Лучший выбор!

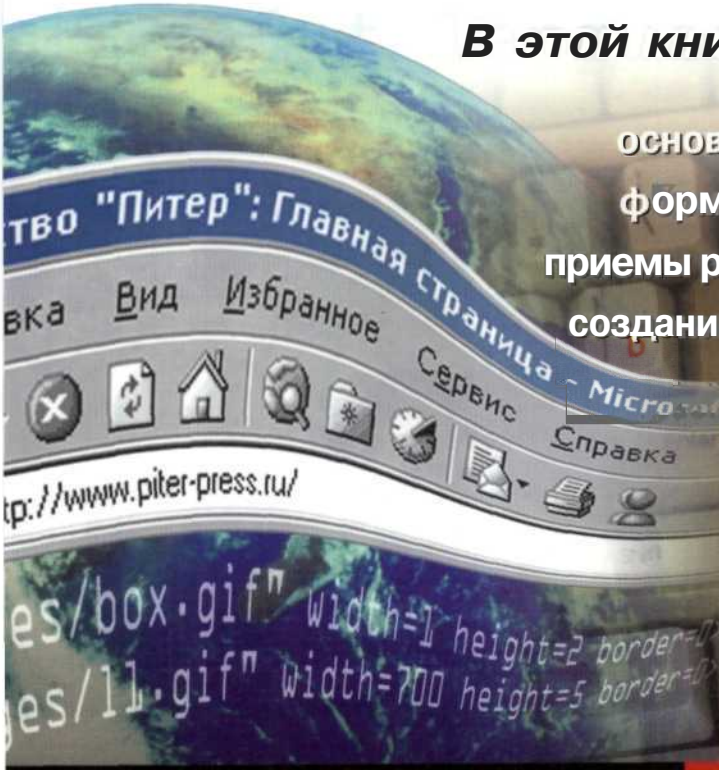
Алексей Гончаров

САМОУЧИТЕЛЬ

HTML

В этой книге вы найдете:

- основные элементы языка
- формы, таблицы, фреймы
- приемы разметки гипертекста
- создание графики для HTML-документов
- справочные таблицы
- обзор редакторов гипертекста
- список полезных Web-узлов
- и многое другое ...



На прилагаемой дискете вы найдете файлы Web-страниц, работа с которыми рассматривается в книге

ПИТЕР

СЕРИЯ

ПИТЕР®

Алексей Гончаров

САМОУЧИТЕЛЬ HTML

ПИТЕР®

Москва • Санкт-Петербург • Нижний Новгород • Воронеж
Ростов-на-Дону • Екатеринбург • Самара
Киев • Харьков • Минск
2002

Алексей Гончаров

Самоучитель HTML

Главный редактор	*Е. Строганова*
Заведующий редакцией	*И. Корнеев*
Руководитель проекта	*И. Корнеев*
Литературный редактор	*А. Жданов*
Художник	*Я. Биржаков*
Корректоры	*Я Лукина, А. Пученкина*
Верстка	*Р. Гришанов*

ББК 32.973-018я7
УДК 681.3.06(075)

Гончаров А.

Г65 Самоучитель HTML. — СПб.: Питер, 2002. — 240 с.: ил.

ISBN 5-272-00072-2

Книга посвящена HTML — популярному языку гипертекстовой разметки документов, позволяющему создавать интерактивные публикации в Интернете. В ней приведено описание самого языка, обсуждаются особенности применения графики на Web-страницах, техника подготовки данных для распространения в Интернете и другие задачи, стоящие перед создателями HTML-документов. Все примеры записаны на прилагаемую к книге дискету.

ISBN 5-272-00072-2

ООО «Питер Принт». 196105, Санкт-Петербург, ул. Благодатная, д. 67в.

Лицензия ИД № 05784 от 07.09.01.

Налоговая льгота - общероссийский классификатор продукции ОК005-93, том 2; 953005 - литература учебная.

Подписано в печать 18.09.02. Формат 70×100 1/16. Усл. п. л. 19,35. Доп. тираж 4500 экз. Заказ № 1304.

Отпечатано с фотоформ в ФГУП «Печатный двор» им. А. М. Горького
Министерства РФ по делам печати, телерадиовещания и средств массовых коммуникаций.
197110, Санкт-Петербург, Чкаловский пр., 15.

Краткое содержание

Предисловие 13

Глава 1 • Введение 14

Глава 2 • Синтаксис HTML 4 35

Глава 3 • Основные элементы HTML версии 4 52

Глава 4 • Объекты и формы 97

Глава 5 ■ Сценарии 115

Глава 6 • Приемы разметки гипертекста 132

Глава 7 • Создание графики 148

Глава 8 • Редакторы гипертекста 184

Приложение А • Шестнадцатеричные числа 209

Приложение Б • Свойства таблиц стилей 210

Приложение В • Состав прилагаемой дискеты 229

Приложение Г • Источники информации в Интернете по тематике книги 231

Алфавитный указатель 232

Содержание

Предисловие **13**
От издательства 13

Глава 1 • Введение **14**
Для кого предназначена эта книга 14
Что есть в этой книге 15
Немного истории 15
HTML как явление нашей жизни 19
Терминология 20
Особенности гипертекста 23
Просмотр Web-страниц 25
Microsoft Internet Explorer 25
Netscape Communicator 30

Глава *2т* Синтаксис HTML 4 ..... **35**
Версии HTML 35
Анатомия Web-страницы 36
<HTML> </html> 38
<HEAD></head> 38
<TITLE> </title> 38
<STYLE></style> 38
<META> 39
<BODY></body> 40
<!-- Комментарий --> 40
<H1></h1> 41
<HR> 41
<A> 42
<BASE> 42
Правила синтаксиса 43
Кодирование символов 45
Использование спецсимволов 46
Типы данных 48
Управление цветом 48

Глава 3 • Основные элементы HTML версии 4 52
Заголовок страницы 52
<TITLE> </title> 52
< STYLE > </style> и <LINK> 52
<META> 53
Стандартные атрибуты 53
Атрибуты событий 55
Форматирование текста 56
<Px/p> 56

 57
<NOBR> </nobr> 57
<PRE> </pre> 58
<CENTER> 58
<B> 58
<BIG></big> 58
<SMALL> </small> 58
<I> </i> 59
<STRIKE> </strike> или <S> </s> 59
<U> </u> 59
 59
 59
<TT></tt> 59
<INS> </ins> и <DEL> </del> 59
<BASEFONT> 60
<FONT> 60
<BDO> </bdo> 61
Табуляция, пробелы, переносы 61
Элементы содержания 62
<EM> </ет> и <DFN> </dfn> 62
<BLOCKQUOTE> </blockquote> 62
<Q> </q> 63
<CITE></cite> 63
<ADDRESS></address> 63
<STRONG> 63
<CODE> </code>, <SAMP> </samp> и <VAR> </var> 63
<KBD></kbd> 64
<ABBR></abbr> 64
<ACRONYM></acronym> 64
Таблицы стилей 64
<STYLE></style> 65
Классы 66
Универсальные классы: атрибут id 66
Каскадные таблицы стилей: элемент <LINK> 67
<DIV> и <SPAN> 68
Выводы 68

Списки 69
<UL> <LI> 69
<OL> <LI> 69
<DL> <DT> <DD> </dl> 71
Гиперссылки 73
<A> 73
<LINK> 76
Таблицы 76
<TABLE> </table> 76
<CAPTION> </caption> 78
Выравнивание данных в ячейках 79
<TR> 79
<TH> 79
<TD> 80
Группы строк: <THEAD>, <TFOOT> и <TBODY> 82
Группы колонок: <COLGROUP> и <COL> 83
Фреймы 84
<FRAMESET> <FRAME> </frameset> 84
<NOFRAMES> </noframes> 87
Организация переходов по фреймам 87
<IFRAME> </iframe> 91
Устаревшие и нестандартные элементы 92
<BGSOUND> 92
<BLINK> </blink> 93
<DIR> </dir> и <MENU> </menu> 93
XMP и LISTING 93
<COMMENT> </comment> 93
<PLAINTEXT> </plaintext> 93
<EMBED> </embed> 94
<NOEMBED> </noembed> 94
<MARQUEE> </marquee> 94
<HPn> </hpn> 96
<BANNER> </banner> 96

Глава 4 • Объекты и формы 97

Общие атрибуты объектов 97
Рисунки и карты 98
<IMG> 98
<MAP> <AREA> </map> 99
Элементы объектов 103
<APPLET> </applet> 103
<OBJECT> </object> 104
<PARAM> 105
Общие атрибуты форм 106

Элементы форм 106
<ISINDEX> 106
<FORM> </form> 107
<INPUT> 108
<LABEL></label> 110
Пример формы 110
<SELECT> <OPTION> </select> 111
<TEXTAREA></textarea> 112
<BUTTON> </button> 113
<FIELDSET> <LEGEND> </legend> </fieldset> 113

Глава 5 • Сценарии 115

Что такое сценарий 115
<SCRIPT></script> 115
<NOSCRIPT> </noscript> 118
Язык JavaScript 118
Синтаксис 118
Управляющие операторы 120
Примеры сценариев 122
Замена изображения 122
Изменение свойств текста 125
Метод setTimeout 127
Управление формами 129
Сценарий для одного элемента 131

Глава 6 • Приемы разметки гипертекста 132

Стиль и традиции 132
Не таблица, а табличка 134
Заголовок и рисунок рядом 134
Мозаичные рисунки 135
Объединение ячеек таблицы 136
Вложенные таблицы 137
Форматирование линии 138
Стихотворный текст 139
Ссылки на файлы мультимедиа 140
Компоновка Web-страниц 141
Собственная Web-страница 143

Глава 7 • Создание графики 148

Форматы графических файлов 148
Создание фона HTML-документа 150
Прозрачность для GIF- и PNG-изображений 152
Программа Gif Construction Set 154
Создание вращающегося значка 162

Компоновка сложного GIF-файла 164
Создание трехмерной вращающейся фигуры 165
Иллюстрация функций Web-страницы 167
Фотоморфизм 168
Преобразование видео в GIF 170
Инструменты рисования в Microsoft Office 2000 171
WordArt 175
Графический редактор MS Image Composer 179
Microsoft GIF Animator 181

Глава 8 ■ Редакторы гипертекста 184

HoTMetaL PRO 5.0 184
Microsoft Word 2000 195
Microsoft FrontPage Express 199
Microsoft FrontPage 2000 201
Netscape Composer 205

Приложение А ■ Шестнадцатеричные числа 209

Приложение Б • Свойства таблиц стилей 210

Единицы измерения 210
Шрифты 210
font-family 210
font-style 211
font-variant 211
font-weight 211
font-size 211
font 211
Форматирование текста 212
text-indent 212
text-align 212
text-decoration 212
text-shadow 212
letter-spacing 213
line-height 213
word-spacing 213
text-transform 214
white-space 214
direction 214
Свойства списков 214
list-style-type 214
list-style-image 215
list-style-position 215
list-style 215

Свойства таблиц 215
display 215
row-span 216
column-span 216
border-collapse 216
border 216
vertical-align 216
table-layout 217
Свойства границ элементов 217
width 217
min-width и max-width 217
height 217
min-height и max-height 217
position 218
top, bottom, right, left 218
margin-top, margin-right, margin-bottom, margin-left 218
margin 218
padding-top, padding-right, padding-bottom, padding-left 218
padding 219
border-top-width, border-right-width, border-bottom-width, border-left-width 219
border-width 219
border-top-color, border-right-color, border-bottom-color, border-left-color 219
border-color 219
border-top-style, border-right-style, border-bottom-style, border-left-style 220
border-style 220
border-top 220
border-bottom 220
border-left 220
border-right 220
border 220
overflow 221
float 221
clear 221
clip 221
visibility 222
z-index 222
Свойства фона и цвета 222
color 222
background-color 222
background-image 222
background-repeat 223
background-attachment 223
background-position 223
background 224

Свойства мультимедиа 224
volume 224
speak 224
speech-rate 224
pause-before, pause-after 224
pause 225
cue-before, cue-after 225
cue 225
play-during 225
azimuth 226
elevation 226
Пользовательский интерфейс 227
cursor 227
color, background-color 227

Приложение В • Состав прилагаемой дискеты 229

Приложение Г • Источники информации в Интернете по тематике книги 231

Алфавитный указатель 232

Предисловие

Появление всепланетной компьютерной сети Интернет (часто именуемой в нашей стране Сетью) внесло существенные изменения в жизнь многих людей. Дело не только в том, что мы получили качественно новое средство связи, реально объединяющее народы. Это само по себе великое достижение. Но Сеть стала, в определенном смысле, образом нашей жизни. Мы привыкаем к Интернет-кафе, журналам об Интернете, анекдотам из Интернета, моде Интернета. Кроме удовольствия от получения новой информации и общения, люди научились извлекать практическую пользу из мира компьютерной связи. В Сети можно рекламировать товары и услуги, продавать их же, искать работу, учиться, заявлять о себе миру (это самый быстрый и дешевый способ!), бронировать места в гостиницах и самолетах и выполнять еще много других полезных дел. Так уж получается, что люди должны осваивать работу в Сети. Но тут есть одна проблема: пользоваться Сетью все-таки несколько сложнее, чем обычным телефоном. Чтобы помочь широкому кругу пользователей освоиться в пока еще новом для них виртуальном мире, предназначена эта книга. Это переиздание моей книги «HTML в примерах», вышедшей в свет в 1997 году. С той поры многое изменилось, и материал был подвергнут существенной переработке. В этом мне очень помогли читательские отзывы. Всем их приславшим я выражаю огромную благодарность! Некоторые интересные разделы я оставил без изменений: ведь не все читали первую книгу.

Если в дальнейшем появятся какие-нибудь дополнения к книге, комментарии или ответы на интересные вопросы читателей, я опубликую их на своей Web-странице (в настоящее время ее адрес http://webcenter.ru/~agonch). На ней есть и электронная почта, при помощи которой можно отослать отзыв или вопрос.

Буду рад получить отзывы об этой книге (или пожелания по тематике следующих) по электронной почте agonch@online.ru.

Желаю удачной работы!

Алексей Гончаров

От издательства

Ваши замечания, предложения, вопросы отправляйте по адресу электронной почты comp@piter-press.ru (издательство «Питер», компьютерная редакция).

Мы будем рады узнать ваше мнение!

Подробную информацию о наших книгах вы найдете на Web-сайте издательства http://www.piter-press.ru.

Глава 1

Введение

Прежде чем начать изучение HTML, нам надо осознать следующее: HTML — это не только язык разметки гипертекста (HyperText Markup Language), а понятие более широкое, включающее в себя Интернет и локальные сети, редакторы, броузеры, разнообразные программные продукты, компакт-диски, обучающие курсы, дизайн и многое другое. Ввести читателя в этот мир можно только одним способом: усадив его за компьютер, подключенный к Сети.

Для кого предназначена эта книга

Войди Интернет в нашу жизнь только как средство коммуникации, по типу телеграфа или телефона, создавать эту книгу не имело бы смысла или, во всяком случае, ее надо было бы писать совсем по-другому. Но глобальная Сеть не только соединила пользователей всего мира, но и утвердилась в виде новых технологий на наших персональных компьютерах. Действительно, пользователь, который приобрел и установил на своей машине Microsoft Office, становится обладателем Интернет-технологии в готовом виде, независимо от того, подключен его компьютер к Сети или нет. Посетитель компьютерной выставки получает в подарок компакт-диск и обнаруживает, что просматривать информацию на нем удобнее всего при помощи Интернет-броузера. Энциклопедия Британика, поставляемая на компакт-дисках, содержит множество материалов в формате HTML.

Иными словами, средства, предназначенные для работы с Сетью, стали использоваться и в других целях, с нею не связанных, а среди программного обеспечения, устанавливаемого на большинство персональных компьютеров, приложения для Интернет заняли свое почетное место.

В результате работа многих пользователей стала иметь большее отношение к Сети, чем они того сами, может быть, желали. Так, одним из способов самовыражения стало создание личных страничек в Интернет. Многие коммерческие фирмы стали использовать Сеть для рекламы и сбыта своей продукции. Люди, занятые поисками работы, получили возможность составлять резюме в формате Web-страницы и размещать эту информацию в Сети. Огромное значение Интернет стал играть для научных, учебных и общественных организаций. Подтверждение тому легко найти, выйдя на просторы киберпространства.

Я адресую эту книгу широкому кругу пользователей. Даже тем, кто не имеет доступа к Интернету. Я убежден, что гипертекстовые технологии — область компьютерного мира, в которой каждый неравнодушный человек может достичь успеха за короткое время. HTML — это своеобразная противоположность сложным языкам программирования (известным только специалистам) и современным прикладным пакетам, па освоение которых требуется затратить (увы!) немало времени.

Что есть в этой книге

Первая глава является вводной. В ней рассказывается об Интернете и о том, как работать с броузерами. Главы 2—3 посвящены собственно HTML: синтаксису языка, описанию элементов, примерам. Вы можете начать работу с создания простой странички, а потом заполнить ее более сложными элементами.

В главах 5—7 обсуждаются различные вопросы создания Web-страниц. Глава 8 содержит описания нескольких гипертекстовых редакторов, которые могут быть полезны разработчику.

Все примеры записаны на прилагаемую к книге дискету, поэтому текст любого HTML-файла можно просмотреть, открыв файл в броузере или текстовом редакторе. Список файлов, которые находятся на дискете, можно найти в приложении В.

Немного истории

Как только на Земле появились первые два компьютера, инженеры задались целью обеспечить связь между ними и передать данные с одного на другой. Разумеется, с технической точки зрения в таком соединении не было ничего невозможного, и, поскольку компьютерная техника развивалась, компьютерные сети (вычислительные машины и линии связи между ними) стали появляться повсеместно.

Идея всепланетной компьютерной сети стала актуальной тогда, когда вычислительные машины перестали быть собственностью только учреждений и ведомств, то есть когда появились персональные компьютеры. Их потребовалось подключить к *глобальной* сети: так же, как это было сделано раньше с телефонами и факс-аппаратами.

В 1994 году, когда я был на стажировке в США, мои американские коллеги постарались приобщить меня к работе с системой, которая тогда еще была недоступна широкому кругу пользователей в измученной кризисами России. Называлась эта система Интернет. Мне выдали книгу Эда Крола «Весь Интернет» (она вышла в свет в 1992 году и еще не успела устареть) и показали, как выходить на просторы киберпространства.

Меня поразили тогда не масштабы Сети, а возможность за считанные минуты связаться с Австралией или Японией, хотя это было как бы само собой разумеющееся удобство. Я всегда понимал, что повсеместное создание компьютерных сетей должно привести к тому, что большинство компьютеров будут подключены к глобальной Сети, а сложности, связанные с протоколами, трафиком и обменом

данными сведутся к минимуму. Иными словами, то, что использование глобальной Сети не будет уделом избранных, соответствующим образом подготовленных программистов и электронщиков, а станет доступным обычным пользователям, занятым решением своих повседневных задач, было для меня очевидным. Поразило же меня другое.

Прежде всего, я совершенно не мог понять, каким образом можно передавать по телефонной линии данные со скоростью 56 000 бит в секунду. И, главное, как после прохождения по реальной линии связи, имеющей довольно узкую полосу пропускания и добавляющей шумы, эти данные оставались пригодными для использования в компьютере, где искажение бинарной информации, строго говоря, вообще не допускается. Не подумайте, что я ничего не знал о помехозащищенном кодировании и коррекции ошибок. Это была моя институтская специальность, так что я вдоволь наслушался лекций и сдал немало экзаменов на эту тему. Но цифры не сходились, и это вызывало недоумение. Уж слишком большой казалась скорость передачи данных. К тому же, как истинный воспитанник застоя, я твердо знал, что нет в мире более ненадежной вещи, чем телефонная связь, тем более связь междугородная. Все это вызывало чувство неудовлетворенности и сильное желание во всем разобраться. Американцы, к которым я обратился с вопросами, только пожимали плечами, что за проблема, мол, и дали мне еще одну книгу (ты инженер — вот и разбирайся, а нам некогда). В конце концов все встало на свои места. Оказалось, что в Америке есть разные телефонные линии: аналоговые, которые предназначены для голосовой связи, и специальные цифровые — для компьютерной техники. А главное, длина линии, используемой для цифровой связи, не превышала 1—2 километров. На такое расстояние высокочастотный сигнал еще можно передать без существенных искажений. А дальше телефонная компания брала все заботы на себя: усиливала сигнал, преобразовывала его и пересылала через оптоволоконный кабель или через спутник. Ну, и наконец, с умом сделанная электроника позволяла добиться максимума возможностей, то есть приблизиться к теоретическому пределу скорости передачи данных для каждой конкретной линии связи.

Ни для кого не секрет, как обстоит дело в России сейчас. Для подключения к Интернету используются обычные российские телефонные линии. У большинства людей просто нет выбора. Многие наши пользователи не могут рассчитывать в ближайшем будущем на его появление. Современные модемы и программное обеспечение построены таким образом, что скорость передачи информации устанавливается автоматически в зависимости от качества линии связи. Если ошибок слишком много, и их не удается исправить, скорость передачи снижается. Это позволяет повысить помехозащищенность всей системы в целом, но время обмена данными, естественно, увеличивается. Мне приходилось наблюдать, как в подобном случае скорость уменьшалась до *двух бит* в секунду. Вы можете прикинуть, сколько времени потребуется для загрузки хотя бы одной картинки. То есть работать связь, конечно, будет, но очень-очень медленно. Впрочем, не везде и не всегда дело обстоит так плохо. В большинстве случаев пропускная способность наших телефонных линий достаточна, чтобы работа с Интернетом доставляла удовольствие. Кроме того, вовсе не обязательно гнаться за рекордными показателями.

Модем со скоростью передачи 14 400 бит в секунду (bps) обеспечивает вполне приемлемые условия работы, а эта скорость — далеко не предел. Я только хочу подчеркнуть, что в нашей телефонии известную роль играет случай: где-то линии хорошие, а где-то никуда не годятся.

Кроме скорости передачи данных меня удивило присутствие в Интернете российских ресурсов. «Путешествуя» по миру, можно было заглянуть и в отдельно взятую страну под названием Россия. У нас одними из первых к Сети подключились физики. Это понятно: World Wide Web изобрели в организации, занимающейся физическими исследованиями — в Женевской лаборатории ядерной физики (CERN). В 1989 году Тим Бенерс-Ли (Tim Berners-Lee) разработал гипертекстовую систему, а в 1990 году там же был создан первый броузер, который так и назывался: WWW. Поэтому российские физические институты были в числе первых организаций, создавших свои серверы. Помимо этого, в крупных городах коммерческие организации создавали собственные сайты (наборы Web-страниц), только вот смотреть их было некому. Из-за рубежа вид был малопривлекательный: вместо кириллицы шла мешанина из каких-то непонятных значков. Эта проблема со шрифтами не разрешена окончательно и по сей день. Но, тем не менее, это был прогресс. Возникал естественный вопрос: сколько еще десятилетий Интернет в России будет оставаться экзотикой? К счастью, судьба оказалась к нам в этот раз благосклонной, и «кончилось все хорошо».

Существует несколько стандартов для организации связи между клиентом (компьютером, принимающим информацию) и сервером (компьютером, который является источником информации). Среда World Wide Web основана на протоколе передачи гипертекста HTTP (HyperText Transfer Protocol), но позволяет использовать протокол передачи файлов FTP (File Transfer Protocol), электронную почту (E-mail) и другие средства.

В любом случае, каждый компьютер, подключенный к Интернету, имеет два уникальных адреса. Первый (IP-адрес), состоящий из четырех групп цифр (например, 123.45.678.9), используется для программной обработки и ничего не говорит пользователю. Этот адрес требуется компьютерам-трассировщикам (почтальонам виртуального мира). Второй вариант адреса несет в себе смысловую информацию. Например, адрес www.piter-press.ru говорит о том, что во Всемирной паутине есть сайт организации Питер Пресс (в данном случае издательства), который размещен в домене русских серверов (ru).

Каждый пользователь Интернета (человек или организация) может получить адрес электронной почты. Адрес строится по схеме:

```
имя_пользователя@имя_домена.имя_домена_верхнего_уровня
```

Из существующих «почтовых» программ можно выделить, например, Eudora Pro или Outlook Express. В некоторых случаях электронная почта может быть даже удобнее телефона, так как не зависит от занятости линий, не требует присутствия абонента в момент получения сообщения, позволяет компенсировать разницу во времени для часовых поясов. Кроме этого, абонент имеет время на обдумывание ответа. При благоприятных условиях с помощью E-mail можно вести диалог

«почти в реальном масштабе времени», с задержкой примерно в десять минут, E-mail — очень демократичный вид связи. Адрес электронной почты Президента США ни для кого (в Америке) не является секретом.

FTP используется для копирования файлов с компьютера на компьютер по Сети. При необходимости, на компьютер-клиент может быть передан список файлов, находящихся в определенной папке компьютера-сервера. Здесь принцип работы с файлами такой же, как в программах Norton Commander или Проводник Windows. Пользователь может переименовывать или удалять файлы, копировать их на свой жесткий диск. Если вам надо «скачать» из Интернета какую-нибудь полезную программу или файл (например, архив), вы не обойдетесь без FTP. Если вы создали свою Web-страницу и хотите поместить ее на сервер, то вам также понадобится протокол передачи файлов.

В 1994 году книга Крола верно отражала суть работы в Сети: в ней последовательно разбирались все средства и протоколы, начиная с FTP и заканчивая World Wide Web. В то время пользователь, желающий поработать, например, с какой-нибудь библиотекой, связывался с ней при помощи протокола telnet (telnet - средство для работы в текстовом интерактивном режиме *с* удаленным компьютером). На время сеанса компьютер клиента превращался в терминал, и на экран выдавались рекомендации по работе с информационно-поисковой системой библиотеки. Они, как правило, представляли собой сведения по несложному, но нестандартному языку, с помощью которого можно было управлять системой и формулировать запросы. «Достаточно» было изучить этот язык и можно было приступать к работе. Некоторые библиотеки предлагали пользователям зарегистрироваться в системе, а другие обеспечивали доступ только после ввода пароля. Найденную информацию можно было загрузить на свой компьютер.

На фоне таких «потрясающих» сервисных возможностей система Gopher казалась намного более удобной. Она предусматривала наличие на каждом сервере собственного набора меню, который позволял получать доступ к необходимой информации, хранящейся на нем и других серверах мира. Пользователю Gopher Сеть представлялась некой гигантской системой меню, по которой можно было «ходить» до бесконечности, в том числе и по кругу, так как все разработчики узлов старались накопить побольше ссылок для заданной предметной области, чтобы обеспечить максимальное удобство своим клиентам. Gopher — симпатичный ушастый зверек, наподобие нашего крота. Предполагалось, что «электронный gopher» должен прокопать всю землю насквозь неограниченное число раз. А работа клиентов заключалась в том, чтобы просмотреть как можно больше ссылок. Самым популярным напутствием в то время было: «Happy gophering!» («Счастливого гоферения!»). Можно добавить, что доступ к графическим данным в Gopher осуществлялся сравнительно редко: в основном работали с текстовыми документами.

Но была в Сети одна новинка. Ее показывали скорее как красивую игрушку, а не как средство повседневной работы. Я имею в виду World Wide Web — Всемирную паутину. Для работы в этой среде использовали броузер Mosaic (Мозаика) - самый новый в то время и самый модный. Его и сейчас еще можно встретить, но он, конечно, безнадежно устарел. О том, что фирмы Netscape и Microsoft будут вести смертельную схватку за право диктовать моду в WWW, тогда *и* подумать

никто не мог. Самым привлекательным в World Wide Web казался доступ к графике. Пользователи загружали в компьютер фотографии земного шара, сделанные со спутников, и сами определяли для себя прогноз погоды. Разумеется, он был точнее, чем прогноз, передаваемый по телевизору! Люди, привыкшие работать с текстами в Сети или даже в текстовом режиме, сразу почувствовали дискомфорт. Mosaic, программа с графическим интерфейсом, работала заметно медленнее, а, кроме того, на загрузку картинок тратилось много времени. Поэтому пользователи не сразу оценили преимущества и перспективность новой системы.

В то время как я знакомился с особенностями Сети, мои американские коллеги организовали в своей лаборатории совещание на тему: «Что лучше на будущее - Gopher или Mosaic?» Аргументы участников той дискуссии были примерно следующие: «Да, Мозаика более современная, но она медленно работает. А к Gopher мы привыкли. И неизвестно, что будет удобнее для клиентов...» На это следовало возражение: «Да, все данные у нас подготовлены для Gopher, но ведь Мозаика более современная. На чем мы будем работать в будущем?» Интересно то, что специалисты рассматривали обе системы, сравнивая их друг с другом. Большинство людей тогда еще не понимали, что одной из этих систем суждено скорое забвение, а другую ждет небывалый взлет. Но это были последние мгновения «старой эры». Уже осенью того же года ставить подобный вопрос было бессмысленно. Пользователей тогда взволновало известие о выходе второй версии броузера Netscape, которую надо было поскорее осваивать, потому что она поддерживала больше элементов языка разметки гипертекста — HTML. Про текстовые режимы все забыли. На повестку дня вышли задачи разработки Web-страниц.

Со временем в Интернете произошли определенные изменения. Электронная почта осталась такой же, как была, но были серьезно улучшены сами почтовые программы. Пользователи были избавлены от рутинных операций и проблем, связанных с декодированием сообщений. Gopher и telnet перестали активно использоваться. FTP прочно удерживает свои позиции как надежное средство загрузки архивных файлов. Броузеры World Wide Web стали основным средством работы с Сетью. Они поддерживают любой из протоколов Интернета из соображений совместимости и удобства. Популярность языка HTML достигла немыслимого уровня, а применение гипертекста уже давно не ограничивается рамками Всемирной паутины.

HTML как явление нашей жизни

HyperText Markup Language (язык разметки гипертекста) давно перестал быть просто языком программирования. Понятие HTML включает в себя различные способы оформления гипертекстовых документов, дизайн, гипертекстовые редакторы, броузеры и многое другое. Человек, изучивший этот язык, обретает возможность делать сложные вещи простыми способами и, главное, быстро, что в компьютерном мире значит не так уж и мало.

Гипертекст как нельзя лучше подходит для включения элементов мультимедиа в традиционные документы. Практически, именно благодаря развитию гипертекста, большинство пользователей получило возможность создавать собственные мультимедийные продукты и распространять их на компакт-дисках. Такие

информационные системы, выполненные в виде наборов HTML-страниц, не требуют разработки специальных программных средств, так как все необходимые инструменты для работы с данными (Web-броузеры) стали частью стандартного программного обеспечения большинства персональных компьютеров. При таком подходе от пользователя требуется выполнить только ту работу, которая непосредственно относится к тематике разрабатываемого продукта: подготовить тексты, нарисовать рисунки, создать HTML-страницы и продумать связи между ними. Неохваченными остались только технические проблемы, такие, как получение видеоизображений, качественного звука, тиражирование дисков и т, д.

HTML является основой моды в Интернете. Все, что обсуждается на страницах виртуальных журналов в Сети, в телевизионных передачах, посвященных Интернету, так или иначе связано с языком разметки гипертекста: красивые рисунки, интерактивность Web-страниц, битва двух фирм-производителей броузеров (Microsoft и Netscape), бизнес, игры, и многое, многое другое. Пользователь, игнорирующий Интернет, добровольно обрекает себя на отлучение от целого мира.

HTML как основа создания Web-страниц имеет прямое отношение и к новому направлению изобразительного искусства — Web-дизайну. Художнику в Интернете недостаточно просто нарисовать красивые картинки, оригинальный логотип, создать новый фирменный стиль. Он должен еще разместить все это в Сети, продумать связи между Web-страницами, чтобы все двигалось, откликалось на действия пользователя, поражало воображение неискушенных клиентов, а у приверженцев Сети вызывало желание создать что-нибудь свое, оригинальное в этой области.

Терминология

Однажды я дал моим немецким коллегам-программистам поработать на своем ноутбуке, на котором была установлена русифицированная версия Windows. Один из этих программистов учился в нашей стране и хорошо знал русский язык.

- Как здорово переведена операционная система! — сказал он мне потом.

— Почему? — удивился я.

- Ну как же, ведь у вас «file» называется файлом, — ответил он.

Я потом часто вспоминал этот разговор, так как тема для технического писателя была весьма интересная. Действительно, англоязычное программное обеспечение попадало в нашу страну самыми разными путями, и в вычислительной технике сама собой складывалась уникальная лингвистическая ситуация. Существует огромное количество терминов, которые соответствуют по произношению английским словам. Программисты и электронщики разговаривают на своем, хорошо понятном им языке. Заимствованные слова имели только одно значение, и это было удобно. Такая система прижилась и стала восприниматься как сама собой разумеющаяся. Трудно было представить, что в других странах все не так. Любые попытки вводить отечественную терминологию глохли на корню и даже подвергались насмешкам. Самый яркий пример — названия клавиш. Нам непривычно видеть на клавиатуре УДЛ вместо Del или ВСТ вместо Ins. Клавиатура так и осталась нерусифицированной. Подобные феномены объясняются прежде всего тем, что

до недавнего времени никто серьезно не занимался локализацией фирменного программного обеспечения. Теперь положение изменилось. Появилось много программных продуктов, использующих русский язык. И вместе с тем возникли чисто русские термины: «папка», «карман», «ярлык», «пиктограмма»... Я не хочу сказать, что это плохо. Просто это пока непривычно. А в некоторых случаях теряется однозначность термина. Вдобавок уже существуют двойные термины: Internet и Интернет. А может, пора уже писать «Виндоус», «Ворд», «Корел Дро»? Жизнь покажет.

Сфере гипертекстовых технологий в этом смысле повезло намного больше. Слава Богу, никто не называет броузер «просматривателем» (хотя иногда называют обозревателем), а для «уорлд вайд веб» использовали красивое русское название «Всемирная паутина». Видимо, в каждом случае необходимо свое решение: использовать русское слово или заимствовать английское. Гипертекст породил много специальных терминов. Часть их я хочу предложить вашему вниманию.

- **Элемент** (element) — конструкция языка HTML. Это контейнер, содержащий данные и позволяющий отформатировать их определенным образом. Любая Web-страница представляет собой набор элементов. Одна из основных идей гипертекста — возможность вложения элементов.
- Тег (tag) — начальный или конечный маркеры элемента. Теги определяют границы действия элементов и отделяют элементы друг от друга. В тексте Web-страницы теги заключаются в угловые скобки, а конечный тег всегда снабжается косой чертой.
- **Атрибут** (attribute) — параметр или свойство элемента. Это, по сути, переменная, которая имеет стандартное имя и которой может присваиваться определенный набор значений: стандартных или произвольных. Предполагается, что символьные значения атрибутов заключаются в прямые кавычки, но некоторые броузеры позволяют не использовать кавычки. Это объясняется тем, что тип атрибута всегда известен заранее. Атрибуты располагаются внутри начального тега и отделяются друг от друга пробелами.
- **Гиперссылка** — фрагмент текста, который является указателем на другой файл или объект. Гиперссылки необходимы для того, чтобы обеспечить возможность перехода от одного документа к другому.
- **Фрейм** (frame) — этот термин имеет два значения. Первое — область документа со своими полосами прокрутки. Второе значение — одиночное изображение в сложном (анимационном) графическом файле (по аналогии с кадром кинофильма).

ПРИМЕЧАНИЕ

Вместо термина «фрейм» в специальной литературе и локализованных программных продуктах иногда можно встретить термин «кадр» или «рамка».

- **HTML-файл или HTML-страница** — документ, созданный в виде гипертекста на основе языка HTML. Такие файлы имеют, как правило, расширения htm или html. В гипертекстовых редакторах и броузерах эти файлы имеют общее название «документ».

- Апплет (applet) — программа, передаваемая на компьютер клиента в виде отдельного файла и запускаемая при просмотре Web-страницы.
- **Скрипт или сценарий** (script) — программа, включенная в состав Web-страницы для расширения ее возможностей. Броузер Internet Explorer в определенных ситуациях выводит сообщение: «Разрешить выполнение сценариев на странице?» В этом случае имеются в виду скрипты.
- **Расширение** (extension) — элемент, не входящий в спецификацию языка, но использующийся, обеспечивая возможность создания нового интересного эффекта форматирования.
- CGI (Common Gateway Interface) — общее название для программ, которые, работая на сервере, позволяют расширить возможности Web-страниц. Например, без таких программ невозможно создание интерактивных страниц.
- **Программный код** или просто код — аналог понятия «текст программы».
- **Код HTML** — гипертекстовый документ в своем первоначальном виде, когда видны все элементы и атрибуты.
- **World Wide Web, WWW** или просто Web — Всемирная паутина, распределенная система доступа к гипертекстовым документам, существующая в Интернете. HTML является основным языком для создания документов в WWW. Изучая его, мы, фактически, изучаем часть этой системы, хотя область применения языка намного шире.
- **Web-страница** — документ (файл), подготовленный в формате гипертекста и размещенный в World Wide Web.
- **Сайт** (site) — набор Web-страниц, принадлежащих одному владельцу.
- **Броузер** (browser) — программа для просмотра Web-страниц.
- **Пользовательский агент** (user agent) — броузер или другая программа, работающая на компьютере-клиенте.
- **Загрузка** (downloading) — копирование файлов с сервера на компьютер-клиент.
- URL (Uniform Resource Locator) или **универсальный указатель ресурса** - адрес некоторого объекта в Интернете. Типичный URL для WWW имеет вид:

  `http://www.название.домен/имя` файла

Здесь `название` — это часть адреса, который часто употребляется для обозначения владельца сайта, а домен — обозначение крупного «раздела» Интернета: страны, области деятельности и т. д. URL используются для того, чтобы указать конкретную Web-страницу или графический файл в гиперссылках, а также везде, где требуется однозначно определить месторасположение Web-страницы или файла.

- **Базовый URL** — часть адреса, которая является общей для всех ссылок текущей Web-страницы.
- **Базовый цвет** — каждый цветовой оттенок на экране монитора получается соединением трех базовых цветов: красного, зеленого и синего.

- Цветовой канал — интенсивность красного, зеленого или синего цвета на экране монитора. Цвет каждого пиксела определяется как комбинация этих трех величин.
- Гамма-коррекция — создание нелинейной зависимости между реальной интенсивностью базового цвета и полученной яркостью на экране монитора. Изображения, полученные без гамма-коррекции, оказываются более темными, так как монитор воспроизводит различные градации яркости нелинейно по отношению к их числовому выражению. Изменение параметра «гамма» часто используют в графических редакторах при работе с изображением. На рис. 1.1 показан пример зависимости между реальной и необходимой яркостью.

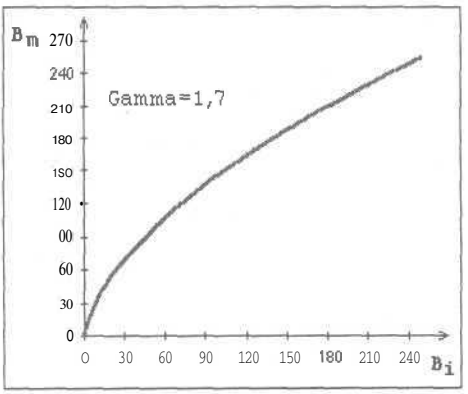


Рис. 1.1. Пример зависимости между исходной (B_i) и необходимой (B_m) величинами яркости пиксела

Особенности гипертекста

Если задаться вопросом, какова основная особенность гипертекстового документа, можно наверняка утверждать, что это — способность получать сложные эффекты форматирования простыми и наглядными методами. Давайте сравним гипертекстовый документ, например, с файлом в формате MS Word. В том и другом случаях можно использовать одни и те же приемы форматирования: выбор шрифта, курсив, выравнивание, вставку таблиц, рисунков и т. д. Но в документе Word механизм форматирования скрыт от пользователя, работать с файлом можно только в самом редакторе или программе, поддерживающей его формат. Подобных продуктов сравнительно немного, и большинство из них разработаны фирмой Microsoft. С гипертекстом дело обстоит иначе. Такой документ можно открыть в любом текстовом редакторе и увидеть, где и каким образом отформатирован текст. Разумеется, просмотреть или распечатать документ в отформатированном виде возможно тоже только в специализированном приложении — гипертекстовом редакторе или броузере.

Открытость структуры гипертекстовых документов позволяет фирмам-разработчикам создавать самые разные программные продукты, а пользователь может выбрать себе подходящую программу.

Разработчик HTML-документа может выбрать способ работы с ним. Теоретически с гипертекстом можно работать даже на уровне MS-DOS в любом редакторе, открывающем ASCII-файлы. Правда, это требует от пользователя обязательного знания большинства элементов HTML. Можно использовать для создания гипертекста и броузер. Любая из этих программ имеет режим редактирования Web-страницы в режиме «источника». Для этого может подключаться один из установленных на компьютере текстовых редакторов. Броузеры имеют и встроенные редакторы гипертекста. Наконец, существуют гипертекстовые редакторы, которые используются только для разработки Web-страниц и создания на последних всевозможных визуальных и звуковых эффектов.

Способ создания гипертекста обеспечивает его абсолютную платформенную независимость. Создавая Web-страницы на компьютере, который работает под управлением Windows, вы можете не сомневаться, что администратор сервера сможет использовать ваши файлы на компьютере, работающем под управлением UNIX или другой операционной системы.

Одной из основных особенностей HTML является принцип, по которому не только допускается вложение одних элементов в другие, но и декларируется необходимость такого вложения. Это отличает HTML от многих других языков, в которых теоретически можно написать код без вложенных конструкций. В данном случае, это невозможно изначально. Каждый элемент допускает непосредственное вложение только ряда элементов, которые, в свою очередь, допускают вложение других, разрешенных для них, и т. д. Таким способом формируется не только общая структура гипертекста, но и создаются разнообразные визуальные эффекты.

Все элементы языка можно условно разделить на три группы. К первой относятся элементы, которые создают структуру гипертекстового документа. Использование таких элементов — необходимая формальность, которой нельзя пренебрегать. Ко второй группе можно отнести элементы, создающие эффекты форматирования. Их использование диктуется конкретными требованиями к документу, фантазией и компетенцией разработчика, К третьей группе относятся элементы, которые позволяют управлять программными средствами, установленными и работающими на компьютере-клиенте. Часто такие элементы создаются автоматически, когда разработчик использует для вставки некоторого объекта в документ гипертекстовый редактор или подобную программу.

Несмотря на то, что спецификация HTML является стандартом, этот язык дополняется новыми элементами (расширениями). Поэтому некоторые Web-страницы удобнее просматривать при помощи определенных броузеров. Расширения создаются только известными фирмами, которые разрабатывают программное обеспечение для WWW, а рядовые пользователи могут совершенствовать свои Web-страницы при помощи программирования. Апплеты позволяют снять ограничения HTML и дают простор фантазии разработчика.

В каждой главе этой книги обсуждаются свои особенности применения HTML, и то, о чем здесь говорится вкратце, далее будет рассмотрено более подробно.

Просмотр Web-страниц

Прежде чем начинать создание собственных HTML-документов, полезно познакомиться с программами, необходимыми для просмотра таких документов — *броузерами*. Последние используются не только как средство просмотра, но и как почтовые программы, а также как средство загрузки файлов посредством FTP. Нам потребуется изучить две основные функции броузеров: просмотр Web-страниц и редактирование их содержимого (элементов HTML).

Несмотря на то, что в мире создано немало программ для просмотра HTML-документов и даже специализированных редакторов, выбор пользователя всегда ограничен. Это объясняется прежде всего тем, что в гипертекстовую технологию постоянно вносятся дополнения, и программы перестают удовлетворять последним требованиям. Сейчас уже почти не осталось приверженцев броузера Mosaic, хотя сравнительно недавно это был очень популярный продукт. И до сих пор есть еще фирмы, которые пытаются продолжать его развитие.

Кто из специалистов по Интернету не слышал таких названий, как Cello, Global Network Navigator или Netcom's NetCruiser! Однако в настоящее время пользователи выбирают продукцию одной из двух фирм: Microsoft и Netscape. Именно им удалось найти решения, завоевавшие всеобщее признание. А тот факт, что обе фирмы постоянно изыскивают возможности реализовывать в своих программах поддержку нововведений конкурента, является причиной быстрого развития гипертекста. Пользователи в этой ситуации только выигрывают, получая в свое распоряжение все новые и новые, самые современные программные продукты.

И все же есть одна область, где даже малоизвестные и примитивные броузеры применяются с успехом. Я имею в виду компакт-диски, на которых размещены гипертекстовые файлы. Здесь разработчик заранее знает, какие документы придется воспроизводить (что напрочь уничтожает проблемы совместимости, такие актуальные для «узкоспециализированных» броузеров!), а необходимость лицензирования приводит к использованию продукции никому не известных доселе фирм. Потребитель, в свою очередь, освобожден от необходимости устанавливать на своем компьютере дополнительное программное обеспечение, так как все, что ему необходимо, записано на диск.

Мы познакомимся с двумя наиболее популярными броузерами, которые обеспечивают корректный просмотр гипертекста в подавляющем большинстве случаев.

Microsoft Internet Explorer

Этот броузер фирмы Microsoft удобен в первую очередь тем, что является полностью русифицированным. Вы можете использовать версию 4 или 5. Microsoft Internet Explorer (MSIE) настолько тесно интегрирован с Windows, что его трудно назвать автономной программой. Скорее, это один из компонентов операционной системы.

Работа с броузером может начинаться с подключения к Интернету или вестись автономно, если все необходимые файлы находятся на локальном жестком диске

(то есть диске своего компьютера). В поле Адрес на панели инструментов указывается URL. Он может совпадать с IP-адресом, а может содержать и дополнительную информацию, необходимую для идентификации определенного документа. Адрес предваряется кодом, который определяет, с какой подсистемой Интернета мы хотим работать. Например, если мы ищем компанию Microsoft, то адрес долженбытьтаким:http://www.microsoft.com.

Код http указывает на то, что программа должна работать с системой гипертекстовых документов и использовать соответствующий протокол (HyperText Transfer Protocol). Но могут быть и другие варианты адреса. Например, адреса с кодом ftp указывают на архивы файлов, доступных по FTP, код mailto обозначает адреса электронной почты, а код file — файлы на собственном компьютере. Для тренировки попробуйте, например, загрузить на свой компьютер какие-нибудь файлы с адресаftp://ftp.microsoft.com.

После указания URL броузер загружает данные из Интернета и демонстрирует нам гипертекстовый документ, находящийся по заданному адресу (рис. 1.2).

Рис. 1.2. Окно MS Internet Explorer

Найдя какую-нибудь организацию, мы вначале увидим ее «домашнюю» страницу (home page). На ней легко обнаружить подчеркнутые надписи. Это ссылки на другие страницы. Весь комплекс страниц (сайт) очень велик, и его нельзя увидеть целиком. Гиперссылки позволяют передвигаться по страницам или частям одной большой страницы. Гиперссылкой может быть не только надпись, но и значок,

рисунок или часть рисунка. О том, что это гиперссылка, можно догадаться по изменению внешнего вида указателя мыши. Передвижение по сайту можно уподобить хождению по лабиринту, поэтому броузер снабжен кнопками Вперед и Назад, которые позволяют вернуться к пройденным страницам. Каждой странице соответствует один или несколько файлов, которые броузер помещает в папку временного хранения (кэш). Поэтому, когда вы хотите вернуться к уже загруженной странице, программа считывает ее не из Интернета, а с жесткого диска. Время доступа к документу значительно сокращается.

Бывает и так, что с загрузкой страницы возникают проблемы. Тогда пользователь может прервать загрузку щелчком па кнопке Остановить.

Применять эту кнопку следует в нескольких случаях. В первую очередь, когда страница еще загружается, а вам уже ясно, что она вам не понадобится. Кнопка бывает полезна и в том случае, когда на странице много графики и нет времени ждать, пока вся она загрузится. Ну, и наконец, бывают ситуации, когда броузер «зацикливается», то есть пытается открыть несуществующий документ. В этом случае также выручает эта кнопка.

И наоборот, загрузку текущей страницы можно повторить, щелкнув на кнопке Обновить. При работе с Интернетом она используется редко, а для разработчика гипертекстовых документов — это важный инструмент. Если вы что-то изменили в своем файле, то увидеть, как он теперь будет выглядеть, можно только перезагрузив документ.

Ниже представлены другие важные кнопки броузера.

Переход на страницу, которая была открыта ранее. Это очень удобное средство передвижения по документам. Движение вперед обеспечивает гиперссылка, а возможность вернуться назад присутствует не во всяком документе.

Движение по цепочке просмотренных документов вперед, если были выполнены переходы назад.

