандартном рабочем окне **Object Layers**. Той же цели служит команда меню **Layer/New Layer**. При выполнении этой команды в окне **Object Layers** появляется новая строка, связанная с новым слоем. Если мы выполним эту команду в тот момент, когда у нас на рисунке находится объект, с заданным цветовым оформлением, но без наложенной текстуры и на обычном белом фоне, то в первом слое у нас будет находиться только цветовое оформление данной формы, а во втором слое сама форма, которая по умолчанию создается черным цветом. Этот пример ярко иллюстрирует концепцию иерархии слоев. В самом верхнем слое хранится информация о текстуре, наложенной на объект, в самом нижнем слое содержится фон объекта. Так как в нашем случае нет ни текстур, ни усложненного фона, в слоях есть лишь сама форма и ее цветовое оформление. При создании дальнейших, более глубоких слоев, они будут лишь повторять второй слой. Всего, повторюсь, может быть только пять слоев. В них могут находиться цветовое заполнение объекта, форма объекта, фон объекта, его текстура. Помимо этого, каждый слой может содержать информацию о цветовой палитре, которая накладывает ограничения по цвету на данный слой.

Слои можно не только создавать, но и копировать. Для этого применяется кнопка **Duplicate Layer** или одноименная команда из меню **Layer**. Для удаления какого-либо слоя достаточно нажать на кнопку с изображением мусорной корзины или выполнить команду **Delete Layer**. Выбор того или иного слоя для работы осуществляется либо одиночным щелчком мыши на его ярлыке в рабочем окне **Object Layers**, либо при помощи команды меню **Layer/Select Layer**.

Необходимо также отметить, что слои могут быть только у единичных графических форм. У объектов, состоящих из нескольких графических форм, созданных при помощи команды **Object/Group**, слоев нет.

Для каждого слоя по отдельности можно задавать его свойства. А комбинация этих слоев уже создаст необходимый графический объект. Для задания свойств слоя используется стандартное рабочее окно **Layer**. Выпадающий список **Fill with** предназначен для установки содержимого слоя. Значение **Color** указывает на то, что в выбранном слое содержится информация о заполнении формы цветом. Если выбрано значение **Image** — в слое будет содержаться сама графическая форма. В том случае, если в данном слое должен находиться фон графического объекта, выбирается значение **Background**. А для текстуры используется вариант **Texture**.

Также в окне **Layer** расположены еще четыре элемента управления, состоящие из ползунков и полей ввода данных, связанных друг с другом. Ползунки **X Offset** и **Y Offset** задают величину сдвига конкретного слоя от верхнего левого угла графического объекта по горизонтали и вертикали соответственно. Проиллюстрировать их действие можно на следующем примере. На

канве рисунка размещаем форму **Profile (man)**. Окрашиваем ее, скажем, красным цветом. При этом автоматически создается слой, содержащий цветовое наполнение объекта. Затем в окне **Object Layers** при помощи кнопки **New Layer** создаем новый слой и в окне **Layer** в выпадающем списке устанавливаем для него значение **Image**. Теперь в первом слое у нас записан цвет, а во втором — сама графическая форма. Если для первого слоя при помощи элемента управления **X Offset** установить смещение по горизонтали на двадцать два пиксела, то красная форма сдвинется вправо (рис. 2.2).

Ползунок **Width** позволяет расширять или уменьшать объект. Значение, установленное им, можно расценивать как ширину границы. Оно может находиться в пределах от десяти до минус десяти. Обычно применяется для создания ореола вокруг верхнего слоя. В нашем случае верхний слой — цветовой. И если мы без смещения слоев относительно друг друга укажем для слоя формы значение параметра **Width**, равное десяти, то вокруг красной графической формы появится черный ореол (рис. 2.3).

Рис. 2.2. Результат смещения слоев графического изображения относительно друг друга

Рис. 2.3. Результат увеличения размеров нижнего слоя графического изображения

При рассмотрении этого рисунка видно, что граница между красным объектом и его черным ореолом достаточно резкая и отчетливая. В тех случаях, когда необходимо добиться эффекта более плавного перехода, стоит использовать элемент управления **Softness**. Значение в поле **Softness** указывает степень размытости границы. Если мы, при сохранении черного ореола для верхнего слоя с цветовым заполнением, установим параметр **Softness**, равный десяти (максимальное значение), то получим максимально мягкий переход от ореола к самому изображению. Границы между ореолом и формой практически не будет (рис. 2.4). Конечно, ограничения полиграфии не позволяют разглядеть рисунок в полном цвете, но идея должна быть понятна.

Теперь рассмотрим проблемы наложения текстур на изображение в качестве отдельных слоев. Прежде всего необходимо осознавать, что текстура накладывается поверх цвета, то есть иерархия слоев сверху вниз выглядит так: фон → графическая форма → цвет → текстура. К каждому из этих слоев

может быть применен любой фильтр и это никак не повлияет на все остальные слои. Именно для обеспечения такой гибкости работы и была создана концепция отдельных слоев. Итак, если мы на готовом объекте, состоящем из двух слоев, создадим новый слой при помощи кнопки **Add Layer**, то в новом слое будет снова только графическая форма. Для создания нового слоя, в который следует вписать текстуру, необходимо выбрать слой, содержащий цветовое оформление и сделать его дубликат при помощи кнопки **Duplicate Layer**. Теперь можно применить необходимую текстуру к графическому объекту. Напомню, что выбор текстуры производится в рабочем окне **Texture**, а ее наложение осуществляется при нажатии кнопки **Apply**. В качестве текстуры для нашего рисунка мы выберем шахматную сетку с мелким шагом. Эта текстура носит наименование **Checker Small**. После этого для нового списка в рабочем окне **Layer** нужно указать заполнение текстурой, то есть в выпадающем списке выбрать значение **Texture**. И все. Теперь в списке слоев видно, что в первом слое находится графическая форма, во втором — цвет, а в третьем — текстура. Мы можем увидеть все слои одновременно в рамках одного рисунка. Для этого достаточно второму и третьему слоям указать соответствующее смещение. Если для цветового слоя установить смещение **X Offset**, равное десяти пикселам, а для текстурного слоя — тридцати, то они будут расположены на рисунке достаточно равномерно.

Рис 2.4. Результат размытия границ верхнего слоя графического изображения

На рис. 2.4 достаточно хорошо видно, что изображение самого нижнего слоя по всем размерам превосходит остальные слои. Связано это с тем, что для этого слоя был установлен параметр **Width**, равный десяти, что и привело к расширению границы на десять пикселов.

Следует отметить, что все текстуры уже несут в себе цветовую информацию, поэтому установить принудительно цвет для нашей шахматной сетки просто так нельзя. Но если для текстурного слоя правильно подобрать параметр прозрачности **Opacity**, то исходный красный цвет начнет "просвечивать" сквозь нашу текстуру, придавая ей соответствующее цветовое оформление. К сожалению, полиграфия не позволит показать нам это на примере.

Иногда может возникнуть ситуация, когда не надо отображать какой-либо слой, но при этом необходимо его сохранить. Например, для дальнейшей работы, для создания нового изображения на базе уже готового. В этом случае приходится работать со стандартным окном **Object Layers**. В левой части каждой строки, содержащей наименование и маленькое изображение содержимого слоя, находится кнопка с изображением глаза. Одиночный щелчок левой кнопкой мыши на ней скрывает соответствующий слой выделенного графического объекта. При этом изображение глаза на самой кнопке также исчезает. Таким образом, подобные кнопки служат еще и индикаторами видимости слоев. Точно так же осуществляется и обратная операция, то есть выведение отдельного слоя из зоны невидимости. Достаточно произвести одиночный щелчок мышью на кнопке и соответствующий слой проявится.

Мы уже рассматривали иерархию слоев, устанавливаемую по умолчанию. Но у пользователя всегда есть возможность поменять ее, изменив порядок расположения слоев. Как мы помним, при работе с несколькими графическими объектами можно переносить их на верхние или нижние планы, регулируя подобным образом перекрывание этими объектами друг друга. Абсолютно такая же технология применяется при работе со слоями. Для того чтобы переместить какой-либо слой на один уровень вверх, применяется команда меню **Layer/Arrange/Bring To Front**. Перемещение слоя на один уровень вниз осуществляется при помощи команды **Layer/Arrange/Send To Back**. Перенос слоя на первый план или на самый нижний уровень производится при помощи команд **Layer/Arrange/Bring Forward** и **Layer/Arrange/Send Backward** соответственно.

В том случае, когда необходимо привести значения всех свойств какого-либо слоя к значениям по умолчанию, стоит использовать команду меню **Layer/Clear Layer Attributes**.

На этом мы заканчиваем рассмотрение слоев и переходим к одной маленькой, но очень интересной возможности Adobe ImageStyler. Это создание дублей графических объектов.

Создание выходного файла

После того, как была проведена вся работа над изображением и авторский замысел был адекватно воспроизведен, необходимо сделать последний шаг. Любой рисунок для размещения на проектируемом сайте должен быть помещен в графический файл. Мы уже говорили, что стандартное сохранение рисунка формирует специальный файл Adobe ImageStyler, который, вообще говоря, не является каким-либо графическим форматом. Если быть точным, генерируемый файл не является графическим.

Для создания графических файлов используется команда **File/Export**, но перед тем, как ее выполнить, необходимо установить некоторые правила и ограничения для экспортируемого изображения. Нам необходимо выбрать

формат выходного файла, подобрать его размер, установить свойства этих файлов. Все эти операции производятся в стандартном рабочем окне **Export**, которое активизируется при выполнении команды меню **File/Export Settings**. Сначала нам необходимо выбрать формат получаемого графического файла. Для этого стоит воспользоваться выпадающим списком, расположенным в верхней части этого окна. Как мы уже говорили, ImageStyler поддерживает три графических формата, используемых в WWW. Однако в списке находится пять значений. Первое из них, **Photoshop**, используется в тех случаях, когда полученное изображение необходимо дополнительно обработать при помощи серьезного графического пакета Adobe Photoshop. Второе значение — **GIF**. Его назначение понятно. Стоит еще раз повториться и упомянуть, что этот формат предназначается для мелких рисунков и изображений, содержащих мелкие и четкие детали. Третье значение, **JPEG** — это графический формат, который предполагает сжатие рисунка с потерей качества. Поэтому его следует применять при работе с изображениями, содержащими плавные цветовые переходы. Четвертое и пятое значения сохраняют рисунок в формате PNG. При использовании значения **PNG-Indexed** указывается количество используемых цветов, а вариант **PNG-Truecolor** сохраняет рисунок с максимально точной цветовой передачей. В этом последнем случае мы не имеем возможности регулировать размер выходного файла, поэтому данный вариант редко используется в работе.

Теперь рассмотрим возможные установки для каждого графического формата. Все они направлены на оптимизацию файла, то есть на уменьшение его размера при сохранении приемлемого качества. Для того чтобы наблюдать результат этих изменений, необходимо установить флажок **Active Preview** или выполнить команду меню **View/Active Export Preview**. При этом в поле **Size** после каждого изменения рисунка или параметров его экспорта в графический файл будет отображаться актуальный размер выходного файла. Чрезвычайно удобная возможность.

Проведем обзор всех параметров, которые влияют на создаваемый графический файл. В том случае, если мы собираемся создавать файл формата GIF, мы можем указать количество цветов, используемых в отображении рисунка. Минимальное значение, естественно, равно двум, максимальное — двумстам пятидесяти шести цветов. Если мы указываем количество цветов заведомо меньшее, чем используется в рисунке, мы сильно выигрываем в объеме файла, но проигрываем в качестве изображения. Улучшить это качество без увеличения количества используемых цветов можно, если установить флажок **Dither**. В этом случае недостающие цвета будут воссоздаваться за счет смешения тех цветов, которые уже есть в изображении. Если установлен флажок **Active Preview**, то мы сразу же увидим наш рисунок именно таким, каким он будет изображаться при загрузке его из графического файла. Таким образом мы сможем сразу оценить величину потери качества и увидеть, как будет выглядеть рисунок с указанным количеством цветов. Как мы уже говорили, в GIF-файлах могут сохраняться рисунки с прозрачным цветом. Если в нашем рисунке планируется использовать прозрачный цвет, необходимо установить флажок **Transparency**.

Формат JPEG, как мы помним, основан на сжатии с потерей информации. Поэтому при выборе этого значения в выпадающем списке появляется ползунок **Quality**, который позволяет указывать уровень сохраняемого качества в процентном отношении. По умолчанию устанавливается тридцатипроцентный уровень. Весьма разумное, взвешенное и сбалансированное значение. Но все равно, для достижения правильного результата необходимо включить режим **Active Preview**, и "поиграть" с ползунком для достижения максимально точного баланса между приемлемым качеством и размером выходного файла. Можно еще немного уменьшить этот размер, если поставить флажок **Reduce Chroma**. Этот режим несколько уменьшает количество информации о цветовой гамме рисунка, но делает это так аккуратно и незаметно, что на качество рисунка это почти не влияет. А вот выиграть при этом еще несколько процентов объема вполне возможно.

Формат PNG пока еще не слишком распространен в Интернете, но и его можно использовать. Adobe ImageStyler поддерживает две модификации этого стандарта. Значение **PNG-Indexed** указывает на то, что будет использоваться формат PNG с указанием количества используемых цветов. По возможностям указания свойств этот вариант полностью эквивалентен GIF-формату. В большинстве случаев, при прочих одинаковых условиях, размер выходного PNG-файла несколько больше размера аналогичного GIF-файла.

Второй вариант PNG-формата не рассчитан на уменьшение цветовой информации. Он так и называется **PNG-Truecolor**. И у него почти нет элементов управления, изменяющих параметры рисунка. В окне сиротливо располагается один-единственный флажок **Transparency**, с помощью которого, как мы уже знаем, фоновый цвет указывается в качестве прозрачного.

После того, как мы выбрали подходящий формат и установили все необходимые значения, нужно выбрать вариант экспорта рисунка. Adobe ImageStyler ориентирован на графику для WWW, поэтому может генерировать помимо графических файлов еще и HTML-файлы, в которых содержатся ссылки на рисунки. Выбор варианта экспорта производится все в том же стандартном рабочем окне **Export**, где для этого приготовлен еще один выпадающий список. Значение **Entire Composition** является простейшим вариантом. При его выборе просто генерируется один выходной графический файл, содержащий все изображение в целом. Если поставить флажок **Make Page**, то будет создан дополнительный HTML-файл, содержащий ссылку на созданный графический файл. То есть предусмотрена возможность сразу создать страницу, которая содержит только изображение и уже готова к просмотру в браузере.

Значение **Trimmed Composition** указывает на то, что в графическом файле будет сохранена область минимального размера, включающая в себя все графические объекты, находящиеся на канве рисунка. То есть в этом случае сохраняется только само изображение, а фон урезается по минимуму. Как и в предыдущем режиме, здесь можно создать HTML-файл, установив флажок **Make Page**.

Вариант **AutoSlice** сохраняет каждый отдельный объект или группу объектов, входящую в состав изображения, в отдельном файле и принудительно генерирует HTML-файл, содержащий ссылки на все эти файлы.

Последнее значение **AutoLayout** сохраняет изображение в целом, с раскладкой слоев и отдельных объектов. Как и в предыдущем случае, здесь самостоятельно генерируется дополнительный HTML-файл, в котором содержится ссылка на искомый графический файл.

Так как последние два режима автоматически создают HTML-файл, при их выборе флажок **Make Page** становится недоступным.

Adobe ImageStyler может экспортировать не все изображение, а лишь текущее выделение. Для этого используется команда меню **File/Export Selection**. При этом в выходной файл попадает не только объект или группа объектов, входящих в выделение, но и части других графических объектов, которые находятся на задних планах, и попали в прямоугольный сектор выделения. Поэтому данный режим стоит использовать весьма осторожно.

Как мы говорили ранее, существует возможность создавать изображения, реагирующие на действия пользователя. Для этого применяется технология Java Script. При сохранении рисунка формируется сценарий на языке Java, управляющий поведением изображения. У такого рисунка может быть максимум четыре состояния. Все они задаются в рабочем окне **JavaScript**. Тут располагается четыре строки просмотра, в которых показываются все фазы изображения. Стандартный вид рисунка показывается в строке **NoAction**. Именно так изображение будет выглядеть, если не производить с ним никаких действий. В строке **OnMouseOver** показывается вид изображения в момент прохождения над ним курсора мыши. То есть, рисунок при этом может измениться. Третья строка — **OnMouseDown**. В ней показывается состояние рисунка в тот момент, если на поле рисунка щелкнуть левой кнопкой мыши. И последнее состояние — **OnMouseOut**. Оно возникает в тот момент, когда курсор покидает поле рисунка.

По умолчанию для каждого изображения формируется только состояние **NoAction**. Для создания нового необходимо нажать кнопку **New JavaScript Action**. При этом для каждого нового состояния рисунок дублируется. Для изменения его вида мы можем использовать все стандартные средства ImageStyler — изменять стили, накладывать текстуры, добавлять графические формы, менять цвета и градиентную цветовую заливку, вращать и смещать изображение. При нажатии на кнопку **New JavaScript Action** автоматически создается новое состояние, учитывая тот порядок, в котором оно указано в окне **JavaScript**. Если какое-либо состояние не нужно, то его все-таки сначала придется создать, а затем удалить. Для удаления состояния и рисунка, ассоциированного с ним, необходимо в окне **JavaScript** выделить щелчком мыши строку с обозначением искомого состояния, а затем нажать на кнопку **Delete JavaScript Action** с изображением корзины. Естественно, состояние **NoAction** удалению не подлежит.

Сохранение подобного изображения может производиться только при использовании HTML-страниц, так как сохранять код сценария на языке Java

в графических файлах пока невозможно. Поэтому при установке параметров экспорта необходимо ставить флажок **Make Page** или выбирать режимы с принудительным созданием страниц.

При сохранении в указанном каталоге создается дополнительная папка с именем images и HTML-файл примерно следующего содержания:

```
<Html>
<Head>
<Title>
form1
</Title>
<Script Language="JavaScript">
<!--Adobe(R) ImageStyler(TM) 1.0 Generated JavaScript. Please do not
edit.
isamap = new Object();
isamap[0] = "_df"
isamap[1] = "_ov"
isamap[2] = "_ot"
isamap[3] = "_dn"

function isimgact(id, act)
{
        if(document.images) document.images[id].src = eval( "isimages." +
id + isamap[act] + ".src");
}

if (document.images) { // ensure browser can do JavaScript rollovers.
isimages = new Object();
isimages.form1_df = new Image();
isimages.form1_df.src = "images/form1image.gif";

isimages.form1_ov = new Image();
isimages.form1_ov.src = "images/form1imageov.gif";

isimages.form1_ot = new Image();
isimages.form1_ot.src = "images/form1imageot.gif";

isimages.form1_dn = new Image();
isimages.form1_dn.src = "images/form1imagedn.gif";
```

```
}
// end generated JavaScript. -->
</Script>
</Head>
<Body BGcolor="#ffffff">


<!-- The table is not formatted nicely because some browsers cannot join
images in table cells if there are any hard carriage returns in a TD. -->


<Table Border="0" CellSpacing="0" CellPadding="0" >
       <Tr>
              <Td Width="27" Height="19"></Td>
              <Td Width="141" Height="19"></Td>
       </Tr>
       <Tr>
              <Td Width="27" Height="176"></Td>
              <Td Width="141" Height="176"><a Href="http://www.adobe.com"
OnMouseOver="isimgact( 'form1',1)"  OnMouseOut="isimgact( 'form1',2)"
OnMouseDown="isimgact( 'form1',3)"  ><Img Src="images/form1image.gif"
Border="0" Height="176" Width="141" Name="form1" Alt=""></a></Td>
       </Tr>
       <Tr>
              <Td><Img Src="images/is_single_pixel_gif.gif" Alt=""
Width="27" Height="1"></Td>
              <Td><Img Src="images/is_single_pixel_gif.gif" Alt=""
Width="141" Height="1"></Td>
       </Tr>
</Table>
<!--Adobe(R) ImageStyler(TM) DataMap1.0 DO NOT EDIT
end DataMap -->
</Body>
</Html>
```

Все, что находится между тэгами `<Script Language="JavaScript">` и `</Script>`, является кодом сценария на языке Java. А технологию внедрения подобных сценариев в Web-страницы мы рассматривали в *главе 1* (*см. раздел "Активные элементы"*). Впрочем, мы получаем уже готовый HTML-документ, из которого можно просто скопировать необходимые блоки кода и вставить в текст проектируемой Web-страницы.

Анимация

Все мы видели, что в оформлении почти каждого сайта используются движущиеся рисунки. Их также часто называют анимационными. Как мы помним, только графический формат GIF позволяет создавать движущееся изображение. При этом в файле хранятся все фазы или кадры рисунка, а браузеры после загрузки файла распознают его тип и поочередно отображают каждый кадр, что и создает эффект мультипликации.

Для создания подобных движущихся изображений мы используем Animagic GIF Animator.

Внешний вид основного рабочего окна этого продукта показан на рис. 2.5. Помимо меню, панели инструментов и строки статуса, в основном окне располагается изображение кадра рисунка, окно, содержащее список искомых кадров, и окно, отображающее палитру рисунка.

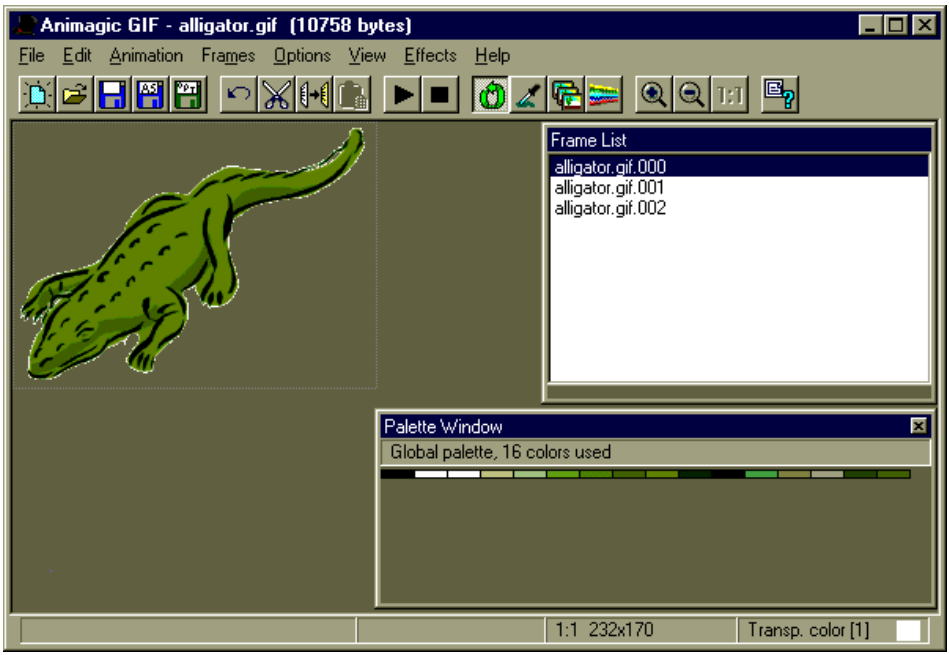

Рис. 2.5. Внешний вид приложения Animagic GIF Animator

Как мы уже говорили, основа анимации — это отдельные кадры. В рабочем окне постоянно показывается только один кадр, но используя окно **Frame List**, можно выбрать именно тот кадр, который мы хотим отобразить.

Основное ограничение Animagic GIF Animator состоит в том, что по сути своей он не является графическим редактором. Это всего лишь инструмент для создания готового анимационного изображения из множества заранее

созданных кадров. Конечно, многие кадры можно создавать в самом Animagic GIF Animator, но только посредством применяемых к готовому кадру эффектов. Можно изменять какие-либо свойства кадра, но изменить само изображение невозможно. Поэтому процедура создания анимационного изображения разбивается на две стадии. Сначала нужно заготовить кадры с использованием какого-либо графического редактора и сохранить их в одном из распространенных графических форматов, а потом загрузить их в Animagic GIF Animator и создать анимационное изображение.

Для создания движущегося рисунка необходимо выполнить команду **File/New** или нажать кнопку **New** на инструментальной панели. При этом очищается окно **Frame List**. Теперь нужно добавлять в него кадры, что можно сделать при помощи команд **File/Append Frames** и **File/Insert Frames**. Отличие их достаточно незначительно. Команда **Append Frames** добавляет новый кадр в конец списка, то есть делает его последним, а команда **Insert Frames** вставляет новый кадр сразу после выделенного в окне **Frame List** кадра.

При выполнении обеих этих команд активизируется стандартное диалоговое окно открытия файла. Animagic GIF Animator распознает файлы форматов GIF, PCX, BMP, JPEG, видеофайлы AVI и текстовые TXT. При открытии графического файла происходит анализ цветов, входящих в состав изображения и все они включаются в состав палитры. При открытии новых файлов цвета, не входящие в состав текущей палитры, добавляются в нее без удаления уже входящих в ее состав цветов.

Необходимо особо отметить возможность использования текстовых файлов в качестве отдельных кадров. Animagic GIF Animator преобразовывает текст в изображение, а затем сохраняет его в виде отдельного кадра.

После выполнения всех фаз для создания анимационного изображения, его необходимо сохранить в GIF-файле. Для сохранения существует четыре варианта. Во-первых, для каждого кадра можно использовать отдельный файл. Для этого применяется команда **File/Save Frames**. При этом активизируется диалоговое окно сохранения файлов. В строке **Имя файла** записывается общее имя для всех сохраняемых файлов, а затем к нему дописывается порядковый номер кадра в общей последовательности.

Команда **File/Save** просто сохраняет все кадры анимационного изображения в одном файле с его родным именем. В том случае, если имя выходного файла нужно изменить, применяется команда **File/Save as**. Необходимо отметить, что при сохранении всего анимационного изображения незарегистрированная копия Animagic GIF Animator принудительно добавляет первый кадр с текстом, содержащим информацию о том, при помощи какого продукта был создан этот рисунок. Это еще один повод для регистрации программы. Но самая интересная возможность кроется за командой **File/Save optimized as**. При ее выполнении Animagic GIF Animator пытается уменьшить размер конечного файла за счет точного подсчета цветов во всех кадрах и удаления лишних цветов из палитры. Также во время оптимизации

может уменьшаться размер некоторых кадров. По минимуму палитра может содержать шестнадцать цветов. Как известно, цветовая палитра хранится вместе с изображением в GIF-файле и уменьшение ее размера может повлиять на его размер. В сторону уменьшения, конечно. Хотя это уменьшение не будет сильно ощутимым, но воспользоваться этой возможностью стоит. Все-таки наши каналы до сих пор не являются образцом для подражания, и до тех пор, пока их пропускная способность далека от идеала, стоит экономить везде, где это возможно, но, конечно, без ущерба для авторского замысла.

Animagic GIF Animator является одним из немногих продуктов, которые позволяют работать с палитрами. Единообразное оформление сайта всегда является хорошим тоном. Естественным продолжением этой идеи является использование общей цветовой палитры, единого цветового набора для всех рисунков, использованных в оформлении сайта. Для того чтобы сохранить цветовую палитру одного рисунка с целью ее последующего использования, применяется команда меню **File/Store Palette**. При этом создается файл с расширением PAL, в котором и будет храниться этот цветовой набор. Обратная операция, то есть импорт палитры, осуществляется при помощи команды **File/Load Palette**. Но при этом может открываться не только PAL-файл, но и графические файлы, которые содержат в себе информацию о применяемом цветовом наборе. Это файлы формата GIF, PCX и BMP. Естественно, при открытии подобных файлов загрузка хранящегося в них изображения не происходит. Загружается только цветовая палитра. При этом та палитра, которая использовалась до момента выполнения этой команды, уничтожается, а вместо нее используется свежезагруженный цветовой набор.

Работа с кадрами

В предыдущем разделе мы рассмотрели, как можно добавлять новые кадры в уже существующую последовательность. Но на этом работа с ними не заканчивается. Все остальные команды работы с кадрами собраны в пункте меню **Frames**. Пройдем их последовательно.

При помощи команды меню **Frames/Shift** можно сдвинуть каждый кадр относительно общего расположения рисунка. При выполнении этой команды активизируется диалоговое окно **Shift frames** (рис. 2.6). Сдвиг выделенного кадра осуществляется путем указания новых относительных координат верхнего левого угла. При этом изменяются значения в полях **Delta**. Выделение кадров производится в списке **Select one or more frames**. В диалоговом окне присутствует также кнопка **Select all**, которая включает в выделение все кадры, но ее применение в данном режиме бессмысленно, так как если изменить координаты у всех кадров сразу, относительные координаты, очевидно, не изменятся, и эффект пропадет. При помощи этой команды можно достаточно легко заставить двигаться изображение, состоящее изначально

из одного кадра. Для этого у каждого последующего кадра указывается все более увеличивающееся смещение, что и заставляет изображение двигаться.