«Домашняя страница» (home page) — понятие условное. Каждая фирма, работающая в Сети, имеет свою собственную домашнюю страницу. Вы можете использовать в качестве таковой свою страницу, страницу своего провайдера, производителя программы или любую другую. Адрес домашней страницы указывается в качестве параметра программы и может быть изменен (команда Сервис ▸ Свойства обозревателя).

Поиск в Интернете осуществляется путем перехода к странице какого-либо поискового сервера. В настоящее время существует множество поисковых серверов, и вы можете выбрать любой, а адрес страницы поиска также задается в качестве параметра.

Каталог ваших любимых или необходимых вам Web-страниц. Используя эту кнопку, вы можете добавить текущую страницу в список избранных страниц или дать команду броузеру загрузить любую из страниц, перечисленных в списке.

Вывод на печать текущего документа. На Web-страницах иногда присутствуют детали, которые не могут быть распечатаны, но в целом броузер производит форматирование для печати вполне удовлетворительно.

Изменение размера шрифта на текущей странице. Это очень удобный инструмент. Не задавая лишних вопросов, программа увеличивает или уменьшает размер букв. Таким образом решается проблема в двух случаях: когда текст на странице слишком мелкий или когда он, наоборот, неоправданно крупный.

Некоторые Web-страницы устроены таким образом, что просмотр их компонентов вызывает открытие еще одного окна броузера. Поэтому, поработав в Сети, вы можете обнаружить, что у вас открыты два или три окна MSIE. За этим нужно следить при помощи значков на панели задач, и закрывать ненужные окна, чтобы не перегружать компьютер.

Web-страницы часто оборудованы средствами электронной почты. Щелкнув на определенной ссылке, вы можете активизировать диалог для создания почтового сообщения. Если вы его напишете и отправите, то программа все сделает автоматически: вам даже не надо будет думать об адресе получателя. Другим средством «обратной связи» являются *формы*. Заполнив соответствующие поля и щелкнув на кнопке «отправить» (которая может иметь в форме название Go, OK, Send, Submit или другое), можно переслать на сервер необходимую информацию. Так, например, происходит заказ товаров в Интернет-магазинах или заполнение анкет во время виртуальных опросов.

С Web-страницами и их компонентами (например, рисунками) можно поступать как с обычными документами: сохранять на диске под определенными именами или распечатывать. Чтобы открыть локальный файл в Internet Explorer, достаточно ввести в поле Адрес ссылку на файл. Простейший вариант — указать путь и имя:

```
C:\Моя страница\Start.htm
```

Можно записать ссылку по всем правилам, как для работы в Сети:

```
file:///C:/Моя страница/Start.htm
```

А сам броузер MSIE «предпочитает» такой формат:

```
file://C:\Моя страница\Start.htm
```

Все три способа записи адреса легко распознаются броузером: программа составлена так, что выбирает из ссылки только полезную информацию, а сколько косых черточек идут подряд и в какую сторону они наклонены — это детали, которыми современное приложение пренебрегает.

Чтобы открыть файл в режиме диалога, надо использовать команду Файл ▸ Открыть. При этом активизируется окно, показанное на рис. 1.3. В нем доступ к папкам осуществляется после щелчка на кнопке Обзор.

Наконец, чтобы совсем не тратить время на поиск файла, пользователи Windows могут создать ярлык Web-страницы на рабочем столе. Для этого надо выполнить следующие действия.

1. Откройте Проводник Windows, а в нем папку с необходимым файлом.
2. Выберите файл (Start.htm) и щелкните на нем правой кнопкой мыши.

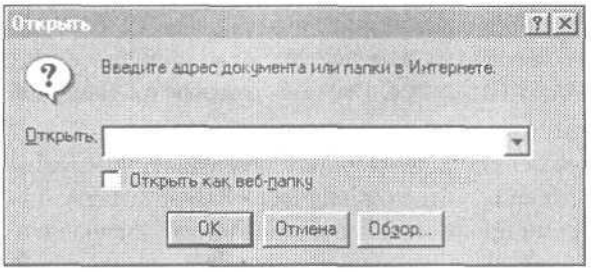


Рис. 1.3. Окно для открытия HTML-документа

3. Выберите в контекстном меню команду Создать ярлык.
4. Перетащите появившийся в папке ярлык на поверхность рабочего стола.
5. Чтобы запустить броузер и сразу же открыть в нем документ, щелкните на этом ярлыке один или два раза (в зависимости от настроек рабочего стола).

Для работы в локальном режиме этот способ запуска броузера — наилучший, так как позволяет избежать потери времени на попытку броузера подключиться к Интернету.

При помощи Web-страниц с ярлыками можно существенно облегчить себе пользование Интернетом или локальными ресурсами. Собранные на странице ссылки по определенной тематике превращают ее в аналог папки Избранное. Иными словами, можно легко сделать набор справочников со ссылками на интересные Web-страницы.

Можно иметь и несколько папок для избранных страниц. Точнее, несколько папок внутри папки Избранное. Для этого надо выбрать команду Избранное ▸ Упорядочить избранное, а затем в открывшемся окне щелкнуть на кнопке Создать папку. После того как вы дадите новой папке имя, можете перетаскивать в нее ярлыки из основной папки. В дальнейшем, при просмотре содержимого папки Избранное, созданные внутри нее папки будут раскрываться после щелчка на них мышью. Папка Избранное хранит не сами документы, а только ссылки на них. Если страница расположена не локально, то для ее просмотра потребуется подключение к сети. В некоторых случаях возникает необходимость иметь копии реальных документов на своем компьютере. Тогда пользователь должен сохранить выбранные файлы в одной из своих папок. На кэш в этом случае полагаться нельзя, так как его содержимое регулярно обновляется.

Существует несколько команд для сохранения просмотренных документов. Файл > Сохранить как позволяет скопировать текущую Web-страницу в папку, выбранную пользователем. При этом копируется не только сам HTML-документ, но и рисунки. Для сохранения одного рисунка надо щелкнуть на нем правой кнопкой мыши и выбрать команду Сохранить рисунок как в контекстном меню.

Средства просмотра HTML-кода дают возможность сохранить документ и даже модифицировать его. Команда Файл ▸ Править в Microsoft Front Page позволяет подключить гипертекстовый редактор, который выбран в окне свойств обозревателя. Команда Вид ▸ В виде HTML открывает выбранную страницу в Блокноте.

Редактирование страниц в режиме источника очень удобно, так как можно сразу увидеть результат внесенных изменений: после модификации страницы ее надо сохранить в текстовом редакторе, а затем в броузере щелкнуть на кнопке Обновить.

Часто при просмотре страниц возникает проблема со шрифтами. Если шрифт на странице слишком мелкий или, наоборот, неоправданно крупный, вид документа легко исправить. Используйте команду Вид ▸ Размер шрифта, чтобы изменить размер шрифта на время просмотра. Буквы можно как уменьшить, так и увеличить, При этом на странице сохраняются пропорции различных шрифтов и заголовков.

Основные инструменты для настройки броузера собраны в окне Свойства обозревателя, которое активизируется одноименной командой меню Сервис. На просмотр HTML-страниц влияют несколько параметров. На вкладке Общие имеются кнопки Цвета, Шрифты, Языки и Оформление, с помощью которых можно установить режимы воспроизведения страниц. В броузере назначаются для использования по умолчанию два шрифта: пропорциональный и моноширинный. Два разных типа шрифта необходимы для прорисовки данных из соответствующих элементов HTML. Какие именно шрифты надо применять, выбирает пользователь. Важен также и выбор кодировки: для русскоязычных документов применяют Windows-1251 или КОИ-8. Можно выбрать и цвета, которыми будут раскрашены текст, пройденные и неиспользованные гиперссылки.

На этой вкладке можно получить доступ к параметрам настройки кэша. Эти параметры позволяют просмотреть содержимое папок кэша (кнопка Настройка в группе Временные файлы Интернета) и скопировать оттуда необходимые файлы, которые были загружены из Сети. При необходимости файлы временного хранения можно стереть, освободив место на диске.

На вкладке Дополнительно можно установить режим, при котором на странице не будут воспроизводиться рисунки или мультимедийные компоненты.

На вкладке Программы можно выбрать приложения, которые будут использоваться по умолчанию совместно с броузером (HTML-редактор, программа электронной почты и др.).

Netscape Communicator

Этот комплект программ (в настоящее время русифицирована его версия 4.06) привлекает к себе внимание многими положительными свойствами. Его популярность во всем мире весьма высока, и не только потому, что пользователи привыкли работать с его более ранними версиями. Интересными, например, представляются комментарии в строке состояния, которые дают представление о процессе загрузки страницы. Удачно, на мой взгляд, организован доступ к параметрам. Броузер называется Netscape Navigator, а встроенный редактор гипертекста — Netscape Composer. В броузере реализована возможность создания закладок, что значительно облегчает поиск страниц. Несомненным удобством является легкость переключения между режимами броузера и редактора. Окно броузера показано на рис. 1.4.

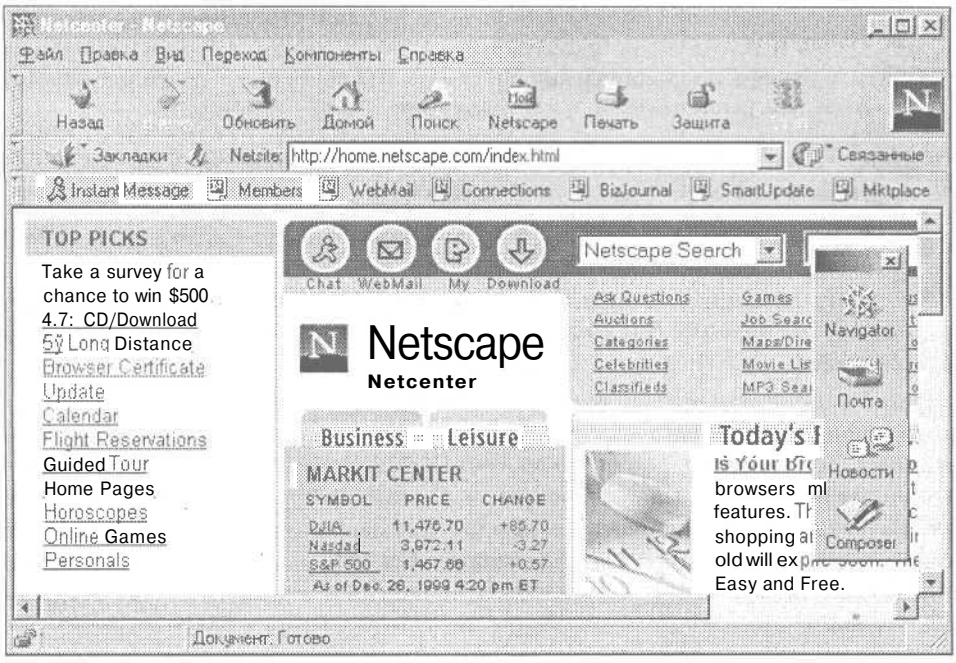


Рис. 1.4. Окно броузера Netscape Navigator

Рассмотрим основные инструменты программы. Они, в основном, такие же, как и в MSIE.

Возврат к странице, которая была открыта раньше текущей.

Переход к странице, которая была открыта позже текущей.

Повторение загрузки страницы.

По умолчанию эта кнопка обеспечивает переход на домашнюю страницу (home page) Netscape, но пользователь может настроить программу на любую страницу, используя окно параметров (команда Правка▸Настройки).

Активизация поиска в Интернете.

Кнопка для перехода на страницу любителей Netscape. «Мой Netscape» — виртуальный клуб, в котором можно сделать свою страничку и получить адрес электронной почты.

Традиционная кнопка для печати текущей страницы. А в меню Файл как и положено современной программе, имеются команды Настройки Страницы, Предварительный Просмотр и Печать, которые позволяют настроить параметры и выполнить просмотр документа перед печатью.

Останов загрузки текущей страницы. Загрузка может быть продолжена после щелчка на кнопке Обновить.

Раньше кнопка с логотипом фирмы-производителя традиционно использовалась для инициирования загрузки страницы. Теперь такая команда практически не нужна, и в данном броузере щелчок на логотипе приводит на домашнюю страницу Netscape. Как правило, логотип снабжается анимацией которая действует в течение загрузки текущей страницы.

В этом броузере, как и в предыдущем, есть поле Адрес. Но в отличие от MSIE, в раскрывающемся списке этого поля можно выбрать все недавно использовавшиеся адреса, а не только те, которые введены в строке вручную. Похожие функции выполняет *журнал* (команда Компоненты ▸ Журнал). В журнале приводится список использованных адресов, а щелчок на адресе позволяет выполнить загрузку. Еще один сервисный инструмент — файл закладок. Он является аналогом папки Избранное в Windows. Для доступа к *закладкам* на панели инструментов есть специальная кнопка.

На формат отображаемой страницы влияют команды меню Вид. С его помощью устанавливается размер шрифта, кодировка страницы и другие параметры. Есть например, команда Остановить анимацию, то есть смену изображений в сложных рисунках формата GIF. Для возобновления анимации необходимо перезагрузить страницу (щелкнуть на кнопке Обновить).

По аналогии с «пользователями» Windows, Netscape Communicator позволяет создавать своих «пользователей», то есть варианты настроек программы (их еще называют профили или конфигурации). Перед началом работы необходимо выбрать одну из этих конфигураций. Для каждого «пользователя», создается собственный файл закладок, журнал и кэш. Журнал связан *с* файлами кэша, так как при помощи журнала открываются страницы, которые когда-то были загружены из Интернета. Как журнал, так и кэш при необходимости можно очистить, но делается это разными путями. Журнал очищается кнопкой в окне настроек (команда Правка ▸ Настройки). Обратите внимание (рис. 1.5), что необходимая панель настроек выбирается при помощи дерева в левой части окна диалога. Для очистки кэша необходимо выбрать ветвь Кэш в разделе Дополнительные настройки.

В Netscape Navigator (NN) можно выбрать те же параметры просмотра, что и в MSIE шрифты, используемые по умолчанию, цвета просмотренных и непросмотренных ссылок. Можно также выбрать определенный фон для всех просматриваемых страниц или вообще отказаться от фона. Это особенно удобно при разработке собственных страниц.

Броузеры позволяют загружать страницы без рисунков. Это, конечно, снижает информативность документов, но ускоряет работу. Для отказа от отображения рисунков в NN надо сбросить флажок Автоматическая загрузка изображений в категории параметров Дополнительные настройки.

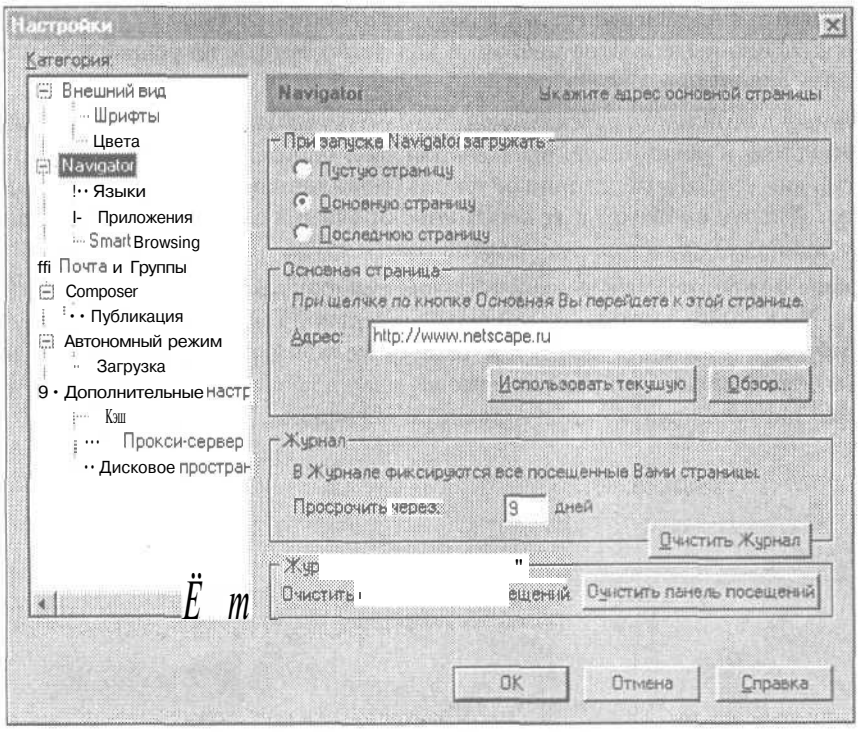


Рис. 1.5. Окно настроек Netscape Navigator

Если флажок сброшен, становится доступной команда Вид > Показать рисунки и кнопка Рисунки, которые позволяют загружать графические компоненты страницы по разрешению пользователя.

Диалог для открытия файлов, расположенных локально, в MSIE и NN имеет особенность: используются два окна диалога, а не одно, как в большинстве программ. Только во втором окне пользователь получает доступ к традиционному списку дисков папок и файлов. Надо заметить, что для задания адреса необходимой страницы, расположенной на жестком диске, достаточно ввести путь к HTML-файлу в поле Адрес.

В меню Файл есть команда Сохранить как, которая используется для копирования текущей страницы в папку, выбранную пользователем. При этом можно выбрать формат файла: HTML или текстовый. В отличие от MSIE, который позволяет сохранить Web-страницу вместе с рисунками, NN копирует только основной файл страницы. Сохранение файла страницы требуется только в определенных случаях, так как вся страница автоматически записывается в кэш. Текстовый формат удобен, например, в тех случаях, когда надо распечатать только текст страницы, не отягощая его элементами оформления и рисунками. Зато если вы откроете страницу в редакторе Netscape Composer, то сможете сохранить ее в новой папке целиком (с рисунками).

Для просмотра элементов HTML в NN служит команда Вид ▸ Код страницы. Эле менты красиво выделяются цветом, и код легко читать, но редактировать его этом случае нельзя.

Поскольку в этой книге рассказывается о самых разных программах для работы гипертекстом, я рекомендую выбрать для себя определенный набор программ решить, как сохранять созданные файлы (файлов со временем накапливаете: много). Иными словами, вам необходим броузер, гипертекстовый редактор набор папок для хранения результатов своего труда.

А вообще, с гипертекстом и с сопутствующими программами надо эксперимента ровать. Ни в одной другой области вычислительной техники вы не найдете такого разнообразия приемов и эффектов. К тому же, для поиска информации вам до ступен целый мир: в буквальном смысле — вся планета, а в переносном — удиви тельный виртуальный мир под названием Интернет.

Глава 2

Синтаксис HTML 4

В этой главе рассматриваются основные принципы создания конструкций HTML. Существуют общие правила составления гипертекстовых документов, которые надо узнать прежде, чем начать изучение элементов языка. Не отчаивайтесь, если назначение каких-нибудь элементов, приведенных в качестве примеров, покажется вам неясным. Большинство элементов будут подробно обсуждаться позднее. В этой главе надо обратить внимание на правила написания тегов и использования атрибутов, а также на проблему кодировок. Приведенный в качестве образца шаблон Web-страницы во многих случаях подойдет в качестве заготовки для конструирования самых различных страниц.

Версии HTML

Первая версия HTML была разработана в начале 90-х годов Тимом Бенерс-Ли для популярного в прошлом броузера Mosaic. Но в те времена ни для броузера, ни для самого языка еще не нашлось достойного применения. В 1993 году появился HTML+, и эта версия также осталась практически незамеченной. Начало широкому использованию гипертекста дала версия 2.0, которая появилась в июне 1994 года. Это был момент начала роста популярности WWW по всему миру. Элементы, включенные в версию 2, в большинстве своем используются и по сей день.

В версии 3.0 HTML, которая появилась год спустя, была реализована возможность прорисовки математических символов (знаков интеграла, бесконечности, дроби, скобок и т. д.) при помощи элементов языка. Под эту версию разрабатывались и броузеры (Arena). Но этот проект оказался тупиковым и не получил дальнейшего распространения.

В 1996 году появился HTML версии 3.2. Это было новаторское решение, достаточно упомянуть, что в спецификацию языка были введены фреймы, которые стали теперь весьма популярными у разработчиков Web-страниц. Даже сейчас на основе этой спецификации можно реализовывать очень неплохие дизайнерские решения. Практически все современные броузеры стопроцентно поддерживают версию 3.2, поэтому у авторов не возникают сомнения по поводу работоспособности заявленных элементов.

Наряду с официальными спецификациями языка, которые разрабатывались организацией W3C (W3 Консорциум), компании-производители броузеров создавали собственные элементы (расширения). Впоследствии некоторые из этих элементов, после получения всеобщего признания, включались в спецификацию

следующей версии языка. Интересно, например, что новаторское решение - фреймы, — которое так полюбилось многим разработчикам, не было включено спецификацию 3.2. Но броузеры поддерживали фреймы, и многие книги, посвящен ные HTML, содержали описания фреймов без упоминания о том, что это нестан дартные элементы. И это было правильно, потому что фреймы стали стандарто де-факто. В версию языка 4 они уже были включены на полном основании.

И наоборот, элементы APPLET и SCRIPT, необходимые для расширения HTML други ми программными кодами, *в* версии 3.2 не сыграли той роли, которую были при званы сыграть. Это объяснялось тем, что броузеры различных версий по-разном интерпретировали программы на языках Java, JavaScript, Visual Basic (VBScript В результате не удавалось получить достаточно надежно работающий код, и дан ные языки использовались любителями HTML в основном для экспериментов.

Официальная спецификация HTML 4 (Dynamic HTML) появилась в 1997 году В это время уже было очевидно, что дальнейшее развитие гипертекста будет осуще ствляться за счет скрипт-программирования. Это оказалось намного более эффек тивным, чем вводить в язык все новые элементы. Появившиеся в то время броузе ры (Netscape Navigator 4, Microsoft Internet Explorer 4 и др.) уже достаточно надежно интерпретировали программный код (был достигнут определенный уро вень стандартизации). Однако проблемы у разработчиков еще остались. В каче стве примера можно отметить, что многие скрипты начинаются с определения версии броузера, чтобы потом использовать тот или иной фрагмент кода. Очевид но, что на программиста ложится обязанность тестирования страниц на все; популярных в настоящее время броузерах. Кроме того, актуальной остается проблема использования старых или не очень популярных программ. Лидерами «броузеростроения» по праву считаются компании Microsoft и Netscape, но вед существуют еще и другие фирмы...

В результате использование всех возможностей Dynamic HTML стало уделом программистов достаточно крупных организаций, где есть условия для разработ ки сложных программ и всестороннего их тестирования. Создателям личных Web-страниц подчас приходится искать компромисс между надежностью и нова торством, чтобы получить достаточно грамотный HTML-код.

Анатомия Web-страницы

Ниже показана заготовка типичного Web-документа. На этом примере мы рас смотрим структуру HTML-страниц. Я назвал этот файл Strukt.htm.

Листинг 1.1. Пример (шаблон) Web-страницы

```
<HTML>
<HEAD>
<TITLE>Структура Web-страницы</title>
<STYLE> H2 {font-family: Arbat;}
CODE {font-family: Arial;} </style>
<META http-equiv="Content-Type" content="text/html; charset=windows-1251">
<META name="Author" content="Alexei Goncharov">
```

```
<META name="Keywords" content="WWW, HTML, document, element">
</head>
<BODY bgcolor=#FFFFFF>
<!-- Комментарий к странице -->
<A name="top"></a>
Переход в <A href="#bottom">конец</a> документа<P>
Переход к <A href="#S001"><B>ссылке 1</b></a><P>
<P>
<HR>
<H1>Заголовок 1</h1>
<H2>Заголовок 2</h2>
<H3>Заголовок 3</h3>
<H4>Заголовок 4</h4>
<H5>Заголовок 5</h5>
<H6>Заголовок 6</h6>
<HR>
Здесь расположена <B>ссылка 1</b><A name="S001"></a>
<HR>
<P>Здесь должен располагаться оригинальный текст Web-страницы
<HR>
<A name="bottom"></a><P>
Переход в <A href="#top">начало</a> документа
</body>
</html>
```

Если рассмотреть исходные тексты различных Web-страниц, то можно легко увидеть схожесть их структур. Это объясняется тем, что документы создаются по определенным правилам. В основу синтаксиса языка HTML лег стандарт ISO 8879:1986 «Information processing. Text and office systems. Standard Generalized Markup Language (SGML)». Правда, существует большое различие между стандартом официальным и стандартом фактическим. HTML постоянно развивается, дополняется новыми элементами, и изучать его надо не по официальным первоисточникам, а на практике, обращаясь к последним разработкам ведущих фирм и специалистов.

Чтобы понять структуру Web-страницы, необходимо рассмотреть все элементы, входящие в приведенный выше листинг. При рассмотрении элементов языка я буду приводить оба тега: начальный и конечный. Например: <I> </i>. Этим я хочу подчеркнуть, что в большинстве случаев разработчик должен использовать два тега для каждого элемента. Число случаев, когда допустим только начальный тег (часть элементов не имеют конечного вообще), невелико, и они специально оговариваются. Для имен тегов можно использовать как прописные, так и строчные буквы латинского алфавита. Некоторые пользователи записывают начальные теги прописными буквами, а конечные теги — строчными. Это помогает разобраться в исходном тексте Web-страницы.

<HTML> </html>

Обозначение документа на языке HTML. Я уже упоминал о том, что одним из принципов языка является многоуровневое вложение элементов. Данный элемент является самым внешним, так как между его начальным и конечным тегами должна находиться вся Web-страница. В принципе, этот элемент можно рассматривать как формальность. Он имеет атрибуты version, lang и dir, которыми в данном случае редко кто пользуется, и допускает вложение элементов HEAD, BODY FRAMESET и других, определяющих общую структуру Web-страницы. Естественно что конечным тегом </html> заканчиваются все подобные документы.

<HEAD></head>

Область заголовка Web-страницы. Иными словами, ее первая часть. Так же, как и предыдущий элемент, HEAD служит только для формирования общей структуры документа. Этот элемент может иметь атрибуты lang и dir, должен включать элемент TITLE и допускает вложение элементов BASE, META, LINK, OBJECT, SCRIPT, STYLE.

<TITLE> </title>

Элемент для размещения заголовка Web-страницы. Строка текста, расположенная внутри этого элемента, отображается не в документе, а в заголовке окна броузера. Эта строка часто используется при организации поиска в WWW. Поэтому авторы, создающие Web-страницы для размещения в Сети, должны позаботиться о том, чтобы эта строка, не будучи слишком длинной, достаточно точно отражала назначение документа.

<STYLE> </style>

Описание стиля некоторых элементов Web-страницы. В файле Strukt.htm назначены шрифты для элементов H2 и CODE. На рис. 2.1 видно, как изменится вид заголовка второго уровня после такого переопределения. Естественно, что для каждого элемента существует стилевое оформление по умолчанию, поэтому употребление элемента STYLE не обязательно, но желательно.

Рис. 2.1. Заголовки, создаваемые при помощи элементов H1...H6. Шрифт второго заголовка переопределен

Интересно, как синтаксис HTML отражает историю развития вычислительной техники. Например, старый, теперь уже не работающий элемент BLINK напоминает нам о тех временах, когда люди использовали дисплеи, которые имели (страшно подумать!) только текстовый режим. При таком положении вещей мигание текста (blink) было, наверное, единственным достижимым визуальным эффектом. В противоположность этому, элемент STYLE, введенный сравнительно недавно, вызывает ассоциации с программами для Windows, так как в них впервые появилось стилевое оформление текста, которое теперь невероятно популярно, и без него уже немыслима работа в таких приложениях, как Word или Excel.

<META>

Этот элемент содержит служебную информацию, которая не отражается при просмотре Web-страницы. Внутри него нет текста в обычном понимании, поэтому нет и конечного тега. Каждый элемент META содержит два основных атрибута, первый из которых определяет тип данных, а второй — содержание. Вот несколько примеров meta-данных.

- Дата, обозначающая «срок годности» документа:

  name="Expires" content="*Дата*"

- Адрес электронной почты:

  name="Reply-to" content="*Имя@Адрес*"

- Имя автора Web-страницы:

  паше ="Author" content="*Имя автора*"

- Набор ключевых слов для поиска:

  name="Keywords" content="*слово1, слово2, слово3 ...*"

- Краткое описание содержания Web-страницы:

  name="Description" content="*Содержание страницы*"

- Описание типа и характеристик Web-страницы:

  name="Content-Type" content="*Описание страницы*"

- Указание приложения, в котором была создана Web-страница:

  name="Generator" content="*Название HTML-редактора*"

Здесь и далее при описании элементов и их атрибутов курсивом выделены фрагменты, которые должны быть заполнены пользователем по его усмотрению. Атрибут name используется приложением-клиентом для получения дополнительной информации о Web-страницах и их упорядочения. Этот атрибут часто заменяют атрибутом http-equiv. Он используется сервером для создания дополнительных полей при выполнении запроса.

Кроме этого, элемент META может содержать URL. Шаблон соответствующего атрибута таков:

URL="http://*адрес*"

<BODY> </body>

Этот элемент заключает в себе гипертекст, который определяет собственно Web страницу. Это та *произвольная* часть документа, которую разрабатывает авто страницы и которая отображается броузером. Соответственно, конечный тег этс го элемента надо искать в конце HTML-файла. Внутри элемента BODY можно ис пользовать все элементы, предназначенные для дизайна Web-страницы. Внутр начального тега элемента BODY можно расположить ряд атрибутов, обеспечиваю щих установки для всей страницы целиком. Рассмотрим их по порядку.

Один из самых полезных Для дизайна — атрибут, определяющий фон страниць Его появление можно уподобить маленькой революции в WWW, так как одина ково серые Web-страницы вдруг расцвели яркими цветными узорами:

```
background="Путь к файлу фона"
```

Более простое оформление фона сводится к заданию его цвета:

```
bgcolor="#RRGGBB"
```

Цвет фона задается тремя двуразрядными шестнадцатеричными числами, кото рые определяют интенсивность красного, зеленого и синего цветов соответствен но. Более подробно о задании цветов будет рассказано ниже.

Оба приведенных выше атрибута не являются альтернативными и часто исполь зуются совместно: если по каким-либо причинам не может быть найден рисуно фона, используется цвет.

Поскольку фон страницы может изменяться, необходимо иметь возможност] подбирать соответствующий цвет текста. Для этого имеется следующий атрибут

```
text="#RRGGBB"
```

Для задания цвета текста гиперссылок используется следующий атрибут:

```
link="#RRGGBB"
```

Точно так же можно задать цвет для просмотренных гиперссылок:

```
vlink="#RRGGBB"
```

Можно также указать изменение цвета для последней выбранной пользователем гиперссылки:

```
alink="#RRGGBB"
```

Гипертекст, расположенный внутри элемента BODY, может иметь произвольную структуру. Ее определяют, в первую очередь, назначение Web-страницы и фанта зия разработчика.

<!-- Комментарий -->

В любом языке программирования есть конструкции, позволяющие создавать произвольные ремарки. HTML в этом смысле — не исключение. Текст, введенный внутри этого элемента, игнорируется броузером. Эти элементы могут располагать-

ся в любом месте Web-страницы. Без закрывающей угловой скобки здесь, по-видимому, не обойтись: комментарий должен быть отделен от основного текста. Признаком комментария служит восклицательный знак, а текст комментария должен обрамляться двойными дефисом. Например:

```
<!-- Начало вывода таблицы -->
```

<Hlx/hl>

Элемент заголовка. Существует шесть уровней заголовков, которые обозначаются H1...H6. Заголовок уровня 1 самый крупный, а уровень 6 обеспечивает самый маленький заголовок. Рис. 2.1 дает представление об относительных размерах букв в заголовках. Для заголовков можно использовать атрибут, задающий выравнивание влево, по центру или вправо:

```
align="left"
align="center"
align="right"
```

<HR>

Горизонтальная линия (horizontal rule) — очень часто используемый элемент. Во-первых, потому что с его помощью очень удобно делить страницу на части. Во-вторых, потому что выбор подобных элементов оформления у автора страницы очень небольшой. Действительно, в HTML практически отсутствуют похожие конструкции, только для горизонтальной линии почему-то было сделано исключение. Правда, несмотря на некоторую скупость языка в этой области, можно придумать немало стандартных графических образов, которые разнообразили бы вид страниц.

Элемент не имеет конечного тега, но допускает ряд атрибутов для выравнивания влево, по центру, вправо, по ширине:

```
align="left"
align="center"
align="right"
align="justify"
```

Можно задавать толщину линии:

```
size=толщина в пикселах
```

Можно управлять длиной линии:

```
width=длина в пикселах
width=длина в процентах%
```

Можно выбрать цвет:

```
color="цвет"
```

<A>

HTML-документ может быть очень большим, и в этом случае пользователю дол жна быть предоставлена возможность быстрого перемещения к нужному раздел документа. Для этого можно использовать механизм гиперссылок. Необходим также в нужных местах текста расставить соответствующие метки. Подробн гиперссылки обсуждаются в разделе «Гиперссылки» главы 3, а здесь мы рассмот рим только шаблон для создания меток:

```
<A name="метка">Произвольный текст</a>
```

В этом случае данной строке документа присваивается имя, и, следовательно, другой части документа или даже на другом документе может быть создана гипер ссылка, приводящая в эту точку.

Например, для перехода внутри документа можно использовать следующую кон струкцию:

```
<P>Переход к <A href="#метка">метке</a></p>
```

Несколько подобных строк могут образовать своеобразное оглавление Web-стра ницы, которое можно разместить в начале и в конце документа.

<BASE>

Элемент для задания базового адреса (URL) для ссылок. Это позволяет опускат] начальную часть адреса в ссылках документа. Для использования этого элемент; необходимо использовать следующую конструкцию:

```
<BASE href="http://компьютер/путь1">
```

Фрагмент адреса *путь1* не является обязательным. При формировании полного адреса он будет отброшен. Так, если в тексте документа встретится относительная ссылка

```
<A href="путь2/имя документа.htm">Видимый текст ссылки</a>,
```

то она будет соответствовать URL

```
http://компьютер/путь2/имя документа.htm
```

В том случае, когда надо задать базовый адрес для локального диска (например, D:), должна быть использована такая конструкция:

```
<BASE href="file://D:\путь\">
```

Тогда при указании относительной ссылки можно будет задавать не только имя файла, но и имена папок, в которых он находится. Иными словами, путь к файлам может быть разбит на две части: абсолютную и относительную. Это полезно в том случае, когда для файлов, указанных в документе, есть общий начальный фрагмент пути.

В выражении абсолютной ссылки можно также опустить указание на схему доступа (file://). В этом случае будет учитываться только левая часть абсолютной ссылки до первого левого символа "\", то есть имя локального диска.

Правила синтаксиса

Теперь, когда мы знаем, как выглядит код Web-страницы, можно сделать некоторые обобщающие выводы относительно синтаксиса HTML. При использовании каждого элемента важно знать, какие элементы могут располагаться *внутри него* и *внутри каких* элементов может находиться он сам. Так, взаимное расположение элементов HTML, HEAD, TITLE и BODY должно быть стандартным на любой странице, правда, в тех случаях, когда не используются фреймы. Если же страница представляет собой документ планировки фреймов (подробнее об этом в разделе «Фреймы» главы 3), то вместо элемента BODY используется элемент FRAMESET.

Существуют группы элементов, которые используются совместно. К ним относятся элементы для создания таблиц, списков, фреймов. В этом случае порядок вложения элементов определяется логикой создания того или иного объекта на странице: тут надо помнить несложные правила конструирования. Таблицы и фреймы часто используются для того, чтобы разместить детали страницы (рисунки, текст и т. д.) в определенном порядке. Например, располагая рисунок *внутри* ячейки таблицы, можно добиться определенного его положения. В таких случаях вложенность элементов определяется разработчиком Web-страницы, и многое зависит от его фантазии и умения.

Большое количество элементов, которые используются для форматирования текста, допускают самые разнообразные варианты вложения. И сами они обязательно должны располагаться внутри определенных элементов. Здесь надо руководствоваться здравым смыслом: каждый элемент выполняет заданную функцию и имеет определенную область действия.

В приведенном ниже примере есть два абзаца (первый в зеленой рамке) и таблица:

```
<P style="border: 3px solid green">Текст абзаца 1</p>
<TABLE> ... </table>
<P> Текст абзаца 2</p>
```

Таблица в данном случае — независимый элемент. Ее можно, например, выравнивать независимо от остального текста.

Можно использовать другой код:

```
<P style="border: 3px solid green">Текст абзаца 1
<TABLE> ... </table>
<P> Текст абзаца 2</p>
```

Исчез конечный тег первого абзаца. Теперь таблица является частью первого абзаца, и зеленая рамка будет охватывать таблицу и текст.

И наоборот, элемент P может находиться внутри таблицы: например, один элемент ячейки TD может содержать несколько абзацев P.

Нарушение правил вложения — одна из наиболее распространенных ошибок при создании Web-страниц. Чтобы избежать таких ошибок, надо пользоваться редакторами гипертекста, которые автоматически контролируют выполнение правил

синтаксиса. Ниже приведена строка, содержащая типичную ошибку вложения элементов:

```
<H1>Заголовок 1 <H2> Заголовок 2 </h1> Заголовок 3 </h2>
```

Надо заметить, что броузеры построены таким образом, что они «стараются» не реагировать на ошибки разметки гипертекста. Если страница может быть отображена, то она выводится на экран без каких-либо предупреждающих сообщений. Программа интерпретирует ошибочно расставленные теги определенным образом и формирует изображение, следуя логике, заложенной в нее разработчиками. При этом вид страницы может и не соответствовать замыслу автора. И только в случае очень серьезных ошибок или явных противоречий броузер выводит сообщение о невозможности отобразить страницу. Косвенным признаком ошибки разметки может служить появление на странице фрагментов кода HTML. Пользователи много работающие с Интернетом, наверняка сталкивались с такой ситуацией.

Правила синтаксиса распространяются и на использование стартового и конечного тегов, атрибутов и содержимого элемента. Не путайте понятия «элемент» и «тег». *Элемент* — это контейнер, содержащий атрибуты внутри стартового тега и полезную информацию между стартовым и конечным тегами. *Тег* — это конструкция, заключенная в угловые скобки и используемая для обозначения области действия элемента.

Некоторые элементы не имеют конечного тега. Очевидно, что элементу BR, обозначающему конец строки, не нужен конечный тег. Некоторые элементы могут использоваться с конечным тегом или без него. Самым ярким примером служит элемент абзаца P. Он может иметь конечный тег, но если этот тег не задан, то признаком окончания действия элемента служит следующий элемент, который может логически определить конец текущего абзаца: другой элемент P, элемент рисунка IMG, элемент списка UL, элемент таблицы TABLE и т. д.

Таким образом, полезная информация одного элемента *должна* находиться или между начальным и конечным тегами данного элемента, или между начальным тегом данного и начальным тегом следующего элемента. Любой произвольный текст, введенный на страницу, воспринимается броузером как подлежащий выводу на экран и, следовательно, форматированию в соответствии с окружающими этот текст элементами. При этом не учитывается разбиение текста на строки, полученное в текстовом редакторе. Теоретически, всю Web-страницу можно уместить в одной длинной строке. Символы конца строки, введенные, например, в Блокноте, могут помочь чтению кода HTML, но не отображаются броузером. Последний, при выводе страницы на экран, может оборвать строку в соответствии с расстановкой элементов Hn, P или BR, а в остальных случаях он форматирует абзацы произвольно, в зависимости от объема текста, размера шрифта и текущего размера окна. Поэтому Web-страницы надо компоновать таким способом, чтобы их вид кардинально не изменялся для разных режимов разрешения монитора, размера экрана, размера окна броузера, а также для полноэкранного или оконного режимов.

Очень важным правилом, которое не имеет исключений, является размещение атрибутов элемента *внутри начального тега.*

Кодирование символов

Я думаю, нет нужды доказывать, что основным языком Интернета является английский. В то же время для гипертекстовых документов предусмотрено использование национальных алфавитов. Стандартным режимом отображения является кодировка ISO Latin 1 (ISO 8859-1). Она подходит как для MS-DOS (набор символов ASCCII), так и для Windows, поэтому набор программ для просмотра и редактирования кода HTML, достаточно широк. В то же время броузеры поддерживают набор символов Unicode 2.0 (ISO10646), что позволяет использовать национальные алфавиты. С практической точки зрения это означает, что символы могут кодироваться однобайтовым числом (коды в пределах 0-255) или двубайтовым (0-65 535). В первом случае для использования национального алфавита необходим атрибут charset (см. листинг 1.1), так как одни и те же коды символов могут быть интерпретированы по-разному для различных кодовых страниц,

А как быть, если атрибут charset не указан? Раскройте в MSIE подменю Вид кодировки меню Вид и проверьте, какие кодовые страницы доступны на вашем компьютере. Наверняка вы найдете команды, отвечающие за отображение русских букв: Кириллица (Windows) и/или Кириллица (КОИ8-Р). Это две наиболее популярные в нашей стране кодировки. Самый простой вариант, когда для Web-страницы не указана ни кодовая страница, ни конкретные шрифты. Тогда броузер будет использовать шрифты, выбранные по умолчанию. Поскольку каждый пользователь настраивает программу для себя и применяет шрифты с национальными алфавитами, то с отображением отечественных ресурсов проблем обычно не возникает. Русский будет читать русские, а немец — немецкие тексты. Если страница загружается из Интернета, то проблема тоже может быть решена: броузер анализирует текст страницы и пытается подобрать необходимую кодировку. Если он делает это неправильно, пользователь всегда может применить упомянутую выше команду и исправить положение.

Если в документе есть *указание* на определенную кодовую страницу, выбор шрифта (в данном случае — некоторого подмножества символов, которые будут служить для отображения кодов 128-255) будет предопределен. Коды 32-127, то есть знаки препинания, цифры и буквы латинского алфавита, отображаются в подавляющем большинстве случаев правильно, а коды 128-255 могут отображаться по-разному. Обычно программы поддерживают большое число национальных алфавитов. Во время инсталляции программного обеспечения автоматически устанавливаются и необходимые для этого шрифты.

Проблема чаще всего возникает, если кодовая страница указана неправильно. Например, при создании гипертекстовых документов в MS Word или MS FrontPage Express в текст страницы автоматически добавляется конструкция типа charset=xxxxx, которая не позволяет использовать кириллицу. В этом случае необходимо правильно указать значение атрибута: charset=windows-1251 (или другое, если вам нужна другая кодовая страница). Если вы посмотрите «фирменные» русские сайты в Интернете, то обнаружите, что большинство переключателей кодировки предлагают два варианта использования кириллицы: Windows и КОИ-8.

Почему же возникают такие сложности, когда существует система Unicode? Ответ прост: из-за стремления разработчиков обеспечить себе комфортные условия работы. Действительно, все стандартные программы рассчитаны на однобайтное представление символов. А редактировать код HTML удобнее всего, просто вводя символы с клавиатуры. Если же документ использует кодировку Unicode, то для работы с ним не подойдут такие средства, как Блокнот, Norton Commander или WordPad, и придется остановиться на гипертекстовом редакторе. В этом случае русская буква А будет выглядеть в режиме «источника» так: &#1040 (в десятичной кодировке). Такую страницу будет сложно читать и редактировать. Вы можете столкнуться с подобной кодировкой, если будете набирать кириллицу в неруси фицированном гипертекстовом редакторе. Он может выполнить автоматическое преобразование символов. Поэтому каждый новый редактор надо тестировать на возможность использования русских букв: набрать небольшой текст, сохранить документ, а затем просмотреть его в режиме источника.

Использование спецсимволов

В HTML и, соответственно, в броузерах реализована возможность прорисовки символов по их кодам. Символы могут быть общепринятыми, вводимыми с клавиатуры, нестандартными и используемыми в HTML в качестве служебных. Все их будем называть *спецсимволами*. Первый способ ввода спецсимвола заключается в указании его кода. Например, латинскую букву А можно задать так: &#65. Для некоторых символов предусмотрена *мнемоническая* кодировка. В табл. 2.1 приведен набор часто используемых спецсимволов (по стандарту ISO 8859-1). Для отделения кода символа от последующего текста надо вводить точку с запятой.

Таблица 2.1. Спецсимволы

Код символа	Числовой код	Мнемонический код	Название	Символ
34	&#34	"	Прямая кавычка	"
38	&#38	&	Амперсанд	&
60	&#60	<	Знак «меньше»	<
62	&#62	>	Знак «больше»	>
153	&#153	™	Торговая марка	™
160			Неразрывный пробел	
162	&#162	¢	Цент	¢
163	&#163	£	Фунт	£
164	&#164	¤	Знак валюты	¤
165	&#165	¥	Йена	¥
166	&#166	¦	Вертикальная черта	¦
167	&#167	§	Знак параграфа	§
169	&#169	©	Знак копирайта	©

Код символа	Числовой код	Мнемонический код	Название	Символ
171	&#171	&laquo;	Левая типографская кавычка	«
172	&#172	&not;	Знак отрицания	¬
174	&#174	&reg;	Знак «зарегистрировано»	®
176	&#176	&deg;	Знак градуса	°
177	&#177	&plusmn;	Знак «плюс минус»	±
178	&#178	&sup2;	Степень 2	²
179	&#179	&sup3;	Степень 3	³
181	&#181	&micro;	Знак «микро»	µ
182	&#182	&para;	Знак абзаца	¶
183	&#183	&middot;	Точка-маркер	·
185	&#185	&sup1;	Степень 1	¹
187	&#187	&raquo;	Правая типографская кавычка	»
188	&#188	&frac14;	Одна четвертая	¼
189	&#189	&frac12;	Одна вторая	½
215	&#215	&times;	Знак умножения (крестик)	×
247	&#247	&divide;	Знак деления	÷

Зачастую мнемонические коды символов бывают очень полезны: особенно для тех символов, которые нельзя ввести с клавиатуры. Например, для указания разрешения экрана можно использовать такую конструкцию:

```
640&times;480
```

Двойная кавычка, угловые скобки и амперсанд (", <, >, &) являются служебными в HTML. Если их надо использовать в обычном тексте на Web-странице, то они должны быть указаны только при помощи кодов.

Интересно, что кириллица тоже может быть реализована в виде набора спецсимволов. Русские буквы расположены (при кодировании Windows-1251) в том месте кодовой таблицы, где первоначально располагались латинские буквы с тильдами, умляутами и т. д. Поэтому англоязычные редакторы HTML при переключении клавиатурного регистра часто записывают кириллицу в виде кодов. Эта особенность рассмотрена в разделе «HoTMetaL PRO 5.0» главы 8.

Таблицы спецсимволов расположены в файле Spec.htm на прилагаемой дискете. Вы можете проделать эксперимент с этим файлом. Создайте его копию и удалите из текста указание на кодовую страницу (charset=windows-1251). Затем откройте страницу в броузере и посмотрите, как изменятся образцы символов. Например, вместо русских могут появиться символы европейских алфавитов (буквы с тильдами, умляутами и т. д.). Может измениться и вид кириллицы, набранной при помощи мнемокодов. Изменив вид кодировки в броузере, можно увидеть самые разные буквы. Если же вид некоторых спецсимволов не зависит от выбранной кодировки, то эти символы можно с уверенностью использовать в своих Web-страницах.

Типы данных

Выше упоминалось, что основным типом данных Web-страницы является текстовый. Существует много элементов, которые являются контейнерами для текста или используются для форматирования текста. Числовые и мнемонические коды, обозначающие символы, предваряются знаком амперсанда &. Таким образом, амперсанд и угловые скобки могут быть введены в текст не напрямую, а только при помощи кодов. Текстовые данные, являющиеся значениями атрибутов, заключаются в прямые двойные кавычки.

Числовые данные требуются только для указания значений атрибутов и записываются без кавычек.

Гиперссылки (координаты переходов) обычно состоят из двух частей: текста подсказки и адреса, который определяет координату перехода. Текст подсказки или заменяющий его рисунок располагается внутри элемента A, а адрес задается при помощи атрибута href. Гиперссылка обычно выделяется на фоне обычного текста цветом или подчеркиванием. Кроме того, вид указателя мыши обязательно изменяется, когда пользователь помещает его над гиперссылкой.

На Web-страницах используются и другие формы ссылок. Так, графические данные хранятся в отдельных файлах, и для указания места расположения этих файлов служит элемент IMG с атрибутом src. Для указания Web-страниц, размещаемых внутри фреймов, служит элемент FRAME с атрибутом src.

Существуют правила для определения размеров элементов. Размер, заданный обычным числом, выражается в пикселах. Так, например, задается длина горизонтальной линии в 100 пикселов:

```
<HR width=100>
```

Горизонтальный размер может быть задан и в процентном отношении к ширине окна броузера:

```
<HR width=50%>
```

Если размер окна будет изменен, то изменится и длина горизонтальной линии.