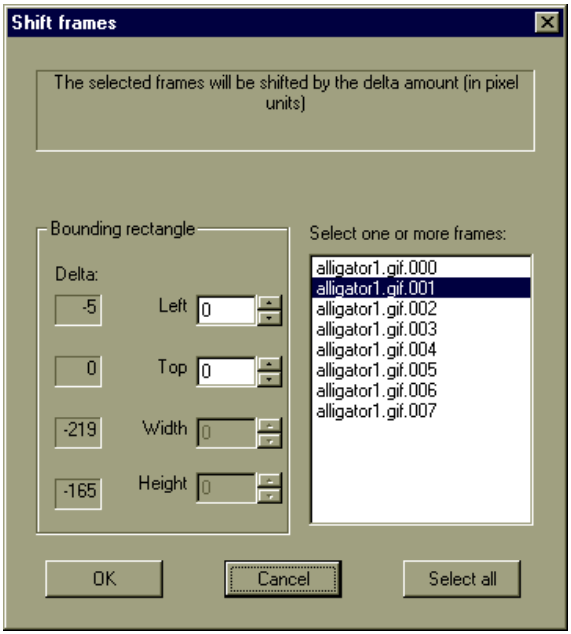

Рис. 2.6. Диалоговое окно **Shift frames**

Иногда бывает необходимо изменить размеры рисунка без изменения масштаба самого изображения. Для этого используется команда **Frames/Crop**. При ее выполнении визуализируется диалоговое окно **Crop frames**, являющееся функциональным аналогом рассмотренного ранее окна **Shift frames**. В этом окне нам доступны все поля ввода из группы **Bounding rectangle**. В них указывается количество пикселов, на которое будет обрезан выделенный кадр по соответствующим осям координат. Общие координаты кадра при этом не изменяются, поэтому возможно несовпадение с остальными кадрами. Если необходимо обрезанные кадры выровнять по верхнему левому углу всех кадров, не подвергшихся этой процедуре, следует поставить флажок **Align to top left corner after cropping**.

Animagic GIF Animator позволяет производить отражения и повороты кадров относительно их главных осей. Для этих целей используется команда меню **Frames/Flip and rotate**, при выполнении которой визуализируется одноименное диалоговое окно (рис. 2.7). Как видно, пользователю предлагается выбрать те кадры, над которыми будет производиться операция, а затем уже в выпадающем списке **Operation** выбрать необходимое преобразование изображения. Первые два значения списка **Flip (vertical mirror)** и **Mirror (horizontal)** отвечают за зеркальные отражения рисунка. Остальные три предполагают разворот рисунка на девяносто, сто восемьдесят и двести

семьдесят градусов. Направление вращения можно выбрать при помощи группы радиокнопок, которые становятся доступными после выбора какой-либо операции.

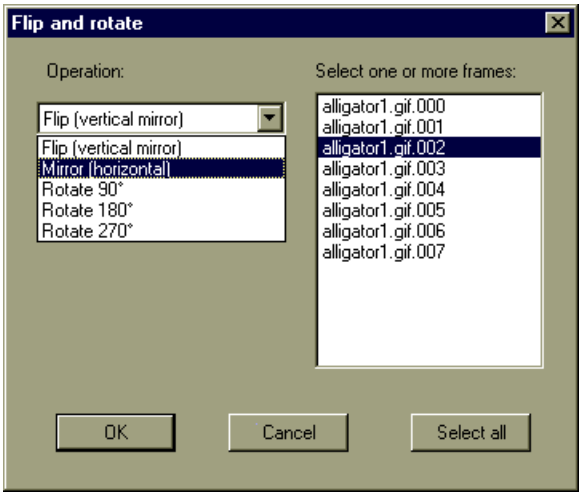

Рис. 2.7. Диалоговое окно **Flip and rotate**

Для того чтобы продублировать какой-либо кадр, используется команда **Frames/Duplicate**. Удаление кадров производится при помощи команды **Frames/Delete**. При этом активизируется уже знакомое нам диалоговое окно со списком кадров. В списке выбираются кадры, подлежащие удалению, и нажимается кнопка **OK**.

Также есть возможность добавить кадр без изображения, целиком залитый одним цветом. Для этих целей используется команда **Frames/Insert solid color**. При этом курсор принимает форму пипетки и цвет заливки кадра можно взять как с самого рисунка, так и из окна палитры.

Иногда бывает необходимо поменять последовательность воспроизведения кадров. Команда **Frames/Reverse order** позволяет осуществить это действие. В списке появившегося диалогового окна выбирается несколько кадров и по нажатию кнопки **OK** они меняются местами.

С помощью команды **Frames/Moving sprite** можно создавать движущиеся изображения автоматически. Это более продвинутый аналог команды **Shift frames**. Достаточно задать шаг смещения по вертикали и горизонтали, количество кадров, к которым будет применен этот эффект, и каждый выделенный кадр будет сдвинут относительно предыдущего на заданную величину шага.

И последняя команда **Frames/Gray scale** предназначена для того, чтобы переводить кадры в полутоновую палитру. То есть выбранные кадры преобразовываются таким образом, что изображение показывается в оттенках

серого цвета. Необходимо отметить, что для каждого кадра, попавшего под действие этой команды, формируется отдельная полутоновая палитра.

На этом возможности манипуляций с отдельными кадрами заканчиваются. И мы переходим к установке параметров воспроизведения анимационного изображения.

Параметры анимации

Все операции по установке параметров воспроизведения анимационного изображения рисунка осуществляются при помощи команд меню **Animation**. Первые команды этого пункта достаточно просты. Команды **Animation/Play** и **Animation/Stop** запускают и останавливают воспроизведение изображения. Впрочем, намного удобнее пользоваться соответствующими кнопками на панели инструментов, чем этими командами. Следующие две команды тоже, скорее всего, не будут часто использоваться. Это **Animation/Next frame** и **Animation/Previous frame**. Они делают текущим следующий и предыдущий кадр соответственно. То же самое гораздо быстрее осуществляется при помощи одиночного щелчка мыши в окне со списком кадров.

Следующая возможность уже представляется достаточно важной. Команда **Animation/Reduce color depth** предназначена для уменьшения цветовой насыщенности всего анимационного изображения. При выполнении этой команды активизируется диалоговое окно **Reducing color depth** (рис. 2.8). В поле ввода **Number of colors in the new global palette** указывается новое количество цветов, которое будет содержать общая цветовая палитра. Напомню, что общая цветовая палитра является объединением палитр каждого кадра. Изменение цветового набора общей палитры отражается и на каждом локальном цветовом наборе. Применяется эта возможность для уменьшения размера конечного GIF-файла. Как мы уже говорили, уменьшение размера палитры не слишком сильно повлияет на общий размер файла, но ситуация сейчас такова, что бороться стоит за каждый байт.

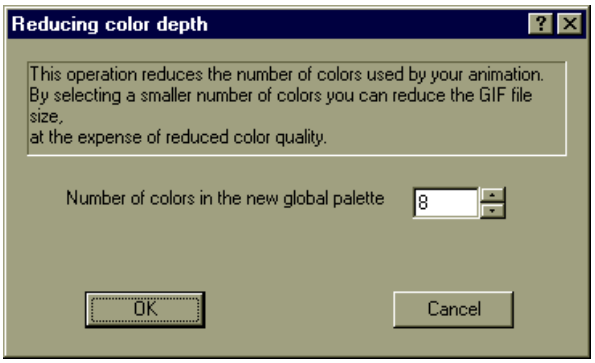

Рис. 2.8. Диалоговое окно **Reducing color depth**

Одним из наиболее часто упоминаемых преимуществ GIF-формата является наличие прозрачного цвета. С помощью программы Animagic GIF Animator мы также можем объявлять какой-либо цвет прозрачным. Для этого существует команда меню **Animation/Transparent color**. При выполнении этой команды курсор принимает форму пипетки, что позволяет осуществить выбор цвета либо на изображении активного кадра, либо из окна палитры. В том случае, если текущий кадр использует свою локальную палитру, прозрачность не распространится на аналогичные цвета других кадров. Если же применяется глобальная палитра, то для всех остальных кадров этот цвет тоже становится прозрачным.

У нас также есть возможность управлять циклами воспроизведения анимационного изображения. Для этого используется команда **Animation/Loop count**. Выполнение этой команды активизирует диалоговое окно **Loop iteration count** (рис. 2.9). С ее помощью мы можем выбрать одну из возможностей — либо мы заставим цикл мультипликации воспроизводиться постоянно, либо остановим воспроизведение после некоторого количества повторов. Первый вариант устанавливается выбором радиокнопки **Infinite loop**, второй — **Stop after**. С этой радиокнопкой связано поле, в котором указывается количество циклов, после которого воспроизведение мультипликации остановится.

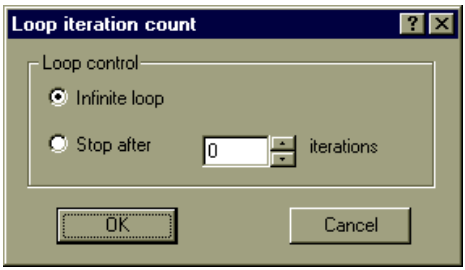

Рис. 2.9. Диалоговое окно **Loop iteration count**

Одним из параметров воспроизведения анимации является частота смены кадров. Это свойство изменяется при помощи команды меню **Animation/Frame Rate**, которая вызывает одноименное диалоговое окно (рис. 2.10).

Прежде всего, необходимо установить общую скорость воспроизведения анимации. Для этого используется поле ввода **Milliseconds per frame**, находящееся в группе элементов управления **All Frames**. В данном поле указывается количество миллисекунд, которое отведено для отображения каждого кадра. Таким образом, изменяя этот параметр, можно регулировать количество кадров, показываемых в течение одной секунды. Скорость воспроизведения — количество кадров в секунду — показывается в информационном поле **Frames per second**.

Все эти свойства влияют на общие параметры анимации, которые относятся ко всем кадрам сразу. Но у нас есть возможность указывать время воспроизведения каждого кадра в отдельности. Для этого служит общий список всех кадров и группа элементов управления **Selected Frame**. После того, как

мы задали время воспроизведения каждого кадра, можно установить количество повторов каждого отдельно взятого кадра. Для этого в списке выбирается необходимый кадр, а в поле ввода **Repeat** устанавливается количество повторов. Таким образом, можно увеличивать время экспозиции каждого кадра по отдельности. Время отображения этого кадра показывается в информационном поле **Milliseconds this frame**. Существует некоторое ограничение по времени — мы можем устанавливать только промежутки, кратные основному времени воспроизведения кадра.

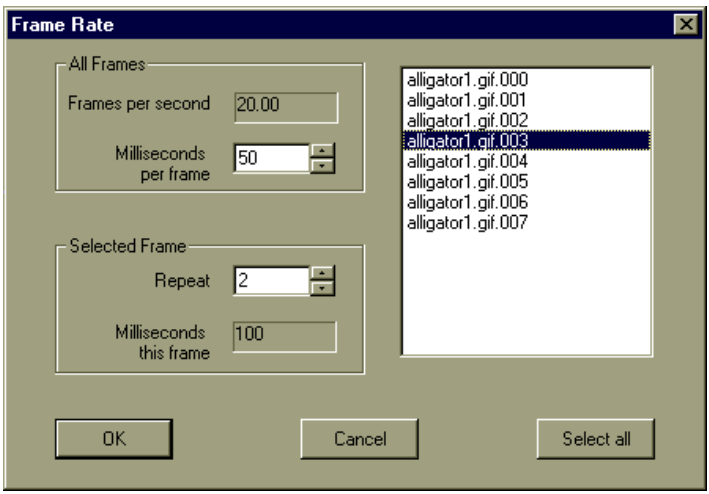

Рис. 2.10. Диалоговое окно **Frame Rate**

В списке возможностей воздействия на ход воспроизведения анимации стоит, пожалуй, отметить еще одну возможность. Команда меню **Animation/Unoptimize** позволяет частично отменить результаты оптимизации, если она к тому моменту уже была произведена. При этом уменьшенные кадры будут приведены к своему исходному состоянию. Но урезанная цветовая палитра не будет дополнена до исходной, так как информация об исходной палитре при оптимизации теряется безвозвратно. Данная возможность деоптимизации может применяться в том случае, если скорость соединения в сети заведомо велика. Например, если созданное анимационное изображение будет применяться для отображения во внутренней интрасети.

"Эффектные" кадры

Animagic GIF Animator содержит в себе возможности создания некоторых графических эффектов. Например, с их помощью производится плавный переход от последнего кадра к первому, сглаживая таким образом неизбежный скачок изображения в момент этого перехода. Рассмотрим эти возможности по очереди.

Команда меню **Effects/Fade in** реализует эффект, который заключается в том, что изображение постепенно проявляется на однотонном фоне. Этот эффект обычно применяется к первому кадру анимации. При выполнении команды **Effects/Fade in** активизируется диалоговое окно **Effects dialog** (рис. 2.11). Для выполнения этого эффекта генерируются новые кадры, которые будут добавлены перед целевым кадром. Количество новых кадров, а значит и гладкость воспроизведения эффекта, указывается в поле ввода **Number of new frames to insert**. Можно регулировать скорость воспроизведения эффекта за счет изменения времени, отводимого на отображение каждого генерируемого кадра. Для этого используется поле **Miliseconds per frame**. Как мы уже говорили, изображение кадра, к которому применен этот эффект, постепенно проявляется на однотонном фоне. Для выбора цвета фона из общей палитры используется кнопка **Select from palette**, но если в палитре нет требуемого цвета, его можно задать прямым указанием его RGB-кода. Для этого используется кнопка **Select RGB value**. По умолчанию в генерируемых кадрах в качестве исходного цвета применяется белый цвет с RGB-кодом (255, 255, 255).

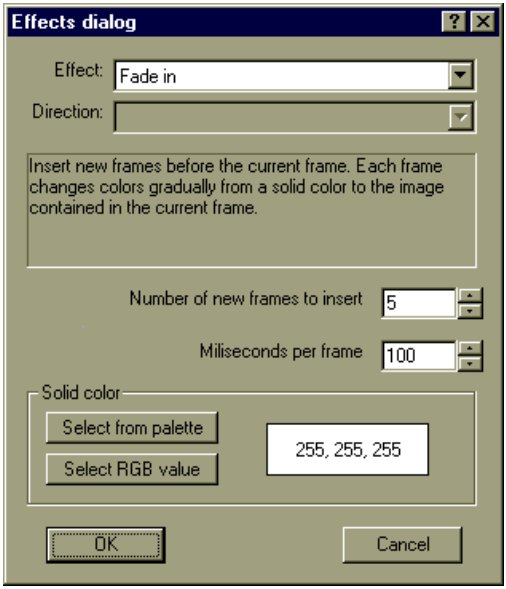

Рис. 2.11. Диалоговое окно **Effects dialog**

В коллекции Animagic GIF Animator есть и обратный эффект, вызываемый командой **Effects/Fade out**. При осуществлении этого эффекта изображение кадра постепенно заливается выбранным цветом. Установка свойств этого эффекта производится при помощи того же самого диалогового окна, что и для предыдущего. В выпадающем списке **Effect** сразу установлено значение **Fade out**. Кстати, при помощи выбора из списка можно установить любое направление действия эффекта, независимо от того, какая команда — **Effects/Fade in** или **Effects/Fade out** — активизировала диалоговое окно.

Следующий эффект достигается при помощи команды меню **Effects/Dissolve**. Эта команда активизирует диалоговое окно **Effects dialog** (рис. 2.12). Эффект представлен в трех вариантах. Конкретный вариант этого эффекта выбирается в выпадающем списке **Effect**. Этот эффект проявляется тем, что кадр разбивается на ячейки и кусочки изображения, содержащиеся в них, заменяются на ячейки, содержащие соответствующие части того кадра, который завершает эффект. То есть происходит замена некоторой части ячеек одного кадра на соответствующие ячейки другого кадра. Честно говоря, проще посмотреть один раз на то, как это выглядит, и сразу все сразу станет понятно.

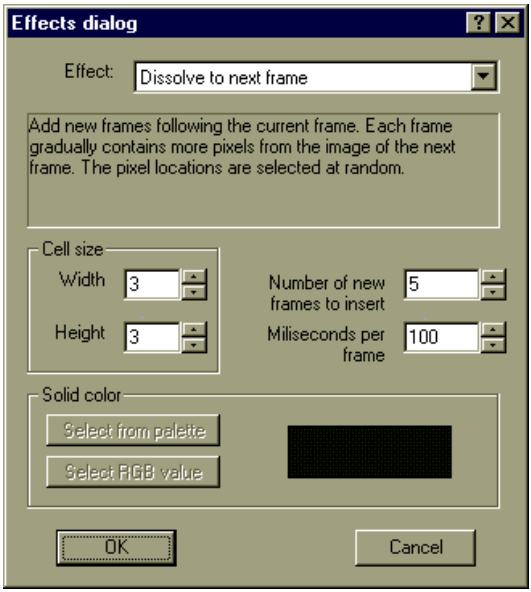

Рис. 2.12. Диалоговое окно **Effects dialog**

Эффект, вызываемый командой **Dissolve**, применяется к выбранному заранее кадру, но как расположатся добавленные кадры эффекта — зависит от выбранного вами варианта, который, как мы помним, указывается в основном выпадающем списке диалогового окна. Если выбрано значение **Dissolve from solid color**, то новые генерируемые кадры расположатся до выбранного кадра, к которому применяется эффект. При этом в качестве стартового будет применен чистый кадр, залитый каким-либо цветом, а из него уже будет постепенно проявляться рисунок искомого кадра.

Значение **Dissolve to solid color** порождает эффект, по своему действию обратный предыдущему. В этом варианте генерируемые кадры оказываются после стартового, и изображение этого кадра постепенно размывается до однотонного.

Но есть и третий вариант этого эффекта, который позволяет постепенно переходить от изображения одного кадра к следующему. Для этого зарезер-

вировано значение **Dissolve to next frame**. Как всегда, чаще всего этот вариант применяется для перехода от последнего кадра к следующему.

Гладкость перехода зависит от количества кадров, отводимых для работы эффекта и размера ячеек. Чем больше количество кадров и меньше размер ячеек, тем более гладко происходит переход. Естественно, большое количество кадров увеличивает объем графического файла, содержащего рисунок, поэтому, как всегда, приходится искать баланс.

Размер ячеек указывается при помощи полей ввода, объединенных в группу **Cell size**. В поле **Width** указывается ширина таких ячеек, а в поле **Height** — высота. Значения указываются в пикселах. По умолчанию используются квадратные ячейки размером три на три пиксела, но всегда можно изменить как размеры, так и пропорции ячеек. Справа от этой группы элементов управления расположены поля ввода, управляющие количеством генерируемых кадров и длительностью их воспроизведения. С ними мы уже знакомы.

В самом низу диалогового окна расположена группа **Solid color**. Элементы управления, находящиеся в ней, применяются в том случае, если эффект использует одноцветный кадр. Эти две кнопки позволяют устанавливать цвет этого кадра. В том случае, если в каждом из кадров используется одна цветовая палитра, доступна только кнопка **Select from palette**, позволяющая выбрать необходимый цвет из общей палитры. Если же для каждого кадра используется локальная палитра, можно указать RGB-код нужного цвета при помощи кнопки **Select RGB value**.

За командой **Effects/Banner scroll** скрывается эффект, заставляющий изображение выбранного кадра скользить в выбранном направлении. Выбор варианта скольжения производится все в том же выпадающем списке **Effect**. Значение **Banner scroll out** указывает на то, что на них изображение будет смещаться, постепенно выходя за пределы рисунка. Для того чтобы изображение, наоборот, вплывало в кадр, используется значение **Banner scroll in**. Совмещение этих двух возможностей позволяет изображению двигаться циклически. То есть в одну сторону оно будет уходить и одновременно наплывать с противоположной стороны. Этот вариант носит название **Banner rotate**.

Направление движения изображения выбирается в списке **Direction**. Доступны четыре варианта: движение слева направо, справа налево, сверху вниз и, конечно, снизу вверх. Для этого используются значения **from left to right**, **from right to left**, **from top to bottom** и **from bottom to top** соответственно. Все остальные доступные элементы управления нам уже знакомы.

Подобно эффекту, вызываемому командой **Dissolve**, действует и эффект, реализуемый при помощи команды меню **Effects/Wipe**. Но в данном случае переход от одного кадра к другому организуется не хаотически, а упорядоченными полосами. Направление появления полос, содержащих части изображения целевого кадра, указывается в списке **Direction**, который содержит все те же значения, что и для предыдущего эффекта. Порядок действия

эффекта устанавливается при помощи списка **Effect**. Итак, значение **Wipe from solid color** добавляет новые кадры перед стартовым, и на них изображение постепенно проявляется полосами, начиная с однотонного кадра. Цвет заливки этого кадра выбирается при помощи элементов управления группы **Solid color**. Значение **Wipe to solid color**, наоборот, исходное изображение постепенно закрашивает полосами выбранного цвета. Толщина линий заливки напрямую зависит от количества кадров, реализующих данный эффект. Пункт **Wipe to next frame** позволяет делать плавный переход с помощью эффекта от изображения выбранного кадра к изображению следующего.

Развитием этой же темы служит эффект спирального проявления, который активизируется командой **Effects/Spiral**. При этом изображение стартового кадра проявляется или закрывается прямоугольниками, образующими спираль. Эффект **Spiral to solid color** закрывает изображение прямоугольниками выбранного цвета, а второй вариант эффекта — **Spiral from solid color** на чистом кадре, залитом выбранным цветом, подобным же образом проявляет изображение искомого кадра. При этом в первом случае генерируемые кадры помещаются после стартового, а во втором — перед ним. Значение **Spiral to next frame** в качестве конечного кадра, к которому будет приводиться изображение, выбирает кадр, следующий за стартовым. Если стартовый кадр является последним в списке, а для всего анимационного рисунка не задано циклическое воспроизведение, то данный пункт нельзя будет выбрать из списка. Направление движения эффекта выбирается в списке **Direction**. Значение **from inside out** заставляет заполняющую область распространяться из центра стартового кадра к его границам. При этом изменяющаяся область представляет собой прямоугольник, пропорциями полностью совпадающий с пропорциями кадра и расположенный точно по его центру. С каждым новым кадром размеры этого прямоугольника увеличиваются. Таким образом, последний генерируемый кадр практически полностью закрывает стартовый. Значение **from outside in**, наоборот, предполагает заполнение, двигающееся от границ к центру. При этом область наложения нового эффекта представляет собой рамку с увеличивающейся от кадра к кадру толщиной. Таким образом, на последнем кадре эффекта эта рамка практически полностью закрывает стартовое изображение.

И последний эффект, входящий в коллекцию Animgic GIF Animator, очень похож на эффект Wipe. Но вызывается он командой **Effects/Blind**. Если эффект Wipe сначала рисовал одну большую полосу, потом добавлял к ней другую, и так далее до полной заливки кадра, то эффект Blind формирует множество вертикальных или горизонтальных узких полос, а затем постепенно расширяет их, заполняя таким образом пространство кадра. На последнем кадре эффекта эти полосы практически сливаются. Стартовое расстояние между полосами, а значит и их толщину нельзя задавать явно, но она регулируется при помощи указания количества кадров, отводимых для реализации эффекта. Применяются три стандартных варианта эффекта:

Blind to solid color, **Blind from solid color** и **Blind to next frame**. Первый закрашивает изображение кадра, приводя его к одноцветному фону, второй, наоборот, на этом одноцветном фоне проявляет изображение исходного кадра, а третий вариант заменяет изображение стартового кадра рисунком, находящимся в следующем кадре. Как уже говорилось ранее, расширяющиеся линии могут иметь как вертикальную, так и горизонтальную ориентацию. Эта ориентация указывается при помощи списка **Direction**. Значение **vertical** устанавливает вертикальные полосы, а значение **horizontal**, естественно, горизонтальные.

На этом коллекция эффектов Animagic GIF Animator исчерпана и мы можем, пользуясь полученной информацией, создавать свои анимационные рисунки.

Теперь, после того, как мы узнали, как проектировать Web-сайты и изготавливать графические изображения для них, самое время перейти к рассмотрению вопросов, связанных с размещением сайта и обслуживанием его. Как решить эти проблемы, мы узнаем в следующей главе.

Глава 3

Администрирование сайта

Механизм работы сервера

В самом начале книги мы разобрались с механизмом получения ресурсов из Интернета. Но рассматривали мы эту проблему только со стороны удаленного пользователя. Сейчас же нас интересует совсем другая сторона. Мы создали свой сайт. Как теперь заставить его работать? В этой части мы многое узнаем о работе сайтов, о проблемах, связанных с ними, о процедурах их поддержки и многом другом.

Прежде всего, стоит разобраться с тем, где физически хранятся сайты и как они делаются доступными удаленному пользователю. Для хранения сайтов и отдельных HTML-документов необходим, естественно, компьютер, на котором будет установлена особая программа, называемая Web-сервером. Эта программа и принимает запросы, приходящие от удаленных пользователей, формирует ответы и отсылает их, пользуясь протоколом TCP/IP (Transmission Control Protocol/Internet Protocol). Этот протокол является некоторым сводом правил, которые определяют порядок связи между отдельными компьютерами и передачи данных между ними. Этот протокол состоит из двух частей. TCP занимается обменом данными. Для этого передаваемые данные разбиваются на отдельные части — пакеты, которые помимо самих передаваемых данных содержат сведения, позволяющие осуществлять их передачу и получать подтверждение об их прибытии. А за передачу по сети разбитых подобным образом данных отвечает уже протокол IP. Он занимается маршрутизацией, или, как ее еще называют — роутингом. Основываясь на этом протоколе работает Web-сервер. Однако для пересылки HTML-документов используется протокол HTTP, который мы рассматривали ранее. Помимо отсылки HTML-документов, он обычно позволяет создавать и администрировать FTP-серверы. То есть, в дополнение ко всему, хороший Web-сервер должен еще поддерживать протокол обмена файлами FTP (File Transmission Protocol).

Как мы уже говорили, основной проблемой Web-страниц является их предопределенная статичность. Для ее решения были разработаны стандарты

программ, которые функционируют при поддержке сервера и позволяют создавать интерактивные странички. Одним из таких протоколов является CGI (Common Gateway Interface). В отличие от различного рода скриптов, CGI-приложения выполняются не на компьютере пользователя, а на самом сервере. CGI-приложения являются одной из разновидностей активных элементов, которые мы рассматривали в первой части, при знакомстве с редактором FrontPage 2000. Обычно они применяются для обработки данных, введенных при помощи форм.

У Web-сервера обычно есть некая структура каталогов, которая упорядочена и приспособлена для работы сервера. В одной из папок находятся HTML-документы, в другой — хранятся файлы для архива FTP и отдельная папка предназначена для исполняемых модулей. Если на одном сервере располагается несколько сайтов, то для каждого сайта выделяется отдельный каталог и необходимая структура папок формируется в каждом таком каталоге.

Обычно Web-сервер позволяет присваивать папкам любую комбинацию из двух признаков: доступ на чтение и доступ на выполнение. Доступ на чтение присваивается папкам, в которых расположены документы, а доступ на исполнение — папкам с исполняемыми модулями.

Самый простой и очень часто используемый Web-сервер для малых и средних сайтов поставляется вместе с Windows 95 и Windows 98. Это Personal Web Server. Этот сервер поддерживает две службы: HTTP и FTP. Установить режим работы сервера можно, вызвав окно его свойств либо из панели управления, либо двойным щелчком на значке Web-сервера (TrayIcon) на панели сервисов (рис. 3.1). Как видно, там указан Internet-адрес персонального Web-сервера (в данном случае, это http://localhost). Помимо этого, в редакторе свойств персонального Web-сервера нас будет интересовать вкладка **Службы**. На ней показывается текущий статус служб FTP и HTTP. Позаботьтесь о том, чтобы в моменты разработки ваших сайтов сервис HTTP был запущен. Логичным представляется решение запускать его автоматически при запуске Web-сервера.