Символ * может использоваться для разделения пространства страницы в определенных пропорциях. Ниже показаны два способа определения фреймов (атрибут cols создает фреймы с вертикальным делением), причем левый фрейм в данном случае всегда будет в три раза уже, нежели правый.

```
<FRAMESET cols="25%, 75%">
<FRAMESET cols="1*, 3*">
```

Управление цветом

Кодирование цвета используется для раскрашивания шрифтов, горизонтальных линий и фона, других элементов страницы. Цвета обозначаются английскими названиями или числовыми шестнадцатеричными кодами. Существует несколь-

ко атрибутов, значениями которых являются параметры цвета. Самый простой способ определить цвет — написать название цвета на английском языке. Так, например, задается красный цвет шрифта в элементе FONT:

```
color="red"
```

В табл. 2.2. представлены все допустимые названия цветов. Но в общем случае цвет определяется так называемым RGB-кодом, шаблоны которого уже упоминались выше. Любой цвет представляется в этом случае как комбинация красного (R), зеленого (G) и синего (B) цветов, взятых в определенных пропорциях.

Таблица 2.2. Названия и коды цветов

Русское название	Английское название	RGB-код
Аквамарин	aqua	#00FFFF
Белый	white	#FFFFFF
Желтый	yellow	#FFFF00
Зеленый	green	#008000
Золотой	gold	#FFD700
Индиго	indigo	#4B0080
Каштановый	maroon	#800000
Красный	red	#FF0000
Оливковый	olive	#808000
Оранжевый	orange	#FFA500
Пурпурный	purple	#800080
Светло-зеленый	lime	#00FF00
Серебристый	silver	ttCOCOCO
Серый	gray	#808080
Сизый	teal	#008080
Синий	blue	#0000FF
Ультрамарин	navy	#000080
Фиолетовый	violet	#EE80EE
Фуксиновый	fuchsia	#FF00FF
Черный	black	#000000

Доля каждой цветовой составляющей определяется интенсивностью цвета и выражается двуразрядным шестнадцатеричным числом. В десятичном исчислении эти числа соответствуют диапазону от 0 до 255. Легко подсчитать, что комбинируя интенсивности трех базовых цветов, разработчик Web-страницы имеет возможность запрограммировать любой из 16 777 216 доступных оттенков. В Windows 95 это соответствует цветовому режиму монитора True Color (24-разрядное двоичное кодирование цвета). Это наилучший на сегодняшний день режим цветопередачи. Таким он, видимо, останется и в дальнейшем, так как возможности цветового зрения человека и возможности мониторов в смысле точного воспроизведения

цветовых нюансов подходят здесь к своему пределу. Таким образом, пользователи и разработчики Web-страниц нисколько не ущемлены в своих «цветовых правах» по сравнению с остальным компьютерным миром.

Если вы внимательно посмотрите на коды, приведенные в табл. 2.2, то обнаружите, что для формирования стандартных цветов часто используются или крайние значения интенсивности базового цвета 00 и FF, или среднее значение 80. Многие современные приложения имеют средства для работы с цветом, предоставляя пользователю возможность, выбрав в палитре цвет, увидеть его численные характеристики. И наоборот, задав численные значения получить новый оттенок. Нельзя, правда, сказать, что все подобные программы совместимы между собой в смысле генерации цвета. Создав некоторый цвет в одной программе и задав его RGB-код в другой, вы не обязательно получите тот же самый оттенок. Цветовые нюансы для Web-страниц лучше всего проверять на самих страницах. В файле Color.htm на прилагаемой дискете приведен пример подбора оранжевого цвета. Он труднее всего воспроизводится на компьютере и требует визуального тестирования. Подобную страницу можно использовать и для подбора других оттенков. Фрагмент HTML-файла для этого приведен ниже:

```
<TABLE border=3 width=200>
<TR>
<TD align="center" bgcolor="white" ><B>Код</b>
<TD align="center" bgcolor="white" ><B>Цвет</b>
<TR><TD>#FFB000 <TD bgcolor=#FFB000 >1
<TR><TD>#FFA800 <TD bgcolor=#FFA800 >2
<TR><TD>#FFA000 <TD bgcolor=#FFA000 >3
<TR><TD>#FF9800 <TD bgcolor=#FF9800 >4
<TR><TD>#FF9000 <TD bgcolor=#FF9000 >5
<TR><TD>#FF8800 <TD bgcolor=#FF8800 >6
<TR><TD>#FF8000 <TD bgcolor=#FF8000 >7
<TR><TD>#FF7800 <TD bgcolor=#FF7800 >8
<TR><TD>#FF7000 <TD bgcolor=#FF7000 >9
<TR><TD>#FF6800 <TD bgcolor=#FF6800 >10
<TR><TD>#FF6000 <TD bgcolor=#FF6000 >11
<TR><TD>#FF5800 <TD bgcolor=#FF5800 >12
</table>
```

Палитра оформлена в виде таблицы, часть ячеек которой раскрашена при помощи атрибута задания фона:

```
bgcolor=#RRGGBB
```

Из листинга видно, что задача подбора нужного оранжевого оттенка сводится к подбору интенсивности зеленой составляющей при максимальном значении красной. Вы можете и сами поэкспериментировать с подобной палитрой. В приложении А приведены все значения двуразрядных шестнадцатеричных чисел и их десятичные эквиваленты.

Еще один аспект применения цвета. Выше упоминалось, что элемент HR, создающий горизонтальную линию, допускает использование ряда атрибутов. С их помощью линию можно превратить в цветной прямоугольник. Вот, например, прямоугольник светло-зеленого цвета, выровненный влево, высотой 20 и шириной 18 пикселов:

```
<HR color="lime" size=20 width=18 align="left">
```

Подобные изображения можно использовать для разделения частей страницы или в качестве маркеров списка. Теоретически их можно указывать и в гиперссылках (как деталь, на которой надо щелкать мышью), но это не совсем удобно: элемент HR всегда размещается на отдельной строке.

Ну, и наконец, осталось привести HTML-код, рисующий радугу на Web-странице:

```
<FONT color="red"> РАДУГА </font>
<TABLE border=0 width=100% >
<TR><TD bgcolor=#FF3030 Ж
<TR><TD bgcolor=FFDOOO >0
<TR><TD bgcolor=#F3FF5F >Ж
<TR><TD bgcolor=#00FF00 >3
<TR><TD bgcolor=#6FD3F7 >Г
<TR><TD bgcolor=#5F72FDF >С
<TRXTD bgcolor=#B568F4 >Ф
</table>
```

Можно сделать вывод, что «раскрашивание» Web-страницы — хороший способ придать ей современный, профессиональный вид. Но в данном случае большое значение имеют не только изобретательность и вкус, но особенно чувство меры. Цветовая проработка Web-страницы является альтернативой использованию многочисленных рисунков и позволяет обеспечить более быструю загрузку документа. Так уж повелось, что многие мелкие детали (например, маркеры) представляют собой файлы формата GIF, и часто приходится долго ждать, пока загрузятся все кружочки, квадратики, черточки и другая графическая мелочь. Солидные фирмы могут себе это позволить, так как заинтересованному клиенту не остается другого выхода, как ждать окончания загрузки нужной страницы. Если вы создаете личную страницу, то есть резон сделать ее быстро загружаемой. Иначе ваши потенциальные читатели, не дождавшись прорисовки всех деталей, уйдут «бродить» по другим уголкам Сети.

Я думаю, не будет большой ошибкой утверждать, что разработчики HTML серьезно просчитались, не включив в язык элементы, позволяющие строить примитивные графические объекты. Это практически безграничное поле деятельности для совершенствования гипертекста. Важность графики, поддерживаемой броузером, явно недооценили, а теперь, видимо, слишком поздно наверстывать упущенное, так как признание завоевали другие подходы.

ПРИМЕЧАНИЕ

К сожалению, данный раздел по технологическим соображениям невозможно дополнить цветными иллюстрациями, но такие иллюстрации можно найти в примерах в файле Color.htm на прилагаемой дискете.

Глава 3

Основные элементы HTML версии 4

Главной проблемой при описании и изучении HTML является определение набора атрибутов и их значений, допустимых для каждого из элементов. Очень часто можно столкнуться с ситуацией, когда некий хорошо известный атрибут не использовался в каком-нибудь элементе, а потом вдруг его использование стало давать эффект при просмотре страницы в новом броузере. Поддержка дополнительных атрибутов — скорее дело совершенствования броузеров, а не языка. Например, фирмы Netscape и Microsoft постоянно развивают свои программы. Поэтому я рекомендую вам экспериментировать. Если при разработке Web-страницы вам потребуется известный атрибут, которого нет в спецификации элемента, проверьте — а вдруг заработает?

Заголовок страницы

Заголовок Web-страницы представляет собой информацию, заключенную внутри элемента (секции) HEAD. В разделе «Анатомия Web-страницы» главы 2 мы уже познакомились с элементами заголовка, и сейчас их надо обсудить более подробно.

<TITLE> </title>

Элемент TITLE определяет текст, который появляется в заголовке окна броузера во время просмотра страницы. Этот текст не только служит подсказкой, но может использоваться и поисковыми машинами для анализа страниц.

Существует три способа для поиска страниц в Интернете на основе текстовых данных: по ключевым словам элемента META, по тексту, размещенному на странице, и по строке заголовка внутри элемента TITLE.

< STYLE > </style> и <LINK>

Элемент STYLE тоже должен располагаться внутри элемента HEAD. Если вы хотите разобраться, какие нестандартные форматы используются на странице, надо просмотреть содержимое этого элемента. В нем будут указаны необходимые форматы. Если таких форматов нет, значит стили страницы записаны в отдельном файле. Ссылка на такой файл должна находиться в элементе LINK. Подробнее о стилях рассказывается ниже в разделе «Таблицы стилей».

<META>

Секция заголовка может содержать несколько элементов META, каждый из которых отвечает за определенный набор параметров. Использование элементов META не является обязательным, но некоторые настройки могут быть весьма важны. Так, например, известно, что броузер в некоторых случаях способен автоматически определить вид кодировки страницы. Пользователь, работая с броузером, может выбрать в меню определенную кодировку. Чтобы исключить неопределенность при просмотре конкретной страницы, на ней целесообразно разместить указание на кодовую страницу. Для документов в кодировке Windows оно должно быть таким:

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

Информация, сосредоточенная в элементах META, определяет общие настройки Web-страницы и называется *профилем*. Профили можно хранить в отдельных файлах и присоединять к определенной странице при помощи специального атрибута элемента HEAD:

```
<HEAD profile="URL">
```

В секции HEAD могут располагаться элементы, которые имеют отношение ко всей странице целиком. Так, если для последней создано звуковое сопровождение, то его параметры определяет элемент BGSOUND (см. ниже раздел «Устаревшие и нестандартные элементы»).

Стандартные атрибуты

Существует ряд атрибутов, которые могут использоваться во многих элементах. Часть этих атрибутов очень важна для конструирования Web-страниц, а часть подходит только для решения определенных задач. -

Атрибут id выполняет функции уникального имени элемента. В зависимости от типа элемента, этот атрибут выполняет различные функции (см. раздел «Таблицы стилей» текущей главы и раздел «Элементы форм» главы 4).

Атрибут classid задает программу или объект, которые могут использоваться в определенных элементах.

Атрибут style может использоваться со многими элементами. Он предназначен для определения формата конкретного элемента и может принимать самые разные значения. Подробно он рассмотрен ниже в разделе «Таблицы стилей».

Похожие функции выполняет атрибут class. Его можно указывать, если в секции HEAD расположен элемент STYLE или использована ссылка на каскадную таблицу стилей (см. ниже раздел «Таблицы стилей»).

Атрибут align используется для выравнивания текста, объектов или элементов целиком. Выравнивание может выполняться относительно границ окна, рамки

таблицы и т. д. Каждый элемент позволяет указывать определенные значения для этого атрибута. В общем случае значения могут быть такие:

- left — выравнивание по левому краю;
- right — выравнивание по правому краю;
- justify — выравнивание по ширине (для текста);
- center — выравнивание по центру (по горизонтали);
- middle — выравнивание по центру (по вертикали);
- top — выравнивание по верхней границе;
- bottom — выравнивание по нижней границе.

Атрибут lang определяет, на каком языке набран текст внутри текущего элемента:

lang="*код языка*"

Вот некоторые коды:

- ar — арабский;
- de — немецкий;
- el — греческий;
- en — английский;
- en-us — американская версия английского языка;
- es — испанский;
- fr — французский;
- he — иврит;
- hi — хинди;
- it — итальянский;
- ja — японский;
- nl — голландский;
- pt — португальский;
- ru — русский;
- zh — китайский.

В некоторых языках применяется обратное привычному направление текста: справа налево. При помощи атрибута dir можно указать необходимое направление:

- dir="LTR" — слева направо;
- dir="RTL" — справа налево.

Атрибут dir теоретически может использоваться в разных элементах, но не все броузеры обеспечивают его работу. Более надежный способ — применение специально предусмотренного для подобных случаев элемента BDO (см. ниже).

Атрибут type определяет тип документа, который указывается в ссылке. Здесь используются так называемые типы MIME (Multipurpose Internet Mail Extensions).

Первоначально они предназначались для определения формата отправлений электронной почты, но сейчас служат для указания форматов документов в составе Web-страниц. Наиболее часто используемые типы:

- text/plain — обычный текст;
- text/ess — каскадная таблица стилей;
- text/html — документ в формате HTML;
- application/postscript — документ в формате PostScript;
- image/gif, image/j pg, image/png — изображения в формате GIF, JPG или PNG соответственно;
- video/mpeg — видеоролик;
- application/Java — апплет;
- text/javascript — программа (сценарий) на JavaScript;
- text/vbscript — программа (сценарий) на VBScript.

Для форм атрибут type имеет другой смысл: тип определенного элемента формы (кнопка, поле ввода и т. д.).

Атрибут charset необходим там, где надо указать вид кодировки, например: charset="ISO-8859-1".

Атрибут longdesc (long description) может быть полезен в тех случаях, когда для какого-нибудь элемента необходимо использовать описание (комментарий) большого объема. Тогда документ присоединяется при помощи ссылки:

```
longdesc="URL"
```

Атрибут title, наоборот, позволяет создать короткую всплывающую подсказку. Она появляется на экране, когда пользователь располагает указатель мыши над элементом. Значение атрибута — произвольная текстовая строка.

Атрибуты событий

Для страницы могут быть определены программы, которые выполняются только в случае определенных действий пользователя. В этом случае моменты запуска программ надо «привязать» к определенным *событиям*. Например, если надо изменить внешний вид элемента, когда пользователь укажет на него мышью (очень модный ныне прием), то для такого элемента должны быть указаны два атрибута:

```
onmouseover="Программа1("параметр1")"
onmouseout="Программа2("параметр2")"
```

Первая программа (сценарий) изменит вид элемента, когда указатель мыши окажется над ним, а вторая вернет элементу прежний вид, когда указатель мыши будет убран. Разные элементы позволяют использовать разные события.

События, связанные с мышью:

- onclick — щелчок мышью на элементе;
- ondblclick — двойной щелчок мышью на элементе;

- onmousedown — кнопка мыши нажата;
- onmouse up — кнопка мыши отпущена;
- onmousemove — указатель мышиперемещен в область элемента;
- onmouseover — указатель мыши расположен над элементом;
- onmouseout — указатель мыши перемещен за границы области элемента.

События, связанные с выбором элементов и редактированием форм:

- onfocus — элемент выбран (получен фокус);
- onselect — часть текста внутри элемента выделена;
- onchange — данные в элементе были изменены;
- onblur — элемент перестал быть выбранным (потерян фокус).

События, связанные с клавиатурой:

- onkeydown — клавиша нажата;
- onkeyup — клавиша отпущена;
- onkeypress — клавиша нажата и отпущена.

Форматирование текста

Текст — единственный объект Web-страницы, который не требует специального определения. Иными словами, произвольные символы интерпретируются по умолчанию как текстовые данные. Но для форматирования текста существует большое количество элементов. Большинство из них, кроме специальных, поддерживает стандартные атрибуты: id, class, lang, dir, title, style и атрибуты событий.

Изначально в HTML было введено меньше возможностей для форматирования текста, чем в обычные текстовые редакторы. В результате авторам гипертекстовых документов приходилось прибегать к различным ухищрениям, чтобы придать тексту заданный вид. Сейчас положение изменилось, но все дополнительные возможности осуществляются за счет применения таблиц стилей. Например, только с помощью свойства text-indent можно задать величину отступа первой строки абзаца.

Форматировать текст можно и с помощью традиционных элементов: выделять фрагменты курсивом, полужирным, выбирать шрифт и т. д. Рассмотрим эти элементы. Для них могут быть использованы стандартные атрибуты id, class, lang, dir, title, style, атрибуты событий, а также атрибуты, определяющие уникальные свойства определенных элементов.

<P> </p>

Элемент абзаца (paragraph) — один из самых полезных. Он позволяет использовать только начальный тег, так как следующий элемент P обозначает не только начало следующего абзаца, но и конец предыдущего. В тех случаях, когда по смыс-

лу необходимо обозначить завершение абзаца, можно использовать и конечный тег. В некоторых случаях начальный тег удобно ставить в конце строки: он не только обозначит конец абзаца, но и выполнит функцию тега
 (разрыв строки). Например:

```
<P>Текст первого абзаца.
<P>Текст второго абзаца.</p>
Текст третьего абзаца.<P>
```

Вместе с элементом абзаца можно использовать атрибут выравнивания align:

- align="left" — выравнивание по левому краю;
- align="center" - выравнивание по центру;
- align=" right" — выравнивание по правому краю.

Для центрирования абзаца следует использовать такую конструкцию:

```
<P align="center"> Текст абзаца
```

Абзацы форматируются броузером, и их вид зависит, в частности, от размера окна программы. Три следующих элемента позволяют внести некоторую определенность в формат абзаца.

Элемент, обеспечивающий принудительный переход на новую строку. Он имеет только начальный тег. В месте его размещения строка заканчивается, а оставшийся текст печатается с новой строки.

Атрибут clear позволяет выравнивать объекты (например, рисунки) относительно текста, в котором использован элемент BR. Если элемент объекта содержит атрибут align, то в расположенных рядом элементах BR должен присутствовать атрибут clear, например:

```
<BRclear="right">
```

Значения атрибута:

- none — значение по умолчанию;
- left — если объект выровнен влево;
- right — если объект выровнен вправо;
- all — для объекта, который может быть выровнен по любому краю.

Стандартные атрибуты: id, class, title, style.

<NOBR> </nobr>

Этот элемент по своему действию является прямой противоположностью предыдущего. Текст, заключенный между его тегами, будет выведен в одну строку. Если длинная строка не уместится на экране, для ее просмотра придется использовать горизонтальную полосу прокрутки.

<PRE> </pre>

Элемент для обозначения текста, отформатированного заранее (preformatted). Подразумевается, что текст будет выведен в том виде, в котором был подготовлен автором. Например, учитываются символы конца строки, появившиеся при наборе текста в редакторе. Во всех других случаях броузер игнорирует эти символы. Возможен и обратный эффект: если пользователь введет текст как одну длинную строку, то она не будет разорвана броузером, а уйдет за край окна программы. В этом смысле элемент PRE работает так же, как элемент NOBR. По умолчанию для отформатированного заранее текста выбирается моноширинный шрифт. Этот элемент удобно использовать для показа листингов программ или для вывода текстовых документов, переформатирование которых может привести к искажению их смысла.

Элемент PRE позволяет набрать текст с использованием специальных символов форматирования, таких как «line feed» или «carriage return» (см. табл. 3.1 ниже).

Теоретически можно представить ситуацию, когда разработчику Web-страницы потребуется показать, как создавали линии таблиц в далеком прошлом, когда текстовый режим уже существовал, а символы псевдографики еще не были изобретены. В ход шли плюсы, восклицательные знаки и тире. В этом случае элемент PRE также окажется незаменим, хотя я не рекомендую поддаваться ностальгическим порывам: лучше сделать черно-белый рисунок формата GIF.

Для этого элемента определен специальный атрибут, который позволяет задать ширину блока текста в символах:

```
width=число-символов
```

Этот атрибут не поддерживается многими броузерами. Стандартные атрибуты: id, class, lang, dir, title, style, атрибуты событий.

<CENTER>

Элемент для центрирования текста, а точнее — любого содержимого. Этот элемент не является общеупотребительным. В тех случаях, когда это возможно, вместо пего в элементах текста используют атрибут align="center".

<B>

Выделение текста полужирным шрифтом. Очень популярный элемент. Использование полужирного шрифта — прием, позаимствованный из текстовых редакторов.

<BIG> </big>

Увеличение размера шрифта.

<SMALL> </small>

Уменьшение размера шрифта.

<I> </i>

Выделение текста курсивом.

<STRIKE> </strike> или <S> </s>

Зачеркнутое начертание текста. В настоящее время элемент STRIKE заменяют более простым в написании элементом S.

<U> </u>

Подчеркнутое начертание текста.

Элемент, создающий эффект нижнего индекса (subscript).

Элемент, создающий эффект верхнего индекса (superscript).

Действие двух последних элементов иллюстрирует фрагмент файла гипертекста Text.htm, показанный на рис. 3.1. Оба этих элемента обеспечивают уменьшение размера шрифта. Поэтому их можно использовать и для форматирования абзаца целиком, если надо, чтобы он был выведен мелким шрифтом.

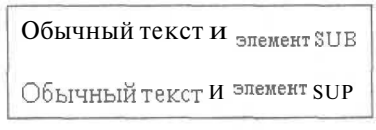


Рис. 3.1. Использование элементов SUB и SUP

<TT> </tt>

Элемент, обозначающий текст телетайпа (teletype). Его особенность заключается в том, что он обеспечивает использование моноширинного шрифта.

<INS> </ins> и <DEL> </del>

Эти элементы позволяют выделить текст, который надо обозначить как вставленный (элемент INS) или удаленный (элемент DEL). Визуально вставленный текст выделяется подчеркиванием, а удаленный — зачеркиванием.

Для указания источника изменений, то есть для документа, в котором находится данный фрагмент либо дано объяснение, почему в тексте появилась такая вставка, может быть использован атрибут:

```
cite="Адрес (URL)"
```

Для даты изменения тоже предусмотрен специальный атрибут:

```
datetime="Дата"
```

В результате начальный тег может иметь такой вид:

```
<INS datetime="2000-04-26" cite="file:///C:/Pages/Дополнения.htm">
```

<BASEFONT>

Элемент, определяющий базовый (основной для всей страницы) размер шрифта. Внутри элемента необходимо указать атрибут:

```
size=базовый размер шрифта
```

Величина для этого атрибута может лежать в пределах от 1 до 7. По умолчанию используется величина 3. Установка, выполняемая этим элементом, имеет значение для элемента FONT (см. ниже), который позволяет задавать относительный размер шрифта. Другие атрибуты у этого элемента такие же, как и у элемента FONT.

<FONT>

Определение типа, размера и цвета шрифта. Все эти характеристики определяются при помощи соответствующих атрибутов. Абсолютный размер шрифта задается атрибутом size (размер):

```
size=абсолютный размер шрифта
```

Этот атрибут может принимать значения от 1 до 7. На рис. 3.2 показаны несколько образцов надписей, выполненных шрифтами разного размера.

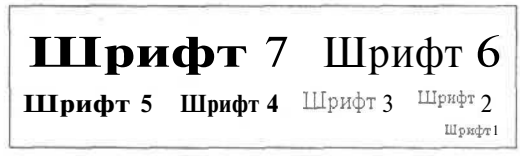


Рис. 3.2. Возможные варианты размеров шрифта

Размер шрифта может также задаваться *относительно* базового:

```
size=+число
size=-число
```

При назначении величины для этого атрибута надо учитывать величину базового размера. В сумме эти две величины должны соответствовать одному из абсолютных размеров. Так, для базового размера, равного 3, относительный размер может находиться в пределах от -2 до +4. Если величина выходит за допустимый предел, то используется либо шрифт размера 7, либо шрифт размера 1. На рис. 3.3 показаны надписи, выполненные шрифтами с заданным относительным размером.

Для элемента FONT можно использовать атрибут цвета:

```
color="цвет"
```

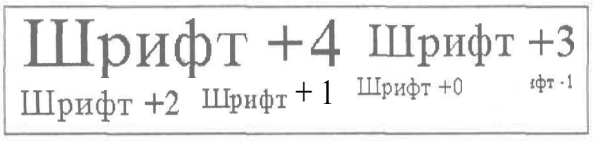


Рис. 3.3. При помощи относительных величин тоже можно получить семь градаций размера шрифта

Атрибут face (вид) позволяет задавать определенный шрифт или несколько шрифтов (через точку с запятой), например:

```
face="Arial; Verdana; Tahoma"
```

Правда, есть одна проблема. Web-страницы просматривает множество людей и нет гарантии, что у каждого из них окажется нужный шрифт. Если в системе не установлен шрифт точно с таким же названием, то броузер использует стандартный шрифт из числа назначенных по умолчанию: один пропорциональный, другой моноширинный.

Элемент FONT может с успехом заменять элементы заголовка H1...H6. Для последних, например, не предусмотрено задание цвета букв. Чтобы заголовок, созданный на основе элемента FONT, хорошо смотрелся, этот элемент необходимо комбинировать с другими: CENTER, B, I, P и т. д.

Дополнительные атрибуты: id, class, lang, dir, title, style.

<BDO> </bdo>

Этот элемент позволяет изменять направление текста. Он используется совместно с атрибутом dir, которому может быть присвоено одно из значений: LTR (слева направо) или RTL (справа налево). Например:

```
<BDO dir="RTL">Направление текста можно изменить</bdo>
```

Табуляция, пробелы, переносы...

В табл. 3.1 приведены коды некоторых, по большей части невидимых в обычном режиме просмотра, символов. Ряд символов (таких как «line feed» или «carriage return») используется в текстовых редакторах для форматирования текста, а пользователь их не видит. Такие символы можно указывать внутри элемента PRE, где они будут выполнять свою функцию.

В тексте могут быть использованы символы мягкого переноса. Они показывают, где может быть закончена строка, но такие символы не учитываются при выполнении операций поиска и сортировки. Если в месте расположения мягкого переноса строка не заканчивается, символ переноса виден не будет. Это отличает мягкий перенос от обычного дефиса, который виден всегда.

Интересен и символ пробела нулевой длины. Такие символы используются для скрытого разделения слов.

Таблица 3.1. Коды символов

Код символа	Числовой код (HTML)	Название
9	&#9	Табулятор
10	&#10	Конец строки (line feed)
12	№2	Конец страницы (form feed)
13	&#13	Возврат каретки (carriage return)
32	&#32	Пробел
45	&#45	Дефис
160		Неразрывный пробел
173	. &#173	Мягкий перенос
8203	&#8203	Пробел нулевой ширины

ПРИМЕЧАНИЕ

Примеры форматирования текста записаны в файл Text.htm, который можно найти на прилагаемой дискете.

Элементы содержания

Существует большая группа элементов, которые используются не столько для форматирования текста, сколько для выделения смысла абзацев и слов. Поскольку для таких элементов не определены заранее эффекты форматирования, разные программы могут по-разному воспроизводить текст, заключенный внутри этих элементов.

<EM> </ет> и <DFN> </dfn>

Элементы, обозначающие выразительность (emphasis) данного фрагмента текста и определение чего-либо (definition). Оба элемента аналогичны по своему действию элементу I, то есть в большинстве случаев позволяют выделить текст курсивом.

С точки зрения дизайна документа эти элементы ничем особенным не отличаются. Они могут пригодиться только для того, чтобы единообразно выделить одинаковые по назначению (или смыслу) фрагменты текста, находящиеся в разных частях документа или даже на разных страницах. Разработчик, в этом случае, не может точно знать, какой именно шрифт будет использован: это определяется каждым броузером по-своему. Но он может быть уверен, что все фрагменты текста будут отформатированы одинаково. В языке можно найти еще несколько элементов, которым можно дать такую же характеристику.

Эти и другие элементы содержания могут иметь стандартные атрибуты: id, class, lang, dir, title, style, атрибуты событий.

<BLOCKQUOTE> </blockquote>

Обозначение цитаты. Этот элемент требует наличия конечного тега. Текст не претерпевает никаких изменений, но абзац располагается с отступом. К кавычкам этот

элемент тоже не имеет никакого отношения: если в цитате имеются кавычки, они должны быть проставлены явным образом. Визуально форматирование этим элементом заключается только в отступе слева, поэтому элемент может быть использован в самых разных случаях.

Этот элемент имеет собственный нестандартный атрибут, который позволяет указать источник цитирования:

`cite="Адрес документа-первоисточника"`

Предполагается, что адрес задается в виде URL.

<Q> </q>

Элемент, похожий на предыдущий. Но если BLOCKQUOTE позволяет создать отдельный абзац с отступом, то Q используется для выделения цитаты *внутри* абзаца (или строки).

<CITE> </cite>

Предполагается, что этот элемент может быть использован для форматирования цитат и ссылок в обычном понимании этого слова. Текст, расположенный внутри этого элемента, печатается по умолчанию курсивом.

<ADDRESS> </address>

Этот элемент подобен элементу CITE и отличается только предусмотренным содержанием. Он также обеспечивает форматирование курсивом. В некоторых справочных руководствах можно встретить информацию о том, что данный элемент поддерживает атрибут align. Легко убедиться, что это не так.

Вообще, к подобным элементам надо относиться с определенной долей недоверия. Я имею в виду элементы, предназначенные для размещения определенной информации, а не для создания эффектов форматирования. Мы уже рассмотрели элементы ADDRESS и CITE, а элементы CODE, KBD, SAMP и VAR рассматриваются ниже. Дело в том, что они не относятся к основному направлению совершенствования HTML и пользовательских агентов. Поэтому броузеры интерпретируют их по-разному и не в обязательном порядке. Другое дело, если стиль для этих элементов определен в самой Web-странице.

<STRONG>

Элемент, отвечающий за выделение текста. Обычно его применение равносильно использованию элемента B для выделения текста полужирным начертанием.

<CODE> </code>, <SAMP> </samp> и <VAR> </var>

Элементы, предназначенные для вывода фрагментов программ. CODE используется для форматирования текста программы. SAMP предполагается задействовать

при иллюстрации примеров (sample) вывода данных на экран. VAR был создан для выделения переменных (variable). Как правило, все эти элементы обеспечивают вывод моноширинного шрифта.

<KBD> </kbd>

Этот элемент предназначен для выделения текста, который пользователь должен ввести с клавиатуры (keyboard). Можно рассчитывать, что текст, выделенный с помощью этого элемента, будет выведен моноширинным шрифтом в полужирном начертании.

<ABBRX/abbr>

Элемент для обозначения аббревиатур, например: MSIE, HTML, WWW.

<ACRONYM></acronym>

Этот элемент можно использовать для выделения акронимов (сокращений, образованных несколькими словами), например: и т. д.; и т. п.

ПРИМЕЧАНИЕ

Примеры элементов содержания находятся в файле Phrase.htm на прилагаемой дискете.

Таблицы стилей

Таблицы стилей (style sheets) являются одним из самых эффективных нововведений HTML 4. Они позволяют изменять свойства элементов в соответствии с желаниями разработчика страницы.

К обычным таблицам таблицы стилей не имеют никакого отношения. В общем случае шаблон таблицы стилей выглядит так:

```
Элемент. имя_стиля {свойство1: значение; свойство2: значение; ...}
```

В результате для определенного элемента задается набор свойств (ассортимент которых весьма значителен). Тем самым снимаются ограничения HTML, а для дизайнера (автора страницы) открывается широкое поле деятельности.

Одна из важнейших особенностей стилевого оформления заключается в том, что преобразованию подвергаются все элементы, заключенные внутри элемента с заданным стилем. Так, определив некоторый стиль для элемента BODY, вы *присваиваете* его всему содержимому Web-страницы. По аналогии с объектно-ориентированными языками программирования это качество называется *наследованием*.

В соответствии с правилами HTML, автор, использующий стили, должен включить в заголовок документа (элемент HEAD) соответствующее мета-определение:

```
<META http-equiv="Content-Style-Type" content="text/css">
```

Броузер получит информацию, какой язык определения стилей использован. «CSS» в данном случае означает «каскадная таблица стилей» (Cascading Style Sheets). Это одновременно стандарт и язык, расширяющий традиционный HTML. В настоящее время существует две спецификации (CSS1 и CSS2), в которых перечислены свойства элементов. Эти свойства очень похожи на атрибуты, но есть два различия: свойств намного больше и правила синтаксиса несколько другие. Перечень свойств приведен в приложении Б.

Надо отметить, что есть еще один способ определения свойств элементов: при помощи языка (например, JavaScript). В этом случае правила синтаксиса свои, хотя суть остается прежней. Этот вопрос обсуждается в главе 5.

Простейшим способом определения стиля является задание его непосредственно для выбранного элемента, например:

```
<P style="font-size: 10pt; font-style: italic; color: blue">
```

Здесь для абзаца выбран размер шрифта, курсив и синий цвет букв. В отличие от атрибутов имя свойства и значение разделяется двоеточием, а свойства отделяются друг от друга точкой с запятой. Для переопределения свойств элемента использован *стандартный* атрибут style, который может указываться со многими элементами.

<STYLE> </style>

Если атрибут style позволяет задать свойства одного конкретного элемента, то элемент STYLE определяет свойства всех одноименных элементов, например:

```
<STYLE type="text/css">
H1 {border-width: 1; border: groove; text-align: center; color: green}
H2 {color: maroon; font-style: italic}
CODE {font-family: Arial, Verdana; background: white}
P {text-indent: 10; color: green; font-size: 12pt}
P CODE {font-weight: bold; color: violet; font-size: 12pt}
</style>
```

В этом случае для всей страницы создаются новые стили для заголовков первого и второго уровня. Для элементов CODE выбирается шрифт и цвет фона. Для абзацев (P) выбирается величина отступа первой строки, цвет и размер букв. Этот простой пример показывает преимущества стилей: автор страницы может легко создать эффекты, недоступные при использовании только одних атрибутов.

Последним в нашем примере идет стиль для вложенных элементов. Он будет использован броузером только в том случае, когда элемент CODE окажется внутри элемента P. При этом свойства текста будут *унаследованы*: от элемента абзаца — отступ (text-indent: 10); от элемента CODE — фон (background: white). А стиль, который можно условно назвать «P CODE» внесет дополнительные форматы: полужирное начертание (font-weight: bold); фиолетовый цвет букв (color: violet) и размер символов (font-size: 12pt). Тут уместно вспомнить о многочисленных

элементах содержания, которые пришли из предыдущих версий языка и вроде бы бесполезны: задачи, для которых они предназначались, кажутся сейчас архаичными. Действительно, зачем предусматривать отдельные элементы для таких объектов, как «текст телетайпа» или «символы, введенные с клавиатуры»? Стили дают таким элементам вторую жизнь. Вложения двух элементов позволяют создать любое количество комбинаций форматов.

Обратите внимание, что в последней строке приведенного выше примера элементы P и CODE разделены пробелом. Когда элементы отделены друг от друга пробелами, это означает, что ряду элементов присваивается один и тот же стиль.

ПРИМЕЧАНИЕ

Как это все выглядит на практике, можно увидеть в файле Style.htm на прилагаемой дискете

Использование элемента STYLE не исключает возможность указания атрибута style. Все элементы, которые будут модифицированы при помощи атрибута style, получат форматирование, отличное от того, которое может быть задано при помощи элемента STYLE. Иными словами, атрибут style имеет более высокий приоритет.

Элемент STYLE может использовать стандартные атрибуты: type, lang, dir, title.

Классы

Ограничения по использованию стилей в последнем примере очевидны. Одному элементу можно присвоить только один стиль. Гораздо удобнее назначать для стилей имена и указывать последние вместе с элементами. Такие универсальные стили называются *классами*. Вот простой пример. Создадим, используя элемент STYLE, стиль заголовка с буквами красивого красного цвета:

```
<STYLE type="text/css">
H1.red1 {color: RGB(215,40,40); text-align: center}
</style>
```

Цвет задан с помощью функции RGB(). Ее аргументами являются *десятичные* числа. Теперь, если надо использовать заголовок, его конструкция должна выглядеть так:

```
<H1 class="red1"> Текст заголовка </h1>
```

При желании можно создать класс red2, red3 и т. д. Очевидно, что такой подход более удобен: можно иметь сколько угодно вариантов форматирования для одного элемента.

Универсальные классы: атрибут id

Следующим шагом, расширяющим возможности разработчика, является использование универсальных классов, то есть таких, которые не связаны с конкретными элементами. Вот простой пример. Допустим, надо создать формат, который позволял бы придать буквам серо-стальной цвет. Назовем этот формат «steel». Для него (внутри элемента STYLE) можно ввести таблицу стиля:

```
#steel {color: RGB(155,180,190); font-weight: bold}
```

Поскольку оттенок надо подбирать очень точно, без функции RGB() опять не обойтись. Учитывая, что буквы получатся достаточно светлыми, можно добавить в стиль полужирное начертание. Обратите внимание, что о размере символов в формате данных нет. Поэтому такой формат (стиль) можно применять для *различных* элементов: заголовков и обычного текста. Но для указания стиля надо использовать атрибут id, например:

```
<H2 id="steel"> Заголовок формата "steel" </h2>
<P id="steel"> Абзац, отформатированный универсальным стилем "steel" </p>
```

В результате размер букв будет определяться другими настройками: значениями по умолчанию или стилями элементов. Это очень удобно: таким способом легко выдержать стилевую целостность текста: известно, что текстовые документы смотрятся лучше, когда для них используется один шрифт и один цвет шрифта. Обилие различных шрифтов и эффектов форматирования может только повредить восприятию.

Каскадные таблицы стилей: элемент <LINK>

Таблицы стилей могут сохраняться в отдельном файле и использоваться для разных Web-страниц. Такие таблицы называются *каскадными* (CSS). Они названы так потому, что несколько CSS-файлов (стили автора страницы, сервера, пользователя) могут использоваться одновременно. В этом случае стили «накладываются» друг на друга, и более поздние определения отменяют принятые ранее установки.

Чтобы подключить к странице файл с таблицами стилей, надо использовать элемент LINK в секции HEAD:

```
<LINK href="имя_файла.css" rel="stylesheet" type="text/css">
```

В этом случае стили используются так же, как и в предыдущих примерах этого раздела. Например, CSS-файл может содержать следующие определения:

```
P.spec1 {color : green;font-variant: small-caps;}
P.new1 {color : maroon;font-style: italic;}
P.new2 {color : maroon;font-style: italic;letter-spacing: 2pt;}
```

В результате на странице могут оказаться три стиля абзацев, показанные на рис. 3.4.

ТЕКСТ АБЗАЦА (CLASS="SPEC1") - ЭФФЕКТ «МАЛЫЕ ПРОПИСНЫЕ»

Текст абзаца (class="new1")- курсив и цвет букв.

Текст абзаца (class = "new2") - добавлена разрядка.

Рис. 3.4. Стили абзацев

ПРИМЕЧАНИЕ

Примеры использования классов и стилей находятся в файлах Class.htm и Formats.css на прилагаемой дискете.

<DIV> и <SPAN>

Существует два элемента, которые специально предназначены для применения стилей. Вы уже знаете, что таблицы стилей универсальны: их можно использовать со многими элементами. Разберем особенности данных элементов.

Для определения стилей допустимы все синтаксические правила, о которых рассказывалось в этом разделе. Особенностью элемента DIV является то, что он предназначен для указания в качестве контейнера элементов, то есть его собственный стиль должен влиять на стили элементов, находящихся внутри. Вот шаблон для использования элемента DIV:

```
<HEAD>
<STYLE type="text/css">
DIV.Имя { Свойство: значение; ... }
</style>
</head>
<BODY>
<DIV class="Имя">
<H1> Заголовок </h1>
<P>Первый абзац
<P>Второй абзац
</div>
</body>
```

Элемент SPAN, наоборот, предназначен для включения в состав других элементов:

```
<HEAD>
<STYLE type="text/css">
SPAN.Имя { Свойство: значение; ... }
</style>
</head>
<BODY>
<H1> Заголовок </h1>
<P><SPAN class="Имя">Текст 1</span> Текст 2 </p>
</body>
```

Выводы

Теперь можно сделать некоторые выводы относительно таблиц стилей.

- Свойства, задаваемые при помощи стилей, позволяют использовать практически неограниченное число вариантов форматирования. Таблицы стилен снимают ограничения, накладываемые традиционной спецификацией HTML.
- С точки зрения синтаксиса, свойства можно задавать разными способами: в одном элементе (inline); для всех одноименных элементов на странице; для *класса* элементов; для ряда докумеитов, с использованием CSS-файла.

- Перечень свойств достаточно велик, и в дальнейшем он, видимо, будет расширяться. Для различных групп элементов (текст, списки и т. д.) существуют свои наборы свойств.
- Теперь существует возможность разделить сам документ и стилевое оформление. Иными словами, для страницы можно применять разные стили, не меняя сам документ.

Списки

Списки (list) были введены в HTML, несомненно, под влиянием успеха текстовых редакторов. Список отличается от обычного текста прежде всего тем, что пользователю не надо думать о нумерации его пунктов: эту задачу программа берет на себя. Если список дополняется новыми пунктами или укорачивается, нумерация корректируется автоматически. В случае ненумерованных списков программа ставит перед каждым пунктом маркеры: кружки, прямоугольники, ромбы или другие изображения. В результате список принимает удобочитаемый, «фирменный» вид. Теги для создания списков можно условно разделить на две группы: одни определяют общий вид списка (и позволяют указывать атрибуты), а другие задают его внутреннюю структуру. В списках можно использовать стандартные атрибуты. Существует несколько разновидностей списков.

<UL> <LI>

Самым простым является ненумерованный список (unordered list). Его шаблон представлен ниже:

```
<UL>
<LI>Пункт 1 списка
<LI>Пункт 2 списка
<LI>Пункт 3 списка
</ul>
```

Элемент UL является своеобразным обрамлением списка. Он позволяет отделять один список от другого. Элемент LI обозначает каждый из пунктов. Вид ненумерованного списка показан на рис. 3.5.

- Пункт 1 списка
- Пункт 2 списка
- Пункт 3 списка

Рис. 3.5. Ненумерованный список

<OL> <LI>

Структура нумерованного списка (ordered list) похожа на предыдущую:

```
<OLtype="I">
<LI>Пункт 1
```

```
<LI>Пункт 2
<LI>Пункт 3
<LI>Пункт 4
</ol>
```

Для него используется другой внешний тег: OL. В этом случае каждый пункт маркируется элементом упорядоченной последовательности: арабскими или римскими числами, буквами латинского алфавита. На рис. 3.6 показан образец списка по приведенному выше примеру.

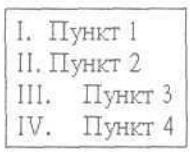


Рис. 3.6. Нумерованный список

Способ нумерации задается при помощи атрибута type. В табл. 3.2 приведены все способы нумерации.

Таблица 3.2. Значения атрибута type

Атрибут	Вид нумерации
type="1"	1, 2, 3, 4, ...
type="i"	i, ii, iii, iv, ...
type="I"	I, II, III, IV, ...
type="a"	a, b, c, d, ...
type="A"	A, B, C, D, ...

Существует атрибут, который позволяет задать начальное значение для нумерации списка:

```
start=номер
```

Например, если использована буквенная нумерация, то значение атрибута start=4 означает, что нумерация списка начнется с литеры D.

Для элемента LI может быть использован атрибут, который определит номер для текущего пункта списка:

```
value=номер
```

Соответственно, следующий пункт списка будет иметь очередной номер и т. д. При помощи атрибута value, если использовать его для первого в списке элемента LI, можно добиться того же эффекта, что и при помощи атрибута start, или нарушить последовательность номеров, если переопределить другие элементы списка.

<DL> <DT> <DD> </dl>

Списки с определениями (definition list) создаются при помощи тегов трех видов:

```
<DL>
<DT>Пункт 1
<DD>Определение пункта 1
<DD>Другое определение пункта 1
<DT>Пункт 2
<DD>Определение пункта 2
<DT>Пункт 3
<DD>Определение пункта 3
</dl>
```

Лучше всего дает представление о смысле списка с определениями рис. 3.7. Каждый пункт списка может быть дополнен одним или несколькими блоками текста при помощи тега (тегов) DD. Каждый блок автоматически размещается с новой строки. Термин «определение» носит условный характер. Абзацы, размещенные в списке, могут быть определениями, дополнениями, разъяснениями пунктов. По сути, пункт представляет собой заголовок, а определение — произвольный текст под заголовком.

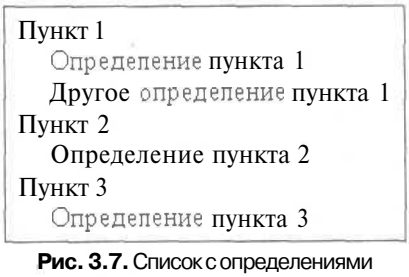


Рис. 3.7. Список с определениями

Другой способ создания сложных списков — использование принципа вложения. Каждый элемент, определяющий пункт списка, может содержать еще один список. Вложенный список располагается с небольшим отступом вправо относительно списка верхнего уровня. Каждый из списков может быть создан и отлажен отдельно, а затем все составляющие могут быть объединены в один большой список. В том случае, когда структура списка ясна, удобно воспользоваться шаблоном. В нем должны быть указаны все пункты на всех уровнях. Доработка такого шаблона сводится к вписыванию конкретных строк текста в соответствующие пункты. Ниже дан пример такого шаблона:

```
<UL>
<LI>Пункт 1
<OL>
<LI>Пункт 1.1
<LI>Пункт 1.2
```

```
</ol>
<LI>Пункт 2
<OL>
<LI>Пункт 2.1
<LI>Пункт 2.2
</ol>
<LI>Пункт 3
<OL>
<LI>Пункт 3.1
<LI>Пункт 3.2
</ol>
</ul>
```

В этом шаблоне нумерованные списки размещены внутри пунктов ненумерованного списка. Рис. 3.8 дает представление о том, как выглядит такая конструкция.

- Пункт 1
  1. Пункт 1.1
  2. Пункт 1.2
- Пункт 2
  1. Пункт 2.1
  2. Пункт 2.2
- Пункт 3
  1. Пункт 3.1
  2. Пункт 3.2

Рис. 3.8. Вложенные списки

В таблицах стилей для списков предусмотрены специальные свойства (см. приложение Б). Например, в секции страницы HEAD можно определить стиль списка с маркерами в виде окружностей (circle):

```
<STYLE type="text/css">
<OL>c01 {list-style-type: circle}
</style>
```

Тогда в секции BODY может присутствовать список с маркерами в виде кружков:

```
<OL class="c01">
<LI> Пункт 1
<LI> Пункт 2
</ol>
```

ПРИМЕЧАНИЕ

Примеры списков можно найти в файле List.htm на прилагаемой дискете.

Гиперссылки

<A>

Один из самых важных элементов языка, обеспечивающий создание гиперссылок. Чаще всего используется такой шаблон:

Произвольный текст <A href="*Адрес ссылки*"> *текст для щелчка*

Или такой:

<A href="*Адрес ссылки*"> <IMG src="*Ссылка на рисунок*">

Первый шаблон используется в том случае, когда гиперссылка встречается в тексте. Атрибут href может указывать на ресурс Интернета, файл на локальном диске или на *метку* внутри текущей страницы. Текст, расположенный внутри элемента A, представляет собой видимую часть гиперссылки. На нем должен щелкнуть пользователь, чтобы осуществить переход. Броузер выделяет этот фрагмент цветом, а после использования гиперссылки меняет цвет, чтобы обеспечить подсказку.

Второй шаблон задается в том случае, когда видимая часть гиперссылки представляет собой рисунок. Если для последнего определена рамка, то она тоже меняет цвет после использования гиперссылки. Если ссылка указывает на рисунок, который находится на локальном диске, она обязательно должна начинаться со слова file, то есть содержать указание на протокол:

href="file://*Диск**Путь к файлу*"

или

href="file:///*Диск*:/*Путь к файлу*"

По умолчанию используется ссылка на файлы текущей папки (той, где расположен файл Web-страницы). В этом случае просто указывается имя файла, например: page2.html, strelka.gif, photo35.jpg.

Часто используются относительные ссылки на папки: это позволяет легко менять местоположение комплекса страниц на диске. Если в текущей папке есть другая, в которой размещены необходимые файлы, ссылка строится по такому шаблону:

href="./*Папка*/*Файл*.*тип*"

Здесь на структуру вложенных папок указывает точка перед наклонной чертой. Если необходимо указать папку, которая находится на том же уровне вложения, что и текущая, то добавляют еще одну точку:

href="../*Папка*/*Файл*.*тип*"

Подобно многим другим элементам языка, элемент A требует использования атрибутов. Атрибут гиперссылки мы уже знаем, шаблон его таков:

href="*URL*"

или

href="*Протокол*://*Адрес ссылки*"

Например:

```
href="http://www.netscape.com"
```

Кодовое слово, стоящее в начале URL, обозначает так называемую схему доступа. Она определяет тип сервера, доступный при помощи данной ссылки. Для пользователя это представляется как доступ к одной из разновидностей Интернета. В этом смысле можно сказать, что Интернет — это как бы несколько сетей в одной. У каждой из этих частных сетей существуют свои правила доступа, достоинства, недостатки, свои приверженцы и противники. Но все пользователи используют одни и те же каналы связи. Похожая ситуация наблюдается и в обычных телефонных сетях. Они могут служить для связи голосом, передачи факсов, межкомпьютерной связи и т. д.

WWW как самая современная система, должна обеспечивать совместимость с более старыми системами, поэтому от старых протоколов не отказываются, а стараются приспособить их к современным нуждам (например ftp). Существуют следующие схемы доступа:

- file — доступ к файлу на локальном диске;
- ftp — доступ к архивам файлов по протоколу передачи файлов (file transfer protocol);
- http — доступ к WWW;
- mailto — отправка сообщения по электронной почте;
- news — доступ к новостям USENET;
- nntp — доступ к новостям USENET по протоколу NNTP;
- telnet — подключение по протоколу telnet;
- wais — подключение к системе поиска WAIS.

Когда гиперссылка используется для указания адреса электронной почты, ее выбор обеспечивает не переход к новому документу, а запуск диалога для отправки сообщения указанному адресату. Обычно такую ссылку размещают в конце страницы для обеспечения связи с Web-мастером или автором страницы. Для своей страницы я бы мог составить такую ссылку:

```
<A href="mailto:goncharov@online.ru">Алексей Гончаров</A>
```

В том случае, когда используются переходы внутри текущей страницы, на ней должны быть расставлены метки:

```
<A name="Метка"> </a>
```

В больших сайтах часто используются метки для перехода к определенной части некоторой страницы:

```
<A name=" http://Адрес/Файл.html#метка "> </a>
```

Для перехода к метке используется ссылка по такому шаблону (пример приведен в разделе «Анатомия Web-страницы» главы 2):

```
Текст подсказки <A href="#Метка"> Текст для щелчка </a>
```

Для элемента A предусмотрены различные атрибуты. Атрибут hreflang, по аналогии с атрибутом lang, позволяет указать язык, который используется на адресуемой странице.

В структуру гиперссылок заложена возможность создания сложных текстовых документов, доступных через Интернет. Предполагается, что такие документы будут состоять из многих HTML-страниц с перекрестными ссылками. Чтобы пользователь мог эффективно управлять документом, броузер должен оптимизировать работу с отдельными страницами, например, загружать страницы, которые могут понадобиться пользователю, в фоновом режиме. Для этого необходимо снабдить страницы информацией о назначении ссылок.

Для решения этой задачи гиперссылки подразделяются на прямые (forward) и обратные (reverse). Ссылка, вызывающая переход с текущей страницы, называется прямой. Соответственно, при помощи броузера или другой ссылки может быть выполнен и обратный переход. Для определения более точного типа ссылки используются два атрибута (один для прямых, другой — для обратных ссылок).

```
rel="Тип прямой ссылки"
rev="Тип обратной ссылки"
```

Определены следующие стандартные типы ссылок:

- alternate — другая версия документа;
- stylesheet — таблица стилей в виде отдельного файла;
- start — первая страница в структуре документа;
- next — следующая (в смысле выполнения переходов) страница;
- prev — предыдущая (в смысле выполнения переходов) страница;
- contents — страница, на которой находится оглавление всего документа;
- index — страница, на которой находится алфавитный указатель;
- glossary — страница, на которой находится словарь терминов;
- copyright — информация об авторских правах на документ;
- chapter — признак главы документа;
- section — признак раздела документа;
- subsection — признак подраздела документа;
- appendix — признак приложения документа;
- help — справочные данные документа;
- bookmark — закладка внутри документа.

Существуют атрибуты, которые характерны только для определенных конструкций. Атрибуты Shape и coords используются в *картах* (см. раздел «Рисунки и карты» главы 4). Атрибут target бывает весьма полезным при создании *фреймов* (см. раздел «Фреймы» текущей главы). Атрибуты accesskey и tabindex можно указывать, если элемент A входит в состав *форм* (см. раздел «Элементы форм» главы 4).

Элемент A позволяет использовать и стандартные атрибуты: id, class, lang, dir, title, type, style, атрибуты событий.

<LINK>

В отличие от атрибута A, который указывается в тексте страницы, элемент LINK используется в заголовке страницы, то есть внутри элемента HEAD.

```
<HEAD>
<TITLE>Глава  1</title>
<LINK rel="prev" href="Введение.htm">
<LINK rel="next" href="Глава2.htm">
<LINK rel="index" href="Указатель.htm">
</head>
```

Элемент LINK не создает гиперссылок в тексте страницы, поэтому для определения объекта, на котором можно щелкнуть мышью, необходимо использовать элемент A с атрибутом href, который имеет то же назначение, что и в элементе LINK.

Атрибуты используются, в основном, такие же, как и в элементе A: charset, href, hreflang, id, class, lang, dir, media, rel, rev, style, target, title, type, атрибуты событий.

Таблицы

Таблицы являются очень удобным средством форматирования данных на Web-странице. Основное удобство заключается в том, что броузер берет на себя заботу о прорисовке рамки таблицы. Размер рамки может быть автоматически согласован с размером окна просмотра в броузере и, разумеется, с размером находящихся в ячейках таблицы строк текста и рисунков. Кроме того, таблицы позволяют решать чисто дизайнерские задачи: выравнивать части страницы друг относительно друга, размещать рядом рисунки и текст, управлять цветовым оформлением и т. д. При создании таблиц используется принцип вложения: внутри основного элемента таблицы (TABLE) создается ряд элементов, определяющих строки (TR), а внутри этих элементов размещаются элементы для описания каждой ячейки в строке (TD,TH).

Чтобы разобраться в структуре существующей таблицы или создать новую таблицу, необходимо помнить, что последовательность элементов описывает таблицу сверху вниз и справа налево. Например, если после элемента TABLE указан элемент TR, это означает, что начинается описание новой строки таблицы. Все, что расположено за этим элементом, будет размещено в одной строке (справа налево). Это может быть последовательность элементов TD (ячеек), другая таблица и т. д. После того как встретится новый элемент TR, начнется описание следующей строки, и т. д. до конца таблицы (тега </table>).

<TABLE> </table>

Внешний элемент таблицы. Он позволяет задавать общие свойства таблицы и отделяет структуру таблицы от остальной части Web-страницы. Рассмотрим атрибуты этого элемента. Большинство атрибутов могут использоваться и в других элементах таблицы.

Таблицу можно выровнять по горизонтали при помощи атрибута align:

- align="left" — влево;
- align="center" - по центру;
- align=" right" — вправо.

Ширину таблицы можно задать точно в пикселах или в процентном отношении к ширине страницы в окне броузера. Например:

```
width=400
width=50%
```

Для управления видом рамки используются два атрибута. Дело в том, что броузер создает изображение рамки, имитируя ее трехмерность (выпуклость) при помощи различия в освещенности граней. На рамке можно различить фронтальную и боковую наклонную грани (рис. 3.9).

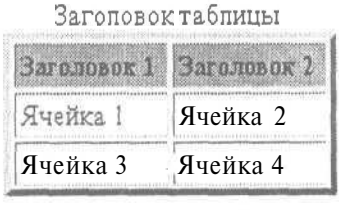


Рис. 3.9. Пример таблицы

Шириной боковой грани управляет атрибут:

```
border=ширина
```

При задании нулевого значения для этого атрибута рамка исчезает совсем.

Шириной фронтальной грани управляет атрибут:

```
cellspacing=ширина
```

Если значение этого атрибута равно нулю, рамка получается тонкой, заостренной.

Для всех ячеек таблицы можно задать размер пустого пространства, окружающего данные в ячейках:

```
cellpadding=число-пикселов
```

или

```
cellpadding="15%"
```

Задание этого атрибута делает ячейки больше. Между рамкой таблицы и данными всегда сохраняется определенное расстояние. В некоторых случаях это позволяет улучшить восприятие таблицы, сделать текст в ячейках легко читаемым.

Для всей таблицы может быть задан цвет фона:

```
bgcolor="Цвет"
bgcolor=#RRGGBB
```

Вместо цвета допускается использовать рисунок:

```
background="Файл"
```

Атрибуты bgcolor и background можно указывать и с другими элементами таблицы, кроме элемента CAPTION.

Атрибут frame (используемый только для элемента TABLE) позволяет задать вид рамки таблицы:

```
frame="параметр"
```

Существуют следующие стандартные параметры:

- void — рамка отсутствует;
- above — верхняя сторона рамки;
- below — нижняя сторона рамки;
- hsides — части рамки сверху и снизу;
- vsides — части рамки слева и справа;
- lhs — левая часть рамки;
- rhs — правая часть рамки;
- border или box — рамка показана полностью.

Разумеется, если атрибут frame отсутствует, рамка вокруг таблицы выводится целиком.

Атрибут rules определяет вид сетки таблицы внутри, то есть между ячейками. Он тоже имеет несколько параметров:

- none — сетка отсутствует;
- groups — сетка вокруг групп ячеек;
- rows — горизонтальные линии между строками;
- cols — вертикальные линии между колонками;
- all — обычная сетка.

Существует атрибут комментария к таблице. Его текст не выводится на экран и может воспроизводиться только специальными программными средствами (например, программой-синтезатором речи):

```
summary="Текст комментария"
```

Допустимые стандартные атрибуты: id, class, lang, dir, title, style, атрибуты событий.

<CAPTION> </caption>

Элемент для задания заголовка таблицы. Несмотря на то что этот элемент располагается внутри элемента TABLE, заголовок выводится на экране вне рамки таблицы (см. рис. 3.9). Положением заголовка можно управлять:

- align="top" — заголовок над таблицей;
- align="bottom" — заголовок под таблицей;

- align="left" — заголовок вверху и выровнен влево;
- align=" right" — заголовок вверху и выровнен вправо.

Другие атрибуты: id, class, lang, dir, title, style, атрибуты событий.

Выравнивание данных в ячейках

Существует набор атрибутов, предназначенных для выравнивания данных в ячейках таблиц. Атрибут align позволяет выравнивать данные в ячейках по горизонтали. Он принимает следующие значения:

- left — выравнивание влево;
- right — выравнивание вправо;
- center — центрирование.

Атрибут valign позволяет выравнивать текст по вертикали. Значения могут быть такие:

- top — выравнивание по верхнему краю ячейки;
- bottom — выравнивание по нижнему краю ячейки (не всегда работает);
- center — выравнивание по центру;
- baseline — выравнивание по первой строке.

Для примера можно продемонстрировать использование значения baseline:

```
<TR valign="baseline"> <TD>Строка 1<br>Строка 2 <TD>Ячейка 2
```

Здесь первая ячейка содержит две строки текста, а вторая — одну. С помощью атрибута valign строки Строка 1 и Ячейка 2 будут расположены на одном уровне.

<TR>

Элемент, создающий строку таблицы. Он не имеет конечного тега. Строка заканчивается там, где начинается следующая, то есть следующий элемент TR. Внутри элемента располагаются элементы TH и TD, определяющие одиночные ячейки. Для выравнивания содержимого всех ячеек в строке можно использовать хорошо известный нам атрибут align и присваивать ему значения left, center и right.

Кроме этого, содержимое ячеек можно выравнивать по вертикали:

- valign="top" - по верхнему краю;
- valign="center" - по центру;
- valign="bottom" — по нижнему краю.

Другие допустимые атрибуты: id, class, lang, style, dir, title, char, charoff, атрибуты событий.

<TH>

Элемент ячейки, которая является заголовком столбца или строки таблицы. Этот элемент должен располагаться внутри элемента TR. Ячейка-заголовок отличается

от обычной тем, что броузер выводит текст внутри нее выделенным (как правило, полужирным) шрифтом. Для элемента ячейки предусмотрено несколько атрибутов.

Если в ячейку введено большое количество текста, броузер разбивает его на строки так, чтобы сохранить требуемую конфигурацию таблицы. Конфигурацию может определять заданная фиксированная ширина таблицы, необходимость согласовать размер таблицы и области просмотра, заданная ширина ячейки. При помощи атрибута nowrap (он не имеет параметров) можно запретить форматирование текста. В этом случае в ячейке будет создана одна строка, а таблица может уйти за край окна.

Атрибуты rowspan и colspan позволяют создавать ячейки, которые в несколько раз больше других ячеек таблицы. Иными словами, ячейки в таблице можно объединять.

При задании атрибута

```
rowspan=n
```

и условии, что n>1, соответствующая ячейка займет не одну, а n строк, и, соответственно, будет иметь размер в n раз больший, чем обычная ячейка данного столбца.

Аналогично, при помощи атрибута colspan можно создавать ячейки, расположенные сразу в нескольких столбцах. Подробнее об объединении ячеек рассказывается в главе 6.

Хорошо известный нам атрибут align используется и для одной ячейки. Он может принимать значения left (выравнивание по левому краю), center (выравнивание по центру) и right (выравнивание по правому краю). Обычно по умолчанию используется выравнивание влево. Элемент TH в этом смысле — исключение. Он обеспечивает центрирование текста, если атрибут align отсутствует.

Для элемента TH можно указать атрибут valign таким же образом, как и для элемента TR.

Размеры ячеек можно задавать точно:

```
width=ширина
height=высота
```

<TD>

Этот элемент определяет обычную ячейку таблицы. Для него допустимы те же атрибуты, что и для элемента TH.

Оба элемента — TH и TD — могут не иметь конечных тегов. Функцию конечного тега выполняет следующий элемент, который определяет структуру таблицы.

Теперь, зная, какие элементы используются для создания таблицы, мы можем создать простейшую таблицу:

```
<TABLE  border=4  cellspacing=3>
<CAPTION> Заголовок таблицы </caption>
<TR><TH bgcolor="yellow">Заголовок 1
<TH bgcolor="yellow">Заголовок 2
```

```
<TR><TD>Ячейка 1
<TD>Ячейка 2
<TR><TD>Ячейка 3
<TD>Ячейка 4
</table>
```

Из примера видно, что первая строка таблицы содержит только ячейки-заголовки. Внешний вид таблицы показан на рис. 3.9. Текст, расположенный после элементов TD, представляет собой содержимое ячейки. Таблица может форматироваться автоматически (если не заданы атрибуты) с учетом объема данных в ячейках. Последний пример можно несколько усложнить. При необходимости можно создать заголовки как для столбцов, так и для строк:

```
<TABLE border=4 cellspacing=3>
<CAPTION> Заголовок таблицы </caption>
<TR><TH bgcolor="yellow">
<TH bgcolor="yellow">Заголовок 1
<TH bgcolor="yellow">Заголовок 2
<TR><TH bgcolor="yellow">Заголовок 3
<TD>Ячейка 1
<TD>Ячейка 2
<TR><TH bgcolor="yellow">Заголовок 4
<TD>Ячейка 3
<TD>Ячейка 4
</table>
```

Эта таблица показана на рис. 3.10. Обратите внимание: несмотря на то, что левая верхняя ячейка не используется, для нее задан цвет фона так же, как и для других ячеек-заголовков. Это необходимо сделать для того, чтобы рамка таблицы в этом месте была правильно прорисована.

Заголовок таблицы

	Заголовок 1	Заголовок 2
Заголовок 3	Ячейка 1	Ячейка 2
Заголовок 4	Ячейка 3	Ячейка 4

Рис. 3.10. Таблица с заголовками столбцов и строк

Еще один пример таблицы. В некоторых случаях возникает необходимость объединения ячеек. Тогда можно использовать атрибуты rowspan и colspan, как показано в этом примере:

```
<TABLE border="4" cellspacing=3 background="fon01.gif">
<CAPTION>Таблица с объединенными ячейками </caption>
<TR><TH rowspan="2"> <TH colspan="2">Заголовок 1
```

```
<TR><TH>Заголовок 1.1<TH>Заголовок 1.2
<TR><TH>Заголовок 2<TD>Ячейка 1<TD>Ячейка 2
<TR><TH>Заголовок 3<TD>Ячейка 3<TD>Ячейка 4
</table>
```

Эта таблица показана на рис. 3.11. Обратите внимание, что в ячейке, не содержащей текста, помещен символ неразрывного пробела . Это необходимо для того, чтобы сетка таблицы была правильно прорисована.

Таблица с объединенными ячейками

Рис. 3.11. Таблица с объединенными ячейками

Группы строк: <THEAD>, <TFOOT> и <TBODY>

Существует возможность группировки строк таблицы. Для этого предусмотрен элемент блока заголовка THEAD, элемент обычных блоков строк TBODY и элемент нижнего блока строк TFOOT. В каждом блоке может присутствовать любое количество строк (элементов TR). Эти три элемента могут использоваться как с конечными тегами, так и без них. В качестве примера можно привести шаблон таблицы:

```
<TABLE border=2>
<THEAD>
<TR> <TD>Заголовок 1<TD>Заголовок 2
<TFOOT>
<TR> <TD>Нижний блок таблицы<TD> 
<TBODY>
<TR> <TD>Строка 1 <TD>Ячейка 1.2
<TR> <TD>Строка 2 <TD>Ячейка 2.2
<TBODY>
<TR> <TD>Строка 3 <TD>Ячейка 3.2
<TR> <TD>Строка 4 <TD>Ячейка 4.2
<TR> <TD>Строка 5 <TD>Ячейка 5.2
</table>
```

В данной таблице отсутствует форматирование (рис. 3.1,2), поэтому внешний вид ее очень простой. С другой стороны, элементы групп строк дают возможность для определения дополнительных стилей.

Заголовок 1	Заголовок 2
Строка 1	Ячейка 1.2
Строка 2	Ячейка 2.2
Строка 3	Ячейка 3.2
Строка 4	Ячейка 4.2
Строка 5	Ячейка 5.2
Нижний блок таблицы	

Рис. 3.12. Таблица с группами строк

При использовании этих элементов надо придерживаться следующих правил.

- В таблице можно указывать по одному элементу THEAD и TFOOT, но несколько элементов TBODY.
- Последовательность задания элементов следующая: THEAD, TFOOT, TBODY. Но в таблице на экране блок TFOOT окажется самым нижним.
- Все блоки должны содержать одинаковое количество колонок.

Группы колонок: <COLGROUP> и <COL>

Элемент COLGROUP позволяет создавать группы колонок с одинаковыми свойствами. Рассмотрим пример таблицы:

```
<TABLE border=4>
<COLGROUP span=1 width="30" bgcolor="lime">
<COLGROUP  bgcolor="yellow">
<COL span=2 width="30">
<COL width="60">
<COLGROUP bgcolor="aqua">
<COL width="50">
<TR><TD> 1-1 <TD> 1-2 <TD> 1-3 <TD> 1-4 <TD> 1-5
<TR><TD> 2-1 <TD> 2-2 <TD> 2-3 <TD> 2-4 <TD> 2-5
</table>
```

Ее внешний вид показан на рис. 3.13. Каждый элемент COLGROUP позволяет назначить свойства определенному числу колонок, задаваемому атрибутом span. Все эти колонки будут одинаковые. Можно также использовать элемент COL для указания свойств одной колонки. Тогда часть свойств будет совпадать для всех колонок, относящихся к одному элементу COLGROUP, а часть может отличаться. В таблице могут быть определены свойства для любого количества колонок, и если реальных колонок будет меньше, то некоторые (последние) определения окажутся невостребованными. Это не является ошибкой. Для задания свойств могут использоваться те же самые атрибуты, что и для других элементов таблицы.

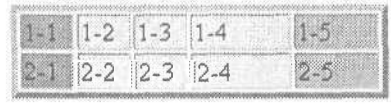


Рис. 3.13. Таблица с элементами COLGROUP и COL

Есть несколько правил задания элементов. В смысле описания свойств элемент COLGROUP обладает более высоким приоритетом, а элементы COL располагаются внутри элементов COLGROUP. В таблице могут присутствовать несколько элементов COLGROUP. Если число колонок в одном таком элементе задается атрибутом span использовать в нем элементы COL не имеет смысла. Если элементы COL существуют, то атрибут span соответствующего элемента COLGROUP игнорируется, то есть число колонок определяется числом элементов COL. Для отдельных элементов COL можно вводить собственные атрибуты span.

ПРИМЕЧАНИЕ

Примеры таблиц можно найти в файле Table.htm на прилагаемой дискете.

Фреймы

<FRAMESET> <FRAME> </frameset>

Фреймы — это области, которые создаются в окне броузера для одновременной демонстрации нескольких документов. Не все броузеры позволяли разделять область просмотра на части, но эта идея завоевала всеобщее признание. Новые версии броузеров поддерживают фреймы в обязательном порядке. При создании страниц с фреймами разрабатывается несколько Web-страниц. При этом HTML-файлы отличаются по типам. Документы *раскладки* (layout) используются для создания структуры окна, то есть для описания того, как оно должно быть разделено. Документы *содержания* (content) предназначены для заполнения информацией каждой из областей, Итак, как же создать Web-страницу с фреймами?

Вначале необходимо продумать, какие области потребуются. Горизонтальное деление экрана задается при помощи атрибута rows, а вертикальное — при помощи атрибута cols. Значения атрибутов могут быть выражены в пикселах или процентах. Кроме того, используется символ * для обозначения оставшейся части экрана. Приведем несколько примеров:

- cols=50%, 50% — деление области просмотра по вертикали пополам (принцип программы Norton Commander);
- cols=25%, 75% — левая вертикальная область в три раза уже правой. Такой стиль избрали многие фирмы, имеющие свои сайты в Интернете;
- rows=150, 30%, * — для верхней горизонтальной области отведено 150 пикселов, для средней — тридцать процентов доступного пространства, а для нижней - все, что останется;
- cols=*, 4* — стиль для любителей головоломок. Правая вертикальная область в четыре раза шире левой. Эту формулу можно записать и так: cols=20%, 80%.

В элементе FRAMESET можно использовать и стандартные атрибуты id, class, title, style, onload, onunload.

Вторым этапом является подготовка отдельных HTML-файлов для каждой области. Они создаются по таким же правилам, что и другие гипертекстовые документы. Нужно только учитывать размер области, в которой они будут демонстрироваться. До тех пор, пока эти файлы не будут созданы, открывать документ раскладки в броузере не имеет смысла: вы ничего не увидите. Кстати, из этого следует одна особенность общения с Интернетом. Если вы набрели на сайт с фреймами и хотите заполучить в свою личную папку понравившийся документ, не пытайтесь сохранять основной HTML-файл. Вместо этого просмотрите его в режиме источника и найдите ссылку на конкретный документ содержания. Затем загляните в папку кэша, где последний и должен находиться.

В документе раскладки секция FRAMESET используется вместо секции BODY.

Атрибуты элемента <FRAME>

Кроме стандартных атрибутов — id, class, title и style — этот элемент имеет ряд атрибутов, позволяющих усовершенствовать оконную структуру.

Внутри элемента FRAMESET должна быть создана ссылка на каждый документ содержания, входящий в сложную страницу. Кроме того, каждый элемент FRAME полезно снабдить *именем* с помощью атрибута name. Имя можно указывать в гиперссылках (см. листинг 3.5). В результате элемент FRAME может выглядеть так:

```
<FRAME src="Имя файла.htm" name="имя фрейма">
```

После того как все страницы загружены, пользователь имеет возможность передвигать границы фреймов при помощи мыши. Атрибут no resize запрещает делать это для определенного фрейма.

Атрибут scrolling управляет прокруткой внутри одной области. Он может принимать значения YES (полосы прокрутки создаются в обязательном порядке), N0 (прокрутка запрещена) и AUTO (полосы прокрутки появляются, когда необходимо). Если этот атрибут отсутствует, броузер создает полосы прокрутки для тех документов, которые не умещаются целиком в отведенных им областях. Запретив прокрутку, можно создать так называемый баннер.

Значением атрибута longdesc является ссылка на другой файл (URL). Таким способом для фрейма создается описание любого объема. Это своего рода альтернатива использованию стандартного атрибута title, при помощи которого обычно задается короткий текстовый комментарий.

При помощи атрибута frameborder указывается, нужна или нет рамка вокруг фрейма. Значение 1 создает рамку, значение 0 — отменяет. Если необходимо убрать границу между фреймами, надо учитывать, что она создается рамками двух смежных областей.

Атрибут marginheight задает величину отступа страницы от верхнего и нижнего краев фрейма. Значение указывается в пикселах, например:

```
marginheight="75"
```

Другой похожий атрибут, marginwidth, создает поля слева и справа, например:

```
marginwidth="10"
```

В качестве примера, поясняющего конструкцию фреймов, рассмотрим шаблон (файл Frame.htm) для создания сложной Web-страницы.

Листинг 3.1. Шаблон страницы с фреймами

```
<HTML>
<HEAD>
<TITLE> Фреймы </title>
</head>
<FRAMESET rows="20%,60%,20%">
<FRAME src="fr1.htm" noresize>
<FRAMESET cols="22%,78%">
<FRAME src="fr2.htm">
<FRAME src="fr3.htm" scrolling="yes" marginwidth="10" marginheight="75">
</frameset>
<FRAME src="fr4.htm" >
</frameset>
</html>
```

В данном примере экран делится на четыре части, как показано на рис. 3.14. Для верхней части страницы запрещено удаление с экрана, а для правой в обязательном порядке создаются полосы прокрутки. Обратите внимание, что для одновременного деления области просмотра по вертикали и по горизонтали следует создать вложенный элемент FRAMESET.

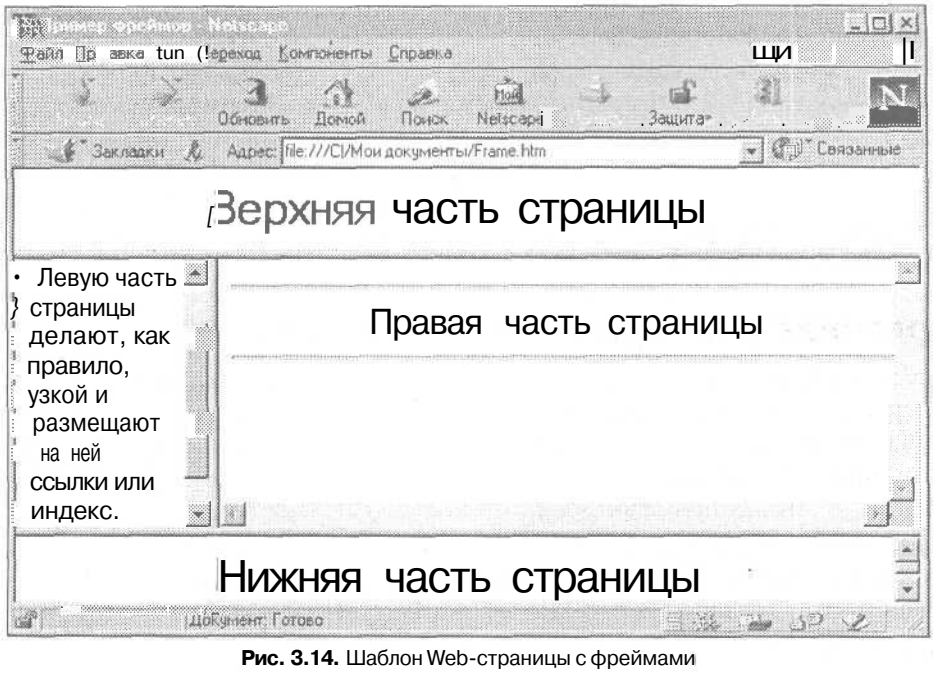


Рис. 3.14. Шаблон Web-страницы с фреймами

ПРИМЕЧАНИЕ

Пример из этого раздела включает файлы Frame.htm, Fr1.htm, Fr2.htm, Fr3.htm, Fr4.htm, записанные на прилагаемой дискете.

<NOFRAMES> </noframes>

Этот элемент используется для того, чтобы предусмотреть ситуацию, когда броузер не поддерживает фреймы. В этом случае надо вывести на экран предупреждающее сообщение или адресовать клиента к другой странице. Фрагмент кода может быть написан следующим образом.

Листинг 3.2. Шаблон страницы с элементом NOFRAMES

```
<HTML>
<HEAD>
<TITLE>Текст заголовка</title>
</head>
<FRAMESET cols="nn%, mm%">
<FRAME src="Страница1.htm">
<FRAME src="Страница2.htm">
<NOFRAMES>
<P>Для просмотра этой страницы необходим броузер,
поддерживающий фреймы</p>
<P>Вы можете посмотреть
<A href="Без-фреймов.htm"> упрощенную версию </a> страницы</p>
</noframes>
</frameset>
</html>
```

Разумеется, броузеры, поддерживающие фреймы, не станут воспроизводить информацию из секции NOFRAMES.

Организация переходов по фреймам

В качестве примера рассмотрим несколько взаимосвязанных страниц (рис. 3.15). Пусть основная страница (Main.htm) не имеет фреймов, а две другие построены по стандартному принципу: слева меню, справа информация. Конфигурацию фреймов в нашем случае задает файл Frame1.htm. Такую структуру имеют многие сайты. Кроме того, «цепочки» страниц удобно использовать для создания виртуальных книг, галерей изображений, то есть там, где пользователю требуется последовательно просматривать ряд страниц. В данном случае важно правильно организовать ссылки.

Ниже приведены листинги используемых файлов.

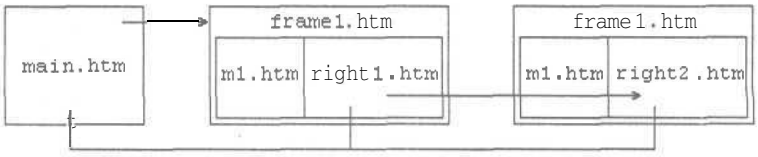


Рис. 3.15. Схема переходов по страницам

Листинг 3.3. Файл Main.htm

```
<HTML>
<HEAD>
<TITLE>Основная страница</title>
</head>
<BODY bgcolor="aqua">
<H2>Основная страница</h2>
<HR>
<A href="frame1.htm"> Следующая страница </a>
<HR>
</body>
</html>
```

Листинг 3.4. Файл Framel.htm

```
<HTML>
<HEAD>
<TITLE>Заголовок для фреймов</title>
</head>
<FRAMESET frameborder=1 framespacing=5 cols="160,*">
<FRAME name="menu01" NORESIZE src="m1.htm">
<FRAME name="info01" src="right1.htm">
<NOFRAMES>
<P>Ваш броузер не поддерживает фреймы
</noframes>
</frameset>
</html>
```

Листинг 3.5. Файл M1.htm

```
<HTML>
<HEAD>
<TITLE>Левый фрейм</title>
</head>
<BODY text="black" bgcolor="gold" link="green" vlink="purple" alink="red" >
```

```
<H3>Меню</h3>
<HR>
<A target=_parent href="main.htm" > Основная </a>
<A target="info01" href="right1.htm" > Правый 1 </a>
<A target="info01" href="right2.htm" > Правый 2 </a>
<HR>
</body>
</html>
```

Листинг 3.6. Файл Right1.htm

```
<HTML>
<HEAD>
<TITLE>Правый фрейм</title>
</head>
<BODY text="black" bgcolor=#9CB3BE link="green" vlink="purple" alink="red">
<H2> Правый фрейм № 1 </h2>
<A href=main.htm target=_parent> Предыдущая страница </a>
    
<A href=right2.htm> Следующая страница </a>
<HR>
</body>
</html>
```

Листинг 3.7. Файл Right2.htm

```
<HTML>
<HEAD>
<TITLE>Правый фрейм</title>
</head>
<BODY text="black" bgcolor="pink" link="green" vlink="purple" alink="red">
<H2>Правый фрейм № 2</h2>
<HR>
<A href=right1.htm> Предыдущая страница </a>
    
<A href=main.htm target=_parent> Основная страница </a>
<HR>
</body>
</html>
```

Теперь разберем, как организованы ссылки. Гиперссылка на основной странице (Main.htm) указывает на документ раскладки: файл Frame1.htm. Он не содержит полезной информации, но позволяет задать размеры областей и указать необходимые документы содержания.

Здесь важно, что каждому фрейму дается имя при помощи атрибута name, например

```
name="info01"
```

Страницы, показываемые внутри фрейма, могут быть разными, а имя останется постоянным и будет использовано в гиперссылках.

В левом фрейме демонстрируется страница M1.htm, которая выполняет функции меню: с его помощью можно вернуться на основную страницу или выбрать страницу для правого фрейма: в нашем примере их две, но можно использовать любое количество. Левый фрейм сделан более узким (рис. 3.16).

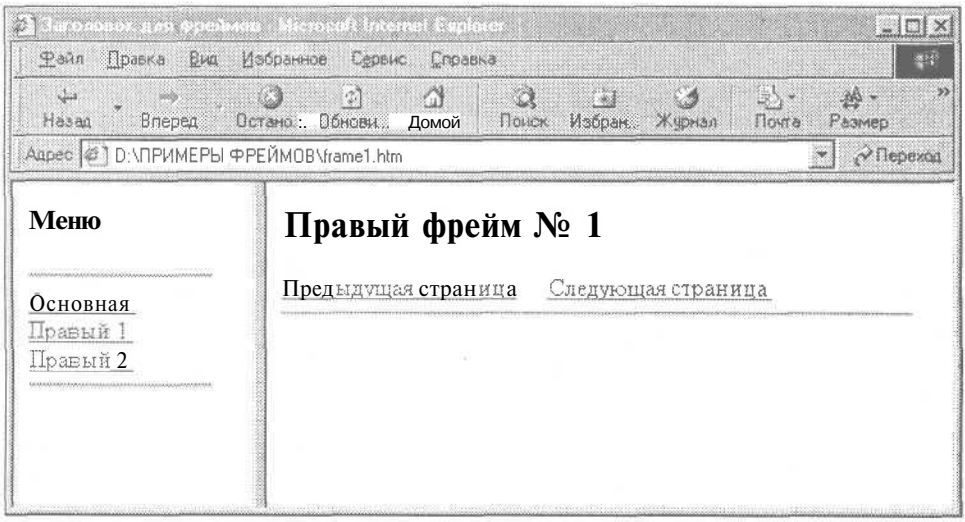


Рис. 3.16. Просмотр файлов M1.htm и Right1.htm

Обратите внимание, что в строке заголовка броузера присутствует заголовок из файла Frame1.htm. Это единственная информация из этого файла, которую мы увидим на экране, но зато все данные из элементов TITLE других страниц будут скрыты.

В реальной задаче файлы Right1.htm и Right2.htm должны быть заполнены полезной информацией. Кроме нее для удобства пользователя можно разместить ссылки типа «переход к следующей странице», «Переход к предыдущей странице», «переход к основной странице». Часто такие ссылки выполняются в виде пиктограмм.

В нашем примере используется *два* независимых способа передвижения по страницам: при помощи ссылок в меню (левом фрейме) и при помощи ссылок в информативных документах (правом фрейме).

Файл M1.htm содержит три гиперссылки. Первая ведет на страницу без фреймов, поэтому в ней обязательно должен быть указан атрибут target со значением _parent. В противном случае при переходе возникла бы ошибка: было бы открыто еще одно окно броузера или страница Main.htm оказалась бы *внутри* фрейма. В общем случае шаблон гиперссылки такой:

```
<A target=_parent href="Имя.htm"> Текст подсказки </a>
```

Другие значения атрибута target:

- _self — как будто атрибута target нет, то есть страница открывается внутри текущего фрейма;
- _top — то же, что и _parent;
- _blank — открытие еще одного окна броузера и показ страницы в нем.

Две другие ссылки позволяют менять содержимое правого фрейма, оставляя левый неизменным. Здесь важно указать атрибут target с именем правого фрейма (используем ссылку в одном фрейме, а страницу загружаем в другой):

```
target="info01"
```

На страницах Right1.htm и Right2.htm гиперссылки построены по такому же принципу, но их синтаксис зависит от того, на страницу какого типа следует выполнить переход. Для перехода на страницу без фреймов используется атрибут:

```
target=_parent
```

Для перехода на «следующую» («предыдущую») страницу внутри текущего фрейма атрибут target не нужен: действует правило умолчания.

ПРИМЕЧАНИЕ

Файлы Main.htm, Frame1.htm, M1.htm, Rightl.htm и Right2.htm имеются на прилагаемой дискете.

<IFRAME> </iframe>

Этот элемент позволяет создать на странице область с полосами прокрутки (inline frame) для демонстрации содержимого другой страницы. Шаблон для использования элемента (просмотра файла Имя файла.htm) такой:

```
<IFRAME src="Имя файла.htm" width="nnn" height="mmm">
Ваш броузер не поддерживает фреймы. Эту страницу можно просмотреть
<A href="Имя файла.htm">отдельно.</a>
</iframe>
```

Атрибуты width и height определяют ширину и высоту фрейма соответственно. Если атрибуты не указаны, используется окно размером 300x150 пикселов.

Атрибут sc rolling управляет полосами прокрутки и может иметь следующие значения:

- auto — броузер определяет необходимость показа полос;
- yes — показ полос прокрутки (вертикальной — обязательно);
- no — без полос прокрутки.

Можно использовать атрибуты target и frameborder, а также другие атрибуты стандартных фреймов:

```
longdesc="URL"
name="имя фрейма"
```

marginwidth="число *пикселов*"
marginheight="*число пикселов*"

Допускается также использование стандартных атрибутов align, class, id, style, title

Устаревшие и нестандартные элементы

Существует ряд элементов, работоспособность которых невозможно гарантировать Их появление обусловлено несколькими причинами. Во-первых, неработающие элементы появляются в результате изменений, вносимых в язык при создании новых версий. Новые броузеры не выводят никаких сообщений об устаревших элементах и игнорируют их при форматировании Web-страниц. Во-вторых, бессмысленность некоторых элементов заключается в том, что результаты форматирования (например изменения атрибутов шрифта) никак не проявляются на экране. Все эффекты, ожидаемые от подобных элементов, можно с успехом получить при помощи других, более популярных. Я думаю, что любой разработчик должен учитывать тот факт, что его Web-страницы будут просматривать при помощи разных программ. Поэтому он должен быть уверен, что документ будет выглядеть так же, как в оригинале.

Строго говоря, понятие версии языка HTML является в достаточной степени условным. Формально на текущий момент существует определенная спецификация языка, но фактически каждый броузер поддерживает свою интерпретацию версии. Элементы, объявленные устаревшими, продолжают использоваться, а фирмы-разработчики стараются вводить в обиход новые, оригинальные элементы.

Сложилась парадоксальная ситуация: спецификация языка как стандарт де-факто разрабатывается в одной организации (в W3 Consortium), а броузеры — в других (фирмах-разработчиках программного обеспечения). Таким образом, версия языка — это, скорее, временное понятие. Можно говорить о состоянии этой области знания только применительно к определенному отрезку времени, то есть периоду декларирования и использования определенной версии HTML. Посмотрим, какие изменения претерпевает язык в процессе своего развития.

<BGSOUND>

Элемент для создания звуковых эффектов помещается в секцию HEAD.

Атрибут src позволяет выбрать звуковой файл. С помощью атрибута loop задается число повторений звукового фрагмента. Обычно используют непрерывное воспроизведение для звукового сопровождения (loop=-1 или loop="infinite") или однократное воспроизведение, если фрагмент является звуковым приветствием или комментарием (loop=1):

<BGSOUND src="*Звук.wav*" loop=1>

Атрибут

volume=*число*

определяет громкость.

Стандартные атрибуты: id, class, lang, title.

<BLINK> </blink>

Элемент, задающий мигание текста. Он был введен в язык, несомненно, из-за существования подобного режима у текстовых мониторов. Компьютерная общественность и фирмы-разработчики давно признали его анахронизмом. Правда, в настоящее время эффект мигания достигается другими способами — при помощи программных надстроек.

<DIR> </dir> и <MENU> </menu>

Элементы, определяющие границы ненумерованного списка. В настоящее время они считаются устаревшими. Броузеры обрабатывают их так же, как элемент UL. Для примера приведу способ использования DIR:

```
<DIR>
<LI>Пункт 1
<LI>Пункт 2
<LI>Пункт 3
</dir>
```

ПРИМЕЧАНИЕ

Примеры использования элементов DIR и MENU можно найти в файле List.htm на прилагаемой дискете.

<XMP> и <LISTING>

Эти элементы еще в версии HTML 3.2 считались устаревшими. Теперь о них никто не вспоминает. Вместо них рекомендуется использовать элемент PRE.

<COMMENT> </comment>

Это контейнер для комментариев. Некоторые броузеры поддерживают его (текст, расположенный внутри этого элемента, не должен быть виден на экране), но в официальной спецификации языка его нет.

Существует одно ограничение: внутри комментария не должны располагаться другие элементы. Так должно быть, разумеется, только в том случае, когда надо, чтобы все содержимое элемента COMMENT не было видно на экране. Если в комментарии будет присутствовать другой элемент, то его содержимое, отформатированное соответствующим образом, может быть выведено на экран. С этой проблемой связана другая, похожая: обычный текст не может содержать фрагменты, имеющие вид тегов (с угловыми скобками).

<PLAINTEXT> </plaintext>

Этот элемент был предназначен для создания текста с конструкциями HTML, которые должны были восприниматься именно как текст. Все теги, заключенные в этом элементе, воспринимаются броузером только как произвольные символы.

Элемент удобен при обсуждении вопросов, связанных с HTML. В настоящее время вместо него надо использовать элемент PRE.

<EMBED> </embed>

Элемент, являющийся контейнером для некоторого объекта. Для функционирования объекта броузер должен быть снабжен соответствующей надстройкой (plug-in). Элемент используется по следующей схеме:

```
<EMBED>
src="Источник данных"
height=высота
width=ширина
attribute_1="Значение первого атрибута"
attribute_2="Значение второго атрибута" ... >
</embed>
```

<NOEMBED> </noembed>

Этот элемент может располагаться внутри элемента EMBED и содержать текстовую подсказку на тот случай, если броузер не может активизировать объект.

<MARQUEE> </marquee>

Элемент, создающий бегущую строку. Сам по себе прием интересен, но наибольшего эффекта можно добиться, если удачно подобрать атрибуты.

Фоновый цвет задается обычным способом:

```
bgcolor="цвет"
```

Если фон задан, то этот элемент выглядит как цветная полоса, вдоль которой бежит текст. Высоту полосы можно регулировать двумя способами:

```
height=высота в пикселах
height=число%
```

Если высота полосы задается в пикселах, то можно порекомендовать задавать ее в диапазоне 30...50. Высоту можно задавать и в процентах. Процент определяет долю от высоты видимой части гипертекста внутри окна броузера. Эта величина, разумеется, не является постоянной и зависит от размера окна. Если высота полосы достаточно большая, имеет смысл использование атрибута align для выравнивания текста по верхнему краю, по середине или по нижнему краю полосы:

```
align="top"
align="middle"
align="bottom"
```

Правда, не все броузеры поддерживают этот атрибут. Вот пример полосы зеленого цвета, высотой 50 пикселов, с выравниванием бегущего текста по середине:

```
<MARQUEE bgcolor="green" height=50 align="middle">
Бегущая строка </marquee>
```

Направление движения строки тоже можно менять:

```
direction="left"
direction="right"
```

Удачным атрибутом, на мой взгляд, оказался тот, который управляет поведением (behavior) строки. По умолчанию создается обычная бегущая строка, какие бывают на табло. Дойдя до края экрана (окна), она уходит из поля зрения, а затем появляется с противоположной стороны. Этот атрибут задается так:

```
behavior="scroll"
```

Второй вариант движения заключается в следующем: строка появляется из-за края окна, достигает противоположного края и останавливается. Атрибут таков:

```
behavior="slide"
```

По третьему сценарию строка не исчезает с экрана, но и не останавливается. Она движется вправо или влево, «отражаясь» от края окна и меняя направление движения. Атрибут в этом случае должен быть задан так:

```
behavior="alternate"
```

Всю полосу можно сдвинуть по горизонтали вправо:

```
hspace=смещение в пикселах
```

Выше и ниже полосы можно создать пустое пространство:

```
vspace=высота в пикселах
```

Количество проходов строки по экрану можно ограничить:

```
loop=число
```

Выполнив необходимое число проходов, строка остановится. В данном случае отсчет начнется только после того, как пользователь увидит на экране бегущую строку.

Скорость движения задает следующий атрибут:

```
scrollamount=число
```

Если число равно 1, то строка будет еле ползти по сравнению с режимом по умолчанию. Если число больше 10, то строка будет двигаться очень быстро. Данный атрибут задает скорость движения как число пикселов, которые проходит строка за каждый шаг.

Существует второй атрибут скорости, определяющий временной интервал (в миллисекундах) между шагами:

```
scrolldelay=число
```

С помощью этого атрибута можно заставить строку двигаться рывками.

ПРИМЕЧАНИЕ

Примеры бегущих строк можно найти в файле Text.htm на прилагаемой дискете.

<HPn> </hpn>

Предполагается, что n = 1, 2, 3... Этот элемент обеспечивает подсветку символов строке. Но он не поддерживается большинством броузеров, поэтому использовать его не рекомендуется.

<BANNER> </banner>

Элемент, который позволяет оставить часть информации на экране вне зависимости от того, как прокручивается документ. Такие детали страницы, называемые баннерами, удобны для размещения логотипов, подсказок, заголовков и т. д. Данный элемент поддерживается не всеми броузерами, и его использование имеет смысл только в том случае, когда заранее известно, что он будет функционировать. Существует более надежный способ создания баннеров: при помощи фреймов.

Глава 4

Объекты и формы

Достаточно трудно определить, что на самом деле является объектом, а что нет. Можно утверждать, что обычный текст объектом не является. С другими деталями документов дело обстоит сложнее. Строго говоря, объектами являются горизонтальные линии, таблицы и окна. Но в HTML существует понятие объекта как некой нестандартной части Web-страницы, которую определил пользователь. К объектам, например, относятся рисунки. Другая большая группа объектов — программы. При помощи программных кодов создаются нестандартные детали страниц. Для их включения в состав страницы используется универсальный элемент OBJECT.

Общие атрибуты объектов

Элементы объектов (IMG, OBJECT, APPLET) позволяют использовать общие атрибуты align, border, width, height, hspace, vspace и ряд стандартных атрибутов.

Как правило, объект занимает прямоугольную область на экране, поэтому одним из самых полезных является атрибут, который определяет ширину рамки (border):

border=*число пикселов*

Рамка нужна не только для красоты. Если объект используется внутри элемента A, то изменение цвета рамки позволяет отличить пройденную гиперссылку от нетронутой.

Размеры объекта можно задавать так:

height=*высота в пикселах*

и

width=*ширина в пикселах*

Можно использовать атрибуты выравнивания:

- align="bottom" — по нижнему краю;
- align="left" — влево;
- align="middle" — по центру;
- align=" right" — вправо;
- align="top" — по верхнему краю.

Размеры объекта могут не совпадать с размерами рамки. Тогда справа и слева о" объекта можно создать пустое пространство:

hspace=*число пикселов*

По аналогии можно создать пустое пространство выше и ниже:

vspace=*число пикселов*

Рисунки и карты

<IMG>

Элемент для создания ссылки на графический файл (image). Он не содержит ко нечного тега — вся необходимая информация задается при помощи атрибутов Этот элемент является универсальным: с его помощью можно использовать изоб ражения в гиперссылках, вставлять картинки в таблицы, просто размещать рисун ки на Web-странице, создавать маркеры, решать задачи дизайна и т. д.

Необходимым атрибутом является src — указатель на файл графики:

src="*Ссылка на файл*"

Ссылка на файл представляет собой URL. В некоторых случаях у пользователя может возникнуть желание скопировать в отдельную папку какую-нибудь Web-страницу или набор страниц из Интернета. Это позволяет в дальнейшем использовать страницы, не подключаясь к Сети. Скопировать HTML-файлы легко: это сделает броузер. Сложнее с рисунками. Сначала их надо найти в папке кэша, скопировать в требуемую папку, а затем откорректировать значения всех атрибутов src, указав для них новый путь.

Очень полезным атрибутом является alt. Он позволяет выводить текст в тех местах, где должны располагаться рисунки. Страница может загружаться достаточно долго, и пока графические файлы на подходе, пользователь должен видеть, какие изображения он сможет получить. В некоторых случаях это позволит ему, не дожидаясь окончания загрузки, прокрутить в окне текущий документ или перейти на другую страницу. Вот несколько примеров:

- alt="My photo" — для фотографии;
- alt="SEND" — если кнопка имеет надпись в виде рисунка;
- alt="www.АДРЕС. com" — если это гиперссылка;
- alt="pict15.gif" —удобно для разработчика страницы.