Теперь рассмотрим структуру каталогов Web-сервера. Обычно в каталоге Web-сервера (по умолчанию — WWWSHARE) находится три вложенных каталога. FTPROOT является корневым каталогом для сервиса FTP. Каталог WWWROOT, как можно догадаться, предоставляет место для HTML-документов, которые будут составлять наполнение личного сайта. Там же располагается и документ default.html, который загружается по умолчанию, если удаленным пользователем указан адрес только сервера, а наименование документа не указано. А вот каталог **Scripts** предназначен как раз для размещения исполняемых программ, поэтому все exe- и dll-файлы должны храниться именно в этом каталоге.

Итак, при указании в браузере имени нашего компьютера, физически мы будем находиться в каталоге WWWSHARE, а затем уже можно указывать наименование того файла, который нас будет интересовать. Естественно,

необходимо помнить, что в полном имени файла и, соответственно, при указании структуры каталогов используется обратный слеш, а в Internet-адресе — прямой слеш. Для обхода этого ограничения применяются псевдонимы каталогов.

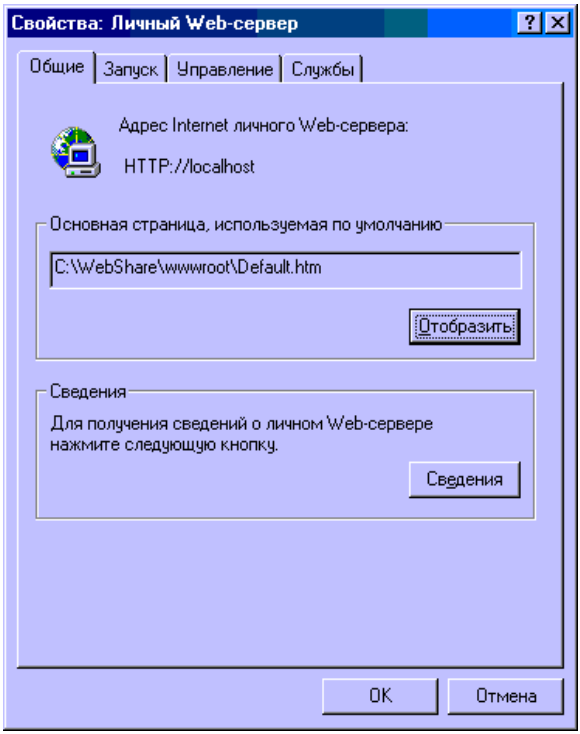

Рис. 3.1. Диалоговое окно установки свойств персонального Web-сервера

Для того чтобы увидеть назначенные псевдонимы для каталогов, необходимо в окне свойств Web-сервера (см. рис. 3.1), на вкладке **Управление** нажать одноименную кнопку. Текущий браузер загрузит страницу **Администратор служб Internet**. На этой странице необходимо будет указать закладку **Каталоги**, и для управления будет предоставлен список необходимых каталогов с их псевдонимами (рис. 3.2).

При этом каждый каталог можно удалить, добавить по своему усмотрению новые установки или изменить текущие. Так, при нажатии на ссылку **Изменить** пользователю предоставляется страница, где можно указать положение данного каталога на диске локальной машины, изменить его псевдоним и права доступа. Для каждого каталога можно указать возможность читать документы, находящиеся в нем и исполнять программы. Для каталога **Scripts** мы, естественно, должны дать разрешение на исполнение программ и запретить чтение из него. Правильно выбранная структура каталогов увеличи-

вает безопасность, что будет заставлять владельца сайта несколько логичнее размещать информацию. В принципе, конечно, можно каждому каталогу дать разрешения и на чтение и на выполнение приложений, но тогда это сильно повлияет на целостность и логическую структуру сайта, размещенного на этом сервере. Давно известно, что разделение прав является очень сильным профилактическим средством, поэтому не стоит пренебрегать им.

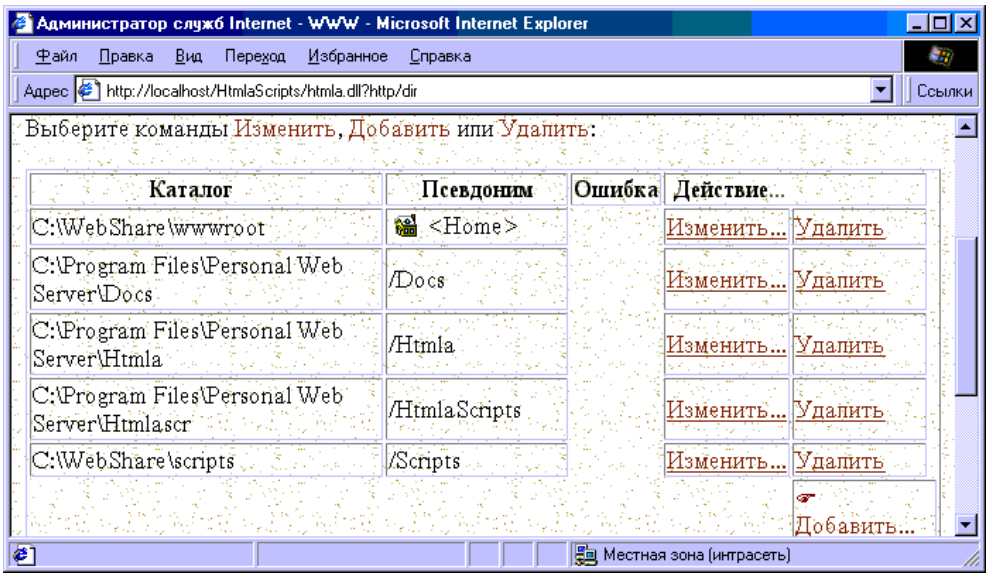

Рис. 3.2. Администрирование структуры каталогов персонального Web-сервера

В псевдонимах каталогов применяется прямой слеш, следовательно, для указания пути к исполняемому приложению на локальной машине необходимо указать адрес **http://localhost/scripts**. Естественно, вместо localhost вы должны будете вставить наименование своей машины.

Как видно, администрирование такого сервера не должно создавать какие-либо проблемы. Все очень просто и понятно. А в силу своей простоты сервер очень редко дает сбои.

Конечно, существуют и другие Web-серверы. Такие как Internet Information Server от Microsoft, известный Apache или маленький, но мощный Falcon Web Server от BlueFace. Выбор всегда есть. Главное, понять, какие именно качества нужны в конкретной ситуации, можно ли будет его поддерживать, не будет ли его администрирование слишком сложным. Вполне вероятно, что средств Personal Web Server вам покажется недостаточно и необходимо будет поставить более сложный сервер. Все это повлечет за собой изучение более глубокого слоя проблем и погружение в них постепенно сделает вас достаточно компетентным специалистом.

Размещение сайта на стороне

Очень часто сайты размещают на дисковом пространстве сервера какого-либо Интернет-провайдера. Причем, это не обязательно именно тот провайдер, который предоставляет вам или вашей организации доступ в Интернет. Доступ — это одно, а размещение сайта — совсем другое дело. Руководствоваться при выборе здесь нужно совсем другими параметрами, нежели при выборе провайдера для доступа в Интернет. Рассмотрим этот вопрос подробнее.

Прежде всего, нам потребуется дисковое пространство для размещения самого сайта и почтовый ящик. Почтовый ящик обычно предоставляется бесплатно, но необходимо обратить внимание на то, что предоставление почтовых адресов и предоставление почтовых ящиков — разные вещи. Очень часто декларируется предоставление большого количества адресов, но далеко не всегда для этих адресов предоставляются почтовые ящики. Получается ситуация, когда при указании одного из этих адресов удаленным пользователем почта действительно приходит на ваш сервер, но он переправляет ее в ящик, который может находиться совсем в другом месте. То есть почтовый ящик как физическое место хранения пришедших писем может прилагаться далеко не к каждому предоставляемому почтовому адресу. Часто бывает, что сами ящики предлагаются за дополнительную плату. Будьте осмотрительны.

Что касается объема дискового пространства, предоставляемого под сайт, то, на мой взгляд, не стоит обращать на этот критерий слишком уж пристальное внимание. Сейчас в сети можно найти столько свободного и практически бесплатного места для размещения сайта, что если выбранный провайдер не удовлетворяет именно по параметру дискового пространства, на это можно закрыть глаза.

Еще один параметр выбора провайдера для размещения сайта — это объем трафика. Мы все предполагаем, что на наш сайт будут заходить посетители. Причем чем больше, тем для нас лучше. При этом возникает некоторый обмен информацией между удаленными пользователями и Web-сервером. Этот объем и называется трафиком. Добейтесь получения четкой шкалы его оплаты. Если вы не оговорили заранее максимальный его предел, то всегда существует риск, что провайдер решит, что объем вашего трафика превысил все мыслимые пределы, мешает работе остальных серверов или что-нибудь еще. В лучшем случае вас пригласят в офис обсудить эту проблему. В худшем вы получите дополнительный счет, подлежащий немедленной оплате под угрозой отключения сервера. Поэтому никогда не доверяйте заявлениям о "неограниченном трафике". Подобные заявления всегда предполагают, что величину "неограниченности" устанавливает сам провайдер, чьи оценки далеко не всегда будут совпадать с вашими. Вообще, всегда требуйте четкого определения ограничений и не менее четкой ценовой шкалы, так как все заранее неурегулированные вопросы провайдеры обычно решают в свою пользу.

Далее необходимо установить способ сопровождения своего сайта. Стандартный вариант — доступ по FTP. В этом режиме вы можете перемещаться по своей структуре каталогов, добавлять, удалять и переименовывать файлы. Для нужд владельца обычного информационного сайта это более, чем достаточно. Часто встречается предоставление Telnet-доступа. В этом случае вы получаете практически полный доступ к серверу. Такой доступ позволяет вам управлять удаленным компьютером так, будто он подключен к вашей клавиатуре. При запуске утилиты Telnet вы попадаете в сеанс DOS, из которого и производится работа. А если у провайдера в качестве операционной системы используется UNIX, то вы попадете в сеанс его командного интерпретатора Shell, работа с которым не слишком удобна. По уровню распространенности и удобства это еще хуже, чем DOS. Но подобный доступ необходим только для отладки программ, работающих в составе сайта. Так как мы не занимаемся программированием, нам такой доступ не нужен. Более того, он снижает безопасность сервера провайдера, поэтому многие провайдеры отказывают своим клиентам в Telnet-доступе.

Иногда встречается и совсем неудобный вариант, когда в качестве средства управления сайтом используется электронная почта. То есть для того, чтобы изменить содержимое сайта, вы должны послать письмо администратору провайдера, в котором попросить его удалить какие-либо файлы или добавить их. При этом добавляемые файлы присоединяются к письму. К сожалению, выполнение этого поручения очень часто затягивается и почему-то сопровождается невероятным количеством ошибок. Не связывайтесь с таким провайдером! Гораздо лучше заплатить немного больше, но быть уверенным в том, что вы всегда сможете правильно сами управлять своим сайтом.

Как мы уже знаем, для работы нашего сайта необходима программа Web-сервер. У провайдера есть две альтернативы. Либо под управлением одного Web-сервера работают все сайты, либо для каждого сайта выделяется своя копия Web-сервера. В том случае, если в оформлении вашего сайта используется много активных элементов, предпочтительнее иметь свою копию сервера. Более того, у каждого Web-сервера есть свои настройки, которые управляют поведением сайтов. Если вы хотите настроить его по своему желанию, просто жизненно необходимо добиться предоставления отдельной копии.

Еще одна грань этой проблемы связана с тем, какое программное обеспечение было использовано при создании сайта. Если использовался Microsoft FrontPage, то для получения максимального эффекта и правильной работы внедренных активных элементов необходимо на сервере установить серверную часть FrontPage. Очень часто провайдеры требуют дополнительную плату за эту услугу. Если нет выбора, то лучше заплатить.

Каждый провайдер физически где-то расположен. При этом на него распространяются законы той территории, где он находится. А эти законы подчас могут ограничивать содержание вашего сайта. Необходимо четко знать, что вам запрещено и, при заключении договора, опираться только на законодательство. Многие российские провайдеры указывают в договоре, что размещаемые материалы не должны противоречить некой сетевой этике. Так вот, решать, какой

материал этой этике соответствует, а какой нет, будет сам провайдер, а точнее, один из его администраторов. И, вполне может случиться так, что его мнение разойдется с вашим. В таких случаях провайдер считает, что прав именно он, и удаляет спорные материалы. Поэтому, при заключении договора вставьте фразу примерно следующего содержания: "Материалы сайта не должны противоречить действующему на территории страны законодательству". В этом случае вы выходите на правовое поле и можете защищать свои права в суде. Это нормальная, здоровая практика.

И, наконец, главное — имя для сайта. Как мы помним, имена имеют вид **www.имя_сайта.домен**. Если быть точным, то префикс www не всегда используется в имени сайта. Домен указывает основную принадлежность сайта. Помимо национальных доменов, которые отданы в собственность отдельных стран, и могут ими использоваться любым способом (так, например, страна Тувалу продала свой национальный домен с очень престижным именем tv за кругленькую с их точки зрения сумму), существуют и шесть общемировых доменов, таких как .com, предназначенный для размещений сайтов коммерческих компаний, или .gov, использующийся правительственными организациями. Домены второго уровня, образуемые припиской еще одного имени слева (имя.домен), являются частной собственностью. Для того чтобы зарегистрировать свое имя домена, необходимо связаться с регистрирующей организацией этой страницы, проверить, не занято ли уже такое имя, оплатить некоторую сумму и этот домен второго уровня — ваш. Внутри этого домена владелец может создавать другие домены третьего уровня. Из этого и происходит стандартное имя **www.имя_сайта.домен**. При этом префикс www просто является именем домена третьего уровня. Очень многие провайдеры, являющиеся владельцами доменов второго уровня по определению, могут за меньшую сумму выделять домены третъего уровня. То есть полученное имя будет иметь вид **ваше_имя.имя_провайдера.домен**. А внутри этого доменного имени третьего уровня уже можно задавать любые префиксы, которые будут являться доменами четвертого уровня. Таким образом, очень часто имена наподобие **www.myname.myprovider.com** ошибочно называются доменами третьего уровня. Подобные домены всегда стоят намного дешевле и их можно даже найти бесплатно. По функциональности они ничем не отличаются от обычных и более престижных (и более дорогих) доменов второго уровня. Они точно так же обрабатываются поисковыми машинами, точно так же идентифицируются. Поэтому, если престижность имени не важна, а сэкономить средства хочется, регистрация для сайта доменного имени третьего уровня вполне оправданна.

В данный момент регистрация и ежегодная абонентская плата за домены второго уровня, принадлежащие России, составляет эквивалент тридцати долларов США. Если не хочется платить провайдеру, который возьмет немного больше (подчеркиваю, немного!), чтобы компенсировать хлопоты по отправке письма в РосНИИРОС, достаточно просто выбрать имя, отправить электронное письмо, оплатить имя, и домен — ваш!

Безопасность в сети

По настоящему безопасной можно считать лишь систему, которая выключена, замурована в бетонный корпус, заперта в помещении со свинцовыми стенами и охраняется вооруженным караулом, — но и в этом случае сомнения не оставляют меня.

Юджин Х. Спаффорд

Не спрашивайте, кого атакуют хакеры, они атакуют вас! Эта фраза, достаточно давно ставшая крылатой, как нельзя лучше характеризует положение дел. Если ваш Web-сервер важен для организации или для вас лично, если информация, размещенная на нем, критична и не предназначена для свободного распространения, если Web-сервер находится не у провайдера, а в корпоративной сети — защищайтесь.

Не жалейте денег и сил на защиту. Убытки от компьютерных атак велики даже в России, не говоря уже о более развитых в этом отношении странах. Если информация, размещенная на сервере, конфиденциальна и предназначается только для зарегистрированных посетителей сайта или для работников и партнеров фирмы, а ее разглашение может нанести убытки, скажем, в 50 000 рублей, не потратить на защиту хотя бы десять процентов от этой суммы — просто глупо. Для получения более точной величины процентного соотношения ценность информации/стоимость защиты стоит обратиться к экспертам по безопасности.

Иногда, конечно, их требования могут показаться немного параноидальными. Они могут посоветовать установить сервер в комнату без окон и запретить доступ уборщиц в эту комнату. Причем подобное требование будет одним из самых умеренных. Остальные могут быть еще жестче. Но дело в том, что любой подобный совет — оправдан. За все промахи уже заплатили другие. Не стоит вам платить за это еще раз. Лучше учиться на ошибках других и следовать указаниям экспертов по безопасности.

В одной из древнегреческих комедий хор пел: "Предусмотрительность лучше, чем непредусмотрительность". Несмотря на время, которое прошло с тех пор, фраза эта не потеряла актуальности и поныне. Если есть возможность выстроить систему безопасности — сделайте это.

Все опасности можно разделить на две категории — перехват внешнего трафика и вторжение в корпоративную сеть. Внешний перехват обычно осуществляется в отношении электронной почты. Причем, что самое неприятное, чаще всего в роли такого перехватчика выступает государство. В России в отношении практически каждого провайдера введен и осущест-

вляется план СОРМ-2 (Система оперативно-разыскных мероприятий). СОРМ-1 действовал в отношении телефонной связи любого типа. К чему это привело, мы все знаем. Сегодня по сотовой телефонной сети практически не передается никакая конфиденциальная информация, так как эта сеть чрезвычайно уязвима и элементарно прослушивается. План СОРМ-2 направлен на получение спецслужбами практически неограниченного доступа к электронной связи любого клиента того провайдера, у которого этот план введен в эксплуатацию. Ваша почта может быть прочитана! Поэтому шифруйте ее. Найдите в Интернете программу PGP, которая ориентирована как раз на защиту электронной почты и является одним из лидеров в своем секторе криптографических средств. Используйте ее.

Да, конечно, использование дополнительных программ создает дополнительные сложности, но альтернатива этому — доступность вашей почты. вас могут атаковать все, — от частных лиц до государства, включая и ваших конкурентов.

Вторая опасность, связанная с проникновением в локальную корпоративную сеть, порождена правилами обустройства Web-сервера. Этот Web-сервер, обычно устанавливается на отдельной машине, которая, естественно, включена в общую сеть корпорации. Следовательно, образуется цепочка: удаленный пользователь → содержимое сервера → Web-сервер → локальная сеть. Несмотря на то, что последние два перехода, особенно последний, далеко не тривиальны для обычного пользователя и доступ по ним весьма затруднен, все равно, их прохождение реально, а значит любой, достаточно подготовленный человек, может получить доступ к вашей локальной сети. Вы действительно этого хотите? Нет? Тогда защищайте себя и свой сервер.

Да, не существует абсолютной защиты, но можно уменьшить риск взлома сети и ощутимо повысить уровень безопасности, если использовать некоторые программные средства. О них и пойдет речь в следующих главах.

Прокси-сервер

Чаще всего для организации успешной атаки на ваш сервер злоумышленнику, которого часто называют хакером, необходимо знать точный IP-адрес. Этот адрес состоит из четырех чисел. Прокси-сервер скрывает подобный адрес или адреса сети, которую он охраняет. Он скрывает специфику сети, ее архитектуру. То есть прокси-сервер позволяет осуществлять доступ к тем сетям, которые действуют на основе TCP/IP (а весь Интернет и является одной большой сетью, работающей на этом протоколе), не позволяя при этом узнать адрес рабочей станции, с которой осуществляется доступ. Но прокси-серверы не ставят заслон различным разрушительным элементам, вирусам, проникающим вместе с присоединенными файлами по электронной почте. Поэтому необходимо выстраивать целую систему безопасности, которая будет включать в себя и антивирусную защиту.

Прокси-серверы ориентированы на работу по протоколу HTTP. В принципе, многие из них позволяют работать и с другими протоколами, но это осуществляется путем их трансляции в HTTP, то есть прокси-сервер создан специально для нужд Web и одной из его специфических особенностей является наличие хранилища для наиболее часто запрашиваемых документов, которое обычно называют кэшем.

Рассмотрим процедуру работы сервера в том случае, если из защищенной сети направляется запрос на получение какого-либо ресурса.

Сначала пользователь делает запрос на загрузку какой-либо Web-страницы по ее URL. Точнее, пользователь задает URL, а его программное обеспечение, то есть браузер, передает запрос на получение информации. Прокси-сервер принимает запрос и анализирует этот URL. В том случае, если документ, ассоциированный с этим URL, есть в кэше, то он отсылается удаленному пользователю и на этом обработка полученного запроса заканчивается. Если же в хранилище этого документа нет, то сервер определяет IP-адрес, соответствующий URL, переданному в запросе и посылает запрос Web-серверу, используя полученный IP-адрес и протокол HTTP. При этом реальный IP-адрес машины пользователя, который запросил получение Web-страницы, заменяется на IP-адрес прокси-сервера. Удаленный Web-сервер принимает запрос от прокси-сервера, формирует ответ и, используя протокол HTTP, отсылает его обратно. Прокси-сервер в свою очередь отсылает полученную информацию пользователю, который ее запрашивал. Таким образом, становится видно, что прокси-сервер является дополнительным звеном в цепочке запроса информации и он подставляет свой IP-адрес вместо такого же адреса пользователя.

Именно на такой технологии построены работающие в Интернете специальные сайты, скрывающие IP-адрес удаленного пользователя. Они позволяют любому пользователю сначала зайти на свой ресурс, а потом уже с него перейти по требуемому адресу в Интернете. При этом в файлах регистрации посещенных сайтов (лог-файлах) отмечается именно IP-адрес сайта-посредника.

Теперь мы рассмотрим различные варианты кэширования. При выборе конкретного прокси-сервера стоит позаботиться о том, чтобы он поддерживал все виды кэширования, которые будут описаны ниже. Даже если какой-либо вид покажется вам ненужным, все равно, стоит хотя бы иметь возможность воспользоваться им. Никто не знает, что произойдет в будущем.

Первый вид — *пассивное кэширование*. В этом режиме прокси-сервер, получая запрос на какой-либо ресурс, достает его и сохраняет в своем кэше, одновременно устанавливая время хранения для полученного ресурса. В следующий раз при запросе этого же ресурса прокси-сервер проверяет, не истекло ли время его хранения. Если оно еще не истекло, пользователю посылается сохраненная копия. В противном случае, при запросе прокси-сервер достает из сети свежий вариант запрошенного ресурса, отправляет его клиенту и сохраняет копию у себя.

Во время работы прокси-сервера в режиме *активного кэширования* он сам решает, как проводить обновление кэша. При этом хороший прокси-сервер понимает, какие именно объекты чаще всего запрашиваются его клиентами и самостоятельно посылает запросы на их получение. Таким образом, весьма ощутимо экономится время получения информации. В этом режиме прокси-сервер учитывает время хранения копий. Объекты с почти истекшим сроком хранения будут обновляться с большей вероятностью. Прокси-сервер учитывает свою загруженность. Если в какой-то момент времени интенсивность клиентских запросов снизится, часть своего рабочего времени сервер отдаст на нужды активного кэширования. Таким образом, можно увидеть, что в режиме активного кэширования прокси-сервер увеличивает вероятность быстрого получения максимально свежих копий наиболее часто запрашиваемых ресурсов.

И, наконец, третий вариант кэширования — *негативное кэширование*. При этом варианте в кэше накапливаются копии объектов, доступ к которым невозможен, то есть нерабочие ссылки, закрытые ресурсы, и т. д.

На рынке программного обеспечения можно найти прокси-сервер практически с любыми возможностями. Спектр предлагаемых решений достаточно велик — от малых и быстрых серверов с минимальными функциями до громадных и дорогостоящих комплексов, которые могут действительно очень много. Ваша задача — найти правильный баланс.

Одно из наиболее приемлемых решений — использование прокси-сервера WinProxy. Версия 2.1g позволяет отслеживать все соединения, предоставляет один IP-адрес для нескольких пользователей, поддерживает протоколы HTTP, FTP, Telnet, NNTP, SMTP, POP3, позволяет удаленное администрирование. Это хороший выбор.

В следующих разделах мы рассмотрим его настройку.

Настройка прокси-сервера

После достаточно быстрой инсталляции прокси-сервера создается новая группа в главном меню с именем WinProxy. В ней и находится ярлык для запуска самого прокси-сервера. После первого запуска прокси-сервера необходимо произвести его настройку. Для этого используется команда меню **File/Properties Wizard**, которая запускает мастера, позволяющего достаточно быстро установить главные свойства сервера.

Первый шаг этого мастера — указание используемых Интернет-протоколов, которые устанавливаются в диалоговом окне (рис. 3.3). Перед теми протоколами, которые будут использоваться в повседневной работе, необходимо выставить флажки. Прежде всего, нужно отметить базовый протокол для передачи Web-данных — HTTP. Следующий пункт — FTP. Если для членов защищаемой сети планируется доступ к файловым архивам под FTP, то и эту возможность следует установить.

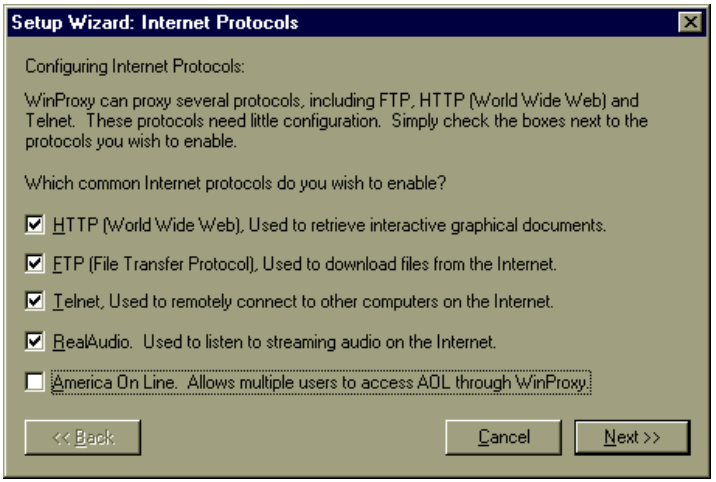

Рис. 3.3. Диалоговое окно **Setup Wizard: Internet Protocols**

Протокол Telnet, как мы уже говорили, предназначен для доступа к конкретным компьютерам по сети Интернет. Если эта услуга не требуется — отключите ее.

Протокол RealAudio применяется для прослушивания звукового потока в реальном времени. То есть, попросту говоря, при помощи этого протокола осуществляется подключение к Интернет-радиостанциям.

Последняя строка в этом окне позволяет регулировать подключение к сервису организации America On-Line. В США это очень популярный сервис, но не думаю, что в данный момент в условиях российской реальности он так уж необходим. Скорее всего, он никогда не понадобится членам защищаемой сети. Тем более, что America On-Line в последнее время отказывается обслуживать удаленных пользователей из России.

Напомню, что несмотря на все это изобилие протоколов, прокси-сервер работает только с одним из них — HTTP. Все остальные транслируются в его команды.

Следующий шаг — установка номера порта, на котором будет действовать прокси-сервер. Для этого также используется свое диалоговое окно (рис. 3.4).

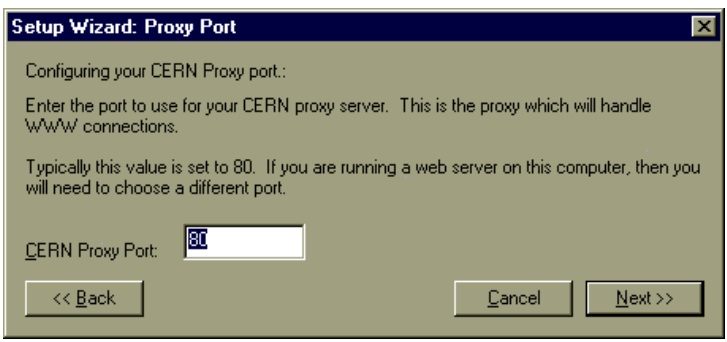

Рис. 3.4. Диалоговое окно **Setup Wizard: Proxy Port**

В поле **CERN Proxy Port** необходимо ввести номер этого порта. По умолчанию это значение равно восьмидесяти и таким его и следует оставить, если только на вашей машине не функционирует какой-нибудь Web-сервер. Дело в том, что точно такое же значение зарезервировано для порта, на котором функционирует Web-сервер. Поэтому, в том случае, когда один компьютер используется и для размещения прокси-сервера и для функционирования Web-сервера, необходимо выбрать другое значение.

Диалоговое окно для следующего шага (рис. 3.5) — настройки прокси-сервера для работы с сервером новостей. В поле **Internet News Server** нужно внести адрес сервера новостей, к которому подключена данная система. В том случае, если, как в примере, указан не полный IP-адрес, а буквенный, прокси-сервер соединяется с провайдером и производит процедуру вычисления IP-адреса по указанному в поле значению. Если не требуется обработка прокси-сервером соединений с сервером новостей, данное поле следует оставить пустым.