Высоту и ширину области, в которой демонстрируется рисунок, задают при помощи атрибутов height и width. В том случае, когда задается один из этих атрибутов, рисунок масштабируется таким образом, чтобы его высота или ширина соответствовали заданной. Второй размер устанавливается автоматически, в соответствующей пропорции. Таким образом, применение только одного из атрибутов изменяет оба размера рисунка. Если задать явным образом оба атрибута, то рисунок будет масштабироваться по двум осям в соответствии с заданным размером. Масштабирование, как правило, ухудшает качество изображения.

Рисунок можно снабдить и стандартными атрибутами: class, dir, id, lang, longdesc, style, title, атрибутами событий.

Элемент IMG позволяет использовать изображения, отдельные части которых связаны со ссылками и позволяют выполнять переходы. Такие изображения называются картами (map). В том случае, когда реакцию на щелчок на карте обрабатывает программа, расположенная на сервере, в элемент включается атрибут ismap. Иногда его записывают так:

```
ismap="ismap"
```

Однако задание значения атрибута совершенно не обязательно.

В том случае, когда карта обрабатывается броузером, используется атрибут usemap. Он определяет имя карты:

```
usemap="#Имя"
```

Это имя ставится в соответствие со значением соответствующих атрибутов элементов AREA и MAP (см. ниже), которые определяют конфигурацию карты.

Интересно, что задание атрибутов usemap придает элементу IMG свойства, характерные для элемента A, то есть возможность осуществления перехода. Кроме того, мы сталкиваемся со случаем, когда необходимо обязательное совместное использование сразу трех элементов: AREA, IMG и MAP.

<MAP> <AREA> </map>

Этот элемент необходим для общего определения карты. Внутри него определяются области карты при помощи элементов AREA и задается имя карты при помощи атрибута:

```
name="Имя"
```

Для каждой области карты должен быть создан свой элемент AREA. Он не имеет конечного тега. Этот элемент должен включать атрибут, определяющий ссылку:

```
href="Адрес"
```

Атрибут для задания текста, заменяющего изображение карты, не является обязательным:

```
alt="Текст подсказки"
```

Он необходим для работы текстовых броузеров, но может быть использован как комментарий.

Атрибуты, определяющие форму области на карте, являются обязательными. Существует три стандартных вида областей: круг (circle), прямоугольник (rect) и многоугольник произвольной формы (polygon).

Для круга необходимо задать координаты центра и радиус (r), выраженные в пикселах. Координаты центра отсчитываются от левого края (x) и верхнего края (y) рисунка (рис. 4.1). Шаблон для задания круговой области таков:

```
shape="circle" coords=x, y, r
```

Для определения области произвольной конфигурации задают координаты (x, *y]* каждого из углов многоугольника, который точно или приблизительно соответствует по форме этой области:

```
shape="poly" coords=x1, y1,  x2,y2,  x3, y3...
```

Пример многоугольника показан на рис. 4.1.

При определении прямоугольной области задают координаты верхнего левого и правого нижнего углов прямоугольника:

```
shape="rect" coords=x1, y1,  x2,y2
```

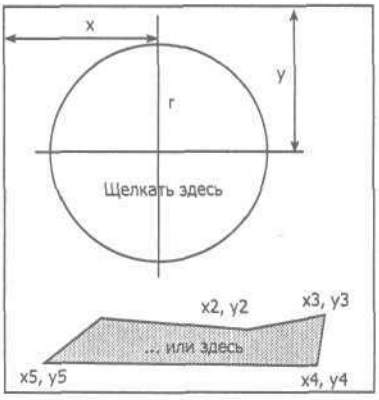


Рис. 4.1. Задание координат на карте

При помощи атрибута nohref (который используется без значений) можно запретить переход по ссылке для определенной области карты.

На листинге 4.1 приведен пример страницы Map.htm, на которой размещены две карты. Карта, имеющая имя karta1, содержит область в виде круга и область произвольной формы. Карта karta2 содержит область прямоугольной формы.

Листинг 4.1. Web-страница с картами

```
<HTML>
<HEAD>
<TITLE>Указатель Web-страниц</title>
</head>
<BODY bgcolor="#FFFFFF" text="#000000" link="#0000FF"
vlink="#FF0000" >
<CENTER><H2>Примеры карт</h2></center>
<A NAME="verh"></a>
<P>Изображения карт иллюстрируют способы задания координат
областей для переходов
<HR>
<H2>Карта 1</h2>
```

```
<MAP name="karta1">
<AREA alt="Круг" shape="circle"
coords="119,114,83" href="http://www.piter-press.ru">
<AREA alt="Многоугольник" shape="poly"
coords="74,242,180,250,249,239,242,278,31,276"
href="Start.htm">
</map>
<MAP name="karta2">
<AREA alt="Переход к карте 2" shape="rect"
coords="27,31,191,101" href="#verh">
</map>
<IMG src="Map1.gif" usemap="#karta1" alt="Карта 1">
<HR>
<P>На этой странице представлены две карты, которые позволяют
выполнять различные переходы. Для правильного функционирования
страницы необходимо проверить все ссылки на файлы, заданные
с помощью атрибутов <B>src</b> и <B>href</b>,
<P>Щелчок по зеленому кругу обеспечит переход к Web-странице
издательства "Питер", Желтый многоугольник вернет вас
на страницу Start.htm. Красный прямоугольник
обеспечивает переход в начало этой страницы,
<HR>
<H2>Карта 2</h2>
<IMG src="Map2.gif" usemap="#karta2" alt="Карта 2">
</body>
</html>
```

Из листинга видно, что описание областей карты и соответствующий элемент IMG могут размещаться в разных частях страницы. Переходы, выполняемые с помощью карты, могут происходить как внутри страницы, так и к удаленному ресурсу,

ПРИМЕЧАНИЕ

Файл Map.htm и необходимые для него изображения можно найти на прилагаемой дискете.

Для обращения к карте можно использовать и элемент OBJECT, например:

```
<OBJECT data="Имя.gif" type="image/gif" usemap="#karta1"> </object>
```

С изображениями карт удобно работать в стандартном для Windows 95 редакторе Paint. Для него изображение должно быть представлено в формате BMP. Этот редактор позволяет использовать сетку в режимах увеличения. Ее можно включить или отключить при помощи комбинации клавиш Ctrl+G. После выбора инструмента Выделение указатель мыши приобретает вид тонкого крестика. Таким образом, положение указателя можно легко установить с точностью до одного

пиксела. В строке состояния редактора будут указаны координаты курсора относительно верхнего левого угла рисунка. Значения координат соответствуют требуемым для атрибута coords величинам и идут в том же порядке (x, y).

Если первоначально изображение создано не в формате GIF, то его можно преобразовать в этот формат, используя любой графический редактор, поддерживающий этот тип файлов. Например, MS Photo Editor, входящий в состав Microsoft Office, или популярный Adobe Photoshop. Достаточно открыть графический файл в редакторе и сохранить его (выполнить команду Сохранить как) в формате GIF, Можно также использовать форматы JPG и PNG.

Реальные карты, созданные с использованием самых разных графических пакетов, смотрятся очень привлекательно. Часто области не имеют четких границ, и неискушенному пользователю «мышечувствительная» карта может показаться последним достижением в области разработки программ или, по крайней мере, хитро придуманным трюком. На самом же деле возможность построения карт была заложена в HTML с самых ранних версий.

Одним из способов применения карты является создание меню страницы. Это обычно горизонтальная полоса, состоящая из цветных прямоугольников с текстом команд. Щелчок на каждом из прямоугольников переводит пользователя на другую страницу или в другую точку текущей страницы. Преимущество графического меню заключается в независимости от используемой кодировки символов: буквы всегда будут видны. Такое меню можно сделать как набор нескольких рисунков, но тогда надо принимать меры к тому, чтобы эти рисунки всегда, при любой конфигурации окна броузера, располагались в ряд. Гораздо проще представить меню в виде одного рисунка-карты и разбить его на несколько прямоугольных областей.

Другой популярной областью использования карт является домашняя страница организации. В этом случае необходимые команды и названия рубрик могут располагаться в произвольном порядке: так, как надо дизайнеру. В качестве примера на рис. 4.2 приведен эскиз подобной карты.

Рис. 4.2. Пример карты

Элементы объектов

<APPLET> </applet>

Этот элемент поддерживается броузерами, в которые встроен интерпретатор Java. Простейший шаблон, который позволит включить в HTML-страницу апплет на языке Java, показан ниже:

```
<APPLET code="Имя файла.class" width=nnn height=mmm>
Произвольный текст комментария
</applet>
```

Атрибут code необходим для задания имени файла, содержащего откомпилированную Java-программу. В отличие от других указателей на ресурсы, значение этого атрибута может быть только относительным, то есть апплет должен располагаться в той же папке, что и страница. Для задания другого базового пути необходимо использовать атрибут codebase. При выполнении апплета создается окно шириной *nnn* и высотой *mmm* пикселов. Внутри элемента APPLET может размещаться произвольный гипертекст. Броузеры, поддерживающие Java, игнорируют все элементы этого гипертекста, включая текст, за исключением элементов PARAM (см. ниже). Броузеры, не поддерживающие Java, игнорируют элементы PARAM и воспроизводят «понятный» для них гипертекст.

Начальный тег элемента APPLET может быть снабжен дополнительными атрибутами.

Атрибут codebase предназначен для указания места расположения апплетов:

```
codebase="URL для апплетов"
```

Если этот атрибут не задан, то по умолчанию используется URL-страница.

Атрибут alt выполняет ту же функцию, что и при выводе изображений. Если броузер не может воспроизвести апплет, то выводится текст, заданный при помощи этого атрибута:

```
alt="Произвольный текст"
```

Атрибут name задает имя апплета. Это необходимо только в том случае, когда несколько апплетов, расположенных на одной странице, взаимодействуют между собой:

```
name="Имя апплета"
```

Атрибуты width и height задают параметры окна апплета: ширину и высоту соответственно.

Поскольку для выполнения апплета создается окно, предусмотрено использование уже хорошо известного нам атрибута align для управления размещением этого окна. Атрибут может иметь следующие назначения: bottom, left, middle, right, top.

Вокруг окна можно создать пустое пространство:

```
vspace=Число пикселов выше и ниже
hspace=Число пикселов слева и справа
```

Допустимые стандартные атрибуты: id, class, title, style.

<OBJECT> </object>

В простейшем случае этот элемент позволяет разместить рисунок:

```
<OBJECT data="Имя файла.png" type="image/png">
```

Но область применения элемента достаточно широка. Он может использоваться, например, для создания окна в документе, то есть выполнять функции элемента IFRAME. Для размещения апплетов рекомендуется указывать элемент OBJECT вместо элемента APPLET:

```
<OBJECT codetype="application/java" classid="java:идентификатор"
width="nnn" height="mmm">
Текстовое описание объекта
</object>
```

Атрибуты:

- classid="*имя объекта*" — уникальный идентификатор объекта или адрес объекта;
- codebase="*URL*" — ссылка на объект или базовый URL, который позволяет указывать местоположение объекта при помощи атрибутов archive, classid или data;
- data="*адрес объекта*" — URL или относительный адрес, позволяющий загрузить объект;
- archive="*список адресов*" — аналог атрибута data, позволяющий указывать несколько адресов объекта;
- codetype="*тип*" — тип объекта (используется совместно с атрибутом classid);
- type="*тип*" — тип (MIME) объекта;
- declare — объект с таким атрибутом только загружается, но не активизируется;
- standby="*текстсообщения*" — текст, который выводится наэкран, пока загружается объект.

В элементе OBJECT можно также использовать атрибуты: id, class, lang, title, dir, style, tabindex, usemap, name, align, width, height, border, hspace, vspace, атрибуты событий.

При размещении объектов необходимо учитывать ситуацию, когда по каким-либо причинам броузер не сможет активизировать объект. На этот случай можно предусмотреть специальное текстовое сообщение, которое размещается внутри элемента, как показано ниже:

```
<OBJECT classid="Идентификатор" data="Адрес/Имя.тип">
Объект не может быть показан
</object>
```

Другим способом реагирования на «аварийную» ситуацию является использование нескольких объектов. При этом один элемент должен быть вложен в другой.

Например, при невозможности активизировать объект можно вставить в документ изображение, которое в какой-то степени будет его заменять. Это делается по следующему шаблону:

```
<OBJECT title="Текст всплывающей подсказки"
classid="Адрес/Имя файла объекта. тип">
<OBJECT title="Другой текст всплывающей подсказки"
data="Имя файла рисунка.gif" type="image/gif">
</object>
</object>
```

<PARAM>

Этот элемент используется для передачи параметров объекту и размещается внутри элемента APPLET или OBJECT. Шаблон его может быть таким:

```
<APPLET code="Имя файла.class" width=nnn height=mmm>
<PARAM name="Имя параметра" value=Значение параметра>
Произвольный текст комментария
</applet>
```

Или таким:

```
<OBJECT classid="Адрес объекта"
standby="Загружаем объект...">
<PARAM name="Имя параметра" value=Значение параметра>
<PARAM name="Имя параметра" value=Значение параметра
valuetype="Тип параметра">
</object>
```

Элемент PARAM должен содержать одну или несколько пар атрибутов, определяющих имя параметра (name) и его значение (value). При выполнении апплета прием параметров в нем осуществляется по следующему шаблону:

```
Переменная=getParameter("Имя параметра")
```

Атрибут valuetype может иметь следующие значения:

- data — параметр передается в качестве строки;
- object — параметр является идентификатором объекта;
- ref — значение параметра является ссылкой (URL).

В том случае, когда значение атрибута valuetype равно ref, с помощью атрибута type можно задать тип параметра:

```
type="тип"
```

Общие атрибуты форм

Предполагается, что форма должна содержать определенное число элементов управления: поля ввода, переключатели, кнопки, флажки и т. д. Каждый элемент управления создается при помощи одного из элементов HTML. Для таких элементов предусмотрены атрибуты, влияющие на работу формы.

Большинство элементов формы может принимать определенные значения. Например, для поля ввода это может быть текст, а для переключателя — номер элемента, который выбран пользователем. Для доступа к этим значениям со стороны программы необходимо использование атрибута name.

В момент активизации формы часть элементов может иметь значения. В текстовое поле может быть введена строка, переключатели могут иметь подписи и т. д. Для задания этих значений используется атрибут value.

Атрибут

```
tabindex=номер
```

позволяет определить, в какой последовательности курсор переходит с поля н; поле при нажатии клавиши Tab. На элемент с атрибутом tabindex=1 устанавлива- ется курсор в момент открытия окна броузера.

В элементах управления, содержащих подписи (например, в меню), часто исполь- зуются сочетания клавиш, позволяющие перевести фокус на определенный эле- мент. Обычно командой служит комбинация клавиши Alt и символа, подчеркну- того в названии команды. В формах HTML тоже можно использовать этот прием при помощи атрибута accesskey. Например:

```
accesskey="R"
```

Атрибут disabled позволяет сделать элемент формы недоступным. Вид элемента при этом не изменяется, но цвет текста становится более бледным. Недоступный элемент нельзя выбрать или изменить его значение.

Существуют атрибуты событий, которые непосредственно связаны с формами:

- onfocus — элемент получает фокус (выбирается);
- onchange — информация элемента изменена;
- onblur — элемент теряет фокус.

Элементы форм

<ISINDEX>

Это самый простой элемент, позволяющий создать подобие формы, то есть конструкции для ведения диалога с пользователем. Он предназначен для ввода строки, содержащей текстовые фрагменты, и генерации *запроса.* Поле ввода можно дополнить строкой подсказки при помощи аргумента prompt:

```
<ISINDEX prompt="Строка для ввода критерия поиска">
```

Данный элемент позволит создать поле ввода, показанное на рис. 4.3. Работа этого элемента связана с определенным на текущей странице базовым URL. Допустим, задан базовый адрес:

```
<BASE href="http://www.название.домен/путь">
```

Значение атрибута href должно представлять собой адрес некоторого поискового средства в Интернете. Если пользователь введет в поле элемента ISINDEX последовательность ключевых слов (слово1, слово2, слово3), то броузер сформирует запрос:

```
http://www.название.домен/?слово1+слово2+слово3
```

Эта строка будет отправлена на сервер для активизации поисковой машины. Теоретически этот метод очень удобен, но на практике его применение ограничивается тем, что не все поисковые программы поддерживают стандартный синтаксис запроса. Я имею в виду использование знаков «?» и «+». Поэтому данный прием годится только для некоторых поисковых серверов.

Стандартные атрибуты: id, class, lang, dir, title, style.

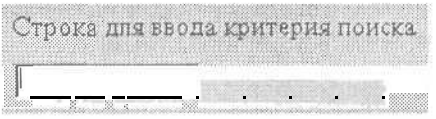


Рис. 4.3. Пример использования элемента ISINDEX

<FORM> </form>

Этот элемент необходим для построения достаточно сложных форм. После заполнения формы и подтверждения ввода со стороны пользователя, введенная информация пересылается на сервер и обрабатывается при помощи CGI-программы, связанной с формой. Атрибут action должен указывать на имя программы, например:

```
<FORM action="http://www.название.домен/имя программы" method="post">
Элементы формы
</form>
```

Одним из вариантов обработки формы может быть пересылка данных по электронной почте:

```
action="mailto:Адрес@сервер.домен"
```

С помощью атрибута method можно задать протокол для пересылки данных на сервер. Протокол GET используется по умолчанию, но в большинстве случаев он не удовлетворяет разработчиков, поэтому чаще используется протокол POST.

Атрибут enctype позволяет указать способ кодирования содержимого формы.

Форму заполняют разные пользователи, поэтому для нее предусмотрен атрибут, который позволяет определить список допустимых кодировок:

```
accept-charset="список кодировок"
```

Можно также определить список допустимых типов данных:

```
accept="список типов данных"
```

Большинство форм снабжаются кнопками, которые позволяют очистить (reset) форму или подтвердить (submit) правильность ее заполнения и отослать данные. Чтобы определить программы-сценарии, которые должны выполняться после указанных действий пользователя, существуют два атрибута событий onsubmit и onreset.

Стандартные атрибуты: id, class, lang, style, dir, title, target, атрибуты событий

<INPUT>

Этот элемент позволяет создавать различные части формы, такие как флажки переключатели, поля ввода. Элемент не имеет конечного тега, так как все параметры задаются при помощи атрибутов.

Вид элемента определяет атрибут type:

- type="text" — создание поля ввода, в котором можно автоматическиразместить произвольный текст, используя атрибут value;
- type="password" — создание поля для ввода пароля, причем введенная информация отображается звездочками;
- type="checkbox" — создание флажка;
- type="radio" — определение одного переключателя. Для создания группы переключателей необходимо использовать несколько элементов INPUT. Вот пример группы из трех переключателей:

```
<H3> Переключатели </h3>
<INPUT type="radio" name="S001" value="Первый">
<INPUT type="radio" name="S001" value="Второй">
<INPUT type="radio" name="S001" value="Третий" checked>
```

Атрибут checked определяет, какой из переключателей должен быть выбран по умолчанию. На рис. 4.4 показан внешний вид этой группы переключателей.

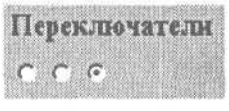


Рис. 4.4. Группа переключателей

- type="button" — создание кнопки произвольного назначения;
- type="submit" — создание кнопки, щелчок на которой подтверждает ввод информации в форму. Атрибут value используется для определения надписи на кнопке;
- type="reset" — тоже кнопка, но для отмены ввода данных в форму;

- type="image" — создание кнопки с рисунком. Для указания графического файла используется атрибут src. Атрибут align предназначен для позиционирования кнопки с рисунком. Значения атрибута уже неоднократно упоминались: bottom, left, middle, right, top. Пользоваться этим атрибутом в данном случае я не рекомендую, так как не все броузеры его поддерживают;
- type="file" — средство выбора файла для присоединения к форме. Пользователю предлагается записать имя файла в поле ввода. Кроме того, броузер автоматически создает рядом с полем ввода кнопку Обзор, которая позволяет запустить стандартный (для операционной системы) диалог выбора файлов;
- type="hidden" — скрытый от пользователя элемент. Такие элементы используются для того, чтобы включить в набор данных формы некую фиксированную информацию. По сути, это определение имени переменной и ее значения.

Остальные атрибуты необходимы для определения свойств элемента. Многие из них являются обязательными, так как обеспечивают обработку данных формы на стороне сервера.

Атрибут name должен присутствовать во всех элементах INPUT, кроме кнопок подтверждения и сброса. Значение этого атрибута определяет имя поля формы, то есть блока данных, введенных в это поле. Программа сервера по этому имени может выделить необходимые данные.

Область применения атрибута value нам уже известна. Значение атрибута задает значение по умолчанию для поля ввода или определяет надпись на кнопке.

Выше было показано, как с помощью атрибута checked создать группу переключателей. Точно таким же способом этот атрибут может быть использован и для флажков. Его наличие показывает, что флажок должен быть установлен по умолчанию. В отличие от переключателей, одновременно может быть установлено любое количество флажков.

Атрибут size позволяет задать длину поля ввода. Длина выражается в символах, но эта величина может быть задана только приближенно. Чтобы разместить в поле ввода определенное количество символов, потребуется подбор значения атрибута. При этом никто не может дать гарантию, что все броузеры обеспечат требуемую длину строки, а не меньшую. Таким образом, длину поля ввода необходимо выбирать с запасом.

Атрибут maxlenght может быть использован двумя способами. Во-первых, он определяет максимальную длину строки, которая может быть записана в поле ввода. Во-вторых, с его помощью можно ограничить размер файла, присоединяемого к форме.

Атрибут readonly позволяет создать элемент, недоступный для редактирования.

Атрибут usemap (см. выше раздел «Рисунки и карты») может использоваться, если в форме создается карта.

Так же, как и в FORM, в элементе INPUT можно указывать атрибут accept.

Допустимые общие атрибуты: accesskey, tabindex, readonly, disabled.

Стандартные атрибуты: id, class, lang, title, dir, style, атрибуты событий.

Используя элементы FORM и INPUT, можно создать объект, который заменит аналогичный объект, созданный на основе элемента ISINDEX. Приведенный ниже код выполняет те же функции, что и код, посвященный элементу ISINDEX (см. рис. 4.3):

```
<FORM action="действие" method="post">
<P>Строка для ввода критерия поиска <BR>
<INPUT type="text">
</form>
```

<LABEL></label>

Этот элемент является контейнером для других элементов формы. Например можно объединить надпись и поле ввода:

```
<LABEL>Ваш адрес: <INPUT type="text" id="adres"></label>
```

Если элемент LABEL и другой элемент находятся отдельно, используется атрибут for, значение которого должно совпадать со значением атрибута id соответствующего элемента:

```
<LABEL for="adress">Ваш адрес: </label>
<INPUT type="text" id="adress">
```

Для каждого элемента LABEL создается один элемент формы. Иногда элементы удобно распределить по ячейкам таблицы, обеспечив тем самым выравнивание данных в форме.

Стандартные атрибуты: accesskey, id, class, lang, dir, title, style, атрибуты событий.

Пример формы

Мы уже знаем достаточно элементов, чтобы создать форму. Часто формы используются для ввода анкетных данных, поэтому наш пример будет имитировать подобный документ:

```
<TABLE border=3>
<FORM action="mailto:Имя@сервер.домен" method="post">
<TR>
<TD><LABEL for="imya">Имя: </label>
<TD><INPUT type="text" id="imya">
<TR>
<TD><LABEL for="familiya">Фамилия: </label>
<TD><INPUT type="text" id="familiya">
<TR>
<TD><LABEL for="telefon">Телефон: </label>
<TD><INPUT type="text" id="telefon">
<TR>
<TD>Пол:   
```

```
<INPUT type="radio" name="pol" value="Male"> M 
<INPUT type="radio" name="pol" value="Female"> Ж
<TD align="bottom">
<INPUT align="bottom" type="submit" value="Отослать">
<INPUT type="reset">
</form>
</table>
```

В этом примере использована таблица для выравнивания элементов формы. В форме есть несколько полей ввода и группа из двух переключателей (рис. 4.5). Для тренировки вы можете заполнить форму информацией, и отослать ее по электронной почте (например, себе). Для этого надо ввести в код страницы реальный электронный адрес. Заполнив форму, следует щелкнуть на кнопке Отослать. Обратите внимание, что подпись этой кнопки задается в элементе на Web-странице. Подпись кнопки Сброс не задана, поэтому она выбирается из настроек операционной системы. Иными словами, в нерусифицированной версии операционной системы эта кнопка может называться, к примеру, Reset.

Рис. 4.5. Пример формы

После того как форма будет заполнена и отправлена щелчком на кнопке Отослать, она будет помещена в папку Исходящие почтовой программы. Чтобы действительно отослать форму, надо отправить это новое электронное послание. Если вы отправляете форму самому себе, то через несколько минут можете проверить свой электронный почтовый ящик: форма в виде письма появится в нем.

<SELECT> <OPTION> </select>

Элемент SELECT предназначен для создания списка или меню на гипертекстовой странице, а элемент OPTION — для создания пункта списка. Пояснить действие этих элементов поможет простой фрагмент, приведенный ниже:

```
<SELECT>
<OPTION value=a>Первый
<OPTION value=b>Второй
<OPTION value=c>Третий
<OPTION value=d>Четвертый
</select>
```

В результате мы получим список, показанный на рис. 4.6. Оба элемента, создающие список, имеют собственные атрибуты.

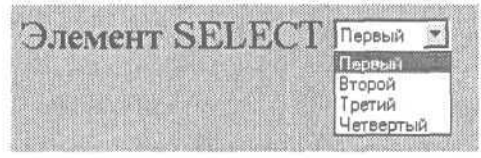


Рис. 4.6. Меню на Web-странице

Атрибуты элемента SELECT могут быть следующими. Атрибут name определяет имя меню (см. выше раздел «Общие атрибуты форм»). При помощи атрибута multiple (который не имеет значений) пользователю разрешается выбрать *сразу* несколько пунктов списка. Атрибут size определяет количество видимых на экране пунктов списка. Когда атрибут size отсутствует, список выглядит обычным образом: вначале видна только первая строка, а щелчок на кнопке со стрелкой раскрывает список. Если значение атрибута задано, то список не раскрывается, а прокручивается, причем пользователь видит только указанное количество строк.

Стандартные атрибуты: disabled, tabindex, id, class, lang, dir, title, style, атрибуты событий.

Элемент OPTION имеет другие атрибуты. Selected (без значений) определяет, какой из пунктов списка должен быть выбран по умолчанию, то есть при загрузке страницы. В списке только один из элементов OPTION может быть помечен таким способом. Атрибут value необходим для обработки данных на стороне сервера. Выбранный в списке пункт характеризуется значениями возвращенных атрибутов name и value.

Стандартные атрибуты элемента OPTION: disabled, id, class, lang, dir, title, style, атрибуты событий.

<TEXTAREA> </textarea>

При помощи этого элемента создается область для ввода или просмотра текста. На рис. 4.7 показан пример такой области, созданный при помощи следующей конструкции:

```
<H2>Элемент TEXTAREA
<TEXTAREA name="text001" rows=5 cols=30>
Область для ввода текста
</textareax/h2>
```

Размеры области задаются атрибутами rows (количество строк) и cols (количество столбцов). Назначение атрибута name такое же, как и в предыдущих случаях.

Подобно элементу INPUT, элемент TEXTAREA может иметь атрибут readonly. Это позволяет создать элемент, недоступный для редактирования.

Стандартные атрибуты: disabled, tabindex, id, class, lang, dir, title, style, атрибуты событий.

Рис. 4.7. Область для ввода или просмотра текста

Элементы SELECT и TEXTAREA могут быть использованы не в составе формы, а как самостоятельные детали Web-страницы. Их применение оправдано в тех случаях, когда надо увеличить плотность размещения данных. При помощи элемента SELECT можно создавать списки, которые получаются более компактными, нежели стандартные списки, рассмотренные в разделе «Списки» главы 3. Область ввода текста также поможет сэкономить место на странице за счет того, что сколь угодно большой текст будет прокручиваться в окне фиксированного размера.

<BUTTON> </button>

Этот элемент позволяет создавать кнопки так же, как и элемент INPUT. Но, в отличие от последнего, он является контейнером (имеет конечный тег). Это означает, что содержимое элемента может быть достаточно сложным, например, комбинацией текста и графики (рис. 4.8):

```
<BUTTON name="Имя" value="submit" type="submit">
Текст<IMG src="Файл.gif" alt="Комментарий"></button>
```

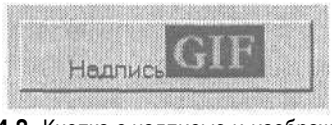


Рис. 4.8. Кнопка с надписью и изображением

Атрибут type может принимать следующие значения:

- button — кнопка, щелчок на которой вызывает запрограммированную разработчиком реакцию;
- submit — кнопка, подтверждающая, что форма заполнена;
- reset — кнопка для очистки формы.

Стандартные атрибуты: accesskey, disabled, tabindex, name, value, id, class, lang, dir, title, style, атрибуты событий.

<FIELDSET> <LEGEND> </legend> </fieldset>

Эти два элемента предназначены для создания группы полей в форме. Чтобы понять, как используются эти элементы, рассмотрим небольшой пример.

```
<FIELDSET>
<LEGEND>Заголовок группы</legend>
```

```
Имя: <INPUT name="imya2" type="text">
Фамилия: <INPUT name="familiya2" type="text"><BR>
Телефон: <INPUT name="telefon2" type="text"><BR>
Текст подсказки
</fieldset>
```

С помощью элемента FIELOSET несколько элементов объединяются: пользователь видит их заключенными в рамку (рис. 4.9). Внутри группы элементы формы используются обычным способом.

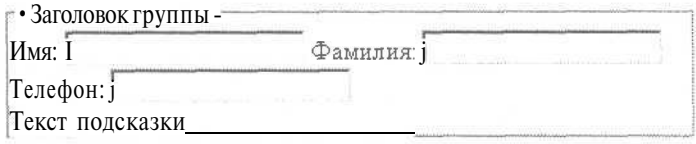


Рис. 4.9. Группа элементов формы

Элемент LEGEND позволяет создать заголовок группы. Поскольку этот элемент является контейнером, в нем можно размещать другие элементы HTML. Например заголовок группы можно составить из двух строк, если использовать тег <BR>. В этом случае размер шрифта заголовка целесообразно уменьшить.

С помощью атрибута align можно регулировать положение заголовка:

- top — заголовок сверху (как показано на рис. 4.9);
- bottom — заголовок внизу (что не всегда удается реализовать);
- left — заголовок вверху и слева (значение по умолчанию);
- right — заголовок вверху и справа.

Стандартные атрибуты элемента LEGEND: accesskey, id, class, lang, dir, style, title, атрибуты событий.

ПРИМЕЧАНИЕ

Примеры элементов форм можно найти в файле Form.htm на прилагаемой дискете.

Глава 5

Сценарии

Многие авторы заинтересованы в том, чтобы их страницы имели современный вид, были многофункциональными и динамичными. Для преодоления ограничений HTML применяются разные средства: апплеты, объекты, каскадные таблицы стилей. Но самым популярным приемом является использование сценариев. Сценарий — это программный код, который включается в текст страницы в виде исходного текста и выполняется броузером при просмотре страницы. Сценарий может быть написан на языке JavaScript, разработанном фирмой Netscape, или на Visual Basic Script (VBScript), разработанном фирмой Microsoft. Поскольку JavaScript является признанным стандартом, и, что немаловажно, стандартом де-факто (этот язык используется на подавляющем большинстве страниц), то мы остановимся на нем при рассмотрении примеров.

Что такое сценарий <SCRIPT> </script>

Этот элемент позволяет отделить текст программы-сценария от остальной информации страницы. Элемент SCRIPT должен включать атрибут language, который определяет язык и может принимать следующие значения:

- `javascript` — код на языке JavaScript;
- `tcl` — код на языке Tcl;
- `vbscript` — код на языке VBScript.

Может оказаться удобным хранить тексты программ в отдельном файле. Тогда элемент SCRIPT надо снабдить ссылкой на этот файл:

```
src="URL"
```

По традиции файлы, содержащие программы на JavaScript, имеют расширение JS.

Атрибут type тоже может указывать на тип языка, хотя его применение не является обязательным. Чтобы соблюсти все правила, внутри элемента можно поместить такое определение:

```
type="text/javascript"
```

Одной из особенностей сценариев является возможность изменения содержимого страницы в результате работы программы. Но это только возможность, а не

правило. С помощью атрибута defer (который не принимает никаких значений) можно «сообщить» броузеру, что таких изменений внесено не будет. В некоторых случаях это позволяет ускорить загрузку страницы.

Из стандартных атрибутов можно использовать атрибут charset.

Элемент SCRIPT (или ряд таких элементов) может располагаться как внутри секции HEAD, так и внутри секции BODY. Если сценарий находится внутри элемента BODY, возможна ситуация, когда какой-либо броузер, не поддерживающий элемент SCRIPT, воспримет программный код как обычный текст и выведет его на экран. Чтобы этого не случилось, код сценария вводят как комментарий:

```
<SCRIPT language="язык">
<!-- Все, что относится к коду сценария -->
</script>
```

Современные броузеры «знают» этот прием и игнорируют символы комментария. Если в тексте сценария нужно ввести комментарий, то для этого используют другое обозначение: в начале строки вводят две косые черты //.

Сценарий выполняется в момент загрузки страницы, то есть когда на экране еще видно ее содержание. В листинге 5.1 представлен пример простейшего сценария.

Листинг 5.1. Вывод сообщения в окне (1 вариант)

```
<HTML>
<HEAD>
<META http-equiv="Content-Type" content="text/html;
charset=windows-1251">
<TITLE>Простейший сценарий</title>
<SCRIPT language="javascript">
alert("Приветствуем вас на этой странице!")
</script>
</head>
<BODY background="fon01.gif">
<P>
<CENTER>
<H1 style="color : maroon">З а г о л о в о к 1</h1>
</center>
</body>
</html>
```

Это обычная страница, но в нее включен сценарий из одной строки. С помощью метода alert() перед загрузкой выводится сообщение (в данном случае приветствие), показанное на рис. 5.1. До тех пор пока пользователь не щелкнет на кнопке OK, загрузка не будет продолжена.

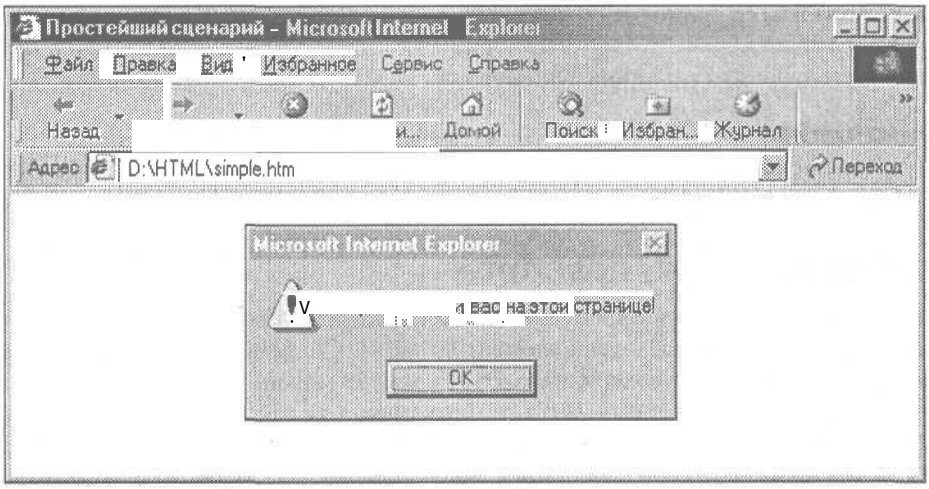


Рис. 5.1. Выполнение сценария

Тот же самый сценарий можно выполнить и другим способом: создать функцию и связать ее с событием. Загрузка страницы соответствует событию onload элемента BODY (листинг 5.2).

Листинг 5.2. Вывод сообщения в окне (2 вариант)

```
<HTML>
<HEAD>
<META http-equiv="Content-Type" content="text/html;
charset=windows-1251">
<TITLE>Простейший сценарий</title>
<SCRIPT language="javascript">
function DoFirst()
{
alert("Приветствуем вас на этой странице!")
}
</script>
</head>
<BODY background="fon01.gif" onload="DoFirst()">
<P>
<CENTER>
<H1 style="color : maroon">З а г о л о в о к 1</h1>
</center>
</body>
</html>
```

Но, разумеется, сценарий не обязательно должен выполняться во время загрузки страницы. В разделе «Стандартные атрибуты» главы 3 приведены события, которые определены для различных элементов HTML.

ПРИМЕЧАНИЕ

Тексты страниц можно найти и в файлах Simple.htm и Simple2.htm на прилагаемой дискете.

<NOSCRIPT> </noscript>

На тот случай, если страница будет просматриваться в броузере, не поддерживающем сценариев, предусмотрен элемент NOSCRIPT. Современные программы просмотра игнорируют его содержимое. Этот элемент можно использовать несколькими способами. Во-первых, внутри него можно разместить предупреждение наподобие следующего: «Ваш броузер не может воспроизвести сценарии, необходимые для просмотра этой страницы!» Во-вторых, внутри элемента можно создать упрощенную версию страницы, без скриптов. В-третьих, можно создать ссылку на другой документ. Элемент NOSCRIPT должен обязательно снабжаться конечным тегом. Из атрибутов допускаются только id и style.

Язык JavaScript

В данной книге не приводится полное описание языка JavaScript. Необходимую справочную информацию легко найти в Интернете и специализированных изданиях. Есть, например, хороший справочник по JavaScript Рика Дарнелла («Питер», 1999 г.). Но познакомиться с основными конструкциями языка, наиболее популярными его возможностями требуется обязательно, тогда дальнейшее его освоение не составит труда. Надо также заметить, что язык постоянно совершенствуется (пополняется новыми конструкциями). В последующих разделах будет рассмотрено несколько полезных примеров.

Синтаксис

Как и всякий современный язык программирования, JavaScript имеет традиционную часть, включающую операторы присваивания, математические и строковые функции, операторы и объектно-ориентированную часть, к которой относятся объекты, события, свойства и методы.

Основной конструкцией языка является *функция*. Она создается по следующему шаблону:

```
function ИмяФункции(параметр1, параметр2...)
{
Текст программы
return выражение
}
```

Обязательными элементами являются: ключевое слово function, круглые скобки и фигурные скобки, определяющие тело функции. Для выполнения функции достаточно указать ее имя в сценарии, например:

```
ИмяФункции(параметр)
```

Внутри функции могут находиться вызовы других функций, и, кроме того, функция может вызывать саму себя. Если функция указана в качестве значения атрибута события, она выполняется, когда происходит соответствующее событие.

С помощью оператора return функция может возвратить определенное значение. Тогда функцию можно использовать в операциях присваивания или проверки условий.

В программе могут использоваться переменные различных типов. Если переменная определена вне функции, она является *глобальной,* то есть доступной для всех функций страницы. Переменные, определенные внутри функции, являются *локальными,* действующими только в пределах функции. Разные функции могут независимо использовать локальные переменные с одинаковыми именами. Для определения переменной можно указать ключевое слово var, хотя это и не обязательно:

```
var Color01 = "red"
Color01 = "red"
```

Тип переменной определяется в момент присвоения ей значения. Для обозначения строковых констант используются двойные или одинарные кавычки. Два типа кавычек необходимы на тот случай, когда строковая константа содержит символы кавычек, например:

```
WelcomeMessage = 'Добро пожаловать на сайт компании "Tip Top"'
```

Числовые значения могут задаваться разными способами:

```
count = 1
X1 = 3.55
F5 = 7.674E-5
```

Логические переменные могут принимать значения false (ложь) и true (истина).

С помощью квадратных скобок определяются элементы массивов:

```
mass[23]
```

Математические операции выполняются следующим образом:

- var01 = var02+3 — сложение;
- var03 = 10*(var04-var05) — вычисления со скобками.

Для вызова математических функций используется объект Math:

var06 = Math.sqrt(var07) — вычисление значения функции (квадратного корня).

Кроме традиционных операций, можно выполнять операцию поиска остатка от деления, например:

```
var08=var09 % 4
```

Существуют унарные операции:

- var10++ — увеличение значения переменной на 1;
- var05 --- уменьшение значения переменной на 1.

В условных или управляющих операторах используются операторы сравнения. Для составления логического выражения, кроме круглых скобок, допустимы следующие знаки:

- == — равно;
- != — не равно;
- > — больше;
- >= — больше или равно;
- < — меньше;
- <= — меньше или равно.

Кроме них, используются логические операторы:

- ! — логическое отрицание;
- | — логическое ИЛИ;
- && — логическое И.

Управляющие операторы

Цикл do...while позволяет выполнять программный код до тех пор, пока выполняется условие.

```
do {
строка 1 ;
строка 2 ;
...
}
while (условие)
```

Обратите внимание, что тело оператора, так же как и тело функции, выделяется при помощи фигурных скобок. Строки кода (команды, операции присваивания, вызовы функций и др.) завершаются (отделяются друг от друга) точкой с запятой.

Данный цикл можно записать и так:

```
while (условие){
строка 1 ;
строка 2 ;
...
}
```

Цикл for используют для того, чтобы выполнить программный код заданное число раз. В качестве примера показана программа, позволяющая создать числовой массив и заполнить его нулями.

```
var Massiv = new Array();
var n = 25;
for (i = 0; i < n; i++) {
Massiv[i] = 0;
}
```

В данном примере показан цикл с тремя параметрами: начальным значением счетчика, условием выполнения цикла и командой изменения значения счетчика. Это традиционный способ использования такого цикла, но ни один из параметров не является обязательным. Если отсутствует условие, то цикл станет выполняться до тех пор, пока не будет прерван другим способом. Если отсутствует команда изменения значения счетчика, ее может заменить аналогичная команда в теле цикла. Если количество параметров меньше трех, символы «точка с запятой» определяют, какой параметр используется.

Есть ряд вспомогательных операторов, используемых совместно с циклами. С помощью оператора break можно прервать работу любого цикла. Оператор continue позволяет прервать выполнение цикла и начать проверку условия. В зависимости от условия цикл может быть прерван окончательно или его выполнение может быть начато еще раз.

Условный оператор if...else используется там, где выполнение программы надо поставить в зависимость от значения выражения (условия). Шаблон оператора таков:

```
if (условие) {
строки кода
}
else {
строки кода
}
```

Блок else (выполняемый, если условие имеет значение false), не обязателен.

Оператор switch. , . case тоже является условным, но выражение, представляющее собой его параметр, не обязательно должно быть логическим. Оно может принимать любые значения. Важно только, чтобы эти значения были указаны как метки в блоках case. Тогда будет выполнен один из многих вариантов кода. Если значение выражения не равно ни одной метке, выполняется блок default. Ниже приведен шаблон оператора. Обратите внимание, что фигурные скобки использованы только один раз, так как варианты кода разделены операторами case.

```
switch (выражение) {
case метка1 :
строки кода
case метка2 :
строки кода
...
default :
строки кода
}
```

Примеры сценариев

Замена изображения

Один из самых модных сценариев — замена одного рисунка другим в тот момент когда над рисунком появляется указатель мыши. Таким способом можно изме нять вид надписи (текст или фон), вводить изображения нажатой и отпущенно кнопки, создавать всевозможные визуальные эффекты, «оживляющие» страницу Листинг 5.3 содержит код подобной страницы. На его примере мы разберем, ка совершается замена изображений.

Листинг 5.3. Сценарий для замены изображения

```
<HTML>
<HEAD>
<META http-equiv="Content-Type" content="text/html; charset=windows-1251">
<TITLE>Замена изображений</title>
<SCRIPT language="javascript">
AgentName = navigator.appName;
AgentVer = parseInt(navigator.appVersion);
if (AgentName == "Netscape" && AgentVer <= 2){
version = "n2";
1
else {
version = "x3";
}
if (version == "x3") {
// Первая кнопка
IMAGE1on = new Image;
IMAGE1on.src = "str1on.jpg";
IMAGE1off = new Image;
IMAGE1off.src = "str1.jpg";
// Вторая кнопка
IMAGE2on = new Image;
IMAGE2on.src = "str2on.jpg";
IMAGE2off = new Image;
IMAGE2off.src = "str2.jpg";
}
// Указатель сверху
function mouse_on(objekt) {
if (version == "x3") {
imgOn = eval(objekt + "on.src");
document[objekt].src = imgOn;
```

```
}
}
// Указатель убран
function mouse_off(objekt) {
if (version == "x3") {
imgOff = eval(objekt + "off.src");
document[objekt].src = imgOff;
}
}
</script>
</head>
<BOOY>
<TABLE align="left" border=0 cellspacing=3 cellpadding=8>
<TR>
<!-- Первая кнопка -->
<TD><A href="URL" onMouseover="mouse_on('IMAGE1')"
onMouseout="mouse_off('IMAGE1')"><IMG src="str1.jpg" name="IMAGE1"
alt="Page 1" border=0 height=36 width=151x/a></td>
<!-- Вторая кнопка -->
<TD><A href="URL" onMouseover="mouse_on('IMAGE2')"
onMouseout="mouse_off('IMAGE2')"><IMG src="str2.jpg" name="IMAGE2"
alt="Page 2" border=0 height=36 width=151></a></td>
</table>
</body>
</html>
```

Весь сценарий находится внутри элемента SCRIPT. Первые строки кода не имеют прямого отношения к замене графики. Дело в том, что броузеры ранних версий не поддерживают некоторые конструкции JavaScript, поэтому, чтобы предотвратить возникновение ошибки, необходимо проверить тип и версию броузера. Чаще всего код сценария надо защищать от броузера Netscape Navigator 2.x. В нашей программе создаются две переменные AgentName и AgentVer, которые позволяют определить название и версию броузера соответственно. Для этого указан объект navigator (представляющий используемый броузер). Кроме этого необходимы:

- appName — свойство, возвращающее название броузера;
- appVersion — свойство, возвращающее номер версии броузера;
- parseInt — функция округления до целого (т. к. номер версии может быть «дробным»).

Теперь, проведя проверку этих переменных, мы можем создать переменную version, которая будет хранить значение п2, если используется вторая или более ранняя версия Netscape Navigator. Значение х3 обозначает все другие программы просмотра.

Теперь можно приступать к рисункам. Чтобы заменить один рисунок другим, надо изменить свойство src графического объекта: ссылку на графический файл. Поэтому вначале создаются две объектные переменные IMAGE1on и IMAGE1off типа Image. Для них определяются необходимые графические файлы:

- str1on.jpg — рисунок, который должен появиться под указателем мыши;
- str1.jpg — рисунок, который сразу помещается на страницу, и который появляется, когда указатель убран.

Теперь мы имеем два изображения кнопки. Саму кнопку (с надписью «Страница 1») можно увидеть в главе 8 на рис. 8.23, но в данном случае иллюстрация мало что объясняет, так как работу сценария нужно наблюдать в действии.

Удобство предварительного определения файлов очевидно: если на странице много кнопок, целесообразно сосредоточить все ссылки в одном месте. В нашем примере используются две кнопки, поэтому для второй тоже задаются два файла формата JPG.

В нашем примере объектом, вид которого мы хотим динамически изменять, является гиперссылка с содержимым в виде рисунка. Для элемента А задано событие «указатель мыши сверху»: onMouseover="mouse_on('IMAGE1')". Естественно, в сценарии должна находиться соответствующая функция. Разберем, как работает функция mouse_on(objekt). Ее аргументом является имя элемента IMG, которое задано с помощью атрибута name="IMAGE1". Таким образом, задается однозначное соответствие между объектами, которыми оперирует сценарий, и элементами, расположенными на странице. Работа функции тоже начинается с проверки версии броузера, так как в теле функции присутствует метод eval, не поддерживаемый ранними версиями Netscape Navigator.

Этот метод работает достаточно интересно. Его аргумент — текстовая строка, которая должна представлять собой команду JavaScript. С помощью метода eval эта команда выполняется. Если в нашем примере такой командой окажется "IMAGEIon.src", то переменная imgOn получит в качестве значения ссылку на графический файл. Это будет не "str1on.jpg", как можно подумать, а URL, например

```
file:///C:/ПАПКА/str1on.jpg
```

Осталось изменить свойство элемента рисунка. Это делается с помощью объекта document и имени соответствующего элемента IMG. Объект document можно считать массивом, в качестве индекса которого выступают имена расположенных на странице элементов. Ну, а свойство src говорит само за себя. Вторая функция (mouse_off) отвечает за возврат рисунка к прежнему виду и работает аналогично.