Рис. 3.5. Диалоговое окно **Setup Wizard: Internet News**

После сервера новостей наступает очередь почтовых серверов. Диалоговое окно, применяемое для указания их параметров (рис. 3.6), позволяет устанавливать два разных сервера для отправки и получения электронной почты. Адрес сервера входящей почты указывается в поле **Incoming Mail Server**, а адрес сервера отсылаемой почты — в поле **Outgoing Mail Server**. Разделение серверов на входящий и исходящий оправданно, так как отправка и прием электронной почты осуществляются посредством разных протоколов. В том случае, если какой-либо из серверов не планируется использовать, соответствующее поле необходимо оставить пустым.

В принципе, сервер WinProxy может поддерживать сразу несколько серверов входящей и исходящей почты, но подобная возможность осуществляется только при помощи так называемой тонкой настройки, которую мы будем рассматривать позже.

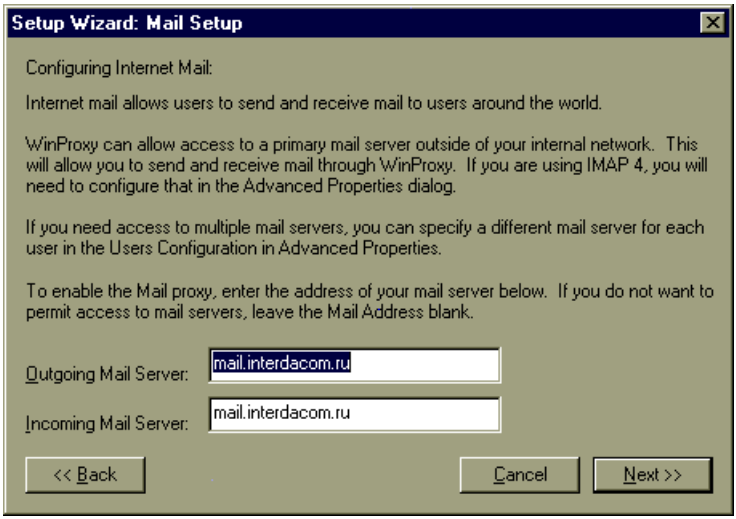

Рис. 3.6. Диалоговое окно **Setup Wizard: Mail Setup**

Как и в предыдущем случае, если вводится символьный адрес серверов, WinProxy предпринимает попытку соединения с провайдером для расчета IP-адресов указанных серверов.

Следующий этап настройки сервера связан с обработкой соединения с сервером распределения имен DNS (Domain Name Server). DNS — это служба распределения символьных имен в сети, использующая протокол TCP/IP. Она устанавливает соответствие между символьными адресами, которые мы часто называем именами и IP-адресами. Таким образом, именно DNS-служба позволяет по имени, набираемому в строке запроса Интернет-браузера, получить правильный числовой IP-адрес. В принципе, эта служба не является обязательной для сети, работающей на протоколе TCP/IP, но практически всегда устанавливается, так как без нее придется производить соединения только указанием полного числового IP-адреса, что, естественно, неудобно. В нашем случае эта служба применяется для обеспечения работы служб SOCKS. Эти службы позволяют использовать протоколы, не поддерживаемые WinProxy напрямую. То есть эта возможность применяется для встраивания ее в приложения. А это уже программирование, которого мы не собираемся касаться. Но все же, если в вашей сети будут создаваться сетевые приложения, использующие другие протоколы, и они должны будут связываться с внешним миром, включите поддержку служб SOCKS4 и SOCKS5. Для этого необходимо выбрать альтернативу **I DO want to enable the Socks 4 and 5 proxy** (рис. 3.7). Также для поддержки этих служб необходимо задать IP-адрес сервера DNS, установленного в вашей сети. В том случае, если используется DNS-сервер, установленный в системе вашего Интернет-провайдера, необходимо получить его IP-адрес. Если эту информацию ваш провайдер не разглашает, попробуйте получить ее при помощи ресурса, находящегося по адресу **http://rs.internic.net/cgi-bin/whois**.

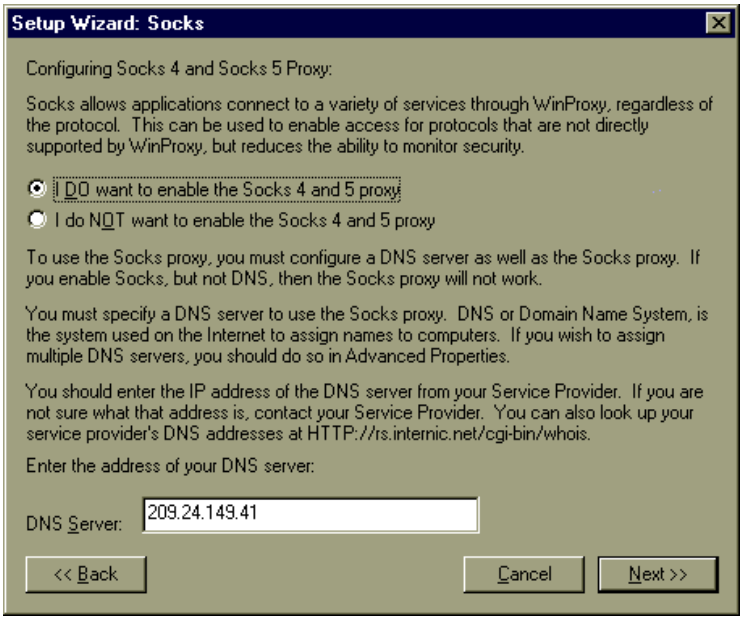

Рис. 3.7. Диалоговое окно **Setup Wizard: Socks**

На следующем шаге необходимо указать порядок доступа в Интернет. В том случае, если для доступа используется стандартный дозвон по обычной телефонной линии, необходимо в диалоговом окне **Wisard Setup: Dial Up Networking** выбрать альтернативу **I AM using Dial Up Networking**. Если доступ осуществляется при помощи более дорогостоящих средств, не требующих дозвона по телефонной линии, например по выделенному каналу, следует выбрать альтернативу **I am NOT using Dial Up Networking**.

В том случае, если выбран вариант с дозвоном, то производится настройка процедуры дозвона с помощью диалогового окна **Wisard Setup: Dial Up Networking** (рис. 3.8). Это необходимо для автоматического соединения прокси-сервера с провайдером доступа в Интернет. Для этого в выпадающем списке **Number to dial** необходимо указать одно из телефонных соединений, которые прописаны в системе. Естественно, стоит выбрать именно то, которое выводит на провайдера. При этом для автоматической авторизации нужно ввести имя пользователя (логин) и пароль. Для этого используются поля ввода **User Name** и **Password** соответственно.

Так как у нас в России пока что очень редки схемы фиксированной оплаты с неограниченным доступом (flat rate), необходимо автоматически отключаться от провайдера, если нет нужды поддерживать соединение. Для этого в поле **Connection Time-Out** указывается время тайм-аута в минутах. Если в течение указанного промежутка времени через прокси-сревер не пройдет ни одного запроса в ту или иную сторону, соединение с провайдером будет оборвано. Если планируется соединение поддерживать постоянно, вне зависимости от его востребо-

ванности, необходимо в этом поле указать нулевое значение. В том случае, если нужно принудительно разрывать соединение при завершении работы прокси-сервера, надо поставить флажок в поле **Terminate connection on exit**.

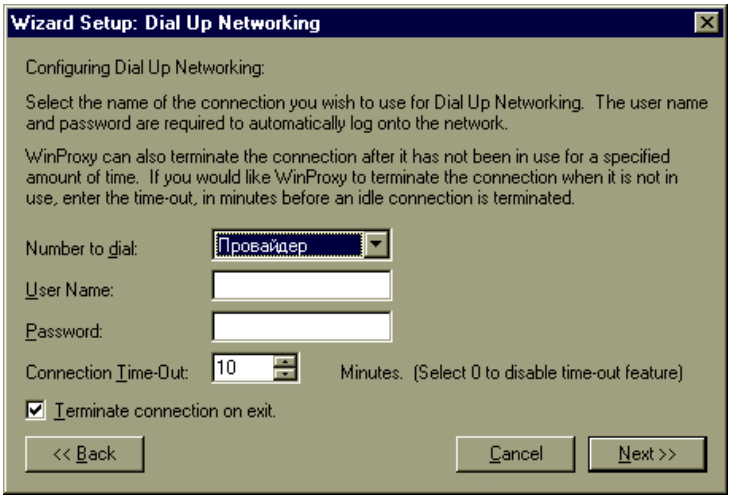

Рис. 3.8. Диалоговое окно **Wizard Setup: Dial Up Networking**

Существует еще один вариант использования прокси-сервера — каскадный. В этом случае один сервер подключается к другому, а уже из второго идет двусторонний трафик. Этот вариант еще сильнее увеличивает степень безопасности, но и отдает некоей "параноидальностью". Тем не менее, если есть желание создать цепочку прокси-серверов, необходимо включить возможность *каскадного соединения*. Для этого используется диалоговое окно **Wisard Setup: Cascading** (рис. 3.9). Сначала необходимо поставить флажок **Enable Cascading**, а потом установить параметры подключаемого прокси-сервера.

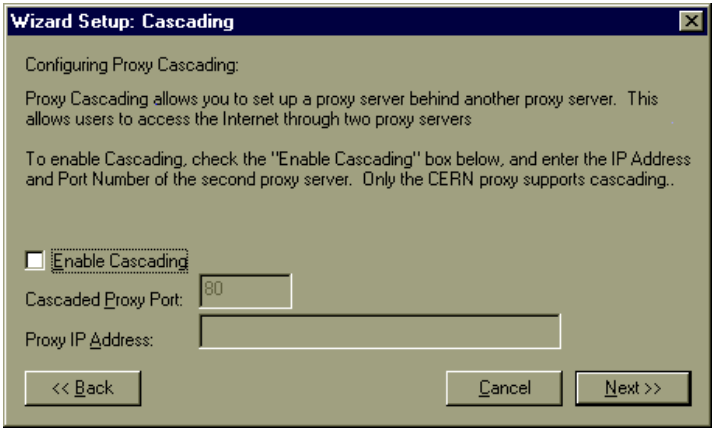

Рис. 3.9. Диалоговое окно **Wizard Setup: Cascading**

Номер порта, на котором он функционирует, указывается в поле **Cascaded Proxy Port**, а его IP-адрес (числовой!) в поле **Proxy IP Address**. Но в качестве каскадного сервера можно указывать только тот, который так же, как и WinProxy, поддерживает протокол HTTP в вариации CERN. CERN (Conseil Europeen pour la Recherche Nucleair), то есть в разработке Европейской лаборатории физики элементарных частиц, в которой и создан первый прокси-сервер. Таким образом, все прокси-серверы, поддерживающие именно этот стандарт, иногда называют еще CERN-совместимыми.

Следующий шаг настройки прокси-сервера реализует некий дополнительный уровень безопасности. У вас есть возможность задать пароль для удаленного доступа к прокси-серверу, чтобы обычный удаленный пользователь не смог получить доступ к его администрированию. Также в этом диалоговом окне (рис. 3.10) можно включить режим проверки обратного адреса для тех запросов, которые поступают в защищаемую сеть извне. В том случае, если этот адрес не проходит процедуру верификации, запрос, исходящий от него, блокируется. Включение этого режима происходит при выборе радиокнопки **I DO want to use Reverse Name Lookup verification**. Если необходимо принудительно отключить этот режим, выбирается, соответственно, другая радиокнопка.

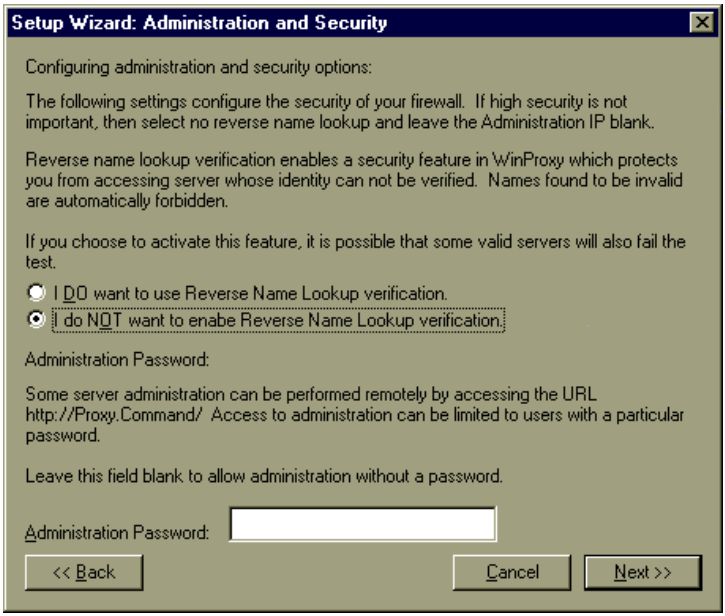

Рис. 3.10. Диалоговое окно **Setup Wizard: Administration and Security**

В поле редактирования **Administration Password** вводится пароль администратора, который будет позволять получать удаленный доступ для изменения настроек и управления сервером.

Весьма настоятельно рекомендуется вести журнал регистрации посещения защищаемой сети. Для этого в сети существуют отдельные службы. Естественно, наш прокси-сервер необходимо настроить на взаимодействие с установленной службой. Сервер, поддерживающий службу ведения журнала регистрации, как и любой сервер в сети имеет свой IP-адрес. Поэтому, для подключения такого сервера к прокси-серверу необходимо в диалоговом окне **Logging** установить флажок **Enable Logging**, указать порт, на котором функционирует сервер, поддерживающий службу ведения журнала регистрации (значение по умолчанию равно 8000), и его IP-адрес.

И последний шаг настройки сервера — это визуализация активных соединений. Она доступна только при определенной производительности системы, так как отнимает некоторые ресурсы. При скорости соединения выше 28,8 Кбит/сек, можно заставить прокси-сервер показывать активные соединения на экране. Для этого необходимо включить радиокнопку **I DO want to view active connections**. После этого можно нажать на кнопку **Finish** и настройка прокси-сервера WinProxy будет завершена.

Тонкая настройка WinProxy

Диалоговое окно для тонкой настройки WinProxy вызывается по команде меню **File/Advanced Properties**. У этого окна есть несколько вкладок, которые определяют группы настроек.

Вкладка **General** практически дублирует процесс первичной настройки, который мы рассмотрели в предыдущей главе. Пожалуй, одним из самых важных инструментов настройки является установка множественных IP-адресов, для которой используется кнопка **Multiple IP Setup**.

Вкладка **Protocols** позволяет конфигурировать протоколы, поддерживаемые прокси-сервером. Помимо того, что мы можем заново установить перечень обслуживаемых прокси-сервером протоколов, мы еще можем установить свойства для каждого из них. Для этого рядом с обозначением каждого протокола находится соответствующая кнопка. Настройка большей части протоколов проста и незатейлива. Достаточно указать номер порта для работы службы, обрабатывающей этот протокол. Но для некоторых из них, например для HTTP, служба настройки достаточно велика и есть возможность даже устанавливать команды этого протокола, которые можно выполнять, а другие запрещать. Таким образом, можно ограничивать входящий или исходящий трафик и накладывать ограничения на действия своих или удаленных пользователей. Однако для правильной конфигурации всех этих служб все-таки необходимо обладать достаточно большим багажом соответствующих знаний, поэтому их конфигурирование следует поручить сетевому администратору. В том случае, если сеть невелика и администратор отсутствует, следует ограничиться условиями, установленными по умолчанию.

Одна из наиболее привлекательных возможностей тонкой настройки — управление отдельными пользователями сети. Для этого используется вклад-

ка **Users**. На ней размещен основной список пользователей и кнопки для редактирования его. В том случае, если получать доступ во внешний мир должны только зарегистрированные пользователи прокси-сервера, а всем остальным в таком доступе будет отказано, следует поставить флажок **Refuse access to all users who are not specifically permitted here**. Если данный флажок не установлен, то незарегистрированные внутренние пользователи будут получать стандартный доступ с общими установками и соответствующими им возможностями.

Для того чтобы завести нового пользователя, необходимо нажать кнопку **New**. При этом будет активизировано диалоговое окно **Edit User** (рис. 3.11). В поле **User/Group Name** вводится имя отдельного пользователя или группы пользователей, которое будет отображаться в основном списке. Для этого элемента необходимо ввести также IP-адрес машины, за которой и будет работать новый пользователь. Этот IP-адрес при помощи кнопки **Add** добавляется в общий список поддерживаемых прокси-сервером IP-адресов. Изъятие какого-либо адреса из этого списка производится при помощи кнопки **Remove**. Всего в списке может находиться и, соответственно, обрабатываться прокси-сервером до пятисот отдельных IP-адресов.

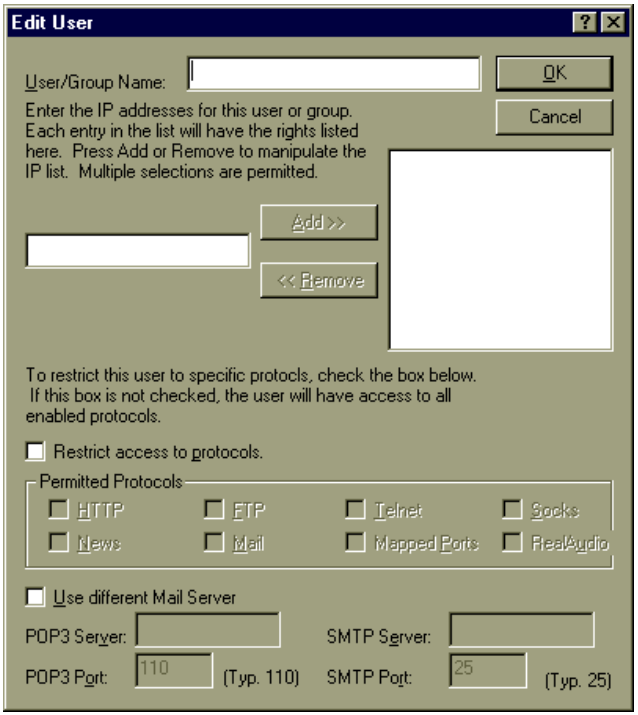

Рис. 3.11. Диалоговое окно **Edit User**

В качестве группы пользователей также может быть добавлена отдельная подсеть. Для ее обработки необходимо ввести ее адресную маску. То есть,

если в подключаемой подсети IP-адреса имеют общий вид, скажем, 100.0.0.X, где вместо X подставляется завершающая цифра, то в общий список вносится адрес 100.0.0.*.

Для каждого отдельного пользователя или подключаемой подсети можно указать список поддерживаемых протоколов и отдельный почтовый сервер, отличающийся от установленного по умолчанию. Для того чтобы настроить список протоколов, доступных для данного пользователя, необходимо установить флажок **Restrict access to protocols**. После этого становится доступной группа флажков, ответственных за выбор протокола. Для поддержки конкретного протокола необходимо выставить соответствующий флажок.

В том случае, если среди протоколов, к которым получает доступ новый пользователь, находится почтовый протокол (переключатель **Mail**), появляется возможность установить для него новый почтовый сервер. Для установки нового почтового сервера, который будет обслуживать этого пользователя или данную подсеть, необходимо установить флажок **Use different Mail Server**. При этом активизируются соответствующие элементы управления, предназначенные для установки значений свойств этого сервера. В поле **POP3 Server** необходимо ввести числовой IP-адрес сервера получения почты, а в поле **SMTP Server** — числовой IP-адрес отправляющего сервера. В полях **POP3 Port** и **SMTP Port** вводятся номера портов, на которых функционируют эти серверы.

После завершения настроек для конкретного пользователя мы снова возвращаемся к вкладке **Users** диалогового окна **WinProxy Advanced Configuration**, в основном списке которого уже отображается новый пользователь или подсеть. Этот список представляет собой таблицу, в которой указывается имя, под которым зарегистрирован пользователь (**Name**), его порядковый номер (**#Users**), доступ к службам (**Access**), адрес POP3-сервера (**POP3 Server**) и SMTP-сервера (**SMTP Server**).

В том случае, если пользователю или подсети был предоставлен полный доступ ко всем службам и протоколам, то есть флажок **Restrict access to protocols** не был выставлен, в колонке **Access** общего списка, которая указывает на характер предоставленного доступа к службам, отображается значение **Full**, характеризующее доступ без ограничений. Если же были установлены ограничения на доступ к каким-либо протоколам и службам, в этой колонке отображается значение **Restricted**. Если для данного пользователя был установлен отдельный почтовый сервер, в колонках **POP3 Server** и **SMTP Server** указываются IP-адреса принимающего и отправляющего серверов.

Если возникает необходимость изменить настройки какого-либо пользователя или подключенной подсети, необходимо в общем списке установить курсор на имени этого пользователя и нажать кнопку **Edit**. При этом активизируется все то же диалоговое окно **Edit User**. Для удаления пользователя из общего списка и, соответственно, отключения его доступа от внешний сети, используется кнопка **Delete**.

Следующая вкладка **Cache** (рис. 3.12) применяется для редактирования настроек прокси-сервера, отвечающих за кэширование информации. Единственная группа радиокнопок, находящаяся на этой вкладке, устанавливает общие сроки хранения копий документов. Если нам необходимо при запросе всегда получать свежую копию документа, нужно отметить радиокнопку **Each time the file is requested**. При этом, естественно, само кэширование практически отключается, так как его смысл именно в сохранении копий запрашиваемых ресурсов и предоставлении их без обращения к их источнику. Если же планируется все-таки хранить копии (что рекомендуется), стоит указать альтернативу **When the file is older then**. При этом в соответствующем поле ввода необходимо указать срок жизни копий в часах. Максимальное значение — 168 часов, что составляет ровно одну неделю. Значение, установленное по умолчанию, — 24 часа.

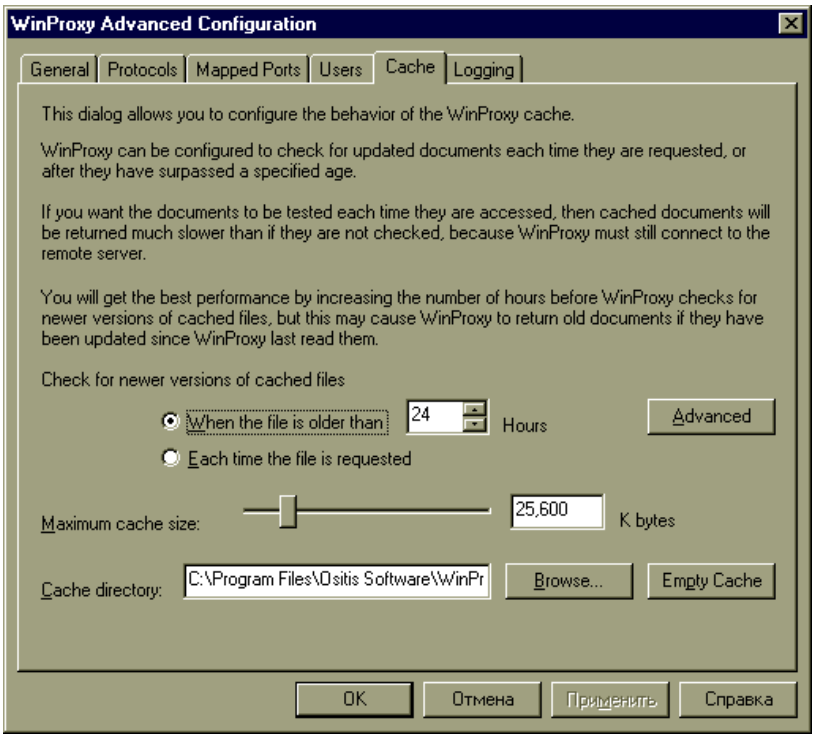

Рис. 3.12. Вкладка **Cache** диалогового окна **WinProxy Advanced Configuration**

Получаемые документы всегда можно отнести к различным категориям, и, соответственно, правила кэширования различных категорий могут различаться. Для установки правил кэширования каждого отдельно взятого ресурса необходимо нажать кнопку **Advanced**. При этом активизируется диалоговое окно **Advanced Cache Properties** (рис. 3.13). Основную часть этого окна занимает список доменов, каждому из которых может быть приписано

свое отдельное правило кэширования. Для добавления нового доменного имени в этот список применяется кнопка **Add**, которая визуализирует диалоговое окно **Edit Cache Exception** (рис. 3.14). В строке **Enter domain names** вводится доменное имя того сайта, к которому будет применяться то или иное правило кэширования. Одно и то же правило кэширования информации может применяться и к нескольким сайтам сразу. В этом случае в строке редактирования вводятся их полные доменные имена, разделенные запятой.

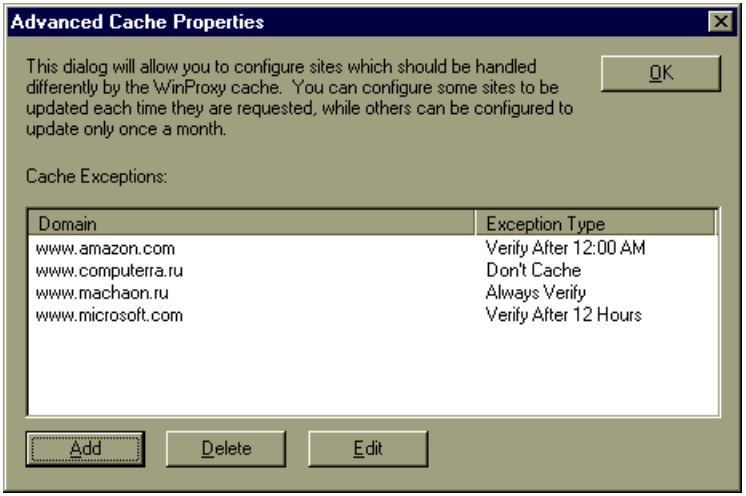

Рис. 3.13. Диалоговое окно **Advanced Cache Properties**

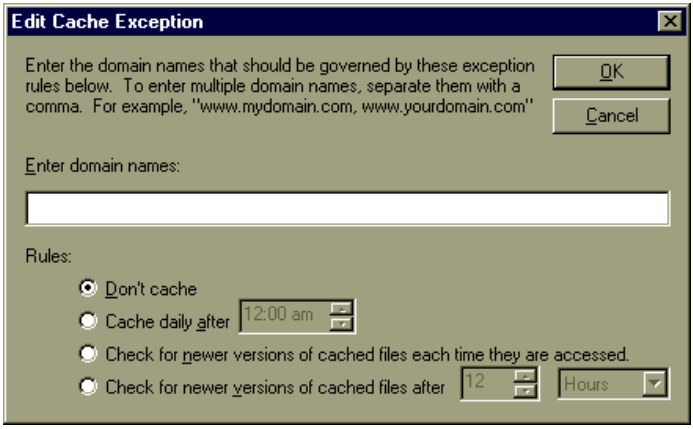

Рис. 3.14. Диалоговое окно **Edit Cache Exception**

После ввода имени сайта необходимо указать одно из четырех правил кэширования. Указание конкретного правила осуществляется выбором одной из четырех радиокнопок. Альтернатива **Don't cache** применяется в том случае, если документы данного сайта не подлежат кэшированию. В этом слу-

чае, в общем списке окна **Advanced Cache Properties**, в колонке **Exception Type** напротив имени этого сайта указывается все то же значение **Don't Cache**. Альтернатива **Cache daily after** заставляет прокси-сервер ежедневно получать копию этого документа. Время получения новой копии указывается в соответствующем поле ввода. Время указывается в часах, с шагом в пятнадцать минут. При этом в основном списке появляется значение **Verify After** в сочетании с временем загрузки новой копии.

Следующий вариант кэширования предполагает проверку наличия новой версии документа каждый раз, когда данный документ запрашивается пользователем. В том случае, если на самом сайте находится обновленная версия документа, скачивается именно она. Иначе пользователю подставляется копия из кэша. Активизирует этот вариант кэширования радиокнопка **Check for newer versions of cached files each time they are accessed**. В основном списке этот вариант кэширования обозначается как **Always Verify**.

И последний вариант кэширования, который предполагает проверку наличия новой версии документа спустя фиксированное время после последнего запроса этого документа одним из пользователей. Для установки этого варианта используется альтернатива **Check for newer versions of cached files after**, а в соответствующем поле ввода указывается время, по истечении которого будет запрошена новая копия. При этом в основном списке этот вариант обозначается как **Verify After** с указанием времени жизни копии. Время жизни копии в этом случае может составлять от одной минуты до семи дней.