Возможно, у вас возникнет подозрение: а не слишком ли сложно запрограммирован сценарий? Действительно, кое-что в тексте программы можно упростить. Например, в сценарии могла бы выполняться такая команда:

```
document["IMAGE1"].src '= "str1on.jpg";
```

В ней все параметры заданы явно, и работать такая команда может только с конкретным элементом IMG.

Вид функции mouse_on можно изменить. Пусть, к примеру, у нее будут два аргумента: имя элемента и имя графического файла:

```
function mouse_on(objekt, risunok) {
if (version == "x3") document[objekt].src = risunok;
}
```

Тогда элемент A должен выглядеть так:

```
<A href="URL" onMouseover="mouse_on('IMAGE1','str1on.jpg')"
onMouseout="mouse_off('IMAGE1','str1.jpg')"><IMG src="str1.jpg"
name="IMAGE1" alt="Page 1" border=0 height=36 width=151></a>
```

В этой конструкции имеются *три* ссылки на графические файлы. Теоретически, для кнопки можно предусмотреть три изображения: «вид после загрузки страницы», «нажатая кнопка», «отпущенная кнопка». Это позволяет визуально различать кнопки, на которых был выполнен щелчок, и те, которые пользователь оставил без внимания.

Изменение свойств текста

Web-страница, текст которой приведен в листинге 5.4, содержит элементы абзаца и заголовка. Для заголовка запрограммированы события мыши так, чтобы цвет и содержание текста изменялись в зависимости от положения указателя.

Листинг 5.4. Сценарий для замены свойств текста

```
<HTML>
<HEAD>
<TITLE>Изменение цвета текста</title>
<META http-equiv="Content-Type" content="text/html; charset=windows-1251">
<SCRIPT>
// Если указатель мыши установлен
function TextOnMouseOver( ){
Element1 = window.event.srcElement ;
Text1 = document.all("Par1") ;
if (Element1.tagName == "H2") {
Element1.style.color = "red" ;
Text1.innerText="Надпись стала красной?"
Text1.style.color="maroon"
}
}
// Если указатель мыши убран
function TextOnMouseOut() {
Element1 = window.event.srcElement ;
Text1 = document.all("Par1") ;
```

продолжение

Листинг 5.4 *(продолжение)*

```
if (Element1.tagName == "H2"){
Element1.style.color = "green" ;
Text1.innerText='Установите указатель мыши на надпись "Заголовок"'
Text1.style.color="blue"
}
}
</script>
</head>
<BODY bgcolor="white" text="green">
<P id="Par1">Установите указатель мыши на надпись "Заголовок"</p>
<TABLE border=0 cellspacing=0 cellpadding=0><TR>
<TD><H2 onmouseover=TextOnMouseOver( ) onmouseout=TextOnMouseOut( )>
Заголовок</h2></table>
</body>
</html>
```

Легко заметить, что в элементе H2 информация о сценарии минимальная: указаны только имена функций. Вся обработка событий сосредоточена в элементе SCRIPT. Когда указатель мыши попадает в область заголовка, выполняется функция TextOnMouseOver( ). Поскольку эта функция не получает никаких параметров, необходимо в первую очередь определить, с каким элементом связано событие. Для этого создается объектная переменная Element1. В тексте программы используются:

- window — объект, определяющий окно броузера;
- event — объект, содержащий информацию о событиях;
- srcElement — свойство, определяющее элемент, в котором произошло событие.

Переменная Text1 связана с конкретным абзацем (элементом P) при помощи атрибута id, задающего имя элемента.

Теперь все готово для активных действий. С помощью свойства tagName проверяется, действительно ли событие связано с элементом H2. Таким образом, сценарий будет выполняться для всех заголовков второго уровня, расположенных на странице. С помощью конструкции style.color для заголовка задается красный цвет. С помощью свойства innerText заменяется текст абзаца. Вторая функция, которая выполняется, когда указатель мыши покидает область заголовка, работает так же, но константы в ней использованы, естественно, другие.

Обратите внимание, что элемент H2 расположен внутри элемента TABLE. Дело в том, что элемент заголовка занимает всю строку, и слева от надписи остается пустое пространство (рис. 5.2). Если пользователь поместит указатель мыши на пустое место, сценарий тоже будет выполнен. Это может показаться неестественным,

так как все привыкли, что реакция возникает, когда выбран конкретный объект (в данном случае надпись). Таблица, которую не видно при просмотре, позволяет ограничить область действия элемента H2 только его текстом.

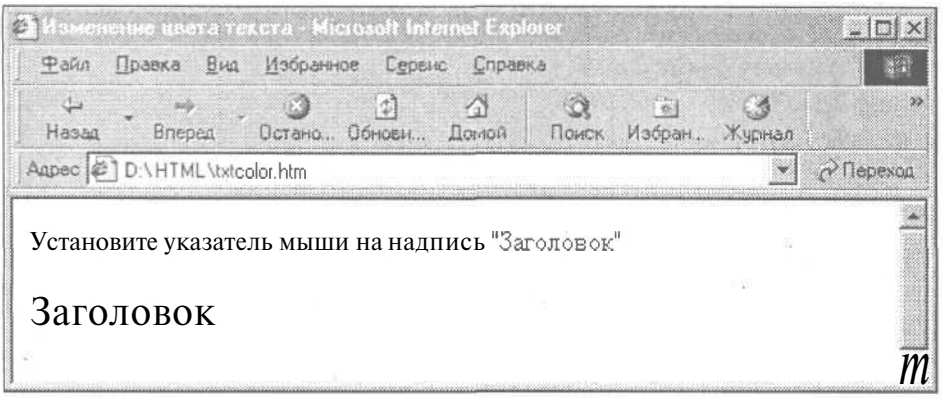


Рис. 5.2. Иллюстрация к листингу 5.4

ПРИМЕЧАНИЕ

Этот пример можно найти в файле Txtcolor.htm на прилагаемой дискете.

Метод setTimeout

В этом примере мы разберем два полезных приема: использование реально действующих часов и метода setTimeout, без которого реализация многих динамических эффектов оказывается невозможной.

Текст страницы приведен в листинге 5.5, а внешний вид — на рис. 5.3.

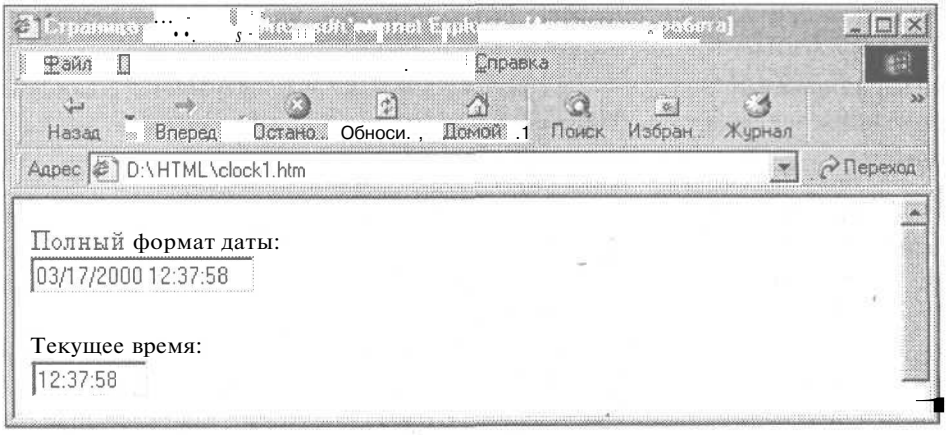


Рис. 5.3. Страница с постоянно обновляемыми элементами

Листинг 5.5. Страница с часами

```
<HTML>
<HEAD>
<TITLE>Страница с часами</title>
</head>
<SCRIPT language="javascript">
function Timer1() {
var D1 = new Date();
var TIME1 = " ";
TIME1 = D1.getHours()+":"+D1.getMinutes()+":"+D1.getSeconds()
document.forma01.dat.value=D1.toLocaleString();
document.forma01.vremya.value=TIME1;
setTimeout('Timer1()',2000)
}
</script>
<body onload=Timer1() bgcolor="white" text="blue">
<FORM name="forma01">
<P> Полный формат даты:<BR>
<INPUT type="text" size=40 name="dat">
<P> Текущее время:<BR>
<INPUT type="text" size=8 name="vremya">
</form>
</body>
</html>
```

Задача данного сценария — вывести на экран значения даты и времени. Такие де тали если и не окажутся очень полезными, украсят страницу, сделают ее «живой» Для вывода данных предусмотрены элементы формы, но могут использоваться и другие элементы, важно лишь, чтобы они имели имена (атрибут `name`).

Сценарий начинает выполняться сразу же после загрузки страницы. Это осуще ствлено по следующему шаблону с применением атрибута события:

```
<BODY onload=Функция()>
```

В функции `Timer1()` переменная D1 хранит объект `Date()` — текущую дату. Для пере менной `TIME1` использованы методы `getHours()`, `getMinutes()` и `getSeconds()` для того, чтобы вывести текущее время в заданном формате. Функция `toLocaleString()` позволяет преобразовать формат даты в строковый.

Если бы мы просто определили дату и текущее время, а затем вывели эти данные на экран, изображение оказалось бы статичным. Оно могло бы измениться только в том случае, если бы пользователь перезагрузил страницу. Поэтому сценарий надо сделать постоянно работающим. Для этого используется метод `setTimeout`

Он позволяет создать паузу в выполнении сценария. Величина этой паузы задается в миллисекундах в качестве второго аргумента функции. В качестве первого аргумента выступает функция, которая должна быть выполнена после истечения заданного времени. В нашем случае используется имя той же функции, в которой находится оператор. Это позволяет функции `Timer1()` бесконечно вызывать саму себя (рекурсивный вызов), и выполнение сценария не будет останавливаться. Поскольку изменение данных (времени) происходит один раз в секунду, величина задержки должна быть меньше секунды. В нашем случае она равна 0,2 с. Метод setTimeout можно использовать для создания таких визуальных эффектов, как бегущая строка, побуквенный вывод надписи, меняющееся изображение и других.

ПРИМЕЧАНИЕ

Эта страница находится в файле Clock1.htm на прилагаемой дискете.

Управление формами

Когда на странице размещают элементы форм, бывает удобно запрограммировать работу этих элементов. В листинге 5.6 приведен пример страницы с двумя кнопками, которые управляют содержимым элемента TEXTAREA. На рис. 5.4 показан пример страницы после щелчка на кнопке Заменить текст.

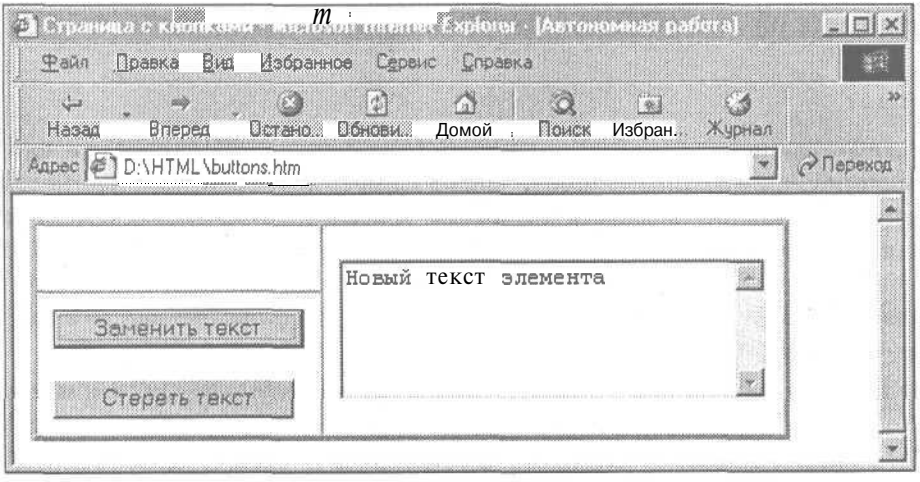


Рис. 5.4. Страница с кнопками

Листинг 5.6. Страница с кнопками

```
<HTML>
<HEAD>
<TITLE>Страница с кнопками</title>
</head>
```

продолжение

Листинг 5.6 *(продолжение)*

```
<SCRIPT language="javascript">
var Stroka = "Новый текст элемента"
function DoFirst(){
document.forma02.text01.value = "Первоначальный текст элемента"
}
</script>
<BODY bgcolor="white" text="blue" onload=DoFirst()>
<FORM name="forma02">
<TABLE border=3 cellspacing=0 cellpadding=10>
<TR><TD>  <TD rowspan=2>
<TEXTAREA name="text01" rows=5 cols=30x/textarea>
<TRXTD>
<INPUT type=button value="Заменить текст"
onClick="document.forma02.text01.value=Stroka">
<P>
<INPUT type=button value="Стереть текст"
onClick="document.forma02.text01.value=' '">
</table>
</form>
</body>
</html>
```

На этой странице создана форма с именем forma02 и с элементами, выровненными при помощи таблицы. Элемент TEXTAREA имеет конечный тег для отделения текста, находящегося в области прокрутки. Однако, поскольку это элемент формы, для изменения его содержимого используется свойство value, а не свойство innerText, необходимое для элементов, служащих контейнерами текста. При загрузке страницы выполняется функция DoFirst(), что позволяет ввести в текстовую область исходную строку.

Для кнопок (элемент INPUT типа button) определены события «щелчок мышью» onClick. Первая кнопка записывает в текстовую область содержимое переменной Stroka, определенной в секции SCRIPT. Вторая кнопка удаляет весь текст из элемента. Поскольку кнопки стандартные, нам не надо думать о том, как будет выглядеть нажатая и отжатая кнопка: все сделает программа. Но при желании можно воспользоваться одним из предыдущих примеров и сделать кнопки на основе рисунков. Но тогда пришлось бы запрограммировать для каждой кнопки несколько событий.

ПРИМЕЧАНИЕ

Эта страница находится в файле Buttons.htm на прилагаемой дискете.

Сценарий для одного элемента

Чтобы использовать сценарии, вовсе не обязательно писать длинные программы. Текст простого сценария можно поместить внутрь начального тега элемента. Здесь необходимо указать атрибуты событий, чтобы запрограммировать реакцию элемента на внешнее воздействие.

Основным средством программирования служит конструкция this — объект, обозначающий текущий элемент.

Например, чтобы запрограммировать изменение цвета элемента в зависимости от положения указателя мыши, достаточно одной команды для каждого события:

```
<H1 onmouseover="this.style.color='green' "
onmouseout="this.style.color='black' ">Заголовок 1</h1>
```

При необходимости для одного атрибута события можно использовать несколько команд JavaScript, но тогда их надо отделять друг от друга точкой с запятой. Можно также использовать условные операторы и, соответственно, фигурные скобки для разграничения необходимых блоков команд.

Если изменение цвета заголовка имеет чисто декоративную функцию, то управление свойствами гиперссылки требуется для подсказки пользователю. Допустим, нас не устраивает традиционный вид ссылки с подчеркнутым текстом. Тогда можно легко изменить вид надписи. В примере, приведенном ниже, для текста ссылки выбран синий цвет (color: blue), полужирное начертание (font-weight: bold), но отменено подчеркивание (text-decoration: none). При размещении указателя мыши над ссылкой цвет текста меняется на красный:

```
<A style="color: blue; font-weight: bold; text-decoration: none"
href="URL" onmouseover="this.style.color='red' "
onmouseout="this.style.color='blue' " >Текст ссылки</a>
```

ПРИМЕЧАНИЕ

Эти примеры записаны в файле Simplest.htm на прилагаемой дискете.

Глава 6

Приемы разметки гипертекста

Элементы HTML изучены, гипертекстовые редакторы освоены. Достаточно л] этого для того, чтобы начать создавать страницы? Увы, нет. Очень важно проник нуться духом HTML, понять, почему Web-страницы делают именно так, а не ина че. Кроме того, многие элементы не всегда используются по прямому назначени Самый очевидный пример — таблицы. Они являются непревзойденным средство организации структуры страницы, когда все детали занимают строго определен ное место, независимо от размеров окна броузера, разрешения монитора и други режимов. Рамку таблицы делают, разумеется, невидимой. Эта глава посвящен; подходам к конструированию страниц.

Стиль и традиции

При создании достаточно крупных сайтов автор всегда должен стараться выпол нить ряд требований: загрузка страниц должна происходить по возможности быс тро, любая страница должна выглядеть одинаково при просмотре в различны программах, пользователь должен легко разобраться, какая информация есть на сайте и как ее найти.

Особенность первых Web-страниц заключалась в том, что они изобиловали под черкнутыми фрагментами — гиперссылками. У авторов не было нынешних воз можностей по созданию сложных визуальных эффектов, но было желание поде литься информацией об интересных местах в Интернете. Такие страницы были удобны именно как текстовые документы.

Здесь явно просматриваются тенденции и мода. Точно так же, как в MS-DOS преобладающим фоном был синий, в WWW вначале преобладали серые страни цы. Разумеется, это мало кого устраивало. Одним из экзотических решений было использование черного фона. Это поначалу впечатляло, но, как всякая экзотика, не прижилось. Самым значительным направлением в экспериментах с фоном было использование рисунков «серым по серому». Мне приходилось видеть не мало подлинных шедевров, построенных только на 2—3 оттенках серого цвета. В такой манере воспроизводили все: от логотипов фирм до лирических пейзажей. Отзвуком этой моды является раскраска панели броузера Internet Explorer 3.0: разработчики этой программы явно хотели, чтобы цветовая гамма панели соот ветствовала виду большинства Web-страниц.

Потом HTML и броузеры были усовершенствованы, и начался бум графики. Разработчики считали своим долгом расположить на своих страницах как можно больше двигающегося, прыгающего и мигающего. Поначалу все это было очень интересно, но со временем «зритель» начал уставать. Кроме того, романтические настроения стали уступать меркантильным соображениям: все больше организаций стали обзаводиться собственными рекламными страницами.

В моду вошли фреймы: левую часть окна отводили для набора ссылок, которые служили путеводителем по сайту — своеобразным меню (рис. 6.1,а). Одновременно вверху и внизу каждой страницы размещали «горизонтальное меню» — набор команд в виде ссылок или рисунков. Такая компоновка страницы оказалась вполне рациональной, но в последнее время от нее отказываются. Возможно, это связано с тем, что из-за разделения окна броузера по вертикали в пропорции 1:4 сайты стали похожи друг на друга, а хочется получить что-нибудь оригинальное. Другая причина, на мой взгляд, — экономия места на экране. Дело в том, что пространство окна просмотра в броузере является весьма «дефицитным». Разработчики всегда стремятся создать как можно больше объектов, которые видны одновременно. Поэтому вертикальная полоса прокрутки в середине экрана многим мешала.

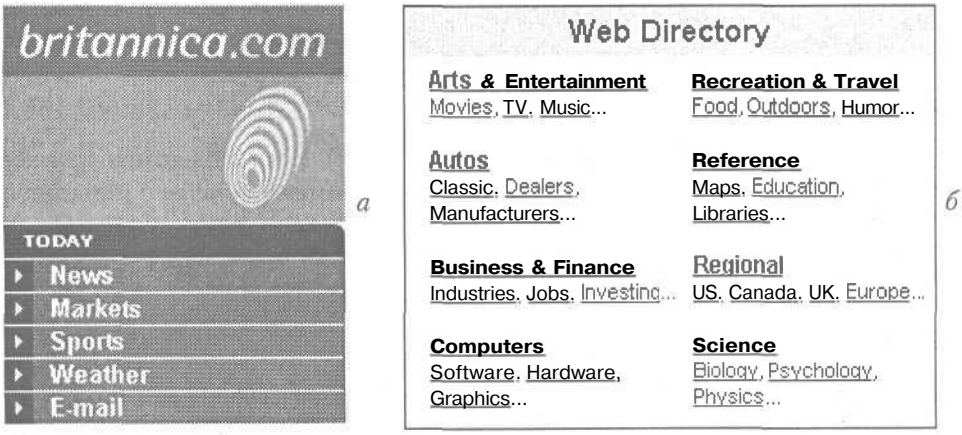


Рис. 6.1. Разные подходы к оформлению страниц: *а* — фрагмент вертикального меню сайта энциклопедии Британника; 5— часть страницы поисковой машины Alta Vista

С точки зрения дизайна многое определяет фон документа. Увлечение рисунками в виде фона тоже оказалось не вечным. С одной стороны, этот прием стал банальностью. С другой стороны, существуют чисто технические причины для отказа от рисунков. Дело в том, что разрешающая способность экрана монитора не может быть произвольно увеличена: разработчик должен учитывать, что страница будет просматриваться в режиме 640x480 пикселов. Это очень маленькая величина по сравнению, например, с журнальной полиграфией. Даже один пиксел легко различим глазом. Чтобы в таких условиях обеспечить сглаживание шрифтов, надо

согласовать цвет шрифта, тени и фона. То же самое можно сказать об изображе ниях красивых кнопок и других графических элементов. Если фон представляет собой рисунок, такое согласование выполнить трудно, поэтому сейчас опять в моде однотонные страницы.

Разработчики по-разному подходят к выбору фона. Все, что связано с моло дежной тематикой, располагается на черном фоне: сайты, посвященные музыке компьютерным играм, авангардной моде и т. д. Респектабельность достигается темно-синим фоном (navy): многие солидные компании выбирают этот цвет Пользователи, создающие личные странички, предпочитают фон пастельны тонов (зеленый, бежевый, голубой). В общем, творческий поиск не прекращает ся ни на минуту. Результат всех этих исканий оказался вполне закономерным дизайнеры пришли к выводу, что самым естественным решением будет ис пользование белого фона. Очень многие титулованные фирмы в настоящее время оформили свои страницы именно таким образом. Если вы посмотрите на рис. 6.1,б, то поймете, почему. На белом фоне четко виден шрифт малого размера, что позволяет разместить большое число гиперссылок на небольшой площади.

В самом низу страницы принято размещать служебную информацию: сведе ния об авторских правах, адрес электронной почты Web-мастера, полезные ссылки и т. д.

Не таблица, а табличка

Простейшая таблица — это таблица из одной ячейки. Если определить для не< атрибуты border, cellspacing и cellpadding, то получится табличка с рельефной рамкой. Код в этом случае может быть таким:

```
<TABLE border=4 cellspacing=3 cellpadding=10>
<TR><TD bgcolor="yellow">Петровым 2 звонка
</table>
```

Табличка, которая создается при помощи этого кода, показана на рис. 6.2.

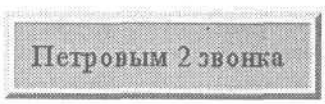


Рис. 6.2. Пример таблицы из одной ячейки

Заголовок и рисунок рядом

Таблица из двух ячеек тоже может пригодиться. Она позволяет, например, разме стить рядом текст и рисунок. Так можно поступить с заголовком и иллюстрацией названием фирмы и логотипом, подсказкой и рисунком-гиперссылкой. Естествен

но, что рамку таблицы в этом случае уместно скрыть, обнулив атрибут border. Код можно использовать такой:

```
<TABLE border=0>
<TRXTD><H3>A03T <I>Две пирамиды</i></h3>
<TD><IMG src="treug1.gif">
</table>
```

В результате текст, отформатированный в виде заголовка, и рисунок расположатся так, как показано на рис. 6.3. Таким способом можно выполнить и более сложное форматирование поверхности Web-страницы.

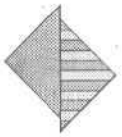


Рис. 6.3. Размещение текста и рисунка рядом

Мозаичные рисунки

Иногда возникает необходимость разбить рисунок на несколько частей. В этом случае при помощи таблицы можно обеспечить точную подгонку фрагментов друг к другу. В этом примере рисунок размером 250x250 пикселов разбит на четыре части:

```
<TABLE border=0 cellspacing=0 cellpadding=0>
<TR><TD><img src="dragon41.jpg" width="125" height="125"></TD>
<TD><img src="dragon42.jpg" width="125" height="125"></TD></TR>
<TR><TD><img src="dragon43.jpg" width="125" height="125"></TD>
<TDximg src="dragon44.jpg" width="125" height="125"></TD></TR>
</table>
```

Чтобы между частями рисунка не было зазоров и смещения, необходимо выполнить ряд условий. Во-первых, значения атрибутов border, cellspacing и cellpadding должны быть равны нулю. В результате рамка таблицы будет невидимой, а пустого пространства между фрагментами не будет. Во-вторых, надо использовать конечные теги ячеек таблицы </TD> и проследить, чтобы в ячейках не осталось пробелов: должны присутствовать только элементы IMG. Рис. 6.4 дает представление об этом приеме. В данном случае для наглядности оставлена рамка таблицы (border=3). Это несколько противоречит изложенному выше, но может служить еще одним способом представления мозаичного рисунка.

Выгоды от использования мозаики следующие. Элементы таблицы загружаются быстрее, и пользователь может их разглядывать во время загрузки. Если рисунок имеет формат GIF, то одну часть мозаики можно сделать с анимацией, а остальные — нет, что позволит уменьшить объем файлов. Наконец, отдельные фрагменты рисунка могут служить гиперссылками.

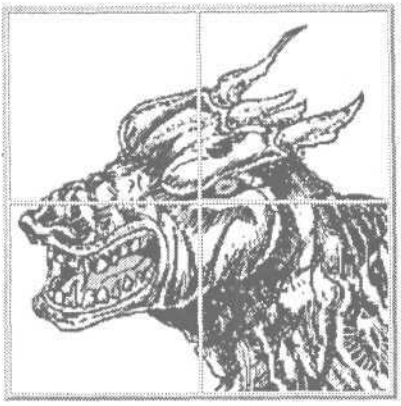


Рис. 6.4. Мозаичный рисунок

Объединение ячеек таблицы

Многие пользовательские задачи требуют создания таблиц со сложной структу рой. Элемент TABLE оказывается в таких случаях как нельзя более кстати, так ка позволяет создавать всевозможные формы таблиц. Самое сложное в разработк рамки таблицы — объединение нескольких ячеек в одну. Разберемся, как это делается На рис. 6.5 показан пример таблицы, в которой объединены несколько ячеек одног столбца и несколько ячеек строки. Ниже приведен код этого фрагмента страницы

```
<TABLE border=4 cellspacing=0 width=70%>
<TR><TD><B>Заголовок 1</b>
<TD><B>Заголовок 2</b>
<TR><TD rowspan=3 >Ячейка 1
<TD>Ячейка 2
<TR><TD>Ячейка 3
<TR><TD>Ячейка 4
<TR><TD colspan=2 align="center">Ячейка 5
</table>
```

Заголовок 1	Заголовок 2
Ячейка 1	Ячейка 2
	Ячейка 3
	Ячейка 4
Ячейка 5	

Рис. 6.5. Таблица с объединенными ячейками

Чтобы объединить ячейки столбца, необходимо использовать атрибут rowspa для элемента TD, соответствующего верхней из объединяемых ячеек. Значени

атрибута определяет количество объединяемых ячеек. Теперь начинается самое сложное. При создании элементов, отвечающих за нижние строки таблицы, необходимо пропускать элементы ячеек, расположенных на месте объединения. В нашем примере эти строки содержат по одной ячейке, несмотря на то, что в таблице две колонки.

Аналогичным образом объединяются ячейки в строке. Для этого используется атрибут colspan. В данной строке также оказалось достаточно одного элемента TD.

Атрибуты rowspan и colspan могут использоваться совместно в элементе TD. В этом случае в одну ячейку объединяются ячейки из нескольких столбцов и нескольких строк. Пример такой таблицы показан на рис. 6.6. Она создана при помощи следующего кода:

```
<TABLE border=4 cellspacing=0 width=70%>
<TR><TD><B>Заголовок 1</b>
<TD><B>Заголовок 2</b>
<TD><B>Заголовок 3</b>
<TR><TD rowspan=4 colspan=2>Ячейка 1
<TR><TD>Ячейка 2
<TR><TD>Ячейка 3
<TR><TD>Ячейка 4
<TRxTD colspan=3 align="center">Ячейка 5
</table>
```

Обратите внимание: несмотря на то, что объединяются ячейки трех строк (как в предыдущем примере), значение атрибута rowspan увеличено на 1.

Рис. 6.6. Одновременное объединение ячеек нескольких строк и столбцов

Вложенные таблицы

В тех случаях, когда структура таблицы достаточно сложна, можно использовать вложение таблиц. В этом случае одна из таблиц размещается вместо данных одной из ячеек. Для облегчения процесса конструирования можно создавать таблицы по отдельности, а затем переносить вкладываемые таблицы через буфер обмена. Пример вложенной таблицы показан на рис. 6.7. Она создана при помощи следующего кода:

```
<TABLE border=4 cellspacing=0 width=70%>
<TR><TD bgcolor="yellow">Таблица 1
```

```
<TD bgcolor="yellow">
<TABLE  border=2>
<TR><TD>Таблица 2
<TD>Ячейка  2-2
<TR><TD>Ячейка  3-2
<TD>Ячейка  4-2
</table>
<TR><TD  bgcolor="yellow">Ячейка  3-1
<TD bgcolor="yellow">Ячейка 4-1
</table>
```

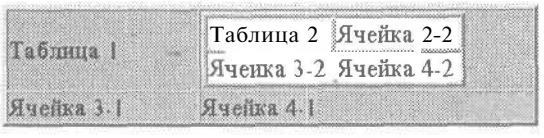


Рис. 6.7. Пример вложенной таблицы

Форматирование линии

При помощи атрибутов элемента HR можно превратить горизонтальную линию в ту или иную деталь Web-страницы. По сути, этот элемент создает прямоугольник, и только параметры по умолчанию делают его похожим на горизонтальную линию. На рис. 6.8 показаны различные форматы линии. Вертикальную линию получить несложно, надо лишь задать большую высоту и маленькую ширину, например: `size=100 width=2`. Задав равные значения для высоты и ширины, получим квадрат, который можно использовать для разделения частей страницы как маркер при создании списков и даже как элемент гиперссылки, на котором надо щелкать мышью. Задав атрибут ширины в процентном отношении (`width=25%`) получим черту, которую иногда ставят в конце текста. Преимущество такого подхода — быстрота создания элемента.

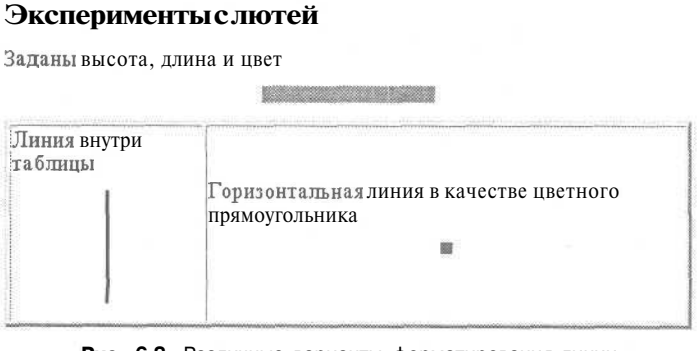


Рис. 6.8. Различные варианты форматирования линии

Стихотворный текст

В стихотворном тексте строфы должны располагаться по центру страницы, а строки выравниваться влево. Чтобы добиться этого эффекта при любом размере окна броузера, удобно использовать элементы CENTER и TABLE, как показано ниже:

```
<CENTER>
<FONT color="maroon"><H3>ОРАНЖЕВЫЙ ЛИСТ</h3></font>
<P>
<TABLE border=0>
<TR><TD><FONT size=4 color="maroon">
Прозрачные осени дали<BR>
Пронизаны солнцем насквозь.<BR>
Но нет, недостойно печали<BR>
Все то, что еще не сбылось.<BR>
<Px/font>
<TR><TD><FONT size=4 color="maroon">
Ветер шумит над кленом<BR>
И кувыркается вниз<BR>
На травы еще зеленые<BR>
Яркий оранжевый лист.<BR>
<P></font>
<TRXTD><FONT size=4 color="maroon">
Я подниму его бережно,<BR>
Возьму из леса с собой.<BR>
Влажный, душистый и свежий -<BR>
Будет теперь со мной.<BR>
<Px/font>
<TR><TD><FONT size=4 color="maroon">
В белую стужу зимнюю,<BR>
С думами о своем<BR>
Свяжутся нитью незримою<BR>
Я, этот лист и клен.<BR>
<Px/font>
</table>
<HR width=100><HR width=50><HR width=25>
</center>
```

Каждая строфа помещается в отдельную ячейку таблицы. Строки должны завершаться элементом BR. Чтобы отформатировать текст, надо использовать соответствующие элементы (например FONT) или создать стиль для таблицы (элемента

TABLE) и элемента CENTER. Например, можно определить цвет текста с помощью атрибута:

```
style="color: maroon"
```

В каждой ячейке должен присутствовать элемент абзаца P, чтобы строфы были отделены друг от друга.

Если стихотворение не имеет названия, в заголовке обычно помещают три звездочки. После текста можно разместить три горизонтальные линии разной длины (элементы HR). Все эти детали должны находиться внутри элемента CENTER.

ПРИМЕЧАНИЕ

Варианты форматирования таблиц можно найти в файле Table.htm на прилагаемой дискете

Ссылки на файлы мультимедиа

Если задаться вопросом, сложно или легко включать в HTML-документы мультимедийные источники данных, то ответ на этот вопрос будет двояким. С одной стороны, сделать это совсем нетрудно, а с другой стороны, внедрение мультимедийных файлов влечет за собой ряд проблем. Броузеры не имеют встроенных средств для воспроизведения таких файлов, но они могут использовать установленное на компьютере клиента программное обеспечение.

Допустим, нам необходимо прослушать на Web-странице звук в формате WAV (звуковой файл Windows). Мы можем создать гиперссылку:

```
<A href="Имя файла.wav"> Щелкни и слушай </a>
```

В том случае, если компьютер оборудован звуковой платой и колонками, если установлен драйвер звуковой платы и если установлена программа для воспроизведения звуковых файлов, мы сможем прослушать содержимое файла. Во всех этих «если» и заключается вторая, негативная сторона вопроса. Нельзя дать гарантию, что у всех пользователей, просматривающих ваши HTML-документы, будет необходимое оборудование и программное обеспечение.

Существует большое количество форматов звуковых и видеофайлов. Это сдерживает распространение мультимедийной продукции, так как трудно дать однозначные и простые рекомендации пользователям, как им сконфигурировать их компьютеры. Зато в случае ограниченного применения, когда можно обговорить условия просмотра (и прослушивания) информации, мультимедиа — вне конкуренции. Например, для пользователей Windows 95 не составляет проблемы просмотреть файл видеоролика в формате AVI. Разумеется, при условии, что в операционной системе установлены соответствующие компоненты. Поэтому, если вы знаете, что читатели ваших страниц в этом смысле экипированы, можете включать в состав документов подобные гиперссылки:

```
<A href="Имя файла.avi"> Щелкни и смотри </a>
```

Компоновка Web-страниц

Как только разработчик Web-страниц создает сайт, количество документов на котором больше двух, перед ним возникает задача определения топологии связей. Гиперссылки, обеспечивающие переход с документа на документ, образуют в реальных задачах сложную схему, которую разработчик обязан хорошо продумать. От того, как соединяются отдельные документы, во многом зависит впечатление гостя, посетившего сайт. Обилие переходов или нерациональное их расположение приводит к тому, что человек быстро устает, а сам сайт ассоциируется у него с лабиринтом.

На рис. 6.9 показаны примеры компоновки сайтов. Пользователь, получивший доступ к странице А, вряд ли запутается, двигаясь по подчиненным ей страницам. Этого нельзя сказать, например, о странице В и связанных с нею страницах. Двигаться по такому лабиринту — сущее мучение. Никакими дизайнерскими идеями нельзя оправдать запутанность гиперссылок сайта.

Рис. *6.9.* Примеры компоновки сайта: А — основная страница рационально спроектированного сайта; В — основная страница сайта с нерациональной структурой; С — «скрытая» страница; 2,3,4 — уровни вложения страниц

В таких случаях не помогает даже маркировка пройденных ссылок. Правда, идеальных вариантов компоновки существует немного, и они трудно достижимы на практике, поэтому автор сайта должен исповедовать принцип разумной достаточности.

Не вызывает сомнений случай, когда вся информация собирается на одной странице. Такой документ может получиться очень большим. Это вполне допустимо. Но на странице должно быть минимум рисунков, иначе время ее загрузки будет приближаться к бесконечности. По правилам хорошего тона в начале такого документа должен располагаться набор местных ссылок для быстрого перехода к разделам документа, а в конце документа и, возможно, рядом с заголовками разделов предусмотрены ссылки на метку в начале страницы. Такой подход хорош для публикации справочных материалов, содержащих большое количество текста. Хотя это не единственный способ представления документов такого рода.

Другой способ создания удобного для просмотра сайта — использование ми нимального числа уровней вложения страниц и размещение большинства гипер ссылок на одной странице. Пользователь во время «хождения» по сайту постоян но возвращается к основной странице, привыкает к ее виду и имеет возможност удержать в памяти примерную конфигурацию сайта.

Но жизнь вносит свои коррективы в идеальные схемы. Так, многие авторы Web страниц размещают в своих документах многочисленные ссылки на ресурсы Ин тернета. Во многих случаях это полезно и дает пользователю массу ценной ин формации. Но это также наилучший способ увести гостя с вашего сайта. Ведь : тех местах, куда он попадет с вашей помощью, будут другие интересные адреса..

Не имеет большого смысла создавать на страницах возвращающие ссылки, кром ситуаций, когда это действительно необходимо. Любой пользователь, поработав ший с броузером, знает, что вернуться к ранее просмотренной странице можн щелкая на кнопке навигации (Back или Назад). Возвращающие ссылки необходи мы в больших документах или в том случае, когда не обойтись без многократног вложения страниц.

Каким образом пользователь попадает на ту или иную страницу? Что касается личных Web-страниц, доступ к ним обеспечивает провайдер или организация, к которой работает автор HTML-документа. Иными словами, владелец сервер; размещает на одной из своих Web-страниц необходимую гиперссылку и коммен тарий. Адреса известных фирм публикуются в книгах и журналах по компьютер ной тематике. Кроме того, набрав в броузере адрес `http://www.`*`название фирмы`*`.com` можно с большой долей вероятности попасть на сайт фирмы, чье название вам известно. Наконец, адрес Web-страницы можно получить, использовав поисковы сервер. Но вполне возможна ситуация, когда на каком-либо сервере размещен; Web-страница, на которую вообще нет гиперссылок. Если адрес страницы доста точно сложный, то вероятность того, что на нее кто-нибудь случайно наткнется крайне низка. Такая «скрытая» страница (см. рис. 6.9) может быть результатом оплошности разработчика. Но, теоретически, в Интернете могут существовать не только скрытые страницы и сайты, но и целые «подпольные» сети (массивы свя занных сайтов). Дело в том, что проконтролировать наличие «скрытых» ресурсов на том или ином сервере достаточно сложно. Во всяком случае, ни один руководи тель предприятия, имеющего сервер, не должен питать на этот счет никаких иллюзий. Попав на сайт организации, имеющей на своем сервере «скрытые» стра ницы, невозможно отыскать последние, но, зная адрес такой страницы, легко установить ее принадлежность, так как начальная часть адреса содержит инфор мацию о домашней странице организации. Эффект «скрытности» можно приме нять как полезный прием, если необходимо ограничить (до определенного пре дела) количество гостей сайта. Сложный адрес страницы играет здесь роль своеобразного пароля, хотя эта система защиты и несовершенна.

Проблема компоновки связей нашла свое отражение и в гипертекстовых редакто рах. В них создаются средства для наглядного просмотра существующих связей. На рис. 6.10 показано окно редактора AOLpress (фирмы America Online) со схе мой справочной системы, выполненной в виде мини-Web (системы HTML-доку ментов, предназначенной для использования в локальном режиме).

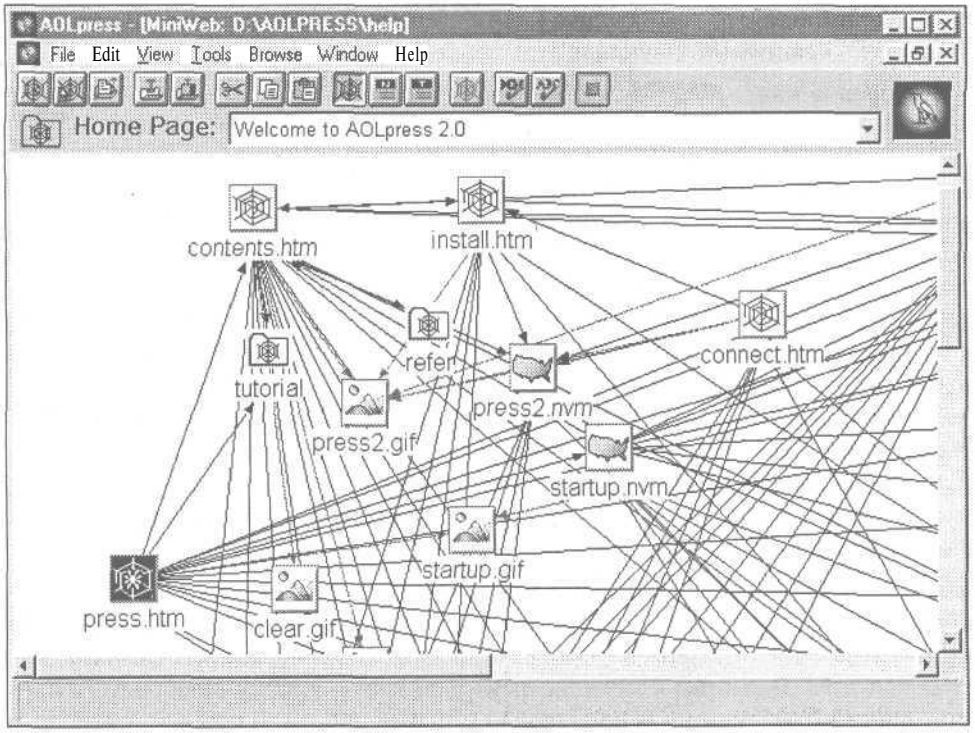


Рис. 6.10. Окно редактора гипертекста AOLpress со схемой сайта

Обилие связей не должно вас смущать. Во-первых, здесь показаны все ссылки, в том числе и на графические файлы. Во-вторых, в справочных системах создают ссылки для терминов, встречающихся в тексте документа. Понятно, что таких ссылок должно быть много. Зато пользователь имеет возможность сразу получать разъяснения по поводу непонятных слов. И наконец, это реальная система, создать которую под силу только команде профессиональных разработчиков.

Этот пример лишний раз убеждает нас в том, что область применения гипертекста шире, нежели Интернет (парадокс: куда уж шире). Способ представления информации в виде набора связанных HTML-файлов, используемых в локальном режиме, бесспорно имеет право на существование и успешно конкурирует с другими, более трудоемкими технологиями.

Собственная Web-страница

Большинство пользователей, имеющих постоянное подключение к Интернету, имеют возможность создать личную страничку объемом несколько Мбайт. Это очень хорошая возможность, и грех ею не воспользоваться. С чего начать? Что делать? Думаю, в первую очередь создатель страницы должен ответить для себя на два других вопроса. Во-первых, какова цель создания страницы, какие функции

она будет выполнять? А во-вторых, каким образом люди будут находить эту страницу среди миллионов подобных?

Когда вышла книга «HTML в примерах» («Питер», 1997), я создал страничку, которая предназначалась, в первую очередь, для пользователей, желающих самостоятельно создавать свои Web-страницы и другие гипертекстовые документы. Причем акцент делался на непосредственное использование языка гипертекстовой разметки HTML, а не на применение сложных гипертекстовых редакторов, хотя этот подход тоже обсуждался. В книге был опубликован адрес этой Web-страницы, которая работала одновременно как электронное приложение и как реально действующий пример. Поскольку страница была оборудована электронной почтой, я получил много отзывов от читателей. Надо сказать, что идея «народного» HTML — с Блокнотом и графическим редактором Paint — многим пришлась по душе. И почти во всех письмах содержались вопросы по оформлению Web-страниц. Большая часть вопросов относилась к области, которую принято называть «быстрым стартом». Читатели в первую очередь интересовались, как начать работу по созданию Web-страниц, а с изучением самого языка (как я понял из писем) проблем обычно не возникало.

Когда я создавал свою страницу, в моде были фоновые рисунки в серых тонах. Надо было придумать что-нибудь другое. Но отказываться от серого фона тоже не хотелось: окно броузера, которое служит естественной рамкой страницы, имеет серый цвет. Я выбрал в качестве фона рельефную плитку серебристого цвета (код цвета "silver" был тогда новинкой) с красным узором. В качестве узора подошли спецсимволы из общедоступного шрифта Wingdings. Я сознательно избегал рисования в графических редакторах и старался использовать только то, что есть под рукой у пользователя Windows. Правда, узор раскрасил в два оттенка красного: вид фона сразу улучшился.

На странице предполагалось использовать таблицы: в первую очередь, для создания рамок вокруг заголовков и ссылок (яркий фон не позволял размещать текст в качестве обычных абзацев). Значит, и другие детали страницы должны были быть выдержаны в этом стиле. В рисунки были добавлены рамки: в основном, это были перекрашенные рамки, которые создаются Internet Explorer и программами для Windows. Получить изображение рамки (как и других элементов пользовательского интерфейса) очень просто: достаточно нажать клавишу Print Screen и вставить содержимое буфера обмена в качестве нового рисунка в графическом редакторе.

Структура заглавной страницы была создана при помощи невидимых таблиц (`border=0`): каждый рисунок или надпись находились в отдельной ячейке. С таким документом очень легко работать, так как в гипертекстовом редакторе невидимые таблицы показаны пунктирными линиями (рис. 6.11). Взаимное положение деталей страницы тоже не изменяется в зависимости от режима просмотра (полноэкранный, оконный) и разрешения монитора.

На титульной странице была оставлена только основная информация (заголовки, фотография, счетчик, электронная почта), а все остальное было распределено по отдельным HTML-страницам и сделано доступным при помощи ссылок.

Рис. 6.11. Все детали размещены в ячейках невидимой таблицы

Определенные проблемы возникли при создании полупрозрачных изображений. Параметры альфа-канала, которые задают степень прозрачности в графических файлах, не используются броузерами. Доступной остается только возможность создания полностью прозрачных фрагментов (одного цвета) в файле формата GIF. Это натолкнуло меня на мысль об использовании рисунка в виде сетки: часть пикселов (расположенных в шахматном порядке) делается прозрачными. Получается, что на странице можно расположить рисунки как минимум в три слоя: фон страницы, полупрозрачный фон элемента, рисунок внутри элемента. Такой прием позволил создавать надписи-таблички (ссылки):

```
<TABLE align="center" border=5 cellspacing=0 cellpadding=3>
<TR><TD background="setka.gif"><A href="knigi.htm">
<IMG src="mybooks.gif" alt="MY BOOKS" border=0 width=181 height=31></a>
</table>
```

Для фона ячейки таблицы был выбран рисунок размером 2x2 пиксела (рис. 6.12,а). В приведенном выше примере это файл Setka.gif. Две точки этого рисунка-сетки сделаны прозрачными, поэтому сквозь фон элемента TD просвечивает фон страницы (рис. 6.12,б). Содержимое элемента TD - рисунок Mybooks.gif, который представляет собой надпись с прозрачным фоном. Эта надпись является и гиперссылкой.

При выборе цвета непрозрачных пикселов рисунка-сетки необходимо учитывать цвет фона страницы. Использование того же самого цвета не дает нужного эффекта. Хорошо, если сетка имеет более темный оттенок, нежели фон. Можно сделать

сетку белой. Элемент с таким фоном будет выделяться ярким пятном на темном фоне страницы. Несомненным достоинством фонового рисунка-сетки является и то, что он очень мал — всего несколько пикселов. Наличие таких рисунков практически не влияет на скорость загрузки страницы.