Также на вкладке **Cache** можно указать размер дискового пространства, отводимого под кэш. Для этого используется бегунок **Maximum cache size**. При этом, естественно, занимается не все указанное пространство, а лишь столько, сколько необходимо. Максимальный размер кэша может достигать одного Мбайта.

В поле ввода **Cache directory** указывается папка, в которой будут храниться копии документов. Если нет желания вручную вводить путь этого каталога, можно воспользоваться кнопкой **Browse**. Расположенная рядом с ней кнопка **Empty Cache** принудительно очищает содержимое кэша.

Для того чтобы получать детальную информацию обо всех произведенных соединениях, необходимо задать параметры процедуры регистрации запросов. Для этого используется вкладка **Logging**. WinProxy поддерживает два независимых варианта этой процедуры. Первый из них — так называемая активная регистрация. Для включения этого режима необходимо поставить флажок **Enable activity logging**. При этом активизируются два поля ввода. В поле **Logging IP Address** указывается IP-адрес лог-сервера, то есть сервера, осуществляющего регистрацию, а в поле **Logging Port** — номер порта, на котором он функционирует. Обычно номер порта совпадает с установленным по умолчанию значением 8000. В этом режиме каждый зарегистрированный пользователь прокси-сервера может обратиться к лог-серверу по его IP-адресу, и получить полную статистику соединений.

Второй вариант — детальный режим отслеживания запросов с сохранением результатов в специальном лог-файле. В этом случае отмечается переключатель **Enable detailed logging to fail**. После этого необходимо в поле **Log file directory** указать папку, в которой будет расположен такой файл.

Еще одна заманчивая возможность, которую дает прокси-сервер WinProxy, — это так называемый "черный список" (blacklist). В него заносятся те сайты, вход на которые будет запрещен для всех пользователей, которых подключает данный прокси-сервер. К сожалению, в обычном диалоговом режиме нельзя задать список этих сайтов и ресурсов. Поэтому, если действительно необходимо установить черный список, нужно осуществить следующую последовательность действий. В рабочем каталоге WinProxy необходимо создать пустой текстовый файл с именем blacklist.pxy. Есть два варианта создания файлов черного списка. В первом варианте достаточно в каждой отдельной строке записать обычное доменное имя, причем его уровень не имеет значения. Это может быть полноценное имя www.playboy.com, либо любой домен первого уровня, например, ru. В первом случае перекрывается доступ пользователей к сайту Playboy (а что им там делать в рабочее время?), во втором случае полностью уходит из зоны доступа весь российский сектор Интернета. Помимо доменных имен в этих строках могут указываться и цифровые IP-адреса. Необходимо отметить, что в одной строке может быть прописан только один адрес, вне зависимости от того, в какой форме он указывается.

Второй вариант оформления файла черного списка позволяет напрямую указывать не только запрещенные адреса, но и те адреса, доступ к которым может быть осуществлен. В каждой строке после адреса через пробел пишется либо Denied, либо Allowed. Первое значение указывает на то, что данный адрес запрещен к посещению, второй вариант разрешает доступ к нему.

Настройка браузера

После того, как прокси-сервер будет установлен и настроен, необходимо настроить все браузеры на тех машинах, которые подключены к нему. Рассмотрим эту процедуру подробнее. В том случае, если используется русскоязычный Internet Explorer, необходимо выполнить команду меню **Сервис/Свойства обозревателя**. В появившемся диалоговом окне **Свойства обозревателя** нужно выбрать вкладку **Подключение** и на ней нажать кнопку **Настройка сети**. При этом будет активизировано диалоговое окно **Настройка локальной сети** (рис. 3.15). В группе элементов управления **Прокси-сервер** необходимо выставить флажок **Использовать прокси-сервер**. После этого становятся доступными все остальные элементы этой группы. В поле ввода **Адрес** вводится IP-адрес машины, на которой установлен прокси-сервер, а в поле **Порт**, соответственно, номер порта, на котором он функционирует.

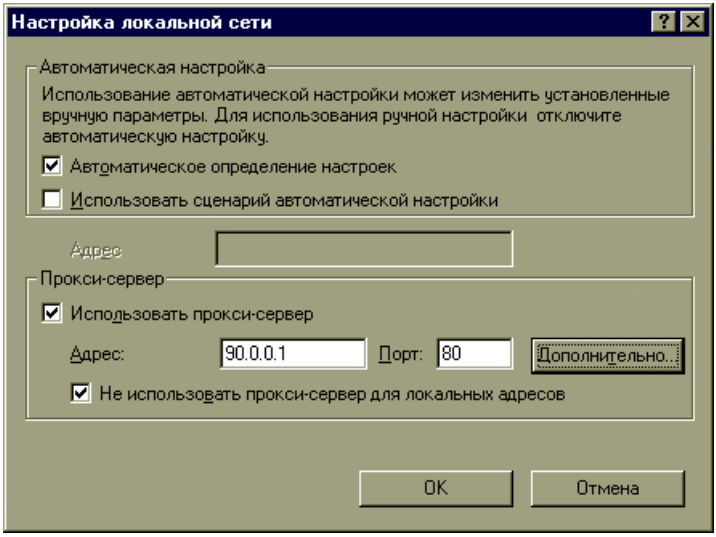

Рис. 3.15. Диалоговое окно **Настройка локальной сети**

Если доступ к машинам внутри защищаемой сети тоже осуществляется по их IP-адресам или соответствующим доменным именам, то по умолчанию эти запросы все равно будут проходить через прокси-сервер. Для того чтобы избежать этой ненужной нагрузки, можно поставить флажок **Не использовать прокси-сервер для локальных адресов**. Для более тонкой настройки браузера используется кнопка **Дополнительно**. При этом визуализируется диалоговое окно **Параметры прокси-сервера** (рис. 3.16). В нем необходимо указать свойства для всех протоколов. Здесь наблюдается некоторое несоответствие терминологии. Мы можем установить отдельный адрес для нескольких прокси-серверов, каждый из которых будет обслуживать один из протоколов. Если же поставить флажок **Один прокси-сервер для всех протоколов**, то будут совпадать и адреса и порты функционирования. Мы же для каждого протокола установили свой порт. Поэтому нужно снять этот флажок, установить один IP-адрес для всех серверов и прописать необходимые номера портов.

Также в этом окне мы можем задать ряд адресов, доступ к которым будет осуществляться без посредничества прокси-сервера. Эти адреса вносятся в текстовое поле, которое находится в группе **Исключения**. Естественно, это должны быть адреса, которым вы полностью доверяете.

Процедура настройки браузера **Netscape Navigator 4.7** не слишком сильно отличается от вышеописанной. Сначала нужно выполнить команду меню **Edit/Preferences**, которая вызовет окно настройки. В левой его части находится древовидная структура, включающая в себя все категории настроек. Нас интересует узел **Advanced**. Необходимо развернуть его щелчком мыши по значку плюса и в появившемся списке второго уровня выделить строку **Proxies**. При этом в правой части диалогового окна появятся элементы

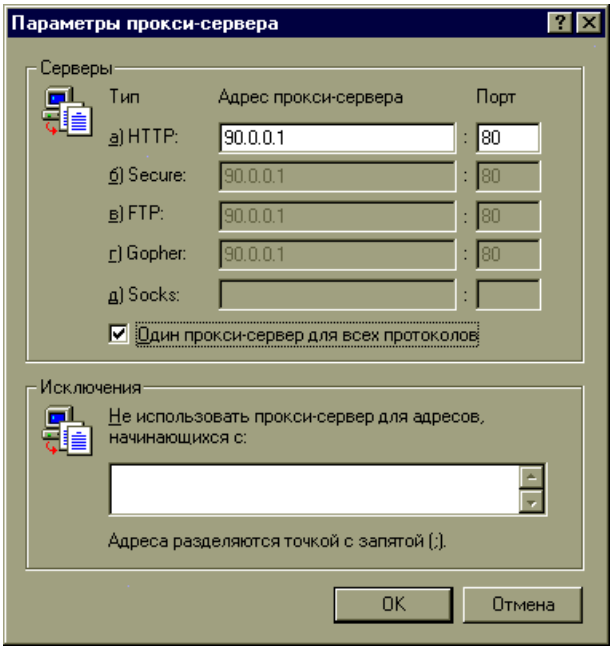

Рис. 3.16. Диалоговое окно **Параметры прокси-сервера**

управления, предназначенные для установки свойств прокси-сервера и все окно примет вид, показанный на рис. 3.17. Основная настройка происходит при помощи группы радиокнопок. Если не планируется использовать прокси-сервер, то следует выбрать радиокнопку **Direct connection to the Internet**. Директива **Automatic proxy configuration** позволяет браузеру произвести автоматическую настройку на прокси-сервер. Для этого необходимо в поле **Configuration location (URL)** указать местонахождение файла конфигурации сервера. Если же вы не уверены в том, что Netscape Navigator совместим с вашим прокси-сервером, стоит выбрать радиокнопку **Manual proxy configuration** и произвести ручную настройку. При выборе этой радиокнопки становится доступна кнопка **View**, при нажатии на которую и появляется диалоговое окно **Manual Proxy Configuration** (рис. 3.18). Легко заметить, что оно практически полностью повторяет свой аналог из Internet Explorer. Соответственно, и работа с ним будет производиться точно таким же образом.

Еще раз повторюсь, что подобные операции настройки нужно провести на каждой машине, которая будет работать под защитой прокси-сервера.

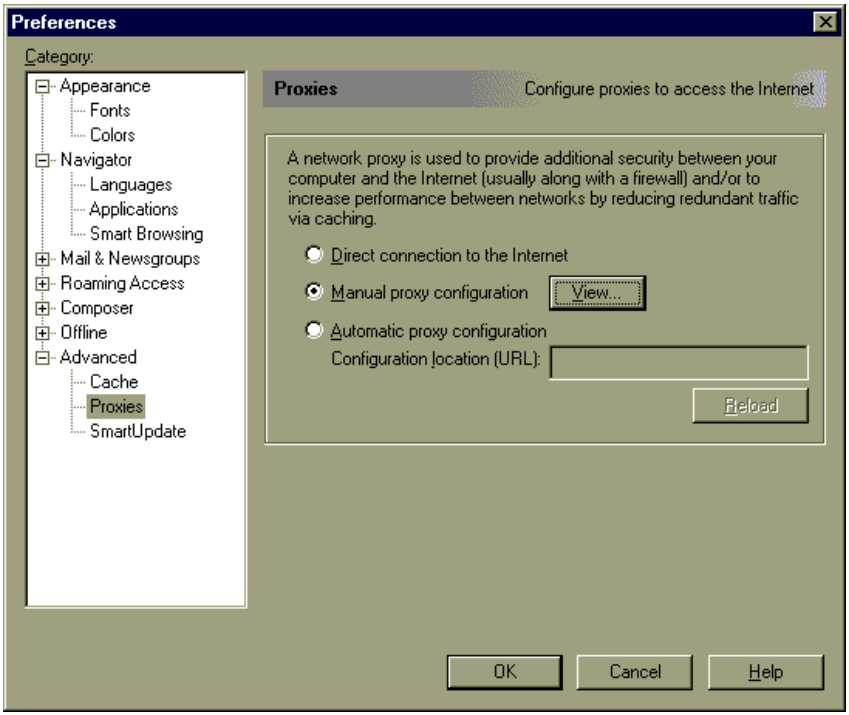

Рис. 3.17. Диалоговое окно **Preferences**

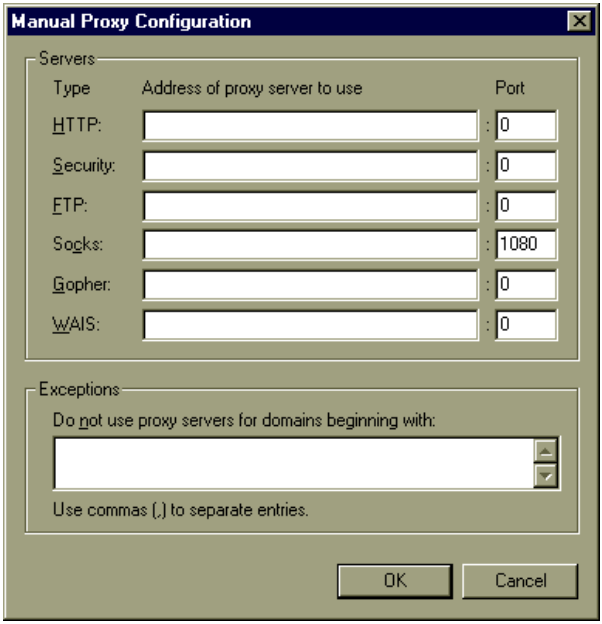

Рис. 3.18. Диалоговое окно **Manual Proxy Configuration**

Огненные стены

Еще одним вариантом защиты вашей сети может быть установка межсетевого экрана (firewall, брандмауэр). Обычно под этими терминами понимают специализированное программное обеспечение, установленное на узловой машине, которая и устанавливает доступ к внешнему миру. Если рассмотренный нами ранее прокси-сервер подменял настоящие адреса внутренних пользователей при доступе в Интернет своим адресом, подставляя себя под возможный удар, то брандмауэры, наоборот, призваны блокировать атаки из внешнего мира. Если вам дорога информация, находящаяся в корпоративной сети, настоятельно рекомендую потратиться на приобретение и установку какого-либо межсетевого экрана. Очень часто хорошие межсетевые экраны берут на себя и функции прокси-серверов, что позволит облегчить конфигурацию и улучшить управляемость системы защиты.

Любой брандмауэр должен не слишком сильно затруднять пользователям сети доступ в Интернет и, в то же время, предотвращать несанкционированный доступ удаленных пользователей к внутренним ресурсам. Он должен запрещать прямой доступ из Интернета к внутренним IP-адресам сети. Эта функция является зеркальной, или дополняющей по отношению к основной функции прокси-сервера. Межсетевой экран позволяет устанавливать правила, по которым будет разрешаться или ограничиваться внешний доступ к защищаемой сети.

Жизненно необходимо, чтобы поставляемая версия межсетевого экрана могла обеспечивать анализ работы сети, формируя журналы подключений (лог-журналы). Здесь нужно отметить, что если служба фиксации запросов прокси-сервера отслеживает соединения, направленные изнутри во внешний мир, то такая же служба брандмауэра направлена на фиксацию подключений извне.

Как минимум, ваш межсетевой экран должен поддерживать протоколы HTTP, FTP и службу DNS. Но, естественно, перечень протоколов должен быть больше, лишние всегда можно отключить. А вот если вдруг понадобится установить поддержку какого-либо протокола, который установленный брандмауэр не поддерживает в принципе, придется приобретать и устанавливать другой экран, что потребует определенных финансовых и административных резервов. Всегда проектируйте свою сеть на вырост, так как запросы ваших пользователей постоянно будут расти.

При выборе межсетевого экрана всегда обращайте внимание на то, есть ли у него графический интерфейс. Многие экраны были созданы достаточно давно и могут до сих пор работать на основе командной строки. Помимо того, что в нынешнюю эпоху графических операционных систем это просто непривычно и неудобно, этот факт может говорить о том, что данная версия просто стара, а, значит, некоторые способы защиты от новых атак в эту версию могут не входить.

В том случае, если в сети функционирует достаточно мощный Web-сервер с закрытыми разделами, доступ к которым осуществляется по зарегистрированному имени и/или паролю, необходимо позаботиться о том, чтобы список зарегистрированных удаленных пользователей Web-сервера совпадал с подобным списком брандмауэра. Соответственно, хороший межсетевой экран должен иметь возможность запрещать или допускать соединения, основываясь на имени пользователя в совокупности с его паролем. Возможны также более серьезные схемы идентификации и аутентификации удаленных пользователей, так как постоянное имя и/или пароль, в конце концов, могут быть перехвачены злоумышленником или узнаны иным способом.

Некоторые последние версии межсетевых экранов могут также осуществлять фильтрацию входящего контента. В этом аспекте экраны практически срослись с системами "фильтрованного доступа" в Интернет. Многие производители программного обеспечения наряду со специализированными программами, которые контролируют доступ локального пользователя в Интернет, запрещая доступ к некоторым ресурсам, создают свои базы данных адресов Web-серверов, содержащих информацию сомнительного свойства. Если вам не хочется выискивать все порно-сайты, а потом долго и нудно вносить их адреса в черный список своего сервера, получите такую базу данных и присоедините ее к своему брандмауэру. А в последующем регулярно ее обновляйте.

И, главное, найдите хорошего и квалифицированного администратора безопасности вашей сети, найдите возможность постоянной связи с ним, даже если он у вас находится в штате работников предприятия. Консультация может понадобиться в любой момент. Все программные ухищрения бесполезны, если нет хорошего, квалифицированного специалиста. Не будет преувеличением заявление, что концепция безопасности сети в качестве одной из главных целей рассматривает поиск подходящего администратора.

Чрезвычайно важно иметь хороший доступ к службе консультаций и технической поддержки фирмы, которая создавала тот межсетевой экран, который вы решили использовать. Сейчас уже и в России есть производители межсетевых экранов. Информацию об этом можно получить на сайте **www.alpha.ru**.

Глава 4

Проектирование виртуальных магазинов

Торговля в Интернете

Прежде чем мы попробуем подступиться к этой проблеме с технических позиций, стоит разобраться с идеологией Интернет-торговли. Весь бизнес целиком и полностью держится именно на продажах. О торговле написано грандиозное количество книг, разработано множество технологий продаж. Применимы ли они к продажам при помощи Интернет? Ответ неутешителен. От готовых наработок используется очень малая часть, так как большинство технологий просто неприемлемы.

Директор центра постоянной подготовки Торгово-промышленной палаты Руана (Франция) Жерар Шандезон и профессор Высшей торговой школы Руана Антуан Лансестр в своей монографии "Методы продажи" приводят следующее определение торговли — "это устный обмен между покупателем и продавцом, в ходе которого продавец делает представление товара с целью заключения сделки". Вы уже видите несоответствие? У нас нет возможности устного обмена информацией. Поэтому на торговый сайт ложатся специфические функции усиленного маркетинга, который должен привлечь покупателей на сайт, рассортировать посетителей оптимальным образом, причем, желательно еще до подхода к основным страницам. Естественно, что этот "усиленный маркетинг" и стоит больше, нежели обычный. Так, по данным агентства Christofer Vroom, Thomas Weitzel Partners, при покупке через Интернет товара на сумму 100 USD затраты на маркетинг составят 17,29 USD против 2,50 USD при покупке на ту же сумму в супермаркете. Хотя, с другой стороны, это данные по США, а в России все может быть немного по-другому.

Следующая проблема — проведение платежей. До августовского кризиса 1998 года можно было воспользоваться стандартными кредитными картами, благо они использовались довольно широкими слоями населения. После кризиса большая часть карточек превратилась в дорогостоящие сувениры.

С тех пор для электронных магазинов в России приходится придумывать другие способы оплаты. Одной из наиболее распространенных стала схема, при которой удаленному покупателю отсылается бланк платежного поручения, тот оплачивает его через банк, а по факту прихода денег ему высылается товар. Увы, подобная схема оплаты превращала online-торговлю в подобие торговли по каталогам, причем пользователь был вынужден платить еще и за возможность просмотра каталога, так как доступ в Интернет у нас пока еще оплачивается на повременной основе.

Сейчас, правда, некоторые банки снова запускают свои карточные проекты, поэтому в свои электронные магазины можно добавлять карточную схему платежа для россиян, но в условиях России пока что нельзя делать основную ставку именно на этот способ оплаты, так как доверие к карточным системам будет восстанавливаться очень медленно. Однако сейчас в России уже снова появились провайдеры электронной коммерции, которые могут проверить платежную карту, предлагаемую к оплате, провести транзакцию и известить продавца.

Так, например, платежная система CyberPlat предлагала очень хороший и юридически выверенный механизм покупок в режиме online без привлечения кредитных карточек, точнее с необязательным их привлечением. Для того чтобы совершать покупки при помощи системы CyberPlat, покупателю достаточно иметь счет в одном из банков, входящих в эту систему.

Сначала покупатель через Интернет заходит на сайт магазина, формирует заказ и направляет магазину запрос на выставление счета. В ответ магазин направляет покупателю счет, заверенный своей электронной цифровой подписью. В счете указывается уникальный номер товара, его стоимость, код магазина, дата и время совершения сделки. Таким образом, этот счет является офертой, то есть предложением заключить договор. Затем покупатель подписывает этот счет своей электронной цифровой подписью и отправляет его обратно в магазин, тем самым подтверждая покупку. Счет, подписанный покупателем, становится чеком для магазина. Сам договор купли-продажи считается заключенным с момента подписания счета покупателем. Подписанный двумя электронными цифровыми подписями чек направляется магазином в банк для авторизации. Банк проверяет коды магазина и покупателя (на наличие таковых в системе), проверяет подписи, остаток средств на счете покупателя или лимиты перерасхода средств и сохраняет копию чека в базе данных банка. На основе результатов проверки формируется разрешение или запрет платежа. При разрешении платежа банк передает магазину разрешение на отгрузку товара и перечисляет на его счет необходимую сумму со счета покупателя, естественно, оставляя некоторый комиссионный процент себе. В случае запрещения платежа банк передает магазину извещение об отказе, а покупателю передает отказ с объяснением причины. Естественно, что при такой схеме и покупатель и магазин достаточно хорошо защищены от возможных эксцессов. В дополнение ко всему и покупатель и магазин могут просмотреть свою историю платежей и проанализировать ее.

Одна из очевидных юридических тонкостей — вопрос силы электронной цифровой подписи. Эта проблема решается при заключении договора об использовании системы CyberPlat. Когда покупатель или продавец заключают подобный договор, они подписывают соглашение о признании действительности электронной цифровой подписи. Соглашение, конечно, делается в традиционной бумажной форме. А в Гражданском кодексе говорится, что при заключении сделки в письменной форме допускается использование аналогов собственноручной подписи, в нашем случае это электронная цифровая подпись, если существует договоренность между сторонами, которая, зафиксирована соглашением.

Еще одна проблема — доставка. В пределах города или мегаполиса еще можно воспользоваться курьерской доставкой. А как быть, если покупатель находится в совершенно другом регионе. Пользоваться почтой? Уточните в ближайшем почтовом отделении, сколько дней будет идти посылка из Москвы, скажем, в Тюмень. А если еще учесть надежность работы нашей системы почтовой связи, то становится ясно, что для того, чтобы удержать реноме магазина на должном уровне, пользоваться услугами почтовой службы России следует с очень большой осторожностью. Можно попробовать воспользоваться системой "EMS Garantpost", но это взвинтит цену товара практически до заоблачной высоты. Именно в силу этих проблем российские Интернет-магазины по большей части переквалифицировались в справочные системы и в системы оформления заказов. Но и эти сложности преодолимы. Вполне возможно разработать проект Интернет-магазина для какого-либо города областного значения с населением более одного миллиона человек, который будет использовать курьерскую доставку. А в силу того, что и клиент и продавец находятся в одном городе, то и проблемы с платежной системой могут быть решены. Есть и решение, находящееся совсем на другом конце спектра возможностей. Сделать магазин, ориентированный на покупателей со всего мира. Тогда за счет общего уровня цен могут быть решены проблемы и с доставкой товара, и с его оплатой, так как в цивилизованном мире различные платежные карты достаточно распространены, и банки охотно работают с ними.

Теперь обратимся к проблемам налогообложения. Если ваш бизнес зарегистрирован на российской территории, то с каждой продажи вы должны будете заплатить налоги. А если ваш покупатель находится за рубежом и оплатил покупку валютой, то придется уплатить налоги и за валютную операцию, причем, сверх уже уплаченных. С другой стороны, если расчетный счет и/или компанию разместить в США, то эти проблемы разом снимаются, так как на данный момент в США не взимаются налоги со сделок, проведенных в Интернете. Хотя к этой информации стоит относиться с осторожностью. Вполне возможно, что в некоторых штатах подобный налог введен, поэтому стоит каждый раз конкретно уточнять налоговое законодательство. В качестве одного из вариантов можно предложить схему, когда в США регистрируется компания типа LLC (Limited Liability Company). Такая компания не облагается налогами как другие. Налоги должны платить

владельцы компании как физические лица. Но в том случае, если владельцы являются иностранными гражданами (в нашем случае, это российское гражданство), то от налогов они освобождаются. Таким образом, мы получаем *легальную* схему безналоговой фирмы.

Итак, мы видим, что при создании Интернет-магазина возникает определенное количество проблем. Но они решаемы. На самом деле, помимо сложностей есть и преимущества. Ведь начальные и текущие затраты достаточно низки. Ваши покупатели живут по всему миру. Даже если ориентироваться только на русскоязычных покупателей, то это уже будет немало. Кстати, представьте себе, что будет, если попытаться продавать последние новинки российского книжного рынка тем, кто проживает в дальнем зарубежье, и попробуйте представить себе объем вашей платежеспособной аудитории и возможную цену книг. Процесс работы почти полностью автоматизирован и, соответственно, оперативный учет и управляемость бизнеса будут явно лучше, чем у большинства серьезных бизнесов. А если учесть еще и возможности легального ухода от налогообложения, то картина получается совсем неплохая.

Итак, если вы уже определились с направленностью Интернет-магазина, внятно представляете себе усредненный портрет своего покупателя и готовы открыть online-торговлю, то необходимо выполнить три шага. Сначала зарегистрировать свою компанию. Затем открыть счет в банке, который работает через Интернет, или выбрать провайдера электронной коммерции. И, наконец, создать сайт и подключить его к торговой системе. Вот о создании такого сайта мы и будем говорить в этой главе.

WebShop 2000. Проба инструмента

Как мы уже говорили, для создания нашего магазинчика мы используем WebShop 2000 от Boomerang Software. Текущая версия продукта — 5.10. Должен сразу предупредить, что этот редактор наиболее массивный из всех, которые мне попадались. Полная версия со всеми мастерами и шаблонами, с библиотекой картинок и тем оформления занимает около 225 (!) Мбайт. Естественно, что код немного тяжеловесен и на машинах, которые оснащены оперативной памятью не очень хорошо, программа работает медленно.

Естественно, что помимо создания систем электронной торговли WebShop 2000 позволяет создавать и обычные сайты. Поэтому мы построим нашу работу следующим образом. Сначала рассмотрим основные возможности программы для построения сайтов. Потом попробуем создать магазин, используя один из прилагаемых шаблонов. А в самом конце узнаем, как делать систему без активного использования встроенных мастеров.

Итак, приступим. При загрузке программы в рабочем пространстве, как обычно, открывается пустой непоименованный сайт, который состоит из одной страницы с названием index и готовой физической структуры, кото-

рая традиционно имеет три папки. В папке CGI-BIN собираются все исполняемые модули, папка **Images** предназначается для хранения графических файлов, используемых при оформлении сайта, а третья папка **NavBars** содержит все необходимое для организации навигационных панелей. Как всегда здесь есть меню, панели инструментов и строка статуса. В качестве основного рабочего поля используются обычно два окна, одно из которых служит для сопровождения всего проекта и носит название **Site Manager**, а второе предназначено для проектирования и отображения содержимого текущей Web-страницы. Естественно, от того, какое окно является активным, зависит содержание меню и панелей инструментов. Верхняя панель инструментов является общей для обоих рабочих окон, а все остальные либо видоизменяются, либо исчезают. Это показано на рис. 4.1.