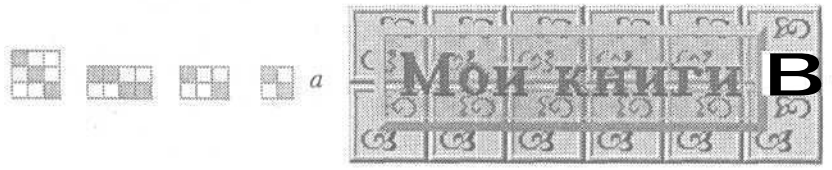


Рис. 6.12. Использование полупрозрачного фона: *а*— примеры рисунков-сеток (масштаб 800%); *б*— элемент с полупрозрачным фоном

Ну, а теперь — руководство к действию. Начинайте разработку собственной страницы с общего плана и схемы: как страница будет разделена на фреймы, где будут находиться крупные рисунки и заголовки, какими будут ссылки и цветовая палитра. В смысле поиска идей можно посоветовать следующее. Некоторые провайдеры создают наборы ссылок для доступа к страницам своих клиентов. Посмотрите, как и что сделано у других, и если вам понравится какой-нибудь эффект, откройте страницу в режиме источника (когда видны элементы HTML), посмотрите код и, при необходимости, оставьте у себя копию. Но не забывайте, что вам доступен целый Интернет примеров. Очень красивые сайты делают известные компании. Там тоже есть чему поучиться. Бывает, правда, что страница создана программным путем, и ее элементы расположены в нескольких очень длинных строках. Тогда, прежде чем разбираться в структуре страницы, надо отделить элементы друг от друга, создав дополнительные строки. Это удобно делать в Блокноте.

Посмотрите сайт своего провайдера. Там наверняка должны быть рекомендации по созданию персональных страниц (как назвать файл титульной страницы, как сделать счетчик посещений и т. д.).

Если требуется разместить на страницах большое количество графики, не следует сосредоточивать много рисунков в одном документе. Загрузка крупных рисунков и фотографий вносит самые большие задержки в просмотр страниц. Поэтому надо принимать меры к тому, чтобы пользователи не ждали подолгу.

Первый шаг в этом направлении — уменьшение объема файлов в формате JPG. Мне больше всего нравится использовать для этой цели графический редактор MS Photo Editor (он устанавливается вместе с программами MS Office). При сохранении JPG-изображения в новом файле можно выбирать показатель качества рисунка в процентах. Чуть-чуть снизив качество, можно получить значительно меньший размер файла. Правда, качество изображения — понятие условное. Разместив на экране рядом две фотографии с качеством 100% и 80%, вы вряд ли заметите различия. Но в 60-процентных изображениях ухудшения уже заметны на глаз. Вид фотографий необходимо проверять в двух режимах монитора: High Color и True Color.

Вторая задача — рациональное распределение графики по страницам. Мы уже знаем, что с помощью фреймов можно сделать легко доступным набор страниц (см. рис. 3.15). По этой схеме можно создать галерею графических объектов так, чтобы пользователь загружал только один рисунок (фотографию) по своему выбору.

Сделайте заготовки страниц, а потом заполняйте их деталями и смотрите, что получается. Страницы легко модернизировать и тогда, когда они уже опубликованы (размещены на сервере провайдера). Как только будет готова новая версия страницы, за несколько минут замените свои файлы на сервере. Для этого можно использовать программу CuteFTP или аналогичную.

Что касается популяризации своего творения, то возможности для этого есть. Сейчас активно используется обмен баннерами, гиперссылками, а также проводятся конкурсы на лучшую страницу. Можно также включить адрес своей страницы в базы данных поисковых серверов и всевозможных «желтых страниц». Я бы не стал исключать из рассмотрения и такой вариант, как реклама в средствах массовой информации (газетах, журналах). Тут многое зависит от личной инициативы.

Глава 7

Создание графики

Рисунки стали неотъемлемой частью HTML-документов, поэтому просто невозможно не рассмотреть вопросы применения графики в гипертексте. Тема, с точки зрения дизайна, поистине необъятная, но, к счастью, с практической стороны дело обстоит намного проще. Так же, как и в случае с гипертекстовыми редакторами, в этой области мы сталкиваемся с большим количеством программных средств, призванных облегчить жизнь (а точнее процесс рисования) авторам Web-страниц. Графика для Интернета несколько отличается от обычной, и эти отличия будут рассмотрены в данной главе.

Форматы графических файлов

Создавать рисунки можно самыми разными способами, используя любые (общепринятые и экзотические) графические пакеты и форматы файлов. Но в конечном итоге потребуется преобразовать результаты своей работы в один из стандартных для Интернета форматов. Сделать это очень легко: надо открыть ваш рисунок в графическом редакторе и выполнить команду Сохранить как (Save As), выбрав для нового файла формат GIF, JPG или PNG. Из числа наиболее популярных программ для обработки графики хотелось бы упомянуть PhotoShop 5.x фирмы Adobe или Microsoft Photo Editor. Кроме того, для иллюстрации практической работы я выбрал интересный, на мой взгляд, пакет Gif Construction Set for Windows 95, созданный фирмой Alchemy Mindworks (см. приложение Г). Наверное, не у каждого читателя этой книги будет возможность установить на своем компьютере мощный пакет для рисования и анимации, а также потратить время на его изучение, поэтому в приведенных примерах я старался использовать простые графические редакторы, например, MS Paint.

Самым распространенным форматом графических файлов для HTML-документов является GIF (Graphic Interchange Format). Для кодирования цвета в нем используется 8 бит, то есть допускается 256 различных цветов или столько же градаций серого. Наборы цветов (палитры) могут быть различными. Один GIF-файл может содержать несколько изображений, позволяющих создавать движущиеся или изменяющиеся образы. В этом случае требуется согласование палитр различных изображений, составляющих один файл. Похожая проблема возникает, если монитор работает в режиме 256 цветов, а на экране одновременно воспроизводятся несколько изображений с разными палитрами. Очевидно, что для части картинок качество цветопередачи будет ухудшено.

Само по себе 8-битовое кодирование цвета предполагает, что размер графического файла должен быть относительно небольшим. Но, кроме того, для уменьшения размера файла используется еще и сжатие изображения. Существует две разновидности GIF-файлов: сжатые и обычные, в которых сжатие отсутствует. Компактность файла и дополнительные преимущества стали причиной того, что данный формат прочно утвердился в качестве стандарта де-факто для Интернета.

Дополнительное преимущество заключаются в том, что хранение информации в файле может быть организовано таким образом, чтобы при выводе рисунка происходило чередование строк. То есть вначале будут выводиться строки с номерами 1, 5, 9 и т. д., затем с номерами 2, 6, 10 и т. д., и так до тех пор, пока весь рисунок не будет отображен. Для наблюдателя такой рисунок вначале кажется нечетким, а затем четкость изображения увеличивается. При передаче данных по сети это особенно выгодно, так как не требует полной передачи файла и позволяет увидеть изображение сразу же, хотя и в несколько размытом виде.

Другое преимущество GIF — возможность сделать часть изображения прозрачной. Прозрачным может стать только один цвет. Так, создание прозрачного фона позволяет более естественно вписать рисунок в документ и избежать появления прямоугольника, обозначающего границы изображения.

Для формата GIF разработано несколько спецификаций. В соответствии со спецификацией GIF89a графический файл может состоять из нескольких блоков.

- Блок заголовка HEADER содержит информацию о размере экрана и палитре.
- Блок текста PLAIN TEXT позволяет добавлять символьные данные к рисунку.
- Блок изображения IMAGE содержит одну картинку, импорт которой может осуществляться не только в формате GIF. При создании движущихся изображений в один файл включается несколько таких блоков.
- Блок управления CONTROL используется для размещения флагов прозрачности и ожидания, а также для определения временной задержки при выводе нового блока изображения.
- Блок приложения APPLICATION предназначен для хранения служебной информации.
- Блок комментария COMMENT используется для размещения произвольной информации. Эти данные не выводятся на экран во время воспроизведения GIF-изображения.
- Блок цикла LOOP необходим для многократного воспроизведения движущейся картинки. В этом блоке задается число повторений при показе «ролика».

Вторым подходящим для Интернета графическим форматом является JPEG (JPG), названный так в честь своего разработчика — Join Photographic Experts Group. Этот формат обеспечивает 24-битовое кодирование цвета и лучше подходит для хранения таких изображений, как фотографии. Недостатком формата является возможность искажения цвета в результате сжатия данных.

Для Интернета был разработан еще один формат графики — PNG (Portable Network Graphics). Он создан с целью замены формата GIF. В отличие от своего

прототипа новый формат позволяет использовать как 8-битовое, так и 24-битовое кодирование цвета. Кроме того, алгоритм создания прозрачности усовершенствован. С помощью альфа-канала прозрачность может быть задана для участков изображения, содержащих разные цвета. Для 24-битовых форматов прозрачность может быть неполной, то есть разработчик устанавливает ее величину в процентах (от 0 до 100). К сожалению, броузеры пока не научились воспроизводить полупрозрачные изображения.

Создание фона HTML-документа

К фоновому рисунку HTML-документа, безусловно, предъявляется ряд требований. В зависимости от того, какой вид хочет придать своей странице автор, выбирается направление конструирования фона. До недавнего времени классическим решением было создание бледно-серого фона с таким же бледным, но рельефным рисунком. Здесь очень многое зависит от художника, но современные графические редакторы позволяют создавать похожие эффекты и автоматически. Такой фон не должен ощутимо снижать контрастность страницы и мешать чтению текста.

В последнее время в моде на фоновые рисунки произошли изменения. Все чаще можно встретить белый фон. Белый цвет вне конкуренции, и тут не надо что-либо объяснять. Другое направление — использование бледного фона, но другого цвета, например, бежевого, голубого или зеленого. Сам рисунок выглядит как произвольный узор из точек, напоминая поверхность кожи или камня. Здесь опять-таки, полет фантазии художника ограничивается необходимостью обеспечения контрастности.

Наконец, для экзотических страниц самым популярным фоном остается черный. Его разнообразят изображениями звезд (для страниц с космической тематикой), создавая ночное небо, или дополняют кроваво-красными надписями или рисунками (если надо сделать «круто»).

В любом случае источником фона служит рисунок небольшого размера. Заранее невозможно предугадать, какой размер на экране займет документ, так как броузер выполняет автоматическое форматирование. В результате рисунок фона (фрагмент) будет тиражироваться так, чтобы заполнить все пространство окна. Разработчик страницы должен решить, как будут совмещаться левая и правая, а также верхняя и нижняя стороны исходного рисунка.

Рассмотрим процедуру подготовки изображения для фона. В качестве примера я хочу использовать текстуру. Как правило она имеет нерегулярный рисунок и не содержит ярко выраженных деталей. Фрагментами могут быть изображения облаков, деревьев, камней, поверхности воды. Основой фона документа может стать и фрагмент фона картины, выполненной акварелью или маслом. Все это легко найти в Сети. Так, например, для получения древесного рисунка вовсе не обязательно искать изображение среза дерева. Можно использовать подходящую фотографию дерева и вырезать кусочек изображения ствола. Один раз мне попалась фотография старого дерева с содранной временем и ветром корой. Кусочек фотографии пригодился для создания «орехового» фона Web-страницы. На рис. 7.1

видно, что рисунок многократно повторяется и границы между фрагментами хорошо заметны.

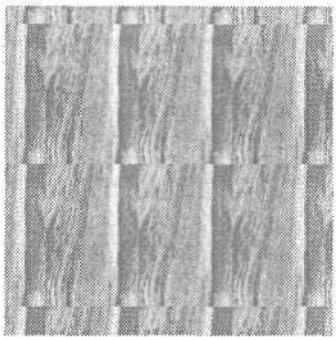


Рис. 7.1. Для фона страницы использован фрагмент фотографии древесного ствола

Чтобы убрать границы, надо использовать четыре одинаковых изображения. На рис. 7.2 показана схема их преобразования и соединения. Для трех фрагментов надо выполнить операции зеркального отображения. В редакторе Paint это можно сделать, выбрав команду Рисунок▸Отразить/повернуть. Затем требуется аккуратно соединить четыре фрагмента так, чтобы между ними не оставалось пустого пространства, но и не было наложения. Это удобно сделать в режиме увеличения. В конце работы полученный рисунок перемещают в верхний левый угол рабочего поля и выбирают размер рисунка так, чтобы не оставалось незаполненного пространства. Осталось конвертировать изображение в формат JPG или GIF.

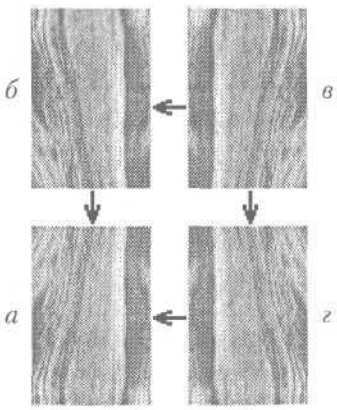


Рис. 7.2. Соединение четырех изображений: а — исходный фрагмент; *б*— фрагмент, отраженный сверху вниз; я— фрагмент, отраженный сверху вниз и слева направо; *г*— фрагмент, отраженный слева направо

На рис. 7.3 показано, как изменился вид фона после всех преобразований. Таким образом, если вы путешествуете по Web, имеет смысл коллекционировать интересные фотографии и рисунки. Они все равно записываются в кэш на вашем диске, и

их надо только периодически копировать в отдельную папку. При выборе и использовании фрагментов изображений необходимо помнить, что существуют авторские права их создателей, и соблюдать меру.

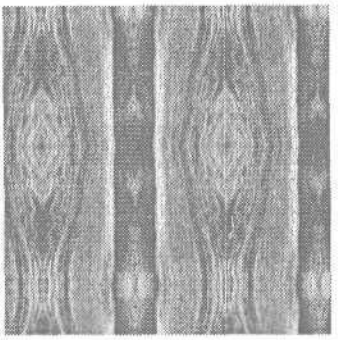


Рис. 7.3. Фон без видимых границ между фрагментами

Мы рассмотрели только один прием создания фона. Существует много других способов подготовки таких рисунков. Например, можно использовать исходный рисунок в виде длинной узкой полосы, которая будет с гарантией превосходить ширину окна броузера. Высота полосы может быть уменьшена вплоть до одного пиксела. Учитывая, что в каждом ряде фона исходный рисунок будет воспроизводиться с самого начала, можно получить такие эффекты, как вертикальные полосы или плавные переходы цвета.

К исходному файлу фона предъявляются два основных требования: файл не должен быть большим, а фон не должен препятствовать свободному чтению документа.

ПРИМЕЧАНИЕ

«Ореховый» фон можно увидеть в файле Graphics.htm на прилагаемой дискете.

Прозрачность для GIF- и PNG-изображений

Выше, в разделе «Форматы графических файлов», уже упоминалось, что один из цветов изображения формата GIF можно сделать прозрачным. Чаще всего прозрачным делают цвет фона. Рисунок с прозрачным фоном, размещенный в документе, смотрится совсем иначе. Он как бы сливается с документом, становясь его неотъемлемой деталью.

Многие графические редакторы позволяют устанавливать прозрачность. Так, на рис. 7.4 показан редактор изображений MS Photo Editor. Для открытия окна, которое позволит выбрать нужный цвет, следует щелкнуть на кнопке Set Transparent Color (установить прозрачный цвет). Затем, когда указатель мыши превратится в

наклонную стрелку, щелкните на нужном цвете. Обратите внимание, что в окне диалога показаны числовые характеристики выбранного цвета. Это можно использовать и для того, чтобы быстро узнать величины базовых цветов для любого оттенка, не устанавливая прозрачность в действительности.

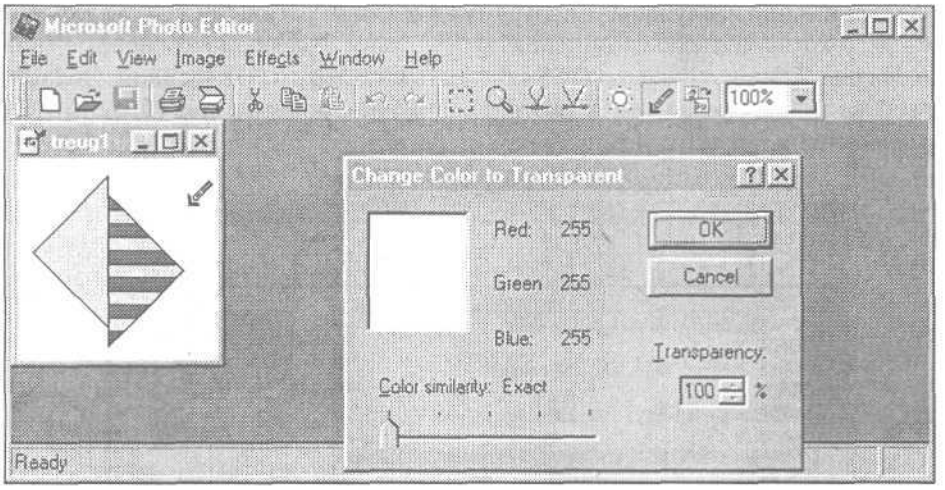


Рис. 7.4. Создание прозрачного цвета в графическом редакторе MS Photo Editor

На рис. 7.5 показан фрагмент Web-страницы Graphics.htm, на которой для сравнения помещены два рисунка: с непрозрачным и прозрачным фоном. При создании эффекта прозрачности в GIF-файле необходимо учитывать, что информация о выбранном цвете теряется, поэтому перед проведением такой операции необходимо создать страховочную копию файла.

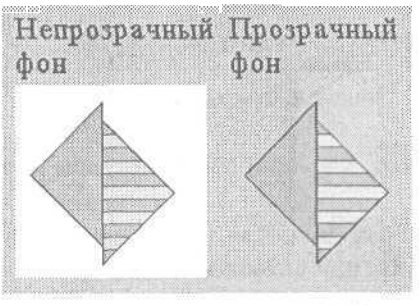


Рис. 7.5. Варианты рисунка с различными видами фона

Редактор MS Photo Editor также позволяет определять прозрачный цвет для файлов формата PNG. На рис. 7.4 видно, что кроме указания цвета можно задать степень прозрачности в процентах. Это возможно только для 24-разрядных файлов. Разумеется, любое изображение может быть конвертировано в такой формат.

Программа Gif Construction Set

В следующих разделах этой главы речь пойдет о создании эффекта движения (анимации) при помощи GIF-файлов. Поэтому вначале нам необходимо познакомиться с инструментарием, который позволяет решать такие задачи. На мой взгляд, внимания заслуживает программа Gif Construction Set (GCS) для Windows 9x, о которой я уже упоминал в начале главы. Разберем ее основные функции. На рис. 7.6 показано окно GCS с открытым сложным GIF-файлом. Область в правой части окна позволяет посмотреть одиночное изображение (блок IMAGE) в уменьшенном формате.

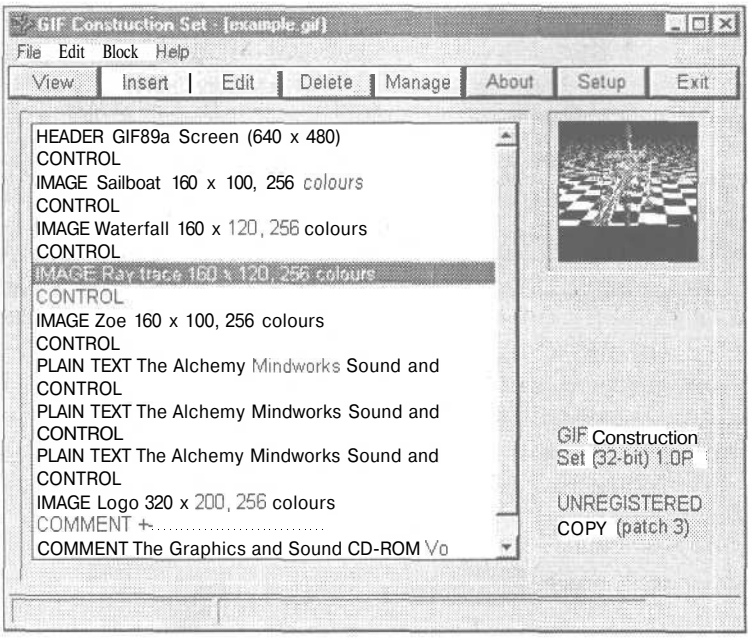


Рис. 7.6. Окно Gif Construction Set

Для выполнения основных операций предусмотрены кнопки, вынесенные на основную панель.

View — Просмотр изображения. Для прекращения просмотра необходимо нажать клавишу Esc или щелкнуть правой кнопкой мыши.

Insert — Добавление нового блока. На рис. 7.7 показана панель, которая позволяет выбрать нужный блок из тех, с которыми мы познакомились ранее (см. раздел «Форматы графических файлов»).

Edit — Редактирование блока. Как правило, для каждого блока необходимо установить набор параметров. Это делается в режиме диалога после выбора блока и щелчке на этой кнопке.

Delete — Удаление выделенного блока.

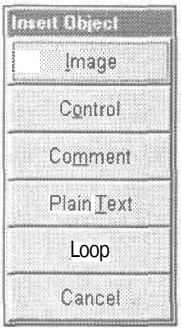


Рис. 7.7. Панель для вставки нового блока в GIF-файл

Manage

1 Работа **с несколькими** блоками CONTROL в одном файле. Соответствующее окно диалога показано на рис. 7.8.

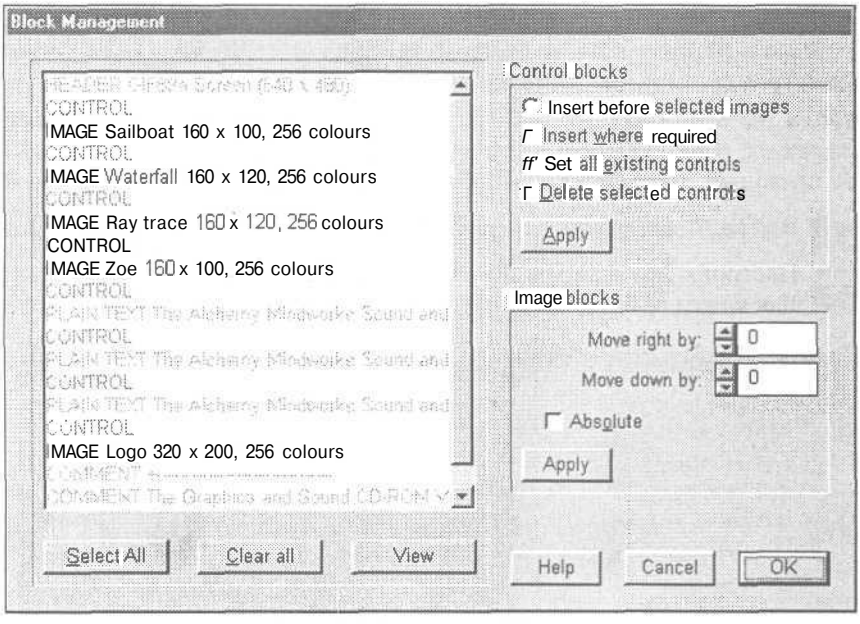


Рис. 7.8. Окно режима Manage

Используя это окно, можно выполнить ряд операций.

- Insert before selected images. Создание нового блока CONTROL перед каждым из выбранных изображений. Как правило, в сложных файлах для каждого блока изображения создается свой блок CONTROL. Это позволяет настраивать параметры каждого изображения отдельно.
- Insert where required selected images. Создание блоков CONTROL перед выбранными блоками изображений там, где блоки CONTROL отсутствуют. Считается, что каждый блок изображения должен предваряться блоком управления. При

нарушении этого условия программа выдает сообщение об ошибке в момент сохранения файла. На практике выполнение этого условия не всегда обязательно: например, при анимации, когда все блоки изображения имеют одинаковые характеристики и выводятся через равные промежутки времени.

- Set all existing controls. Установка одинаковых параметров во всех выбранных блоках. Для этого надо выполнить следующие действия.

  1. Выделить необходимые блоки CONTROL в окне Block Management.
  2. Установить переключатель напротив данной команды и щелкнуть на кнопке Apply.
  3. В открывшемся окне Edit Control Block установить новые параметры и щелкнуть на кнопке OK.
  4. Щелкнуть на кнопке OK в окне Block Management.

- Delete selected controls. Удаление выбранных блоков управления.
- Группа инструментов Image blocks позволяет управлять положением некоторых блоков изображения при выводе на экран всего файла. Перед установкой параметров соответствующие блоки CONTROL должны быть выделены.
- Кнопка Select All позволяет выделить все блоки CONTROL в списке окна Block Management. Кнопка Clear All снимает выделение со всех блоков CONTROL. Кнопка View позволяет просмотреть GIF-файл, не выходя из окна Block Management.

About — Получение справки о фирме-производителе и о регистрации продукта.

( ) — Настройка программы. На рис. 7.9 показано окно диалога режима Setup. Как видим, здесь достаточно много параметров. Рассмотрим их назначение.

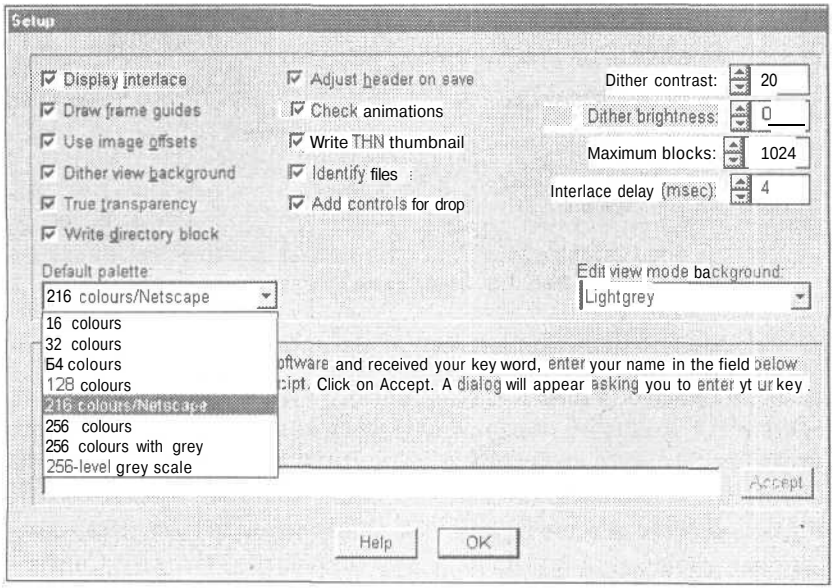


Рис. 7.9. Окно для установки параметров GCS

- Display interlace. Включение и отключение эффекта чередования строк и других похожих эффектов при выводе изображения.
- Draw frame guides. В режиме редактирования блока изображения существует возможность просмотра отдельного блока. В этом случае через границы рисунка проводятся пунктирные линии, которые позволяют лучше понять положение рисунка на экране.
- Use image offsets. При просмотре одиночного блока изображения этот параметр позволяет увидеть картинку в центре экрана или в том месте, где она будет выведена при просмотре всего GIF-файла.
- Dither view background. Этот режим используется, если на компьютере установлен цветовой режим 256 цветов или меньше. Установка данного флажка разрешит выполнение подбора цвета фона с помощью доступных цветов.
- True transparency. Управление режимом создания прозрачности. Чтобы данный эффект был передан верно, помимо установки данного флажка необходим выбор режима цветопередачи монитора High Color или True Color.
- Write directory block. Если этот флажок установлен, то программа создает в файле блок TITLE, в котором размещает сведения о местонахождении файлов изображений, включенных в качестве блоков IMAGE в данный файл. При создании изображений для Интернета этот флажок должен быть снят.
- Adjust header on save. Проверка и вычисление значений для указания ширины и высоты экрана в блоке заголовка во время сохранения файла.
- Check animations. Проверка GIF-файлов, содержащих несколько изображений, на предмет корректности структуры. Проверка выполняется перед сохранением, и в случае обнаружения ошибок выводятся соответствующие подсказки.
- Add controls for drop. GCS-поддержка технологии Drag and Drop (перетащить и оставить) при вставке новых блоков изображений. При установке данного флажка программа будет автоматически создавать для каждого блока изображения блок CONTROL, если блок IMAGE включен в состав файла перетаскиванием.
- Dither contrast и Dither brightness. Выбор параметров контрастности и яркости соответственно для импорта изображения из другого файла. Из-за ограниченности количества цветов в GIF-файле при импорте иногда происходит потеря качества изображения. В таких случаях как раз и необходим тщательный подбор параметров.
- Maximum blocks. Установка допустимого количества блоков в GIF-файле. После изменения этого параметра необходимо заново открыть файл, чтобы новое значение стало актуальным.
- Interlace delay. Установка величины задержки (в миллисекундах) после вывода каждой строки изображения, в котором задан эффект чередования, то есть вывод строк не по порядку.
- Default palette. Выбор типа палитры по умолчанию. На рис. 7.9 этот список показан в раскрытом виде.
- Edit view mode background. Выбор цвета фона для режима просмотра при редактировании блока изображения.

Теперь можно перейти к обсуждению режимов редактирования отдельных блоков. На рис. 7.10 показано окно для редактирования блока заголовка. Напомню, что для перехода в режим редактирования блока надо выделить последний и щелкнуть на кнопке Edit. Установка ширины (Screen width) и высоты (Screen depth) - скорее формальность, чем необходимость. В том случае, когда размер экрана задан меньше истинного, лишнее пространство заполняется черным цветом. Кнопка Background позволяет активизировать палитру для выбора цвета фона. Все эти параметры не имеют смысла, если GIF-файл используется в другой программе. Флажок Global palette позволяет реализовать общую палитру в GIF-файле. Общая палитра создается по умолчанию при формировании файла с несколькими изображениями. Работать с ней надо осторожно, так как с ее помощью легко испортить изображения. Кнопка Save позволяет сохранить палитру. Палитра может быть записана в отдельный файл формата CMP, а затем импортирована в другой GIF-файл.

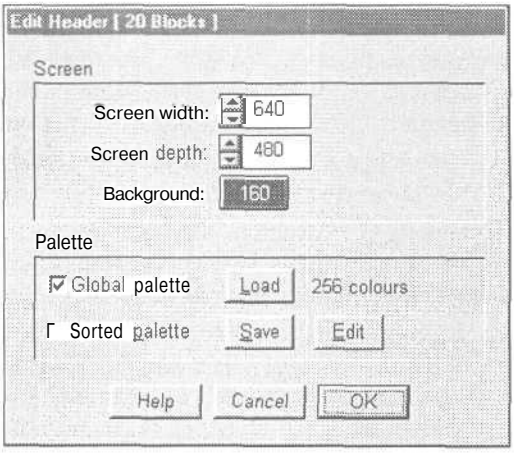


Рис. 7.10. Окно для редактирования блока HEADER

При редактировании блока изображения окно диалога выглядит так, как показано на рис. 7.11. Поля со счетчиками Image left и Image top позволяют задать положение рисунка относительно левого верхнего угла экрана. Установка флажка Interlaced обеспечит вывод рисунка в режиме чередования строк. Группа Palette содержит инструменты для создания собственной палитры изображения. Поле ввода Block title позволяет ввести название блока. Эта информация служит комментарием в GCS и при показе GIF-файла не используется. Кнопка View позволяет просмотреть отдельный блок. Это так называемый «режим просмотра при редактировании» (edit view mode). Он отличается от основного режима просмотра. Средства непосредственного редактирования изображения в GCS крайне бедны. Они доступны в режимах редактирования блоков CONTROL и IMAGE, Так что для подготовки отдельных изображений лучше использовать графический редактор.

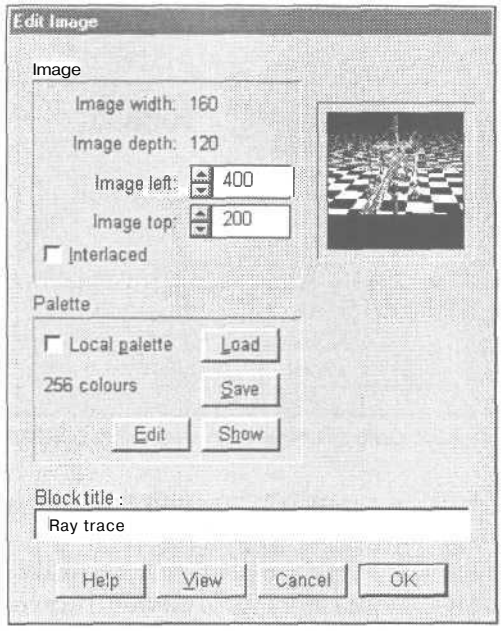


Рис. 7.11. Окно для редактирования блока IMAGE

Окно для редактирования блока управления показано на рис. 7.12. Напомню, что каждый блок изображения может (и должен) дополняться блоком управления. Последний обязан располагаться перед блоком изображения. Для создания прозрачного цвета необходимо установить флажок Transparent colour и выбрать цвет при помощи одного из инструментов, расположенных справа от флажка. Надо заметить, что к выбору прозрачного цвета в данном случае надо подходить с осторожностью. Назначение прозрачности не создает проблем в основном для простых рисунков, в которых использовано мало цветов или существующие цвета легко отличить друг от друга. В некоторых изображениях, таких, как фотографии, могут присутствовать несколько оттенков одного цвета. При этом не имеет значения, что в палитре может быть не более 256 цветов. Оттенки могут быть трудно различимы для глаза. В результате пользователь может думать, что область рисунка закрашена одним цветом. Если для такой области попытаться назначить прозрачность, то прозрачными станут отдельные пикселы и изображение будет испорчено.

Поле Delay позволяет установить величину задержки (в сотых долях секунды) перед выводом следующего блока. Список Remove by дает возможность определить, надо ли удалять изображение после того, как оно показано. Определяется также, каким изображением заменяется удаленная картинка. В списке могут присутствовать следующие значения:

- Nothing — без изменений;
- Leave as is — оставить как есть;
- Background — изображение заменяется фоном;
- Previous image — изображение заменяется предыдущим изображением.

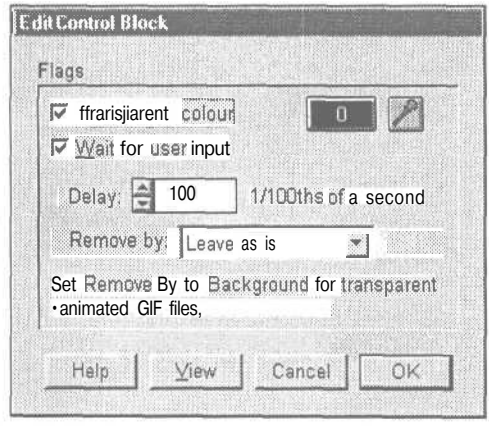


Рис. 7.12. Окно для редактирования блока CONTROL

При разработке GIF-файлов для WWW следует выбрать значение Nothing или Background.

Обратите внимание на подсказку в окне редактирования блока CONTROL: Set Remove by to Background for transparent animated GIF files. На русском языке это напоминание выглядит так: «Выбирайте в списке Remove by значение Background для прозрачных движущихся GIF-файлов». Действительно, это правило должно всегда соблюдаться.

Допустим, мы имеем анимационный GIF-файл и хотим создать в нем прозрачный фон. Для этого надо выполнить следующую последовательность действий.

1. Откройте окно Block Management щелчком на кнопке Manage.
2. Щелкните на кнопке Select All.
3. Установите переключатель Set all existing controls в группе Control blocks и щелкните на кнопке Apply.
4. В открывшемся окне Edit Control Block установите флажок Transparent Colour, щелкните сначала на кнопке с изображением пипетки, а затем — на фоне изображения.
5. Закройте оба окна щелчком на кнопках OK.

В движущихся изображениях с прозрачным фоном можно получить интересный визуальный эффект, если в списке Remove by выбрать значение Nothing. Изображения будут накладываться друг на друга. В некоторых случаях это позволяет создавать эффект следа от движения.

На рис. 7.13 показано окно для редактирования блока текста. В левой части окна находится область, где набирается текст, а все инструменты справа предназначены для его форматирования.

Окно для редактирования блока цикла — самое простое. Оно показано на рис. 7.14. Здесь пользователь может установить число повторений при выводе блоков изображений GIF-файла. По этой величине и длительности задержки вывода одиночного изображения можно определить время воспроизведения анимационного

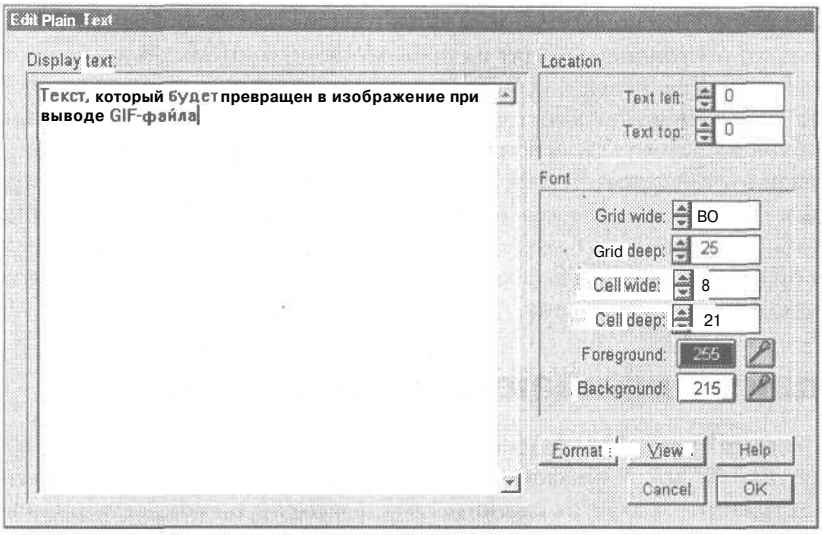


Рис. 7.13. Окно для редактирования блока PLAIN TEXT

файла. После того как нужное количество итераций выполнено, смена изображений прекращается. В общем случае не имеет значения, что представляет собой сложный GIF-файл: набор версий одной и той же картинки или набор разнородных изображений и текстовых фрагментов. Эффекты анимации могут быть использованы с равным успехом как для создания движения, так и для создания сложного мозаичного изображения, в котором отдельные детали выводятся последовательно. Наличие блока LOOP само по себе является причиной цикличности процесса, поскольку без этого блока каждое изображение выводится только один раз.

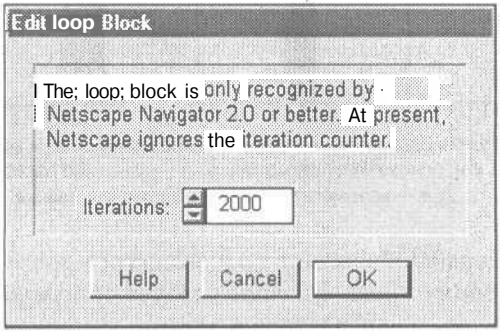


Рис. 7.14. Окно для редактирования блока LOOP

В окне, показанном на рис. 7.14, имеется предупреждение, что блок LOOP поддерживается броузером Netscape 2.0 или более поздней его версией. Получается, что разработчик Web-страницы должен учитывать, в каком броузере будет просматриваться его документ? Это интересный аспект создания Web-страниц, на котором надо остановиться особо. Дело в том, что подобные предупреждения верны

только отчасти. Они позволяют нам увидеть возможную проблему. Что касается броузеров, то постоянно выходят их новые версии, в которых фирмы-производители стараются реализовать все получившие признание нововведения. С другой стороны, никто не может дать гарантию, что при просмотре Web-страницы у какого-нибудь пользователя не возникнут проблемы из-за того, что в странице был реализован нестандартный эффект, а данный броузер не может его воспроизвести. Таким образом, разумное решение, на мой взгляд, заключается в том, чтобы разрабатывать страницы без оглядки на ограничения, связанные с версиями и типами броузеров, но использовать новинки дизайна так, чтобы их отсутствие не помешало прочитать страницу.

Создание вращающегося значка

В этом разделе будет обсуждаться пример изображения, вращающегося в плоскости, перпендикулярной плоскости экрана. Существует много случаев, когда автору Web-страницы требуется оживить свой документ, но нет достаточно времени для создания сложной движущейся картинки. Как быть в таком случае? Самый простой способ — использовать возможности обычных графических редакторов, установленных на большинстве компьютеров. Так, редактор Paint позволяет легко построить вращающееся изображение, основу которого составляет окружность.

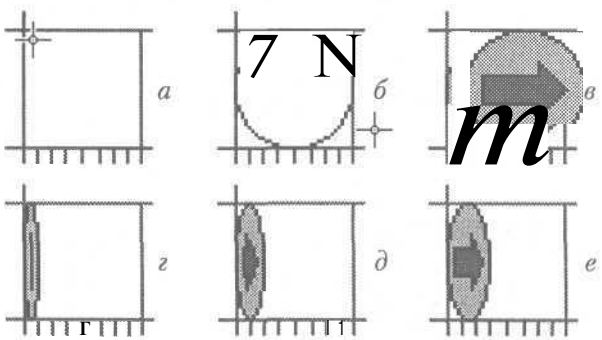


Рис. 7.15. Этапы создания изображений вращающегося значка: *а* — вспомогательная сетка; *б*— окружность как основа будущего значка; *в* — основное изображение значка; *г, д, е*— вспомогательные изображения значка

Создадим для примера изображение вращающегося вокруг вертикальной оси значка, имеющего форму круга. Вначале следует определить размер будущего изображения и нарисовать в редакторе в режиме увеличения вспомогательную сетку, которая должна обозначать границы рисунка (рис. 7.15, а). Точка пересечения левой вертикальной и верхней горизонтальной линий требуется для позиционирования указателя мыши перед началом рисования окружности. Для создания окружности используется инструмент Эллипс. После того как окружность нарисована, создается изображение значка. Эти этапы показаны на рис. 7.15, б, в. На одной из горизонтальных линий сетки наносятся штрихи, которые будут определять ширину вспомогательных рисунков. Отрезки, расположенные ближе к центру

основного рисунка, могут быть несколько шире тех, которые находятся у краев. Затем формируется набор эллипсов разной ширины и создается ряд изображений повернутого значка (рис. 7.15, г-е). Чтобы получить изображения фигуры, совершающей полный оборот вокруг оси, достаточно создать кадры, соответствующие фазам поворота фигуры на 90 градусов, то есть от момента, когда изображение повернуто к наблюдателю ребром, до момента, когда оно полностью развернуто в плоскости экрана.

Теперь осталось выбрать воображаемую ось, вокруг которой будет происходить вращение. Можно задать вращение вокруг вертикальной оси, проходящей через центр основного изображения. В этом случае все промежуточные изображения окружностей и эллипсов должны быть отцентрированы относительно границ кадра. Этот способ имеет недостаток: при использовании симметричных фигур трудно определить направление вращения. Кроме того, в данном случае плохо выражена объемность движения. Последняя исправляется созданием тени или прорисовкой боковой поверхности фигуры. Можно также задать вращение вокруг оси, проходящей через левый или правый край основного изображения.

Далее приступают к созданию отдельных кадров. Все они должны иметь одинаковый размер. Изображения очищаются от вспомогательных линий, вырезаются и сохраняются в виде отдельных файлов (один кадр — один файл). Для сохранения можно использовать форматы BMP, GIF, JPG, PCX, PNG.

На рис. 7.16 показан набор кадров, содержащих изображения значка, у которого на лицевой стороне нарисована стрелка, а обратная сторона закрашена одним цветом. В данном случае вспомогательные изображения выровнены относительно левой и правой границ кадра так, чтобы создать видимость не вращения, а движения вдоль поверхности воображаемого вертикального цилиндра. Это помогло избежать упомянутых выше недостатков.

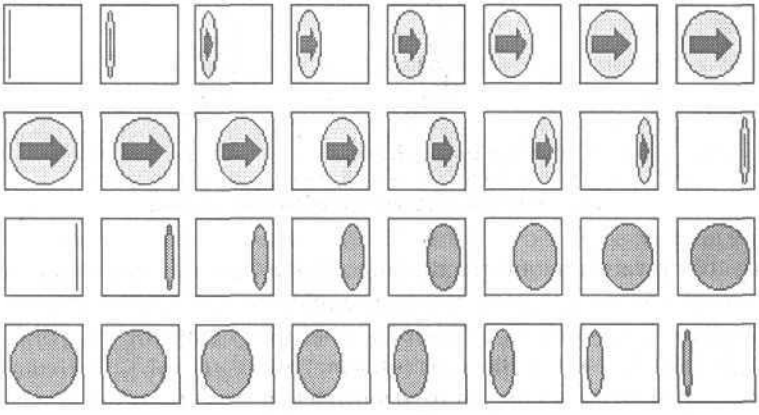


Рис. 7.16. Набор изображений вращающегося значка

В данном примере рассмотрен только один вариант использования инструментов графического редактора. Разумеется, существует множество способов создания разнообразных визуальных эффектов. При помощи инструмента Эллипс редактора

Paint можно создавать и другие виды анимации: изменение размера круга, вращение вокруг двух осей и т. д.

ПРИМЕЧАНИЕ

Вращающийся значок можно увидеть в файле Graphics.htm на прилагаемой дискете.

Компоновка сложного GIF-файла

После того как созданы отдельные кадры движущегося изображения, можно приступать к созданию GIF-файла. Для этой цели можно использовать программу Gif Construction Set. Вся операция выполняется в несколько приемов.

1. Создание нового файла командой File ▸ New. Программа создаст файл с блоком заголовка HEADER.
2. Создание блока LOOP, если необходимо, чтобы движение было непрерывным (зацикленным). Щелкните на кнопке Insert, а затем — на кнопке Loop. Можно сразу же установить число итераций. Щелкните на кнопке Edit и введите требуемое число.
3. Вставка блоков изображения IMAGE. Щелкните на кнопке Insert, затем — на кнопке Image и выберите графический файл, содержащий кадр. Повторите эти действия для всех остальных файлов с кадрами. Перед вставкой изображения программа выводит на экран окно диалога, показанное на рис. 7.17. Здесь пользователю предлагается выбрать способ преобразования палитры изображения. Я рекомендую использовать следующие варианты:
   - D Use a local palette for this image — если блок изображения должен использовать собственную палитру;
   - □ Use a local grey palette for this image — если надо получить изображение в оттенках серого;
   - D Remap this image to the global palette — дает хороший результат в большинстве случаев;
   - П Use this image as the global palette — если необходимо создать общую палитру на основе данного изображения.
4. Теперь можно проверить очередность появления блоков изображения. Выбирайте их последовательно и просматривайте изображение в правой части окна. В случае обнаружения ошибки следует удалить неверно расположенные блоки и вставить их в нужной последовательности.
5. Создание и редактирование блоков управления CONTROL. Щелкните на кнопке Manage. В окне Block Management (см. рис. 7.8) щелкните на кнопке Select All. Установите переключатель Insert where required и щелкните на кнопке Apply. В открывшемся окне Edit Control Block (см. рис. 7.12) установите величину задержки, создайте (если необходимо) прозрачный цвет и выберите значение в списке Remove by. Вопросы редактирования блоков управления обсуждались ранее в разделе «Программа Gif Construction Set». Закройте два последних открытых окна щелчком на кнопках OK.

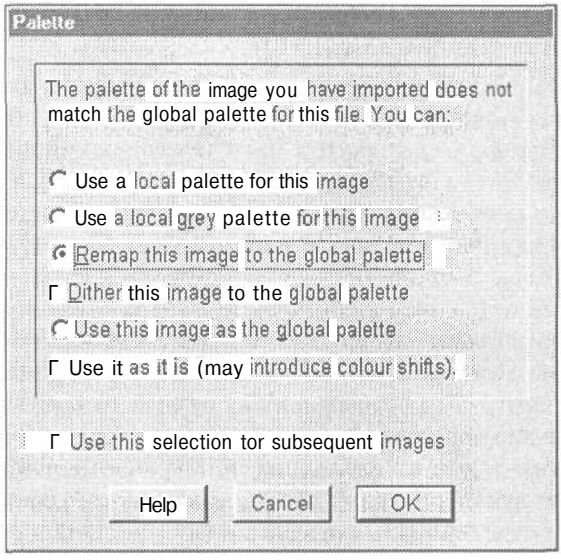


Рис. 7.17. Окно для выбора способа преобразования палитры

6. Просмотр готового изображения. Щелкните на кнопке View на главной панели GCS. На этапе отладки движение можно просматривать и в замедленном темпе, если установить достаточно большую величину задержки (50-1000).

В дальнейшем GIF-файл может быть доработан средствами GCS. Например, для отдельных блоков изображения могут быть установлены различные величины задержки.

Создание трехмерной вращающейся фигуры

Перед тем как браться за выполнение этой задачи при помощи «подручных» средств, необходимо продумать общую идею изображения. Дело в том, что самой сложной проблемой в данном случае является прорисовка повернутого изображения. Поэтому сама идея рисунка должна допускать процедуру упрощенного рисования. Например, удобно использовать геометрические узоры. В любом случае целесообразно нарисовать вначале «проволочный каркас» фигуры, получить набор кадров, а затем раскрасить изображение и нарисовать детали. Предполагается, что в рисунке присутствуют, в основном, плоские поверхности и отсутствуют эффекты тени и освещенности. В некоторых случаях процесс рисования можно облегчить, если предусмотреть набор точек, которые позволят выполнить прорисовку деталей. Так, любой отрезок можно мысленно разделить пополам и для любой его проекции указать точку, соответствующую середине. Существуют и другие «удобные» точки. Разумеется, все это справедливо для изображений небольшого размера. Если же задуманная вами идея не укладывается в рамки

изложенных принципов, для рисования лучше выбрать специализированный графический пакет.