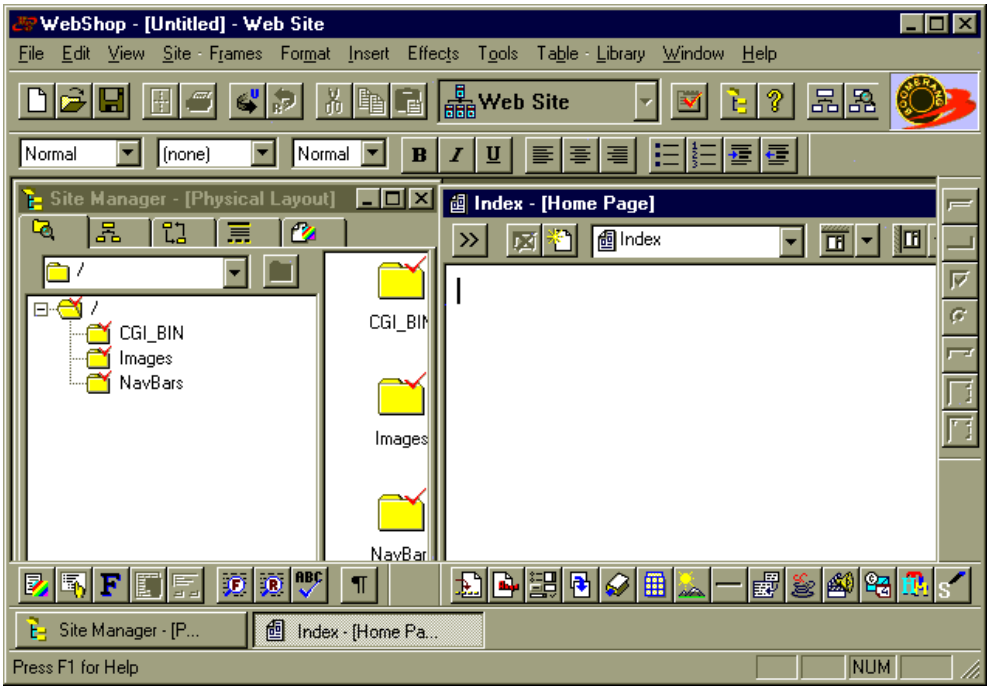

Рис. 4.1. Внешний вид рабочего окна программы WebShop 2000

Схема работы достаточно проста. Сначала в окне **Site Manager** создается необходимая логическая структура, все отдельные страницы, заполняются каталоги **CGI-BIN** и **Images**, а затем оформляется каждая страница. Процесс, естественно, достаточно интерактивный. Придется не раз переходить из окна менеджера проекта к рабочему окну проектирования страницы для внесения изменений, но рано или поздно эта часть работы будет завершена.

Необходимо осознавать, что окно для проектирования страницы не показывает ее в том виде, в котором она будет отображаться браузером, поэтому стоит

время от времени нажимать клавишу <F7> или использовать связанную с этой клавишей команду меню **File/Preview in Browser**, которая запускает встроенный в программу простенький браузер и загружает в него проектируемую страницу. Перед этим программа всегда выдает предупреждение, что если в оформлении вашей страницы использовались какие-либо активные элементы, такие как апплеты, скрипты или конструкции DHTML, то обычный браузер персонального компьютера не всегда будет в состоянии адекватно отображать тестируемую страницу. Поэтому ее стоит просматривать на машине, на которой установлен Web-сервер.

Конечно же, необходимо периодически сохранять работу. Надо отметить, что программа сохраняет не создаваемый сайт, а файл проекта с расширением cnq, который содержит в себе всю схему сайта. А по завершении работы необходимо из файла проекта получить готовый сайт. Для этого используется команда меню **File/Generate Site**, привязанная к клавише <F8>. При активизации этой команды вызывается диалоговое окно (рис. 4.2), в котором необходимо указать папку, предназначенную для размещения сайта. После ее выбора и нажатия на кнопку **OK**, идет процесс генерации сайта, а после его завершения, программа радостно об этом рапортует, выдавая сообщение "The site has been successfully generated". Теперь, если мы просмотрим содержимое той папки, в которую мы помещали сайт, то мы увидим все каталоги, используемые сайтом, HTML-файлы, содержащие все спроектированные Web-странички, и компоненты, реализующие активные элементы страниц. После этого сайт можно выкладывать на сервер и приступать к бета-тестированию.

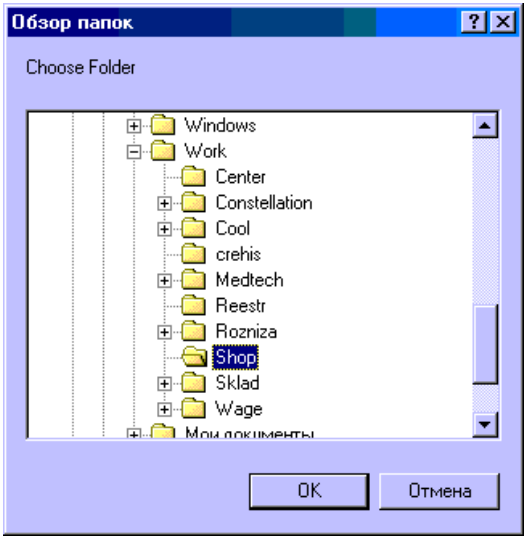

Рис. 4.2. Диалоговое окно **Обзор папок**

Замечание

Все файлы из каталога CGI-BIN должны быть перенесены в папку Web-сервера, для которой установлены права на запуск и исполнение файлов.

Создание Web-страниц

Прежде чем редактировать какую-либо страницу, ее необходимо создать. Для этого в окне **Site Manager** программы WebShop 2000 открывается каталог, в котором она будет находиться, и на инструментальной панели **Physical Layout Toolbar** нажимается одна из кнопок, создающих новые Web-страницы. На выбор предоставляются кнопки **New Frame Set Resource** для создания страниц с фреймовой структурой, **New Page** для организации обычных Web-страниц, кнопка **New Free Page**, создающая страницу со свободной структурой, и **New Dynamic Page** для страниц, содержащих какие-либо активные элементы. Нажатие на одну из этих кнопок приводит к тому, что в окне **Site Manager** на вкладке **Physical Layout** в выбранном каталоге появляется иконка, соответствующая типу созданной страницы. При этом имя HTML-файла для этой страницы находится в режиме изменения, что позволяет тут же установить необходимое наименование файла.

Все инструменты для размещения каких-либо объектов на проектируемой странице находятся на инструментальных панелях, визуализация каждой из которых производится при помощи команды меню **View**. Впрочем, любое действие дублируется соответствующей командой меню. Выбирайте то, что вам нравится. Выключка текста, начертание, имя и размер шрифта указываются при помощи стандартных элементов управления на инструментальной панели **Formatting Toolbar**.

Текст является самым простым объектом и для его вставки достаточно просто щелкнуть мышью на странице, после чего начать ввод.

Одним из вариантов отображения текстовой информации являются списки нумерованные и просто маркированные. Для того чтобы уже написанному тексту придать параметр и вид элемента нумерованного списка, необходимо нажать на кнопку **Numbering** или выбрать команду меню **Insert/Numbering**.

Если нас интересует маркированный список, то, соответственно, надо выбирать команды **Bullets**.

Создание подпункта, то есть повышение уровня вложенности для какого-либо абзаца производится выбором команды меню **Insert/Increase Indent**. Соответственно, для понижения уровня вложенности выполняется команда **Insert/Decrease Indent**. У этих команд также есть дублирующие кнопки.

Для вставки гиперссылок существует команда меню **Insert/Link**, выводящая на экран диалоговое окно (рис. 4.3). Вкладка **To Page** позволяет нам отправить посетителя на одну из страниц сайта. Вкладка **To Resource** позволяет ссылаться на любой ресурс, находящийся в пределах системы каталогов сайта. Вкладка **Relative** предназначена для организации гиперссылок, реализующих навигацию по всему сайту. На ней мы можем установить ссылку на основную страницу сайта, следующую или предыдущую страницы. Полный список посмотрите сами. Ну а вкладка **External** предназначена для реализации самых обычных внешних ссылок. При этом весьма впечатляет набор

префиксов для URL ассоциированного документа. Всего WebShop предоставляет для выбора тринадцать (!) префиксов, каждый из которых, как мы помним, обозначает протокол доступа к ресурсу.

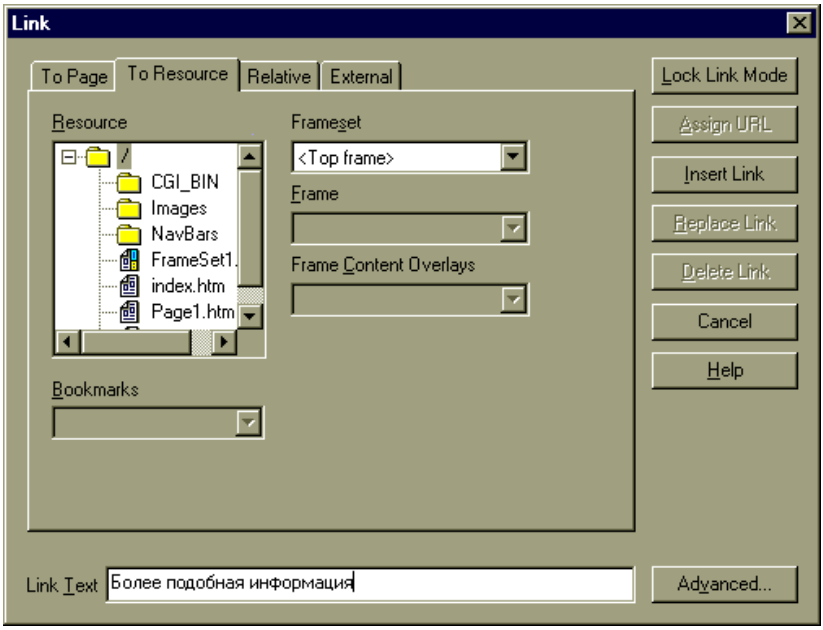

Рис. 4.3. Диалоговое окно **Link**

В том случае, если планируется вставить внутреннюю ссылку, то есть ссылку на конкретное место документа, входящего в состав сайта, как мы помним, в этом месте сначала необходимо установить закладку. Это осуществляется нажатием на кнопку **Bookmark** или выполнением одноименной команды из меню **Insert**. При этом в появившемся диалоговом окне необходимо ввести уникальное имя закладки. Для того, чтобы правильно выбрать ее имя, не повторяя уже существующих, в диалоговом окне есть список имен всех уже созданных в документе закладок.

Вставка графического изображения производится при помощи кнопки **Image** или команды меню **Insert/Image**. При этом активизируется диалоговое окно **Image Attributes** (рис. 4.4). Это одно из самых удобных диалоговых окон для вставки графических ресурсов. Вообще, диалоговые окна WebShop всегда производят вид некой солидности и полной достаточности. Они сразу предоставляют весь заложенный в программе сервис. Основной элемент управления в этом окне — блокнот с тремя вкладками **Image Resources**. В нем мы можем выбрать источник получения графического файла. Страница **Existing Resources** позволяет выбрать графический файл, находящийся в пределах файловой структуры создаваемого сайта. Вкладка **Library** предоставляет доступ к обширной встроенной библиотеке рисунков. При этом по-

является выпадающий список, в котором можно выбрать требуемую категорию рисунка. В соответствии с этим отображаются рисунки в основном окне блокнота. И, наконец, последняя вкладка **URL** используется в том случае, когда требуемый графический файл находится в Интернете и необходимо указать его адрес.

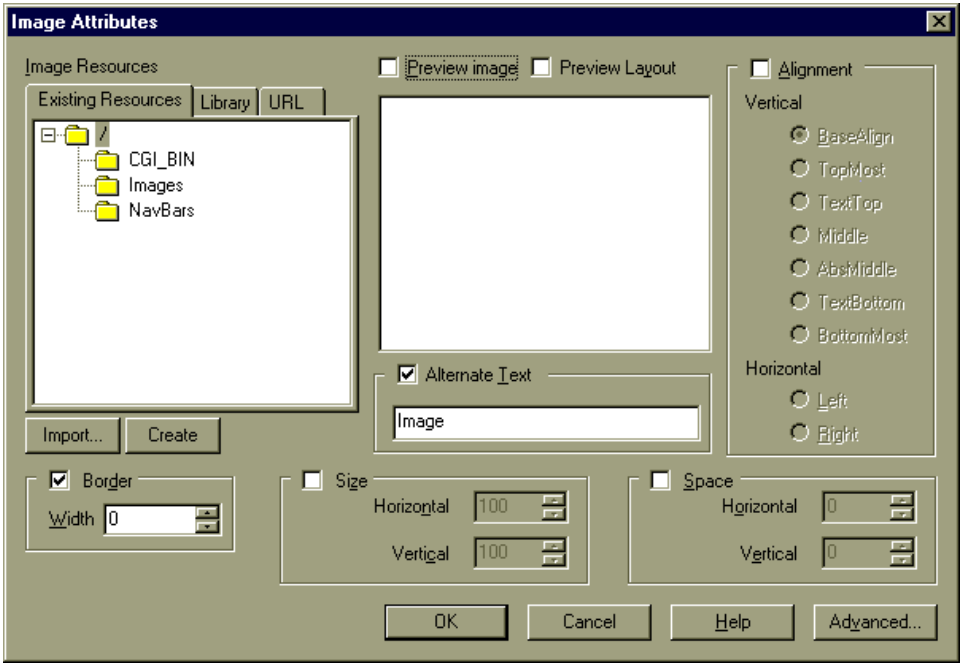

Рис. 4.4. Диалоговое окно **Image Attributes**

Если есть желание несколько подробнее рассмотреть рисунок, не помещая его на страницу, стоит поставить флажок **Preview Image**. При этом включается режим предварительного просмотра рисунков. Установка границы вокруг рисунка производится при помощи независимого переключателя **Border** и связанного с ним поля ввода **Width**, в котором указывается предполагаемая толщина границы рисунка в пикселах. Текст, отображаемый вместо рисунка в том случае, когда в браузере удаленного пользователя отключен режим показа графики, указывается в поле ввода, связанном с флажком **Alternate Text**. Настоятельно рекомендую вам использовать эту возможность. Старайтесь всегда использовать альтернативные средства представления информации, так как даже страницы с фреймовой структурой не всегда могут быть прочитаны.

Выравнивание рисунка относительно текста производится при помощи радиокнопок, находящихся в группе **Alignment**. Так как мы знаем уже, какие виды выравнивания применяются в HTML, нет надобности подробно разбирать каждый вариант. Более того, если поставить флажок **Preview Layout**,

то можно увидеть, как будет размещаться вставляемое изображение относительно основного текста страницы.

Размер рисунка можно выставить принудительно, введя значения в поля группы **Size**, а поля из группы **Space** позволяют устанавливать вертикальный и горизонтальный отступ картинки от обрамляющего ее текста.

Вставка горизонтальной линии производится при помощи кнопки **Horizontal Rule** или одноименной команды меню из блока **Insert**. К сожалению, мы не имеем прямых возможностей установки ее свойств. Даже при выполнении щелчка правой кнопкой мыши на этой линии и выборе из появившегося контекстного меню пункта **Style**, мы получим лишь возможность управлять стилем всей страницы в целом, тем самым задавая единообразное отображение всех одинаковых объектов на странице, в том числе и горизонтальных линий.

В том случае, если текст, предназначенный для размещения на странице, уже существует в виде какого-либо файла, его можно вставить прямо из этого файла, не внося на страницу вручную. Для этого существует кнопка **Insert Text File** или соответствующая команда меню **Insert/Text**. При этом активизируется диалоговое окно **Import Text** (рис. 4.5). Как и ранее рассмотренные диалоговые окна, это окно представляет собой хороший пример

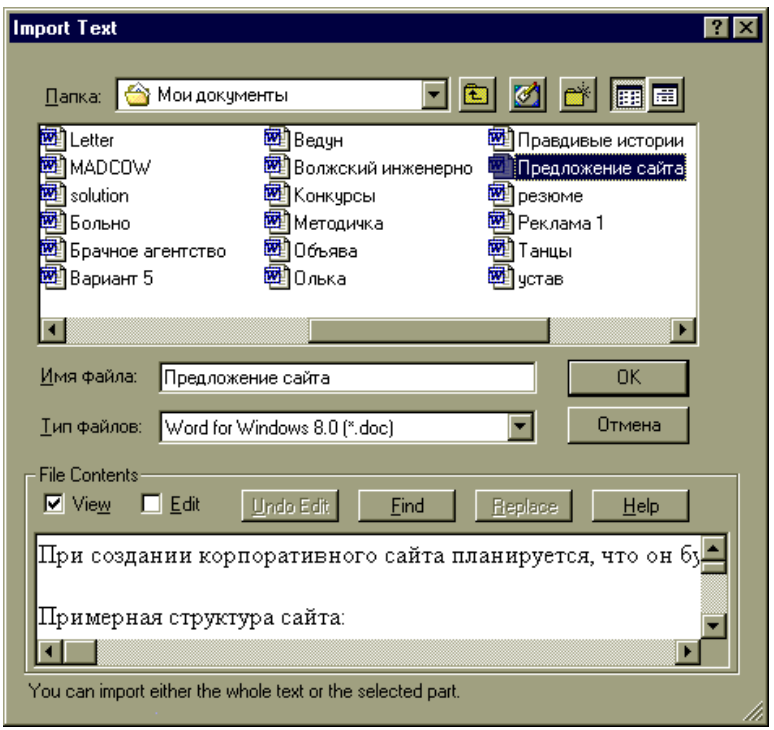

Рис. 4.5. Диалоговое окно **Import Text**

функциональности. Помимо обычного диалога открытия файла, который, кстати, позволяет оперировать файлами Microsoft Word, Microsoft Write, файлами в формате RTF и обычными текстовыми файлами с расширением txt, причем, как в кодировке для Windows, так и в кодировке для DOS, в этом окне присутствует группа элементов управления **File Contents**. Она предназначена для предварительного просмотра необходимого файла и выбора того фрагмента, который будет вставляться в редактируемую страницу. Чтобы в этом окне включить режим предварительного просмотра, необходимо установить флажок **View**. Для получения возможности редактирования текста в окне просмотра выставляется флажок **Edit**. При этом все изменения, введенные в этом окне, не будут распространяться на искомый текстовый файл. Впрочем, если последнее изменение текста оказалось неудачным, всегда есть возможность воспользоваться кнопкой **Undo Edit**, которая отменяет результаты редактирования. Поиск и замена текстовых блоков осуществляется при помощи кнопок **Find** и **Replace**, соответственно. После нажатия на кнопку **OK** выделенный текст из окна просмотра вставляется в текущую позицию на проектируемой странице. В том случае, если выделения текста не было, вставляется все содержимое файла.

WebShop позволяет также вставлять специализированные текстовые блоки. Для этого приготовлена кнопка **Special Text**. Она активизирует одноименное диалоговое окно (рис. 4.6). В нем присутствуют три вкладки, каждая из которых позволяет выбрать один тип текстового блока. На вкладке **Date** мы можем выбрать формат представления даты. Обычно этот блок используется для указания даты последнего изменения страницы. Вкладка **Mail to** вставляет гиперссылку на указанный адрес электронной почты. Вкладка **Copyright** предназначена для вставки информации об авторских правах. Все эти специализированные блоки обычно вставляются в конце страницы.

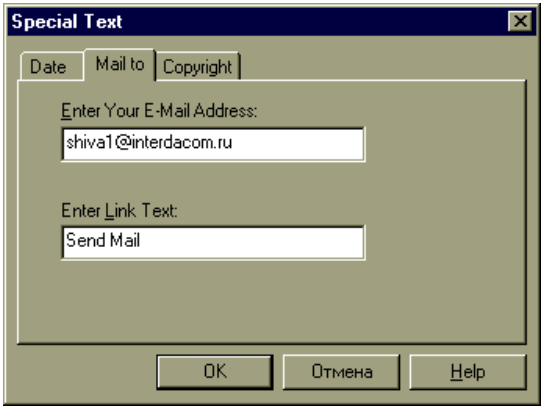

Рис. 4.6. Диалоговое окно **Special Text**

Для этой кнопки предусмотрены три дублирующие команды меню. Для выполнения все тех же действий применяются команды меню **Insert/Date**, **Insert/Mail to** и **Insert/Copyright** соответственно. Хотя каждая из этих команд

открывает все то же окно, где вполне можно выбрать для работы любую вкладку, а не только ту, которая была открыта по умолчанию.

Теперь пришло время рассмотреть специфику использования активных и создаваемых объектов.

Одним из немаловажных разделов любой страницы является панель навигации. Если эта панель не существует в виде отдельного фрейма, всегда отображаемого в составе сайта, ее необходимо устанавливать на каждой странице. Для этого применяется кнопка **Navigation Bar** или дублирующая ее одноименная команда меню **Insert**. В обоих случаях пользователю для работы предоставляется диалоговое окно **Insert Navigation Bar** (рис. 4.7). Его стоит рассмотреть подробно.

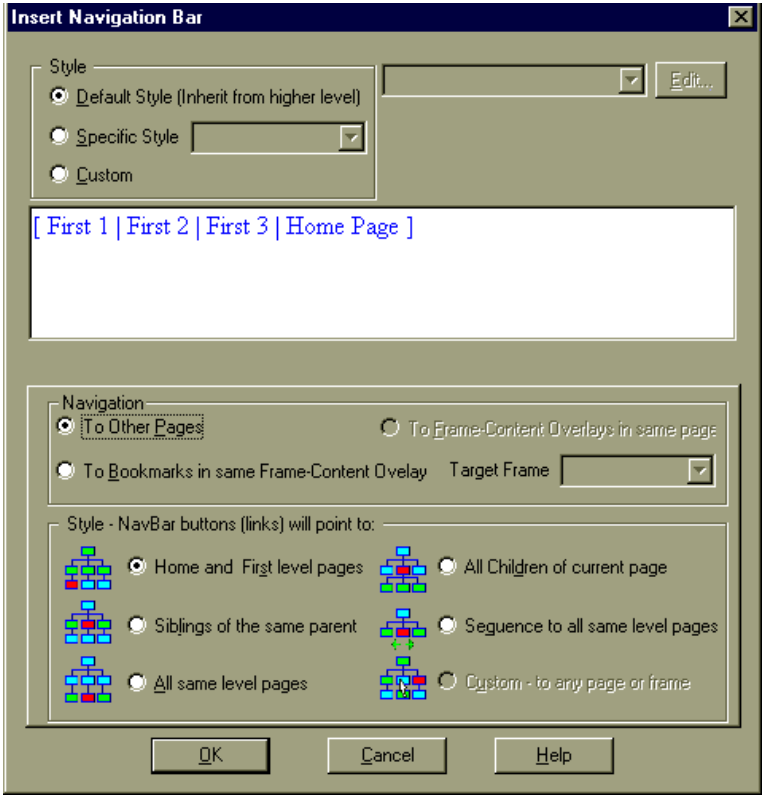

Рис. 4.7. Диалоговое окно **Insert Navigation Bar**

Прежде всего, в верхней части окна мы видим группу элементов управления **Style**, которая предназначена для выбора внешнего вида панели навигации. Более подробно мы рассмотрим эту процедуру в следующем разделе. Сначала нас будет интересовать группа радиокнопок **Navigation**, с помощью которой мы можем выбирать тип панели навигации. В том случае, если предпо-

лагается производить переадресацию пользователя на другие страницы сайта, необходимо использовать радиокнопку **To Other Pages**. Если же навигация будет осуществляться по закладкам внутри одной страницы, выбирается альтернатива **To Bookmarks in the same Frame-Content Overlay**. Радиокнопка **To Frame-Content Overlays in same page** позволяет создавать навигационную панель для перехода по ссылкам, указанным в оглавлении, которые вынесены в отдельный фрейм. При этом в поле **Target Frame** необходимо указать имя фрейма, в котором будет отображаться страница, на которую будет совершен переход.

Следующий блок радиокнопок с иконками, расположенными возле них, позволяет указывать поле навигации, то есть какие именно страницы войдут в создаваемую панель. Рассмотрим возможные варианты.

Радиокнопка **Home and First level pages** выносит в панель навигации ссылки на основную страницу и ссылки на так называемые страницы первого уровня, которые, по сути, являются дочерними страницами домашней странички. Альтернатива **Siblings of the same parent** позволяет осуществлять навигацию между текущей страницей и теми страницами, которые имеют одного с ней родителя. Впрочем, поведение этой радиокнопки зависит от того, какой тип панели мы выбрали до этого. Радиокнопка **All same level pages** включает в создаваемую панель ссылки на все страницы, находящиеся на том же уровне вложенности, что и текущая. Очевидно, что этот вариант оформления включает в себя предыдущий. Альтернатива **All Children of current page** позволяет осуществлять навигацию между дочерними по отношению к текущей страницами. Альтернатива **Sequence to all same level page** создает последовательность из ссылок на страницы, находящиеся на одном уровне. Таким образом, подключается еще один вариант навигации с использованием стандартных кнопок браузера перехода вперед и назад. И, наконец, последняя радиокнопка **Custom – to any page or frame** позволяет пользователю самому выбирать, какие именно ссылки войдут в создаваемую панель навигации.

Как мы уже знаем, одним из наиболее часто используемых средств оформления страницы является баннер, то есть достаточно объемный блок текста с ее наименованием, располагающийся обычно в самом верху страницы. Возможность автоматического создания баннера присутствует и в WebShop. Для этого используется команда меню **Insert/Banner** или одноименная кнопка. Но здесь есть один подводный камень. Текст, размещаемый в баннере задать, нельзя. Он повторяет наименование страницы. А если учесть, что они у нас пишутся только латинскими буквами, можно понять, что для русскоязычных сайтостроителей эта возможность практически неприменима. А жаль, так как при правильном выборе стиля баннер смотрится очень хорошо.

WebShop 2000 содержит также несколько специализированных объектов оформления. Их вставка осуществляется выполнением команды **Site/Insert Component**. При этом активизируется одноименное диалоговое окно,

в котором нам и предстоит выбрать тот компонент, который мы поместим на нашу проектируемую страницу. В выпадающем списке сначала необходимо выбрать тип компонента. На выбор предлагается два типа: часы (**Clocks**) и текстовые компоненты (**Text Components**). Часы, помещаемые на странице, будут всегда показывать текущее время. Нам доступны два вида часов. Аналоговые, то есть с циферблатом и движущимися стрелками, и цифровые. Какие выбрать — решайте сами.

Текстовые компоненты все однотипны и представляют собой реализацию движущегося текста. Компоненты **Horizontal Running Text** и **Vertical Running Text** создают бегущую строку, двигающуюся по горизонтали и вертикали соответственно. Более интересен компонент **Scrolling Text**. Он двигает снизу вверх несколько строк, расположенных горизонтально. Причем появляющаяся снизу строка выплывает из-за нижнего края рабочего поля компонента, постепенно увеличивая свою яркость. Достаточно интересный эффект.

После того, как при помощи кнопки **Insert** мы поместим выбранный текстовый компонент на страницу, необходимо задать отображаемый текст. Для этого компонент необходимо выделить и выполнить команду контекстного меню **Object/Properties**. При этом будет визуализировано диалоговое окно **Java Applet Properties** (рис. 4.8) (а надо отметить, что все компоненты реализованы именно в виде Java-апплетов). Именно в нем мы сможем установить все необходимые свойства для нашего компонента. Как легко заметить, нас будет интересовать список параметров апплета, размещенный в блоке **Parameters** в верхнем правом углу диалогового окна. Для изменения значения какого-либо параметра нужно просто произвести двойной щелчок мышью в колонке **Value** напротив имени этого параметра, и ввести необходимое значение. Осталось только узнать, какой параметр за что отвечает.

Итак, начнем. Параметры `bgcolor` и `textcolor` позволяют устанавливать соответственно цвет фона и цвет самого текста. Впрочем, если первая часть книги была прочитана внимательно, то никаких проблем с идентификацией этих параметров возникнуть не должно. Важная тонкость — значения указываются в шестнадцатеричной системе счисления. Параметр с именем `fadezone` устанавливает размер теневой зоны в нижней части рабочего поля апплета, которая и затеняет выплывающие строки. Размер указывается в пикселах. Скорость движения строк регулируется параметром `speed`. Величина отступа строк друг от друга указывается в параметре `space`. Параметр `fontsize` устанавливает размер шрифта. Значение указывается стандартно, в пунктах. Следующие шесть параметров `line1` — `line6` содержат отображаемые текстовые строки. Вот их-то значения и надо изменить, чтобы установить собственный текст в этом компоненте.

Для других текстовых компонентов текст бегущей строки указывается в параметре `message`. Цвет текста задается параметром `messagecolor`. А за установку конкретного шрифта (точнее, семейства, так как конкретный шрифт устанавливать не рекомендуется), его начертание и направление движения строки ответственны параметры `font`, `fontstyle` и `direction` соответственно.

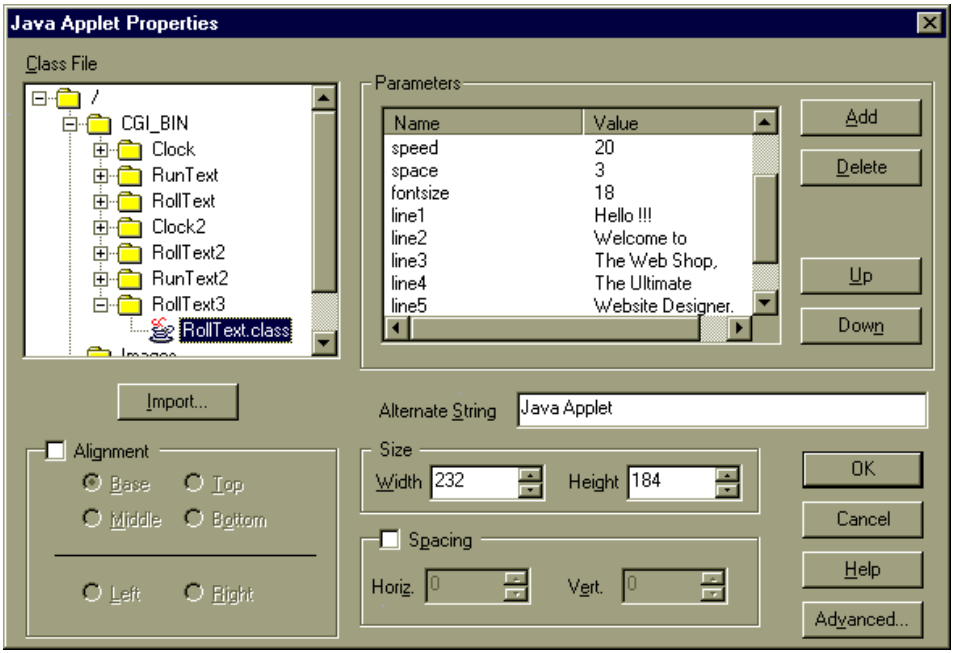

Рис. 4.8. Диалоговое окно **Java Applet Properties**

Естественно, нельзя обойти вниманием таблицы, размещаемые на Web-странице. Вставка таблицы производится выполнением команды меню **Table/Insert** или при помощи соответствующей кнопки. При этом активизируется диалоговое окно **Table Properties** (рис. 4.9). В группе полей ввода **Cells** мы задаем количество и тип строк и столбцов таблицы. Радиокнопки в группе **Dimensions** позволяют указывать размеры таблицы по вертикали и горизонтали. В полях ввода **Sizes** мы можем выставлять значения толщины границы, отступов ячеек друг от друга и отбивки содержимого ячейки от границы ячейки. Группа элементов управления **Caption** предназначена для создания заголовка таблицы и указания его расположения. А в группе **Alignement** (да-да, написано именно так) мы можем указать расположение самой таблицы на странице. Дальнейшая работа с таблицей производится при помощи меню **Table**.

Вставка формы в тело Web-страницы осуществляется нажатием на кнопку **Form** или выполнением команды меню **Insert/Form**. При этом не появляется никакое диалоговое окно, но становится доступной для использования инструментальная панель **Form Fields Toolbar**. На ней находятся семь кнопок, каждая из которых позволяет вставлять тот или иной элемент управления. Впрочем, дублирование этих кнопок командами меню также было любезно предусмотрено разработчиками программы. Существует команда меню **Insert/Form Field**, которая показывает встроенное подменю, содержащее команды вставки отдельных элементов управления.

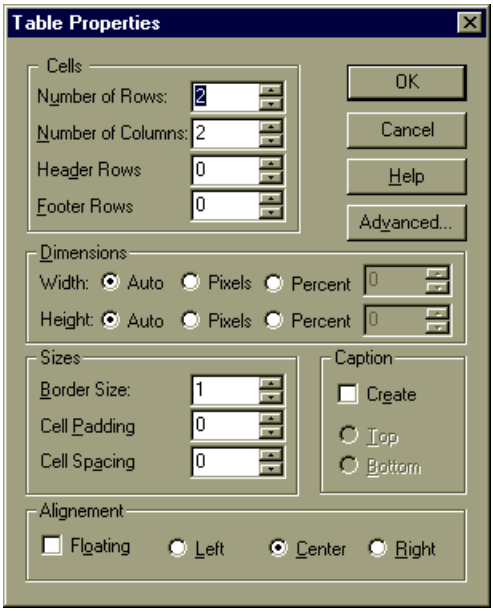

Рис. 4.9. Диалоговое окно **Table Properties**

Если у вас есть какой-либо полезный Java-апплет и есть желание встроить его в страницу, следует выполнить команду меню **Insert/Java applet** или нажать одноименную кнопку. Для работы будет предоставлено диалоговое окно, в котором можно будет выбрать файл, в котором хранится апплет, установить параметры его выполнения и правильно разместить его на странице.

Точно так же следует поступать и с встраиваемыми объектами (plug-in), только для них существует отдельная кнопка **Embedded Object** и соответствующая команда меню. Сама технология встраивания точно такая же, как и в случае с Java-апплетом. Помимо подключаемых модулей, при помощи этого средства подключаются также аудио- и видеоклипы, а также анимационные изображения в формате GIF.

Для вставки сценариев как на языке Java, так и выполненных по технологии VB, зарезервирована кнопка **Script Object**. Однако подключение сценариев требует достаточно детальных знаний этой технологии, поэтому прежде чем встраивать их в свою страницу, подробно разберитесь с тем, какие действия и как он производит.

И последний элемент, который мы рассмотрим в этой главе, — это вставляемый HTML-код, который применяют в том случае, когда при помощи штатных средств программы нельзя добиться требуемого эффекта. Тогда Web-мастер может сам вставить его в HTML-код страницы. Для этого есть кнопка **HTML Code**.

Теперь, после того как мы рассмотрели объекты, используемые для создания страниц, узнаем, как устанавливать свойства для них.

Установка свойств

Прежде всего, нас будут интересовать, естественно, свойства текста. Процедура проста. Выделяем текстовый блок, которому мы хотим приписать какие-либо особые свойства, отличные от установленных по умолчанию. Параметры шрифта изменяются при помощи команды меню **Format/Font**, либо при помощи выбора **Font** из контекстного меню, которое, как мы помним, вызывается щелчком правой клавиши мыши. В появившемся диалоговом окне мы можем устанавливать вид шрифта, его начертание и размеры. Причем размеры можно установить как абсолютные, измеряющиеся в стандартных пунктах, так и относительные, принятые в оформлении Web-страниц. В группе **Effects** можно задать тип применяемого к тексту эффекта, а при помощи выпадающего списка **Color** задается цвет шрифта и фона, на котором он будет расположен. Как мы помним из первой главы, любой текст может быть отображен в нескольких вариантах. Причем реализуется это при помощи тегов HTML. Список подобных представлений текста называется **HTML Logical Style**, и настоятельно рекомендуется к применению. Здесь действует общее правило — добивайся поставленной цели как можно более простым путем. Если необходимо привести текст к какому-либо заранее установленному виду, лучше перед тем, как начинать манипуляции со шрифтом, его размерами и начертанием, сначала посмотреть, нельзя ли это сделать уже существующими средствами.

Расположение абзаца и его свойства можно задать при помощи вкладки **General** диалогового окна **Paragraph Properties** (рис. 4.10), которое активизируется при выполнении команды **Format/Paragraph** или ее двойника

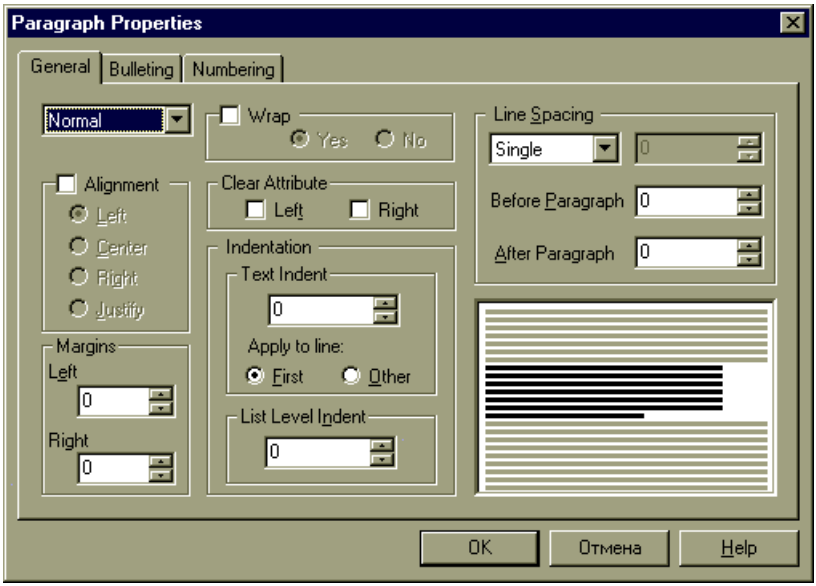

Рис. 4.10. Вкладка **General** диалогового окна **Paragraph Properties**

из контекстного меню. Впрочем, для тех же целей может служить кнопка **Paragraph**. В основном выпадающем списке мы можем выбрать стиль параграфа, а при помощи группы радиокнопок **Alignment** указать его выключку. Отступы всего абзаца слева и справа указываются при помощи полей **Left** и **Right** соответственно, находящихся в группе **Margins**. В том случае, если слова в абзаце должны разбиваться переносами, необходимо поставить флажок **Wrap** и выбрать радиокнопку **Yes**. Принудительное отключение переносов осуществляется при помощи кнопки **No**. Группа независимых переключателей **Clear Attributes** позволяет указывать параметры обтекания текстом графических изображений, если таковые захватывают пространство абзаца. В группе **Indentation** можно задавать отступы для строк, а не для всего абзаца. В поле **Text Indent** вводится величина отступа, а при помощи радиокнопок устанавливается область действия указанного отступа. Если выбрана кнопка **First**, будет реализован отступ красной строки. Если выбрана радиокнопка **Other**, то, наоборот, сдвинуты будут все строки абзаца, кроме первой. Межстрочное расстояние абзаца выбирается в выпадающем списке из группы **Line Spacing**. В нем есть четыре стандартных значения, и одно дополнительное — **Exactly**, которое позволяет пользователю вручную установить любой отступ. В поле ввода **Before Paragraph** указывается величина отступа от предшествующего абзаца, а в поле **After Paragraph** — отступ до последующего абзаца. Впрочем, на этой вкладке всегда присутствует окно с примером того, как будет выглядеть абзац после внесения каких-либо изменений.

Следующий наш шаг — это установка свойств маркированного списка. Для этого используется вкладка **Bulleting** (рис. 4.11) все того же диалогового окна **Paragraph Properties**. Для того чтобы весь абзац стал элементом

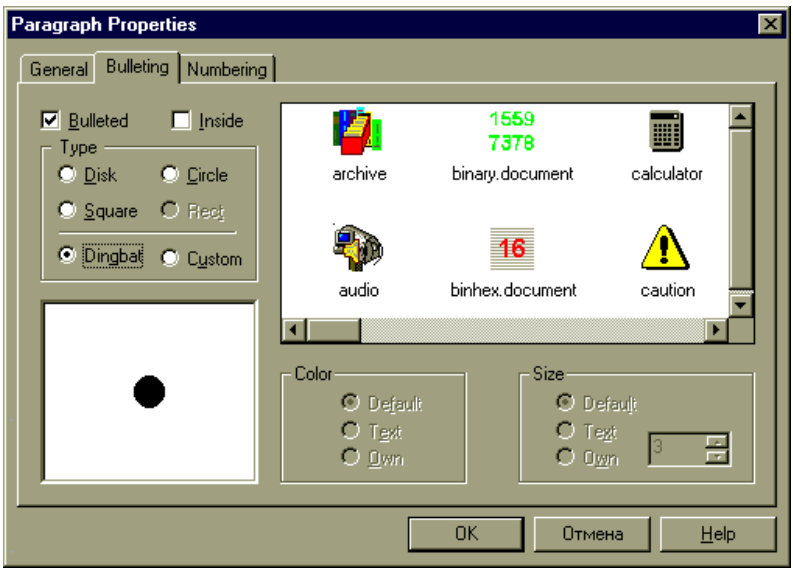

Рис. 4.11. Вкладка **Bulleting** диалогового окна **Paragraph Properties**

маркированного списка, необходимо выставить флажок **Bulleted**. Если же маркеры будут располагаться внутри абзаца, следует выбрать флажок **Inside**. После этого в группе радиокнопок **Type** можно выбрать внешний вид используемого маркера. При этом его вид будет сразу отображаться в окне предварительного просмотра. Радиокнопки **Disk**, **Square** и **Circle** создают стандартные маркеры в виде круга, квадрата и окружности, реализуемые уже известными нам HTML-тегами. Выбор радиокнопки **Dingbat** позволяет в качестве маркеров использовать одну из сорока одной стандартных иконок.

А если выбрать радиокнопку **Custom**, то мы сможем сами установить вид маркера, не прибегая к стандартным и уже порядком надоевшим средствам. При этом в основном окне появляется три вкладки, каждая из которых предлагает свой механизм установки внешнего вида маркера. Вкладка **Existing Resources** предлагает нам выбрать рисунок, находящийся в пределах структуры каталогов сайта, которая отображается в основном окне. Впрочем, если необходимого графического файла еще нет в нашей структуре, его можно либо создать (кнопка **Create**), либо импортировать из любой папки локальной машины (кнопка **Import**). Вкладка **Library** предоставляет для выбора уже готовые изображения, находящиеся в библиотеке WebShop. При этом сначала нужно выбрать из выпадающего списка одну из четырнадцати (!) категорий рисунков, а затем уже в основном окне выбрать наиболее понравившийся маркер. Ну, а если интересующий вас маркер находится где-то в сети Интернет, следует выбрать вкладку **URL** и в поле редактирования ввести URL этого графического файла.

Следующий тип списков — нумерованный. Для установки его свойств используется последняя вкладка этого диалогового окна — **Numbering** (рис. 4.12). Как и в предыдущем случае, если в обрабатываемом абзаце должна

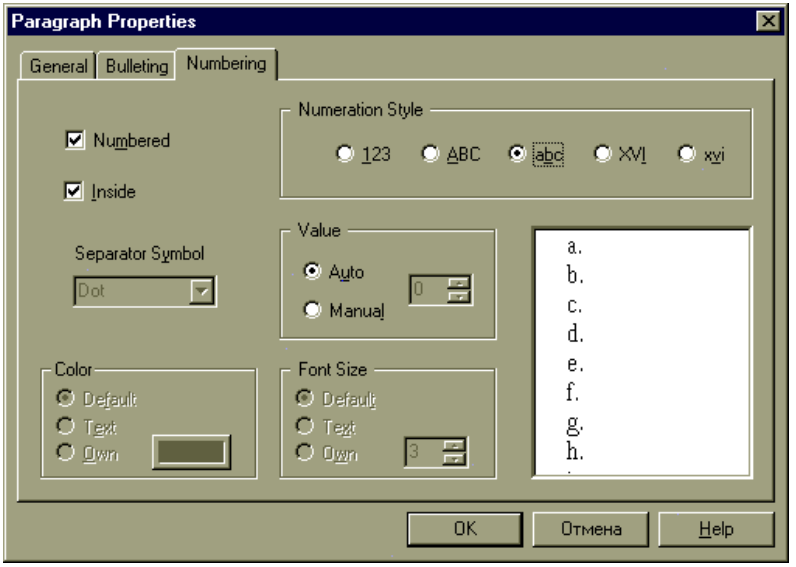

Рис. 4.12. Вкладка **Numbering** диалогового окна **Paragraph Properties**

быть нумерация, устанавливается флажок **Numbered**. Вид нумерации выбирается при помощи радиокнопок **Numeration Style**, при этом внешний вид абзаца сразу отображается в окне быстрого просмотра. Стартовое значение списка выбирается в группе **Value**. Радиокнопка **Auto** указывает на то, что нумерация пойдет с первого пункта, а выбор альтернативы **Manual** активизирует поле ввода, в котором стартовый номер может быть введен вручную и отличаться от единицы.

Свойства самой страницы можно редактировать на вкладке **HTML** (рис. 4.13) диалогового окна **Properties**, которое вызывается по команде меню **Frames/Resource properties**. В поле ввода **Title** вводится наименование страницы, а таблица **Meta Information** предназначена для создания служебных тегов `Meta`. Особенности этих тегов мы рассматривали в *главе 1*.

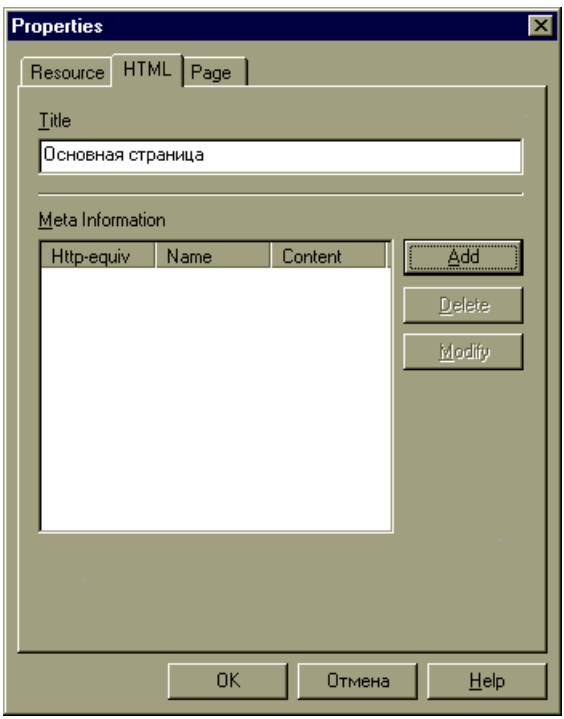

Рис. 4.13. Вкладка **HTML** диалогового окна **Properties**

Общее оформление

Как мы помним, при разработке сайта необходимо уделять особое внимание единообразному оформлению всех страниц, входящих в его состав. Программный комплекс WebShop содержит в себе подобные средства оформления. Во-первых, конечно, это богатейшая библиотека стилей. Ее можно увидеть в окне **Site Manager** на вкладке **Styles**. Эта коллекция для большего удобства навигации разбита на три части и выбор раздела производится из выпадающего списка **Styles Library**. После этого можно уже выбирать конкретный стиль и до-

бавлять его в список стилей, используемых для оформления вашего сайта. Этот вторичный список имеет имя **Site Styles**. Добавление в него выбранного стиля производится нажатием правой кнопки мыши и выбором команды объектного меню **Insert Style**. После этого можно его применять.

Изначально все страницы сайта создаются с оформлением в стиле, установленном по умолчанию. Он генерируется при создании сайта и носит название **Default_Style**. Если мы хотим по умолчанию установить один из тех стилей, которые были выбраны нами из библиотеки и помещены в список, доступных для всего нашего сайта, необходимо щелкнуть на этом стиле правой кнопкой мыши и в появившемся объектном меню выбрать команду **Set as Site Default Style**. При этом все страницы, входящие в состав сайта, мгновенно поменяют оформление.

Впрочем, мы можем изменить тот стиль, который был создан по умолчанию, на свой собственный. Для этого выполняется команда меню **Site/Style** или одноименная команда из объектного меню, относящегося ко всей проектируемой странице. При этом появляется диалоговое окно **Site Style** или **Page Style** (рис. 4.14), в зависимости от того, какая команда была выполнена, но сами по себе эти диалоговые окна являются близнецами.

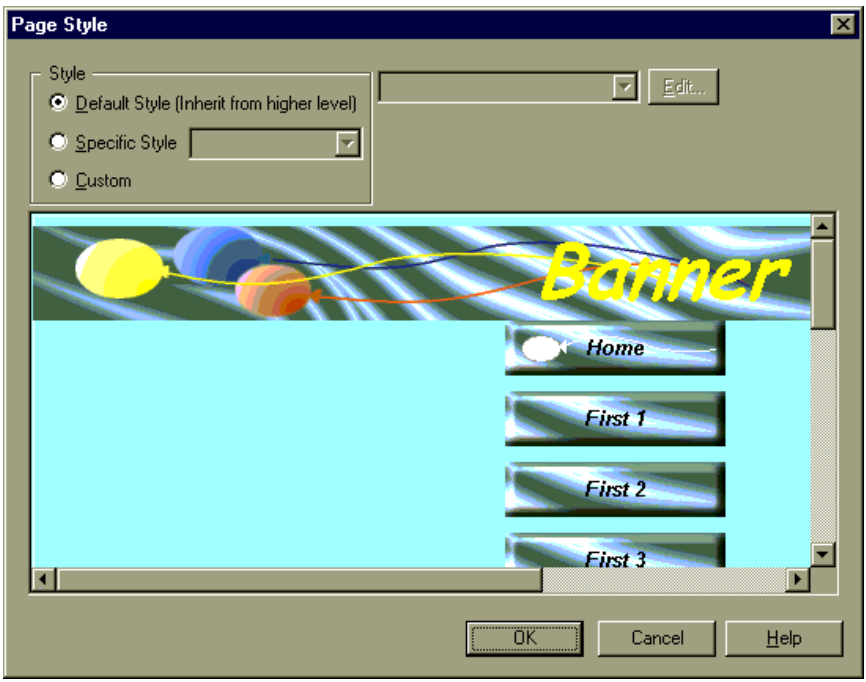

Рис. 4.14. Диалоговое окно **Page Style**

Итак, если у нас задействовано окно **Page Style**, то нам доступна радиокнопка **Default Style**. Выбор ее приводит к наследованию данной страницей

стиля, установленного по умолчанию для ее родительской страницы. Вторая радиокнопка **Specific Style** доступна в обеих модификациях окна. При ее выборе активизируется выпадающий список, содержащий наименования всех стилей, которые используются при оформлении сайта. А вот третья радиокнопка **Custom** позволяет нам менять элементы оформления, прописанные в данном стиле. При этом активизируется выпадающий список, содержащий наименования всех элементов оформления страницы. Для редактирования такого элемента необходимо выбрать его в списке и нажать на кнопку **Edit**. После этого появляется диалоговое окно, предназначенное для изменения свойств именно этого элемента оформления. Еще одним элементом единообразного оформления страниц являются единые колонтитулы. Напомню, что колонтитулами называют нижнюю и верхнюю части страницы, которые несут общую для всех страниц информацию. Мы можем задать некоторое количество единых верхних и нижних колонтитулов и затем использовать их в оформлении Web-страниц.

Для создания колонтитулов необходимо в окне **Site Manager** выбрать вкладку **Common Objects**. Основное окно на этой вкладке разделено по горизонтали на две части. Верхняя часть содержит иконки для верхних колонтитулов, в нижней части показаны иконки для нижних. По умолчанию уже созданы два колонтитула: верхний и нижний, с именами **Header1** и **Footer1** соответственно, но они пусты.

При переходе к окну **Site Manager** появляется соответствующая инструментальная панель. Редактирование существующего колонтитула осуществляется либо двойным щелчком по его иконке, либо при нажатии на кнопку **Edit Object**. При этом появляется окно с рабочим пространством, на котором мы и размещаем ту информацию, которая будет наполнять колонтитул. Создание дополнительного колонтитула тоже не составляет особых проблем. Для этого достаточно на свободном месте окна щелкнуть правой кнопкой мыши и в появившемся контекстном меню выбрать команду **New**. Также можно воспользоваться кнопкой **Create Object**. Если существует несколько верхних и нижних колонтитулов, один из них всегда будет являться основным, устанавливаемым по умолчанию. При этом на его иконке появляется красная галочка. Для того чтобы сделать какой-либо колонтитул основным, достаточно либо нажать на кнопку **Set As Default**, либо использовать контекстное меню этого объекта, выбрав в нем одноименную команду.

Впрочем, помимо создания подобных объектов, необходимо еще включить их в наши страницы. Для того чтобы все страницы, входящие в состав нашего сайта, получили украшения в виде колонтитулов, установленных по умолчанию, необходимо выполнить команду меню **File/Preferences**. В появившемся одноименном диалоговом окне нужно выбрать вкладку **WebSite** и выставить флажок **Headers/Footers**. Если же необходимо колонтитулы применять выборочно, к конкретным страницам, а не ко всему сайту, следует выполнить команду меню **Frames/Resource properties**. При этом будет активизировано диалоговое окно **Properties**. Нас в нем будет интересовать

вкладка **Page**, в выпадающих списках **Header** и **Footer** которой можно выбрать верхний и нижний колонтитулы и применить их к данной странице.

Библиотеки WebShop

WebShop 2000 обладает неплохой коллекцией библиотек, которые необходимо применять в своей работе хотя бы ради того, чтобы не изобретать велосипед. Вполне возможно, что у вас уже есть целый локомотив.

Прежде всего, мы рассмотрим библиотеку цветовых палитр. Работа с ними осуществляется при помощи диалогового окна **Color Library** (рис. 4.15.), активизируемого при выполнении команды меню **Library/Edit Color Library**. К сожалению, недостатки полиграфии несколько меняют его внешний вид не в лучшую сторону. Но все элементы управления видны достаточно отчетливо, и этого нам достаточно.

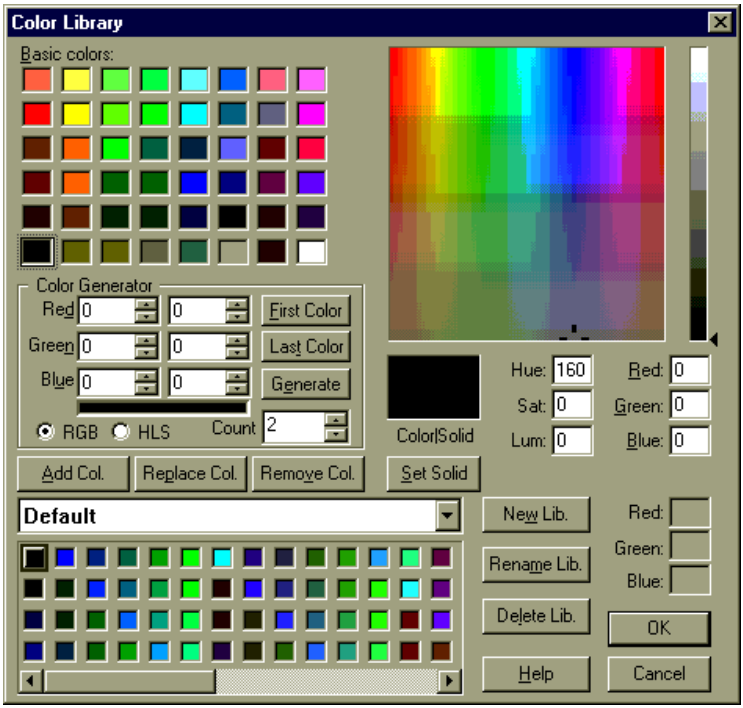

Рис. 4.15. Диалоговое окно **Color Library**

Прежде всего, следует уяснить, что нельзя редактировать стандартную палитру с именем **Default**. Все остальные палитры можно привести в соответствие со своими представлениями о прекрасном. Для выбора конкретной палитры необходимо воспользоваться выпадающим списком в нижней части окна. После этого все ее цвета отображаются в наборе цветовых квадратов в левом нижнем углу

диалогового окна. Теперь, когда у нас есть представление о том, как выглядит данная палитра, можно приступать к внесению изменений.

Любой цвет из палитры мы можем заменить на один из сорока восьми базовых цветов, находящихся в верхнем левом углу окна, в блоке **Basic colors**. Для этого мы выделяем щелчком мыши тот цвет в палитре, который мы хотим сменить, а затем выбираем базовый цвет. После этого нам останется лишь нажать на кнопку **Replace Col.**, и цвет будет сменен.

Мы всегда можем уменьшить глубину палитры путем удаления из нее нескольких цветов. Удаление выбранного цвета производится нажатием кнопки **Remove Col.**. Впрочем, нам доступна и обратная операция. Мы всегда можем добавить новый цвет в палитру, нажав кнопку **Add Col.**. При этом если был выбран один из базовых цветов, то добавляется именно он. Впрочем, их количество ограниченно, поэтому иногда хочется выбрать цвет, не входящий в базовый набор. Для этих целей в правом верхнем углу окна есть обширное цветовое поле, на котором можно выбрать любой подходящий цвет простым щелчком мыши на соответствующем месте. Образец выбранного цвета тут же будет отображен в цветовом квадрате **Color|Solid**. Но если вас не устраивает такой выбор цвета, а хочется большей точности и определенности, всегда можно набрать RGB-код.

Если заполнять палитру по одному цвету слишком тяжело, можно сгенерировать целую цветовую последовательность. Для этого следует воспользоваться группой элементов управления **Color Generator**. В ней есть два столбца полей ввода. В первом столбце мы вводим код стартового цвета последовательности, во втором — код заключительного цвета. Количество цветов в генерируемой последовательности указывается в поле **Count**. Если нет желания подбирать коды, то необходимый цвет можно выбрать непосредственно среди базовых, или на цветовом поле, а затем нажать кнопку **First Color** для стартового цвета или **Last Color** — для заключительного. После того, как установлены стартовый и заключительный цвета, а также глубина цветовой последовательности, нажимается кнопка **Generate** и данная цветовая группа записывается в состав выбранной палитры.

Теперь рассмотрим возможности воздействия на палитры в целом. Так как нам уже не подвластны отдельные цвета, все, что мы можем, это генерировать новые, удалять палитры и изменять их имена. Напомню, что ни удалить, ни изменить имя для палитры **Default** невозможно. Кнопка **New Lib.** диалогового окна **Color Library** создает новую палитру со всего одним цветом. Новые цвета в нее надо добавлять. Кнопка **Delete Lib.** выполняет обратную операцию — удаляет выбранную палитру. А для изменения имени у текущей палитры следует нажать кнопку **Rename Lib.**.

Следующая библиотека, работу с которой мы рассмотрим, содержит в себе все рисунки, поставляемые вместе с WebShop 2000. Ее редактирование можно осуществлять после выполнения команды меню **Library/Edit Image Library** в появившемся окне **Library Edit** (рис. 4.16).

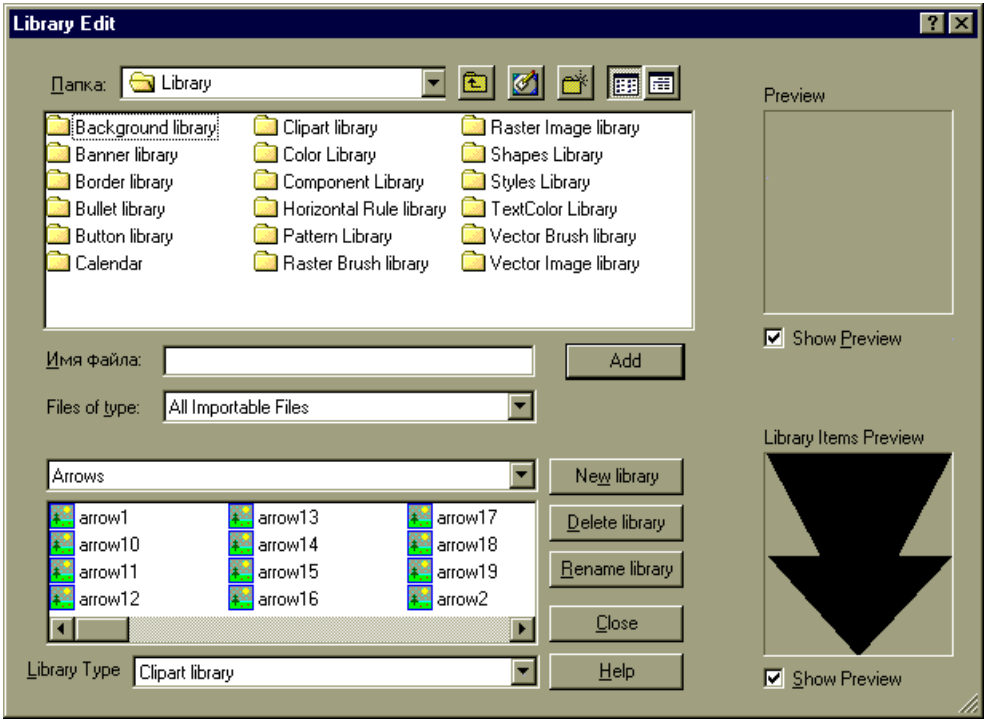

Рис. 4.16. Диалоговое окно **Library Edit**

Необходимо осознавать, что есть несколько типов графических библиотек, а уже внутри каждого типа есть отдельные библиотеки. Тип библиотеки указывается в выпадающем списке **Library Type**, который находится в нижней части окна. После этого, в верхнем выпадающем списке, который находится сразу над окном выбора отдельных рисунков, указывается конкретная библиотека. Теперь мы можем с ней работать и, прежде всего, можем просматривать те рисунки, которые в ней находятся. Для этого надо установить флажок **Show Preview**, находящийся в правом нижнем углу окна. Теперь при указании какого-либо графического файла в общем списке можно будет сразу увидеть изображение, хранящееся в нем.

Создание новой библиотеки данного типа производится в этом окне при помощи кнопки **New library**, а нажатие на кнопку **Delete library** удаляет уже существующую. Кнопка **Rename library** позволяет сменить наименование указанной библиотеки.

После того, как мы создали новую библиотеку, в нее надо импортировать графические файлы. В верхней части окна находится стандартный диалог открытия файла. Мы выбираем папку, указываем имя импортируемого файла и нажимаем кнопку **Add**. Все, графическое изображение добавлено в текущую библиотеку. Для большего удобства поиска графического файла рекомендуется поставить флажок **Show Preview**. Тогда в поле предварительного

просмотра **Preview** мы получим возможность увидеть содержимое присоединяемого файла.

Последняя библиотека WebShop 2000 содержит в себе шаблоны для страниц. Редактировать ее и добавлять новые шаблоны достаточно нелегко, поэтому мы ограничимся лишь обзором тех готовых решений, которые приготовили для нас разработчики.

Набор шаблонов доступен нам по команде **File/New**. При этом активизируется диалоговое окно, в верхней части которого расположен выпадающий список. В нем мы должны выбрать категорию нового сайта. Всего этих категорий четыре. Первая — **_E-Commerce** содержит шаблоны для создания Интернет-магазинов. Вторая категория с наименованием **Art and Education** предоставляет нам шаблоны для создания сайтов, ориентированных на обучение, или действующих в области искусства. Представленные образцы немного академичны, но стиль в них выдержан. Следующая категория — **Business and Finance** включает в себя образцы сайтов, предназначенных для работы в финансовой сфере, либо для делового представительства. И, наконец, последняя категория **Communication and Services** позволяет создавать сайты, нацеленные на работу по связям с общественностью.

Выбор любого из этих шаблонов приводит к генерации готового сайта, в котором необходимо лишь заменить текст и добавить необходимые страницы. Но общая структура и оформление уже будут созданы. Последние три категории автоматически генерируют сайты, а вот для создания магазина придется немного поработать. Но об этом — в следующем разделе.

Создание торговой системы

Подключать вручную базу данных при создании сайта для магазина всегда несколько затруднительно. В нашем случае есть более легкое и простое решение. Мы можем воспользоваться готовым шаблоном из серии **_E-Commerce**. Выполняем команду **File/New**, выбираем шаблон и нам остается лишь отвечать на запросы программы.

С самого начала мастер создания сайта спрашивает вас, хотите ли вы зарегистрироваться в платежной системе. Кнопка **Register Now** запускает процесс регистрации, но мы отложим его, выбрав кнопку **Register Later**, так как нас все-таки будет интересовать подключение к российским платежным системам. Однако, если планируется создание международного магазина, можно подписаться на услуги предоставляемой платежной системы Merchant.

Следующий шаг мастера — заполнение данных о компании. Это делается в появившемся диалоговом окне **Company Information** (рис. 4.17). В принципе, заполнение большинства полей ввода не должно вызвать какого-либо затруднения. Стоит обратить лишь особое внимание на группу элементов управления **Sales: Tax Rate**, в которой устанавливается размер налога с продаж. В нижней части диалогового окна располагается область для ввода данных об администрировании сайта.

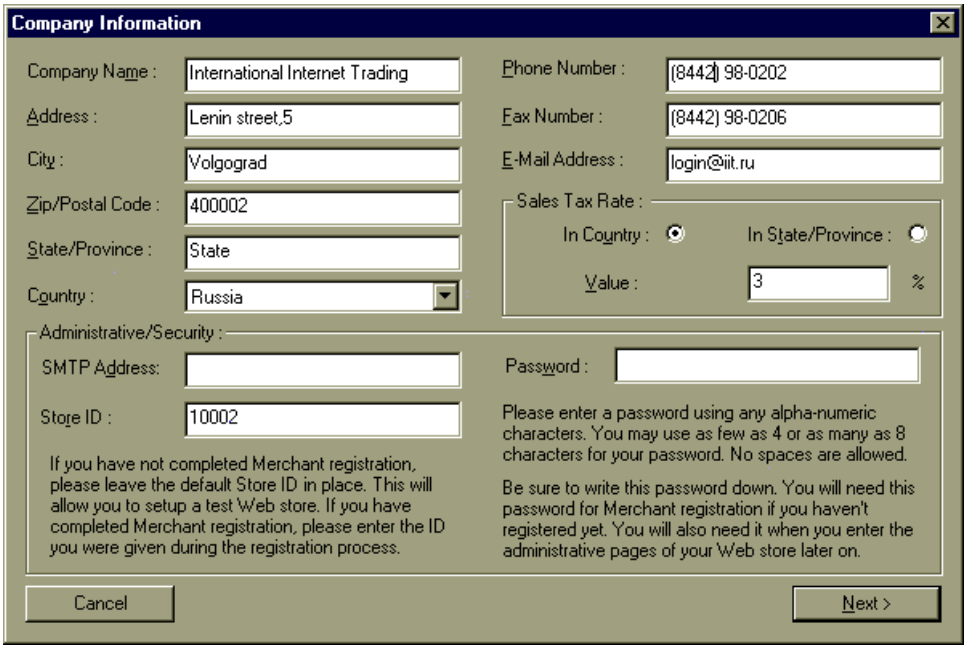

Рис. 4.17. Диалоговое окно **Company Information**

В следующем окне необходимо пометить те виды платежей, которые будет принимать новый магазин. Это могут быть различные пластиковые карточки, чеки или наличная оплата. Выбор за вами.

Следующее диалоговое окно позволяет выбрать способ продажи, то есть будет ли серия заказов одной покупкой, или для каждого товара будет оформляться отдельная покупка. Рекомендуется выбирать способ оплаты **Flat Rate**, как наиболее простой для работы.

Осталось только ввести описания товаров, что можно сделать в предназначенном для этого диалоговом окне (рис. 4.18).

По умолчанию у нас уже есть список товаров, но он, естественно, нам не пригодится. Все, для чего он нужен — это получить пример заполнения полей. В поле **ProductID** вводится уникальный внутренний номер товара, поле **ProductCode** содержит символьно-цифровой код для аналитического учета продаж. В поле **Notes** вводится дополнительная информация о товаре, а его название размещается в поле **ProductName**. В поле **ProductIntroductionDate** указывается дата, с которой товар поступает в продажу. Описание товара вводится в поле **ProductDescription**, а его размеры, необходимые для того, чтобы посетитель сайта полнее представил его себе, вносятся в поле **ProductSize**. Ссылка на графический файл с изображением товара вводится в поле **ProductImageURL**. По сути, при добавлении товаров в базу данных, в это поле вставляется рисунок из графической библиотеки или файловой структуры сайта. Цена товара указывается в поле **UnitPrice**. Наличие в продаже указывается в логическом поле **OnSale**. Вот и весь список критичных для базы данных полей. Все остальные поля не обязательные.

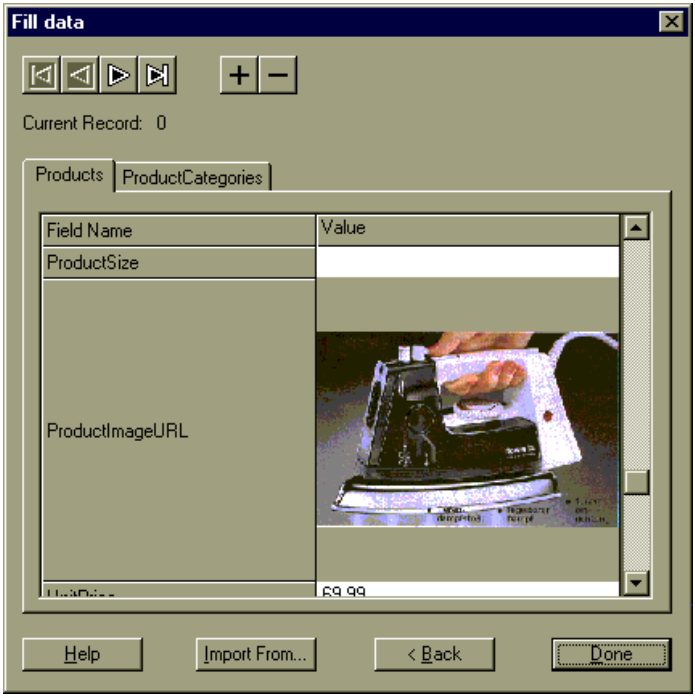

Рис. 4.18. Диалоговое окно **Fill data**

Для перемещения по базе данных товаров используются четыре кнопки в верхнем левом углу. Добавление и удаление записей в базу данных производится при помощи кнопок с изображением плюса и минуса соответственно.

В том случае, если вводить данные о каждом товаре по отдельности слишком долго, можно подключить готовую базу данных в формате Microsoft Access. Естественно, структура этой таблицы должна совпадать с предустановленной в WebShop структурой.

После завершения ввода информации в базу данных мы можем нажать на кнопку **Done** и получить заготовку сайта. Техническую поддержку сайта трогать не стоит, для нас она представляет собой закрытый черный ящик. Впрочем, специалисты могут успешно изменять эти процедуры.

После завершения ввода всех данных и изменения оформления сайта мы можем запускать его в работу. Но ведь данные ежедневно меняются. Для внесения последующих изменений в логику работы и базу данных товаров активизируется пункт меню **E-Commerce**. Так, для регистрации в системе Merchant используется пункт меню **E-Commerce/Merchant Registration**. А остальные пункты меню полностью повторяют порядок действия мастера, вызывая соответствующие диалоговые окна. Но для тонких настроек, подключения сторонних платежных систем и прочих умных вещей необходимо обратиться к квалифицированному специалисту. Мы использовали лишь стандартные решения.

Послесловие

Теперь оглянемся назад и посмотрим, чему же мы научились. Сначала мы просто узнали, как создавать Web-страницы, объединяя их в сайты. Попутно мы совершили экскурс в язык HTML, узнав его основные конструкции и научившись их применять.

Во второй главе мы узнали специфику графических файлов, применяемых для оформления Web-страниц. Теперь мы знаем, какие требования к ним применяются и можем правильно выбрать баланс между объемом рисунка и скоростью его загрузки. Мы умеем пользоваться графическими пакетами, ориентированными на создание графики для Web.

В третьей главе мы провели краткий обзор административных сложностей при размещении сайта в своей сети или у провайдера. Мы узнали, как можно защитить свою сеть от внешнего вторжения.

Ну и, наконец, в четвертой главе мы научились работать с программой WebShop 2000 для генерации сайтов с Интернет-магазином. Все это лишь введение в обширный мир Интернет-технологий. Если это заинтересовало вас — работайте и учитесь дальше. Интернет очень сильно развивается и место под солнцем найдется для каждого. Создавайте свои сайты и добивайтесь достижения поставленных для них целей. И не стесняйтесь привлекать специалистов для реализации тех задач, которые вам пока (!) не по плечу.

Удачи вам в этом нелегком деле.

Глоссарий

- *Active Data Objects* — активные компоненты, при помощи которых посетитель сайта может получать через своего провайдера доступ к базам данных, расположенным на удаленном сервере.
- *Active Server Page* — страница, содержащая встроенные сценарии, исполняемые на стороне сервера. На стороне клиента подобные страницы являются обычными Web-страницами, которые адекватно отображаются любым браузером.
- *ActiveX* — набор технологических стандартов, позволяющих создавать интерактивные элементы, встраиваемые в Web-страницы. Исполняемый код элемента загружается с сайта при первом запросе страницы с встроенным элементом ActiveX. В последующем, при каждом запросе этой страницы исполняется уже сохраненный на локальной машине код.
- *Animated GIF* — файл графического формата GIF, содержащий серию рисунков, которые последовательно отображаются браузером, создавая анимационное изображение.
- *Cascading Style Sheet (CSS)* — правила HTML, которые позволяют присоединять к HTML-документам стилевые таблицы. В этих таблицах указываются параметры текста, позволяющие отображать его максимально близко к замыслу автора страницы, избегая установок браузера по умолчанию.
- *Common Gateway Interface (CGI)* — стандартный интерфейс, позволяющий Web-серверам обмениваться информацией с базами данных и специализированными Интернет-приложениями. Эти приложения при помощи протокола HTTP получают информацию от Web-сервера, обрабатывают ее и возвращают результат обработки либо в виде сформированной Web-страницы, либо в виде ссылки на уже существующую страницу.
- *Dynamic HTML* — динамический HTML, расширение стандарта языка HTML, позволяющее создавать различные динамические эффекты, применяемые к тексту и другим объектам.
- *Embedded Style Sheets (ESS)* — стилевые таблицы, встроенные в конкретные HTML-файлы. Их действие распространяется только на оформление того файла, в который они встроены.

- *External Style Sheet* — файл с расширением css, в котором хранятся стилевые таблицы, подключаемые к текущему HTML-документу.
- *Frequently Asked Questions (FAQ)* — ответы на наиболее часто задаваемые вопросы.
- *FTP (File Transfer Protocol)* — протокол, на котором основан сервис обмена файлами в Интернете. Ссылки на ресурсы подобного типа начинаются с префикса ftp://.
- *GIF (Graphics Interchange Format)* — графический формат файлов, обычно использующихся в оформлении Web-страниц для отображения рисунков с ограниченным (не более 256) количеством цветов. Этот формат хранит данные об изображении в сжатом виде, что и делает его удобным для использования в Интернете.
- *HTML (HyperText Markup Language)* — язык, применяемый для разметки Web-страниц. HTML-код страницы содержит помимо текстовой информации указания по ее отображению, гиперссылки, регулирует вставку графических рисунков, аудио- и видеоклипов, и многое другое. При загрузке Web-страницы на компьютер удаленного пользователя браузер распознает конструкции языка HTML и соответствующим образом обрабатывает их, отображая страницу.
- *HTTP (HyperText Transfer Protocol)* — основной протокол Интернета, предназначенный для передачи гипертекста.
- *IP (Internet Protocol)* — программное обеспечение, которое ответственно за передачу информации в сети Интернет, и других локальных сетях, построенных на основе протокола TCP/IP.
- *IP-адрес* — стандартный способ идентификации машин, входящих в сеть, действующую на основе протокола TCP/IP. Обычно представляет собой комбинацию из четырех чисел, разделенных точкой, каждое из которых не может быть больше 255, например 90.0.0.1.
- *ISAPI (Internet Server Application Programming Interface)* — интерфейс для приложений, базирующихся и выполняемых на стороне Web-сервера. Разработан как вариант дополнения и замены CGI.
- *Java* — язык высокого уровня, разработанный и активно поддерживаемый Sun Microsystems, который очень хорошо подошел для нужд Интернета. Выполняемые модули, написанные на Java, могут выполняться на машине любого удаленного пользователя вне зависимости от того, какая платформа им используется.
- *JavaScript (сценарий Java)* — сценарий в виде текста, включаемый непосредственно в Web-страницу и загружаемый вместе с ней. Исполняется интерпретатором Java, который встраивается практически в каждый современный браузер.
- *Java-апплет* — класс языка Java, встроенный в страницу в виде исполняемого модуля. Загружается с сервера при запросе страницы, содержащей его. Java-апплеты часто применяются в оформлении Web-страниц для придания им интерактивности или для применения мультимедийных эффектов.

- *JPEG (Joint Photographic Experts Group)* — графический формат файлов, применяемый для сохранения изображений с высоким количеством цветов. Формат базируется на сжатии изображения с потерей качества, поэтому может использоваться только для тех изображений, которые не содержат множество мелких, четко очерченных, значащих частей. Чаще всего применяется для сохранения фотоизображений. Степень сжатия и, соответственно, потери качества указывается пользователем при создании файла.
- *news* — протокол для подключения к группам рассылки новостей в Интернете. Может использоваться в качестве префикса (news://) в гиперссылках.
- *ODBC (Open Database Connectivity)* — интерфейс для подключения приложений к базам данных различного формата, созданный для того чтобы авторы программного обеспечения не зависели от конкретной реализации используемой системы управления базами данных.
- *OLE (Object Linking and Embedding)* — технология, позволяющая использовать в рамках одного документа объекты, созданные различными программами.
- *Plug-in* — программное решение, увеличивающее возможности стандартных браузеров. Реализовано, обычно, в виде подключаемых к браузеру программных модулей. Применяется для обработки и отображения информации, не входящей в общепринятые стандарты, но, тем не менее, востребованной удаленными пользователями.
- *PNG (Portable Network Graphics)* — графический файловый формат. Наравне с форматами GIF и JPEG может применяться в оформлении Web-страниц. По своим параметрам весьма похож на GIF-формат. Сжимает изображение без потери качества и имеет возможность использования ограниченного набора цветов. Файлы этого формата обычно несколько "тяжелее" аналогичных GIF-файлов.
- *SSL (Secure Sockets Layer)* — специализированный протокол, предложенный и внедренный Netscape, представляющий собой способ обмена данными между браузерами и Web-серверами с использованием надежных методов шифрования данных. SSL занимает промежуточный уровень между протоколом HTTP и TCP/IP.
- *TCP (Transmission Control Protocol)* — протокол, отвечающий за надежный и подтверждаемый обмен данными между двумя точками сети. TCP разбивает данные на пакеты, снабженные специальным блоком служебной информации, который и позволяет протоколу IP гарантированно доставить их по месту назначения и правильно из пакетов собрать весь массив передаваемой информации.
- *URL (Universal Resource Locator)* — описатель конкретного ресурса в Интернете, указывающий на его местонахождение и позволяющий его отыскать. Включает в себя наименование протокола, по которому происходит доступ (например, HTTP), доменное имя ресурса (www.amazon.com), и путь к самому ресурсу (/content/index.htm).
- *VBScript (Microsoft Visual Basic Scripting Edition)* — подмножество языка Microsoft Visual Basic, предназначенное для написания встроенных в тело

Web-страницы сценариев. Является альтернативой сценариев, написанных на Java.

- *W3C (World Wide Web Consortium)* — ассоциация коммерческих фирм и обучающих организаций, занимающаяся исследованиями и продвижением стандартов в области WWW.
- *WWW (World Wide Web)* — набор взаимосвязанных, ссылающихся друг на друга документов, размещенных на серверах, работающих на HTTP-протоколе, по всему миру. Подобные документы называются Web-страницами. Они написаны на HTML. У каждой страницы есть свой уникальный адрес, называемый URL. Разработку спецификаций WWW произвел Тим Бернерс-Ли в 1989 году, работая в составе Европейской Лаборатории Физики Элементарных Частиц (CERN)
- *Абсолютный URL* — полный Интернет-адрес, включающий в себя протокол, по которому осуществляется доступ к ресурсу, например "http", сетевое имя ресурса и необязательные параметры доступа. Например, http://www.domen.ru/mailform.pl.
- *Активная гиперссылка* — гиперссылка, выделенная в данный момент в браузере. Некоторые браузеры имеют обыкновение подсвечивать такие гиперссылки цветом.
- *Аутентификация* — процесс проверки учетного имени (login) и пароля удаленного пользователя с целью установки их истинности. В момент получения этих данных производится поиск соответствующих значений в учетной базе данных. По результатам проверки осуществляется или запрещается доступ к затребованной информации или сервису.
- *Браузер* — специализированная программа для просмотра Web-страниц и других ресурсов Интернета.
- *Внешняя гиперссылка* — гиперссылка, осуществляющая связь с ресурсом, который находится за пределами сайта, на котором она расположена.
- *Внутренняя гиперссылка* — гиперссылка, указывающая на страницу или файл, находящиеся на этом же сайте.
- *Гиперссылка* — указатель в виде текстовой строки, графического изображения или его части, при нажатии на который происходит переход на страницу или файл в World Wide Web. В Интернете гиперссылка является практически единственным средством навигации и передвижения между сайтами.
- *Гипертекст* — любая текстовая информация, содержащая переходы к другим ресурсам. Сами переходы называются гиперссылками. В Интернете гипертекст является преобладающим способом представления информации.
- *Доменное имя* — имя устройства, имеющего постоянный IP-адрес. Это имя обычно характеризует общее положение узла. Доменные имена различаются по уровням. Доменные имена первого уровня, национальные и международные (ru, com), находятся в ведении специальных организаций, и не могут быть присвоены отдельным устройствам. Доменные имена второго уровня (amazon.com, ozon.ru) могут быть получены

в собственность у этих организаций. Доменные имена третьего уровня предоставляются владельцами имен второго уровня.

- *Закладка* — привязка для локальной гиперссылки, то есть ссылки, действующей в пределах одного документа. Имеет собственное имя, предваряемое знаком решетки (#), по которому и осуществляется переход.
- *Интернет* — всемирная сеть, объединяющая как отдельные компьютеры, так и локальные сети. Обмен информацией происходит по протоколу TCP/IP.
- *интранет* — локальная компьютерная сеть, построенная на тех же принципах, что и Интернет.
- *Клиентское приложение* — программа, выполняемая на машине клиента, которая обменивается данными с удаленным сервером.
- *Межсетевой экран* — специализированное программное обеспечение, установленное на узловой машине, которая и производит доступ к внешнему миру, защищающее внутреннюю корпоративную сеть от вторжения извне.
- *Оборванная гиперссылка* — гиперссылка, указывающая на документ, которого нет, или который в данный момент недоступен.
- *Основная страница* — страница, которая отображается первой в браузере удаленного пользователя, когда тот входит на сайт. Иногда может называться FrontPage.
- *Порт* — один из каналов ввода/вывода информации на компьютере, работающем под управлением протокола TCP/IP. Любой компьютер может иметь множество портов и серверов, но на каждом из портов может действовать только один сервер или служба.
- *Провайдер доступа в Интернет* — фирма, которая предоставляет вашему компьютеру канал и маршрутизацию для выхода в Интернет.
- *Пройденная гиперссылка* — гиперссылка, по которой уже осуществлялся переход. Многие браузеры соответствующим образом меняют ее цветовое оформление.
- *Прокси-сервер* — программное обеспечение, установленное на машине, с которой производится доступ в Интернет. Прокси-сервер во всех исходящих запросах подменяет реальный IP-адрес машины из локальной сети, защищаемой им, на свой собственный IP-адрес, скрывая, таким образом, реальные IP-адреса машин защищаемой сети. Прокси-сервер также обычно хранит в кэше копии наиболее часто запрашиваемых документов и предъявляет их по запросу вместо поиска оригинала в сети, сокращая время ожидания.
- *Путь* — часть URL, характеризующая полное имя папки, в которой находится запрашиваемый файл. Например, в URL http://www.domen.ru/cont/myfolder/page.html путем будет считаться конструкция cont/myfolder/.
- *Сегментированная графика* — графическое изображение, на котором установлено несколько чувствительных областей ("горячие пятна"), причем к каждой области привязывается отдельная гиперссылка.

- *Сервер* — специализированный компьютер в сети, который централизованно предоставляет некоторые службы. В Интернете сервером называют компьютер, на котором действуют программные Web-серверы. Иногда еще их называют узлами или хостами.
- *Скрипт* — особый вид программного кода. Включается в текст Web-страницы в виде текстового исходного кода и интерпретируется системой, установленной на машине удаленного пользователя, запросившего эту страницу. Иногда называется сценарием.
- *Стиль* — набор правил для отображения шрифтов, выравнивания текста, параметров текста, фонового рисунка Web-страницы и других объектов HTML.
- *Тег* — специальная конструкция языка HTML, указывающая на изменение параметров текста или вставку отдельной конструкции в текст.
- *Файл-сервер (файловый сервер)* — компьютер, содержащий файлы для общего пользования и предоставляющий доступ к ним.
- *Фоновый звук* — звуковой клип, связанный с какой-либо Web-страницей. При загрузке страницы в браузер удаленного пользователя одновременно загружается и этот клип. При отображении страницы этот звуковой клип воспроизводится заданное количество раз.
- *Форма* — набор элементов управления, размещенных на Web-странице, которые принимают вводимые посетителем страницы данные и отсылают их на Web-сервер.
- *Фрейм* — область на Web-странице, которая отображается в отдельном окне.
- *Чересстрочный GIF (Interlaced GIF)* — графический файл в формате GIF, отображение которого в браузере происходит постепенно с нарастающим уровнем детализации. То есть в самом начале файла хранятся строки изображения с номерами, кратными восьми, затем с номерами, кратными четырем, и так далее. Соответственно, и загрузка происходит подобным образом. Сначала отображаются строки рисунка, кратные восьми, затем следующие, улучшая детализацию. Данное представление рисунка позволяет увидеть его приблизительное отображение до завершения его полной загрузки.
- *Электронная почта* — система обмена текстовыми сообщениями и присоединенными к ним файлами между компьютерами посредством какой-либо сети, локальной или Интернета.

Предметный указатель