В качестве примера рассмотрим процесс создания вращающегося октаэдра. Вначале необходимо построить вертикальную ось вращения и траекторию движения боковых углов. Используем редактор Paint и уже знакомый нам инструмент Эллипс (рис. 7.18, а). Для работы следует выбрать масштаб 4:1 или 6:1 и включить сетку (при помощи комбинации клавиш Ctrl+G). Обратите внимание, что в масштабе увеличения эллипс представлен в виде набора горизонтальных отрезков. Этим эффектом можно воспользоваться. При расстановке точек, соответствующих боковым углам октаэдра, задействуем края этих отрезков. За каждый кадр углы фигуры должны смещаться на один отрезок. Немаловажное значение имеет тот факт, что длины отрезков в разных частях эллипса неодинаковы. Дело в том, что когда человек смотрит на вращающийся предмет, то движение деталей, перемещающихся мимо наблюдателя, кажется более быстрым, нежели движение деталей по направлению к или от наблюдателя. Неравномерность длин отрезков эллипса как нельзя лучше подходит для реализации этого эффекта. Кроме того, привязка к концам отрезков позволяет обеспечить цикличность движения и указать в каждом кадре положение всех четырех боковых углов фигуры. Сделав полный оборот, фигура должна занять положение, которое было вначале. В данном случае неважно, что некоторые из упомянутых отрезков состоят только из одного пиксела. При необходимости часть таких коротких отрезков можно пропустить.

Как только определены положения вершин фигуры, легко нарисовать контур, используя инструмент Линия (рис. 7.18, б-г). Прямые линии в масштабе увеличения таковыми не кажутся, но в масштабе 1:1 рисунок выглядит более аккуратным. Типичный размер подобного рисунка — 100x100 пикселов.

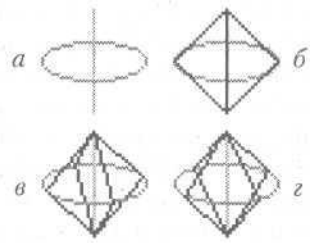


Рис. 7.18. Создание изображений проволочного контура (масштаб 4:1): *а* — ось и траектория вращения; *б, в, г* — варианты изображения

Теоретически, проволочный контур можно и не закрашивать. Подобные изображения, так же как и штриховые рисунки, не утомляют своим движением глаз, особенно если имеют прозрачный фон. В определении количества движущихся на Web-странице объектов необходимо соблюдать меру.

Если посмотреть на вращающийся проволочный контур такой фигуры, как октаэдр, то определить направление вращения практически невозможно. У наблюдателя будет складываться впечатление, что фигура вращается то в одну, то в другую сторону. Этого эффекта можно избежать, если покрасить ребра контрастным

цветом или закрасить часть поверхностей. На рис. 7.19 показан пример изображений, полученных на основе проволочного контура.

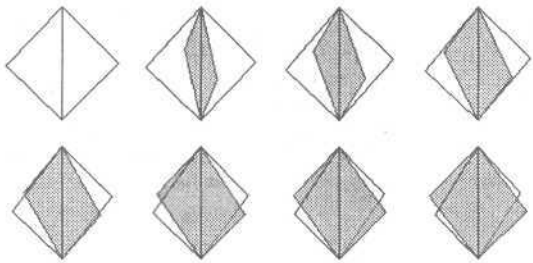


Рис. 7.19. Фазы вращения фигуры, созданной на основе контура октаэдра

ПРИМЕЧАНИЕ

Вращающийся проволочный контур можно найти на Web-странице Graphics.htm прилагаемой дискеты.

Иллюстрация функций Web-страницы

Уже давно стало традицией размещать на Web-страницах небольшие движущиеся изображения, которые прясняют назначение той или иной гиперссылки. Так, для иллюстрации доступа к списку адресов используют изображение записной книжки, в которой сами собой перелистываются страницы. Очень часто рядом с гиперссылкой, позволяющей отправить сообщение по электронной почте, помещают изображение почтового ящика. Разумеется, любой автор Web-страницы хочет (и должен!) опираться на свои, оригинальные разработки. Что касается почтового ящика, то эта задача — одна из самых простых. Создать изображение почтового ящика несложно, надо только придумать, чем его оживить. Самое естественное, хотя и не оригинальное, — движущееся изображение конверта, который падает в ящик. Для этого необходимо, работая в графическом редакторе, передвигать изображение конверта на несколько пикселов и сохранять все полученные таким образом варианты рисунка. Затем можно скомпоновать GIF-файл. Подобные ролики удобнее всего создавать, используя один большой рисунок как источник всех кадров. Основное изображение помещается в левом верхнем углу рисунка, а справа и ниже можно разместить все необходимые фрагменты и детали. Пример такой заготовки показан на рис. 7.20. На рисунке необходимо также обозначить правый нижний угол будущего кадра.

После того как одиночное изображение сформировано, необходимо уменьшить его размер до размера кадра. В редакторе Paint для запуска соответствующего диалога нажимается комбинация клавиш Ctrt+E. Затем выбирается команда Сохранить как и содержащий кадр файл записывается под оригинальным именем. В именах удобно использовать числа, если кадров много. Не рекомендуется создавать кадры при помощи операций вырезки и вставки из буфера обмена, так как в этом случае

трудно обеспечить одинаковое положение рисунков внутри разных кадров. Для работы с каждым последующим кадром необходимо вновь открыть исходный файл.

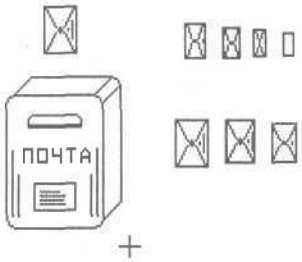


Рис. 7.20. Исходный материал для построения набора изображений

На рис. 7.21 представлены примеры изображений из файла postbox.gif, содержащего «оживленный» почтовый ящик. Для этого файла мне потребовалось создать 16 кадров размером 65x135 пикселов.

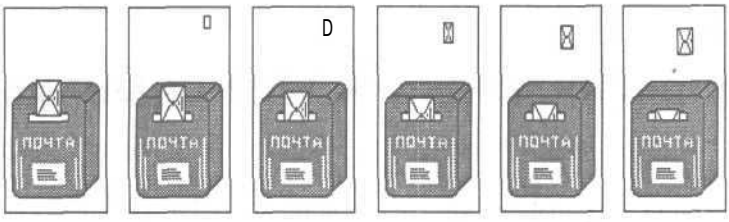


Рис. 7.21. Примеры кадров файла postbox.gif

Если такой файл необходимо использовать в гиперссылке, позволяющей послать сообщение по электронной почте, то на Web-странице должен появиться код наподобие этого:

```
<P>Вы можете отправить сообщение</p>
<ADDRESS><Ahref="mailto:Адрес@сервер.домен">
<IMG src="postbox.gif" width="65" height="135"></a></address>
```

В этом примере элементы P и ADDRESS определяют структуру документа, элемент IMG необходим для вывода изображения, а элемент A позволяет получить доступ к электронной почте.

ПРИМЕЧАНИЕ

Изображение почтового ящика можно найти на Web-странице Graphics.htm прилагаемой дискеты.

Фотоморфизм

Еще один способ быстрого получения набора кадров для движущегося изображения — операция фотоморфизма, то есть автоматического преобразования одного

изображения в другое. Изложенный в этом разделе материал основан на использовании программы PhotoMorph фирмы North Coast Software, но вы можете найти и другие программы, которые выполняют те же функции.

Идея фотоморфизма предельно проста. Вначале берутся два изображения: начальное и конечное. Как правило, это должны быть изображения каких-либо конкретных объектов. Затем пользователь, работая в специальном режиме редактирования, расставляет маркеры на обоих изображениях. Он определяет, где должна оказаться та или иная точка рисунка в конце преобразования, или, другими словами, в какую деталь конечного изображения должна быть преобразована деталь начального изображения. Всю остальную работу выполняет программа. Она не просто подменяет одно изображение другим, а деформирует их в процессе подмены. При этом у наблюдателя должно сложиться впечатление, что один предмет превращается в другой. Конечно, не всякая операция трансформации получается удачной. Для работы с подобными программами нужен навык. Кроме того, необходимо тщательно подбирать рисунки, чтобы превращение выглядело естественно.

На рис. 7.22 показано основное окно проекта. Пользователь должен открыть два файла с изображениями перед началом работы. Основным форматом графических файлов здесь считается JPG.

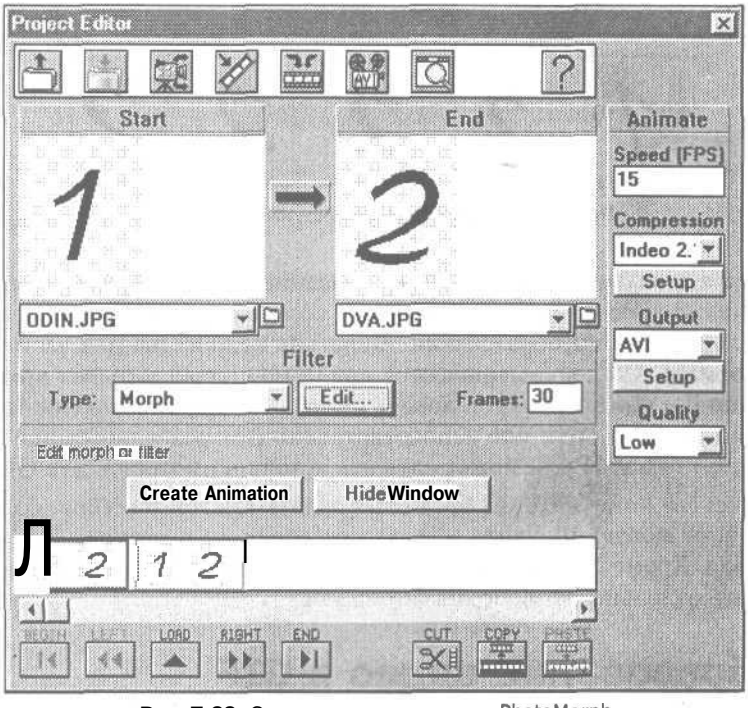


Рис. 7.22. Окно проекта программы PhotoMorph

Кнопка с большой стрелкой между изображениями позволяет перейти в режим редактирования маркеров (рис. 7.23). После того как маркеры расставлены, можно

просмотреть превращение по шагам или создать видеоролик (файл формата AVI). Для пошагового просмотра необходимо использовать кнопку с изображением увеличительного стекла в верхней части окна Project Editor. На рис. 7.24 показаны этапы преобразования единицы в двойку.

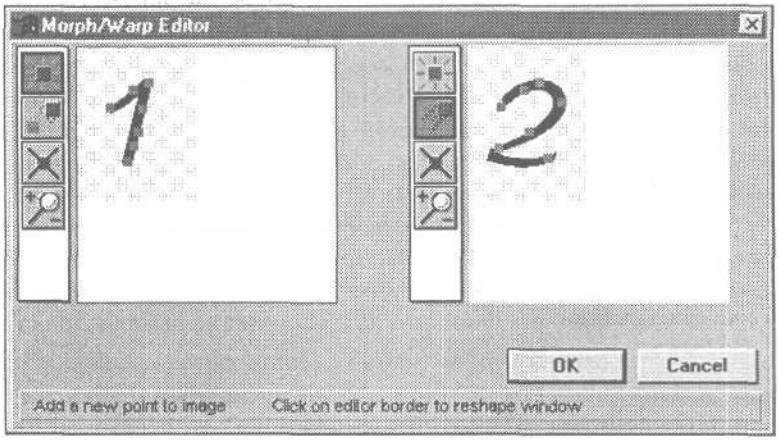


Рис. 7.23. Режим редактирования маркеров

Рис. 7.24. Пример трансформации изображения

Кнопка Create Animation открывает путь к созданию видеоролика. В данном случае операция также проводится автоматически, пользователь должен указать только основные параметры преобразования, такие, как количество или частота кадров. Ролики формата AVI просматриваются при помощи специальных программных средств, которые необходимо установить на компьютере. Видеоролик может быть использован на Web-странице непосредственно (см. раздел «Ссылки на файлы мультимедиа» главы 6) или может служить источником кадров для GIF-файла.

Если в качестве конечного изображения задать пустой рисунок, содержащий только фон, то можно получить эффект постепенного исчезновения («таяния») изображения. Кроме того, изображения с интересными визуальными эффектами могут давать отдельные фазы трансформации.

Преобразование видео в GIF

Программа Gif Construction Set позволяет преобразовать видеоролик в формате AVI в сложный (анимационный) GIF-файл. Вначале необходимо выполнить команду File ▸ Movie to GIF, а затем указать файл видеоролика. Программа активи-

зирует окно, показанное на рис. 7.25. Здесь пользователь должен выбрать способ формирования палитры GIF-файла. Хорошие результаты получаются при использовании режимов Dither to 256-colour orthogonal palette и Remap to 256-colour orthogonal palette. Флажок Loop позволяет поместить блок цикла в файл. Далее программа все делает сама. Единственная проблема, которая может возникнуть, — необходимость использования достаточно большого числа блоков изображения, если видеоролик окажется слишком длинным. Напомню, что максимально допустимое их число устанавливается в режиме Setup. Разумеется, в дальнейшем пользователь может доработать GIF-файл. Например, оставить только часть ролика или вставить другие изображения.

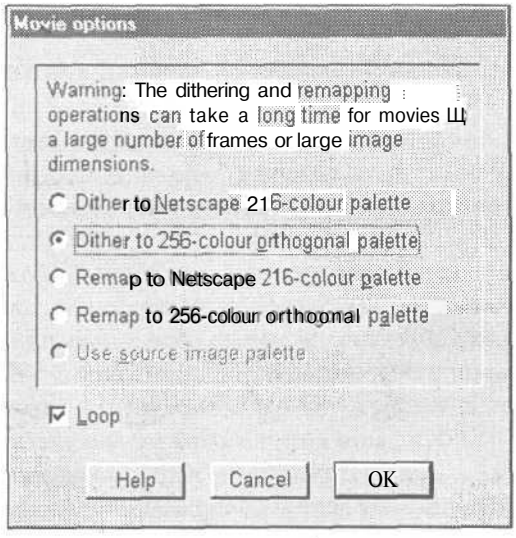


Рис. 7.25. Окно для выбора параметров преобразования видеоролика в GIF-файл

Инструменты рисования в Microsoft Office 2000

Все программы, входящие в состав пакета *MS* Office 2000, имеют единые средства рисования. Популярность пакета ставит меня перед необходимостью дать их общий обзор. MS Office имеется на многих компьютерах, а качество средств рисования заслуживает всяческой похвалы. По сути, этот инструментарий соответствует возможностям хорошего графического редактора. В первую очередь данные средства предназначены для оформления документов и развития самих приложений MS Office: создания рисунков для кнопок, деталей электронных таблиц, форм, окон и т. д. Разработчик Web-страниц может использовать такие эффекты, как объем, тень, изменение освещенности, автофигуры, поворот изображения. На рис. 7.26 показаны примеры рисунков, созданных в MS Office 2000. Основное преимущество этого подхода — быстрота их создания и легкость последующего редактирования.

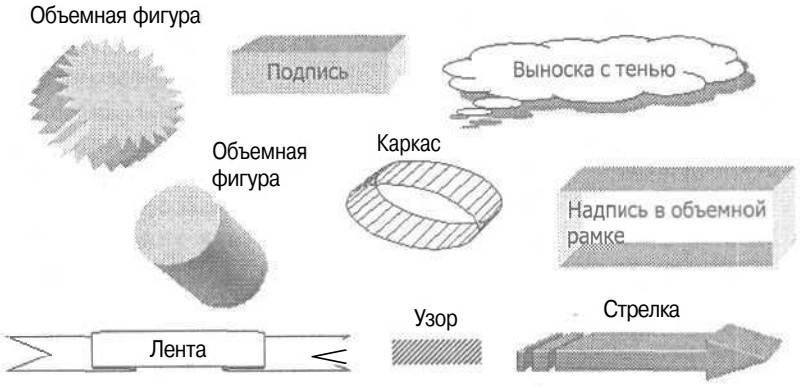


Рис. 7.26. Рисованные объекты, созданные в MS Office

В большинстве приложений MS Office (Access, Excel, Power Point, Word) существует панель Рисование, которая содержит основные инструменты для создания рисунков. Она показана на рис. 7.27, а работа с ней осуществляется следующим образом: пользователь выбирает объект из числа существующих и размещает его при помощи мыши в рабочем поле приложения. Затем он может доработать полученную фигуру: придать ей объем, создать тень, раскрасить, нанести надписи и т. д. Если создано несколько объектов, их можно объединить и выполнять групповые операции выравнивания и перемещения. Если несколько объектов расположены так, что перекрывают друг друга, то можно определить, какой из них будет находиться на переднем плане, а какие будут видны только частично. Размер объекта можно изменить в любое время. Для этого необходимо щелкнуть на нем мышью и воспользоваться появившимися вокруг изображения маркерами. На рис. 7.28 показан набор команд меню Действия, которые позволяют выполнять операции перемещения готовых объектов.

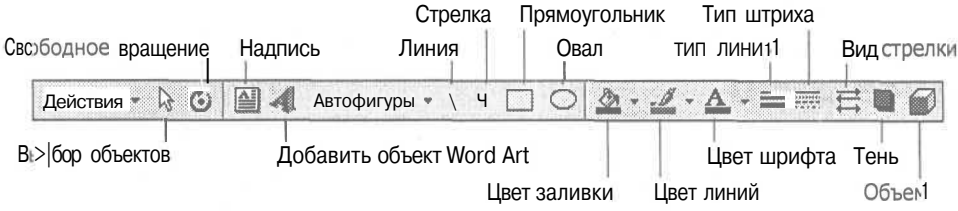


Рис. 7.27. Панель инструментов Рисование

Разработчики MS Office пошли по пути создания и предоставления пользователю большого количества спроектированных и готовых для использования графических объектов, которые могут пригодиться в самых разных ситуациях. Доступ к этим объектам осуществляется через меню Автофигуры, показанное на рис. 7.29. Рисунок дает представление о том, какую форму может принимать автофигура.

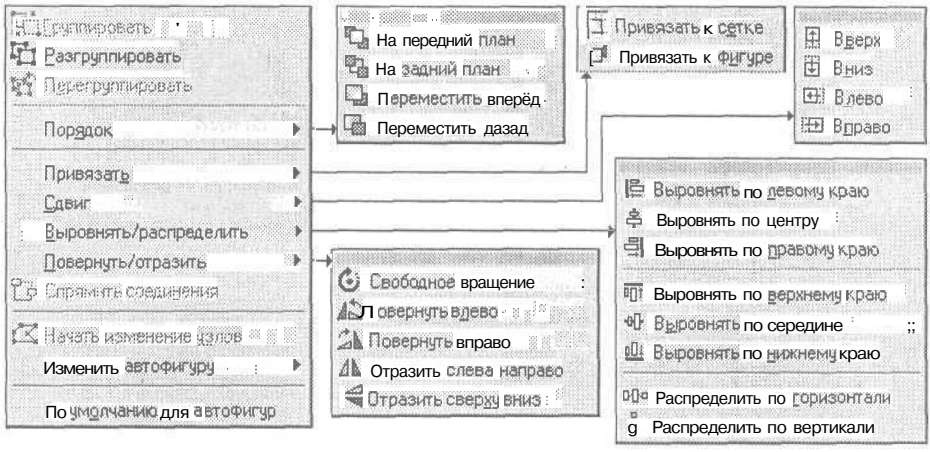


Рис. 7.28. Команды меню Действия

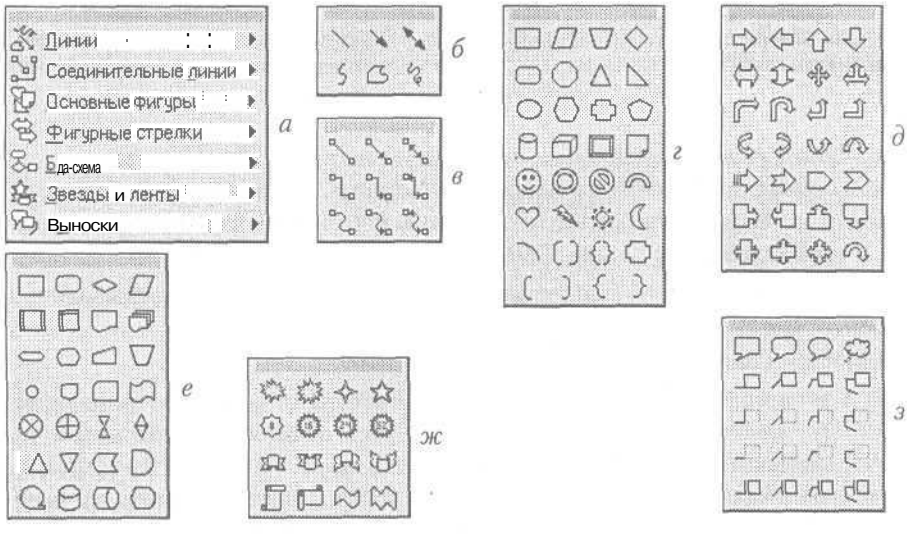


Рис. 7.29. Автофигуры: *а* — список для выбора категорий; *б* — линии; *в* — соединительные линии; *г* — основные фигуры; *д* — фигурные стрелки; *е* — элементы блок-схемы; *ж* — звезды и ленты; *з* — выноски

Любой рисованный объект будет хорошо смотреться только после того, как будет снабжен тенью или станет объемным. В документах MS Office не существует разделения фигур на плоские и объемные. Объемность, как и другие свойства, придаются объекту при помощи соответствующих инструментов. На панели Рисование имеются кнопки Тень и Объем (см. рис. 7.27), которые позволяют не только усовершенствовать рисунок, но и подключить дополнительные средства рисования. На рис. 7.30 показаны панели, с помощью которых производится выбор параметров графических объектов.

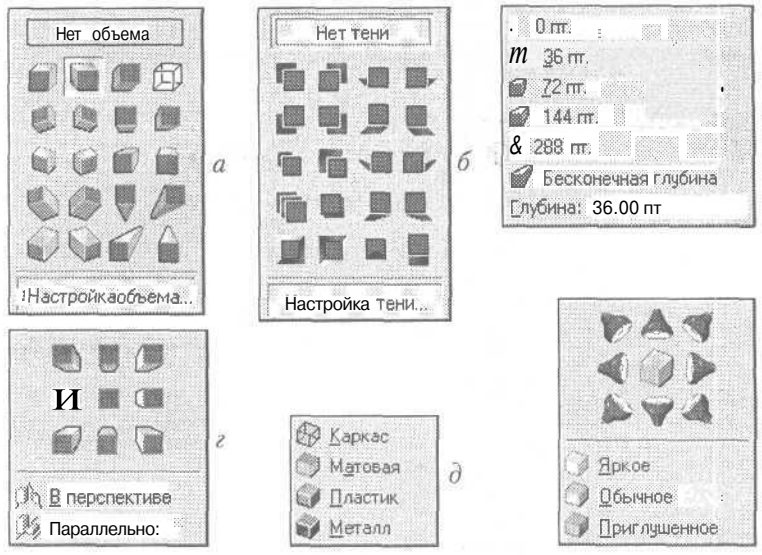


Рис. 7.30. Панели для настройки объема и тени: *а*— варианты проекций; *б*— варианты теней; *в*— параметры глубины изображения; *г* — способы задания перспективы; *д*— варианты фактуры; *е*— панель для выбора освещения

Кнопка Объем раскрывает панель, показанную на рис. 7.30, а. Последняя, в свою очередь, позволяет вывести на экран панель инструментов Настройка объема. Используя ее кнопки, можно вращать объемную фигуру и работать с другими панелями, представленными на рис. 7.30, в-е.

Точно так же обстоит дело с кнопкой Тень. Она раскрывает панель, показанную на рис. 7.30, б. Щелкнув на расположенной на ней кнопке Настройка тени, можно активизировать одноименную панель инструментов.

Объем и тень являются альтернативными свойствами объекта. Придав фигуре объемность, нельзя заставить ее отбрасывать тень, и наоборот. Зато можно очень легко переключаться между этими двумя режимами. На панелях инструментов Настройка объема и Настройка тени находятся кнопки для включения и отключения соответствующих свойств.

Когда в документ внедряется рисунок, расположенный в отдельном файле, на экран автоматически выводится панель инструментов Настройка изображения (рис. 7.31). Далеко не в каждом графическом редакторе существуют инструменты, расположенные на этой панели. А в приложениях MS Office они есть. С их помощью рисунок настраивается примерно так, как мы настраиваем изображение нашего телевизора или монитора. Но все же, какие параметры нужно иметь возможность менять, чтобы адаптировать картинку к текущим условиям просмотра? Эти параметры хорошо известны: яркость, контрастность, цветовая палитра. Все их позволяет регулировать упомянутая панель. Кроме того, она дает возможность сделать прозрачным определенный цвет рисунка, так что через него будет просвечивать фон документа.

Рис. 7.31. Панель инструментов Настройка изображения

Каждому объекту присущ определенный набор свойств, которые могут изменяться в процессе форматирования. Щелчок на объекте правой кнопкой мыши приводит к раскрытию контекстного меню, связанного с ним. В этом меню находится список операций, которые можно выполнить для объекта данного типа.

Что из этих средств можно использовать при создании изображений для Web-страниц? В первую очередь эффект объема, так как он позволяет создавать красивые полутоновые изображения. Можно разместить текст в рисунке, соответствующим образом отформатировав символы. Для текста можно выбрать размер, тип и цвет шрифта, использовать специальные эффекты, такие, как поворот, верхний и нижний индексы и многие другие. Текст может быть отформатирован и при помощи художественных стилей (см. ниже раздел «WordArt»). При создании анимационных изображений доступны инструменты для поворота графического объекта, различные варианты освещенности и т. д. Маркеры самих графических объектов позволяют изменять размеры последних или вращать их в плоскости экрана. Таким образом, в распоряжении разработчика имеются самые разные способы получения кадров для сложных GIF-файлов.

WordArt

В качестве примера, иллюстрирующего возможности приложений пакета MS Office 2000 в нелегком деле разработки Web-страниц, рассмотрим метод получения движущихся изображений при помощи программы для создания фигурного текста WordArt. Она имеет панель инструментов, показанную на рис. 7.32.

Представленные на этой панели кнопки позволяют выполнять различные операции форматирования столь необычного гибрида рисунка и текста.

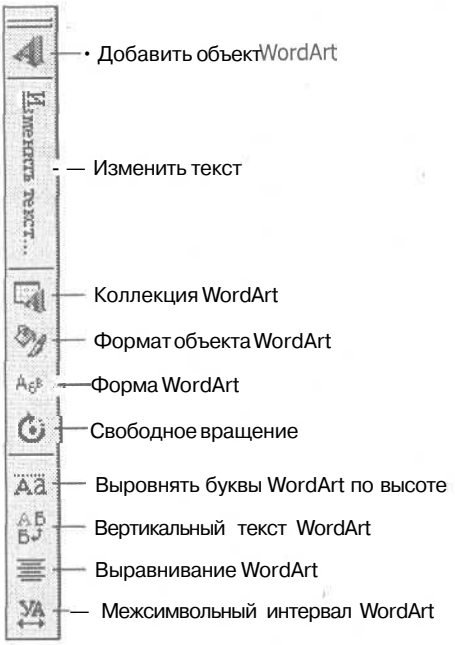


Рис. 7.32. Панель инструментов WordArt

Работа с программой WordArt начинается со щелчка на кнопке Добавить объект WordArt на панели инструментов Рисование. Открывшееся окно диалога показано на рис. 7.33.

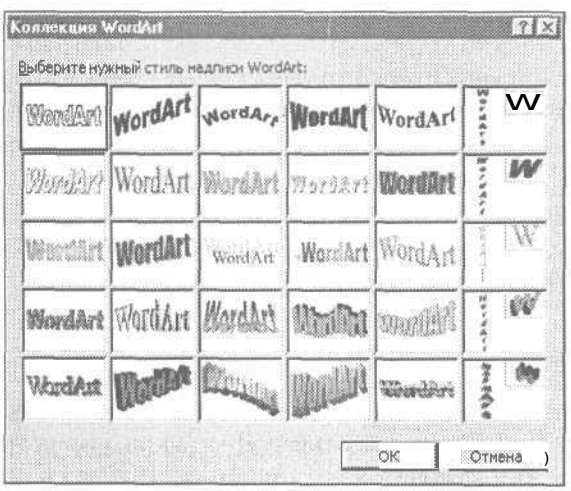


Рис. 7.33. Окно для выбора стиля надписи

Поскольку движение изображения на Web-странице в большинстве случаев должно быть безостановочным, целесообразно задать вращение картинки, хотя WordArt позволяет реализовать и другие виды движения. Щелкнув дважды на значке нужного стиля в окне Коллекция WordArt, мы переходим к окну, показанному на рис. 7.34. В нем надо напечатать текст и выбрать размер шрифта. Высота букв не должна быть слишком большой, чтобы не затруднять дальнейшее форматирование.

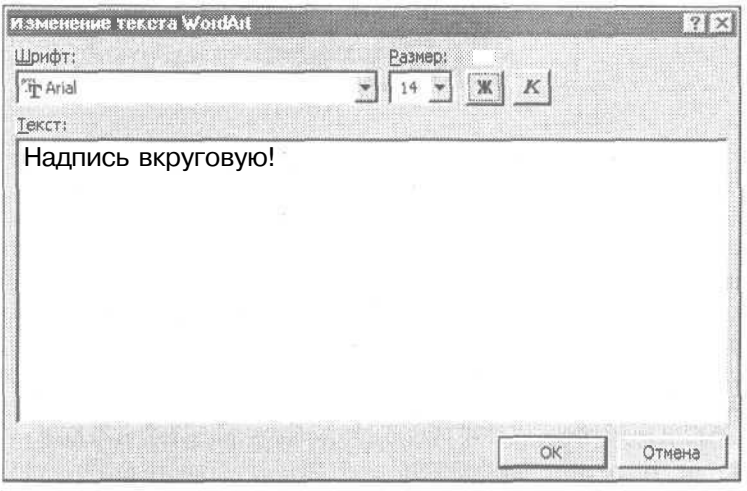


Рис. 7.34. Окно для ввода текста

После закрытия последнего окна объект размещается в документе, а на экран автоматически выводится панель инструментов WordArt. В ней необходимо щелкнуть на кнопке Форма WordArt. Нашим взорам предстанет следующая панель, здесь необходимо выбрать форму текста (рис. 7.35).

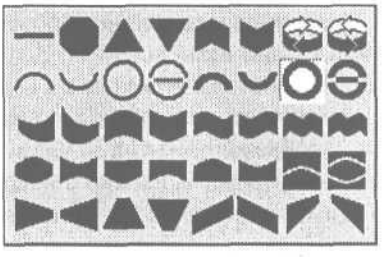


Рис. 7.35. Панель для выбора формы текста

Нам требуется использовать форму кольца. После этого изображение можно привести к виду, показанному на рис. 7.36. Чтобы создать пустое пространство между началом и концом фразы, при вводе текста надо поставить несколько пробелов в начале строки. Кнопка Формат объекта WordArt открывает доступ к окну диалога, которое позволит, в частности, выбрать цвет букв. Окончательно форму необходимо подобрать с помощью окружающих объект маркеров.

Рис. **7.36.** Объект WordArt и его маркеры

Теперь можно приступать к созданию набора файлов изображений. Создайте в графическом редакторе новый файл, который будет выполнять роль заготовки для будущих кадров. Проведите на рисунке вспомогательные линии так, чтобы по ним можно было точно установить изображение кадра. Сохраните файл на диске Переключитесь на приложение, в котором создан объект WordArt, и снимите с объекта выделение. Маркеры должны исчезнуть. Объект WordArt нельзя скопировать в буфер обмена в виде рисунка, поэтому нажмите клавишу Print Screen Вставьте содержимое буфера обмена (копию экрана) в пустой файл в графическом редакторе. Вырежьте изображение объекта и вставьте его в файл-заготовку кадра. После позиционирования удалите вспомогательные линии. Сохраните полученный файл под новым именем. Вернитесь в приложение MS Office. Щелкните на кнопке Формат объекта WordArt и в открывшемся окне диалога перейдите на вкладку Размер (рис. 7.37). На ней можно проконтролировать и точно установить размеры изображения и, что важно в данном случае, задать угол поворота изображения. Измените угол поворота, задав величину, кратную 360. Создайте второй кадр и т. д.

После того как все кадры сформированы, можно собрать их воедино в программе Gif Construction Set уже известным нам способом. Вращение подобного изображения не должно быть быстрым, так как необходимо успевать прочесть текст. Но с другой стороны, чем большую задержку вывода кадра вы укажете, тем меньше должен быть угол поворота изображения за один кадр. В противном случае картинка будет двигаться рывками. Время задержки в нашем случае можно выбрать в пределах 0,1-0,2 секунды, а угол поворота 10-15 градусов.

ПРИМЕЧАНИЕ

Изображение, подученное при помощи WordArt, можно увидеть на Web-странице Graphics.htm прилагаемой дискеты.

В WordArt допустимо использовать и другие приемы форматирования. Например, маркеры, окружающие объект, позволяют менять его размеры, пропорции и выполнять свободное вращение в плоскости экрана. На рис. 7.38 показаны различные варианты форматирования одной и той же надписи.

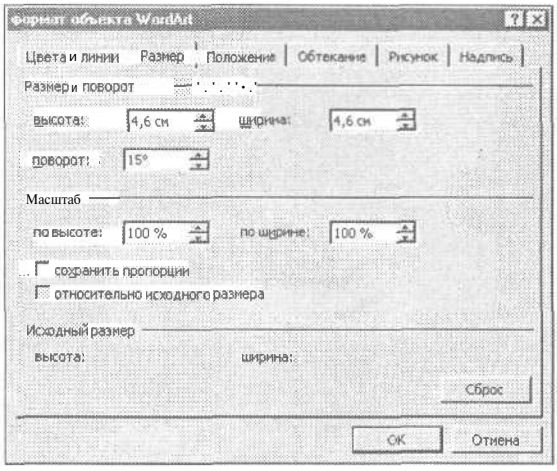


Рис. 7.37. Окно для форматирования объекта WordArt

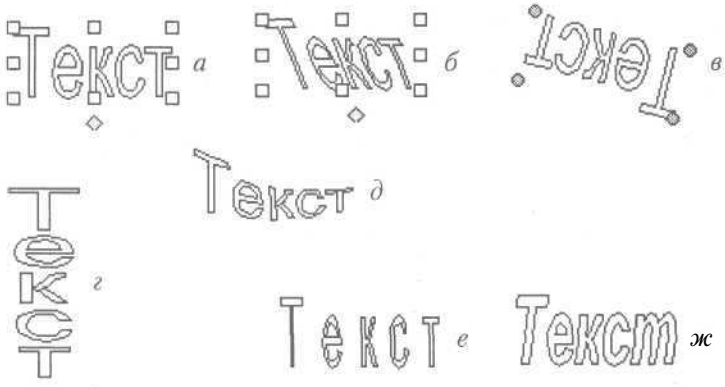


Рис. 7.38. Варианты форматирования объекта WordArt: *а* — исходная надпись; *б*— наклон при помощи маркера-ромбика; *в*— поворот в плоскости экрана; *г*— вертикальный текст; *д*— изменение формы надписи; *е*— изменение межсимвольного интервала; *ж*— выбор начертания полужирным курсивом

Графический редактор MS Image Composer

Графический редактор MS Image Composer (рис. 7.39) обычно поставляется вместе с программой для создания Web-страниц MS Front Page 2000.

Этот редактор позволяет работать со многими графическими форматами и достаточно удобен для работы с рисунками для Интернета. Основные инструменты редактора находятся на главной панели и на вертикальной панели слева. Программа предназначена не только для рисования, но и для работы с готовыми изображениями, причем сразу с несколькими. Поэтому в начале работы целесообразно задать размеры будущей *композиции* командой File ▸ Composition Setup. Затем можно открыть несколько файлов с рисунками.

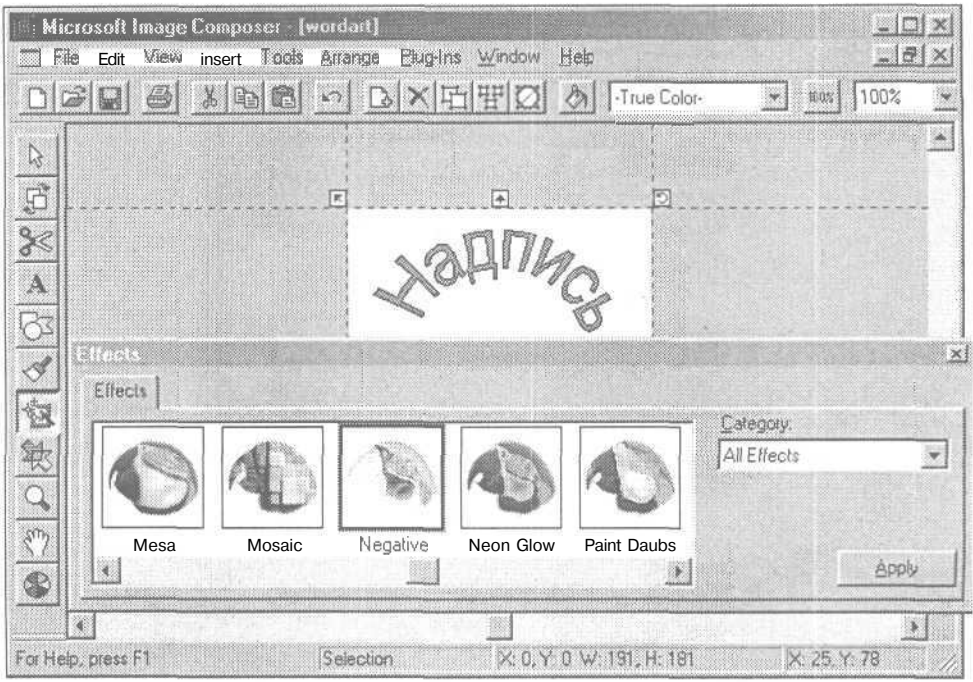


Рис. 7.39. Графический редактор Image Composer

Insert Image File. Эта кнопка позволяет добавить новое изображение в ком позицию. Каждый такой объект представляется в программе в виде спрайта его можно перемещать мышью (даже за границы основного рисунка), накладывать на другие объекты.

Duplicate. Кнопка, которая позволяет быстро создать копию спрайта. На пер вый взгляд, это может показаться лишним, так как рисунок можно вставить в композицию еще раз. Но логика работы с программой подсказывает, что про цесс создания копии объекта должен быть очень простым. В программе суще ствует много инструментов, которые позволяют преобразовать спрайт. Вариан тов очень много, поэтому и могут понадобиться новые заготовки.

Самый простой способ изменения рисунка — использование его рамки с маркера ми (рис. 7.40). Спрайт можно масштабировать по вертикали или горизонтали, также вращать.

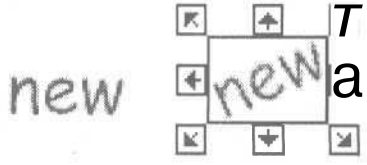


Рис. 7.40. Рамка с маркерами позволяет легко изменить рисунок

Обычно в каждый момент времени выбран один спрайт. Щелкая на объектах при нажатой клавише Ctrl, можно выделить несколько объектов одновременно. Есть даже специальная кнопка Select All (выбрать все). В контекстом меню спрайта есть команда Flatten Selection. Она позволяет преобразовать выбранные спрайты в один спрайт. Если выбраны все объекты, то такая команда конвертирует все спрайты в обычные элементы рисунка, и оставляет один спрайт — саму композицию.

Теперь рассмотрим инструменты вертикальной панели (см. рис. 7.39). Большая часть кнопок открывает окна диалога, позволяющие выполнять различные операции над рисунком. После установки всех параметров, надо щелкнуть на кнопке Apply (применить) и закрыть окно, чтобы увидеть рисунок. Если результат вас не удовлетворяет, выберите команду Edit ▸ Undo. Помните, что отменить можно только результаты последней операции.

Selection. Когда нажата эта кнопка, можно работать со спрайтами: выбирать их или перемещать с места на место.

Arrange. Работа с группой спрайтов. В открывающемся окне диалога есть множество кнопок для их взаимного выравнивания, перемещения, вращения и т. д.

Cutout. Выбор различных способов вырезки фрагментов рисунка. Удобна, например, кнопка Stencil (трафарет). Она позволяет быстро создать трафарет или маску спрайта (рис. 7.41).

Text. Создание надписи. Это стандартный инструмент для всех графических редакторов: выбирается шрифт, размер букв, цвет, начертание.

Shapes. Создание геометрических фигур: прямоугольников, эллипсов, окружностей и т. д. Все они вставляются в композицию в виде спрайтов.

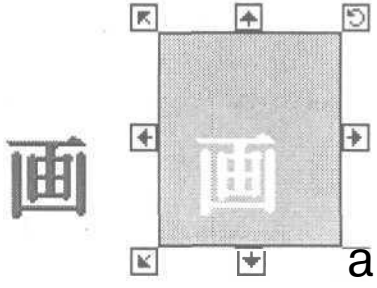


Рис. 7.41. Создание трафарета

Paint. Набор инструментов обычного графического редактора: карандаш, кисть, аэрограф, «палец», ластик и масса других занимательных мелочей. Рисовать можно не закрывая окно диалога, если виден выделенный спрайт. Здесь проявляется удобство редактора: рисование выполняется только в пределах выбранного объекта.

Effects. Набор стандартных визуальных эффектов. В программе Adobe Photoshop они еще называются фильтрами. Каждая операция позволяет преобразовать рисунок по определенному алгоритму. Есть, например, эффекты

размытия, имитации материала, неонового света, изменения текстуры. На рис. 7.39 открыто данное окно диалога.

Effects. Этим диалогом надо пользоваться, выделив несколько перекрывающихся спрайтов. С помощью картинок-образцов можно задать вид перекрытия: полупрозрачное, при помощи маски или другое.

Zoom. Изменение масштаба композиции.

Pan. Перемещение композиции в пределах окна просмотра.

Color Tuning. Цветовая палитра.

Microsoft GIF Animator

Эта программа может существовать как отдельный продукт или устанавливаться вместе с MS Image Composer. Окно программы показано на рис. 7.42. Работа этой программы несколько отличается от работы программы Gif Construction Set. Чтобы создать новый анимационный файл, надо открыть графический файл одного из доступных форматов: видеоролик типа AVI или обычный файл GIF.

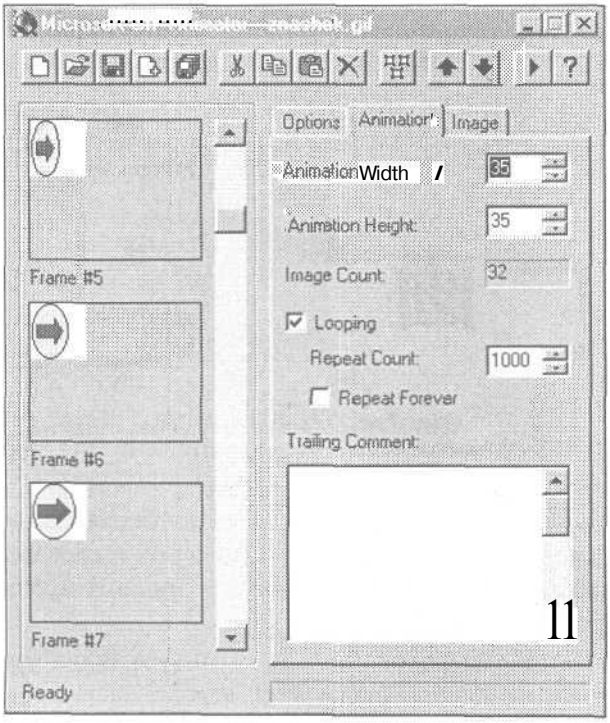


Рис. 7.42. Microsoft GIF Animator

Если открыт файл AVI, вы сразу узнаете, из какого количества кадров он состоит. Преобразовать видеоролик в анимационный файл формата GIF можно, щелкнув на кнопке Save. Новый файл будет создан автоматически с прежним именем.

Если у вас есть набор отдельных кадров в формате GIF, и вы открыли первый, то остальные добавляются щелчком на кнопке Insert. Затем на вкладках можно выбрать параметры движущегося изображения (рис. 7.42). С помощью флажков Looping (зацикливание) и Repeat Forever (повторять постоянно) можно установить число повторений ролика.

На вкладке Image можно посмотреть размер изображения — это пригодится для задания атрибутов HTML. Затем для каждого кадра необходимо установить параметры воспроизведения. Например, можно выбрать прозрачный цвет: щелкнуть на квадратике Transparent Color (прозрачный цвет), чтобы раскрыть палитру. Параметр Duration (продолжительность) определяет время воспроизведения определенного кадра, так как для разных кадров эта величина может отличаться (если надо создать визуальный эффект). Кроме того, в группе Undraw Method надо выбрать способ замены изображения. Значение Leave позволяет оставить старый кадр и наложить сверху следующий. Значение Restore Background обеспечивает обычную смену кадров.

Для просмотра ролика надо щелкнуть на кнопке Preview. Он будет демонстрироваться в специальном окне, в котором его можно прокрутить еще раз или просмотреть отдельные кадры (рис. 7.43).

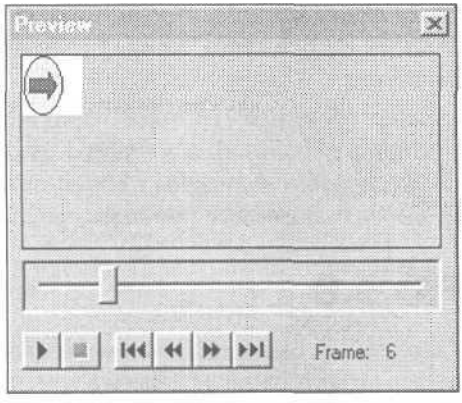


Рис. 7.43. Окно просмотра анимационного файла

Глава 8

Редакторы гипертекста

Как только мировой компьютерной общественности стало понятно, что отныне придется постоянно работать с гипертекстом, многие фирмы начали разработку специализированных редакторов, позволяющих создавать HTML-документы. Эти программы сразу же завоевали популярность, несмотря на то, что гипертекстовую структуру можно создавать в простейшем текстовом редакторе для MS-DOS.

Редактор гипертекста необходим, в первую очередь, для начинающих пользователей, так как позволяет обойтись (на первых порах) поверхностными знаниями о структуре гипертекстового документа и синтаксисе HTML. Поскольку элементы гипертекста создаются в режиме диалога, пользователь может изучать язык параллельно с созданием Web-страницы. Вдобавок, редакторы обладают способностью проверять правильность написания кода HTML. Бывают случаи, когда программа выдает сообщение об ошибке, но при этом не идентифицирует ошибку. Тем не менее, автор Web-страницы может понять, что с разметкой не все ладно, и не станет использовать спорный фрагмент кода. Просматривать страницу броузером в этом случае не имеет смысла: скорее всего, он прорисует страницу, как будто никаких проблем не существует.

Я буду опускать в описании программ сведения о традиционных командах: для открытия и закрытия файлов, использования буфера обмена, поиска и т. д. Любой пользователь Windows, поработав хотя бы с одной из популярных программ прекрасно знает, как применять подобные команды.

HoTMetaL PRO 5.0

Редактор гипертекста HoTMetaL был одним из первых программных продуктов такого рода. Основная идея редактора заключалась в том, чтобы красиво прорисовать на экране изображения тегов. Здесь сказались преимущества графического режима Windows перед текстовым режимом MS-DOS. В случае Windows гипертекст в своем естественном виде лучше читается и редактируется. Одновременно фирмы-разработчики развивали в своих программах возможность анализа синтаксиса HTML, что помогало пользователям избежать некоторых ошибок, таких, например, как неправильное вложение элементов.

Фирма SoftQuad разработала несколько версий этой программы. На рис. 8.1 показано окно редактора HoTMetaL PRO версии 5.0. Доступ к необходимой информации осуществляется при помощи панелей. Например, слева помещена панель с

деревом папок, что позволяет быстро находить нужные файлы. На панели справа создана заготовка Web-страницы. Мы видим, что теги прорисованы особым образом и хорошо заметны. На панелях инструментов размещено большое количество кнопок. Следовательно, разработчики продукта предполагают, что пользователь должен активно использовать их во время создания страниц. Разберемся, для чего они предназначены.

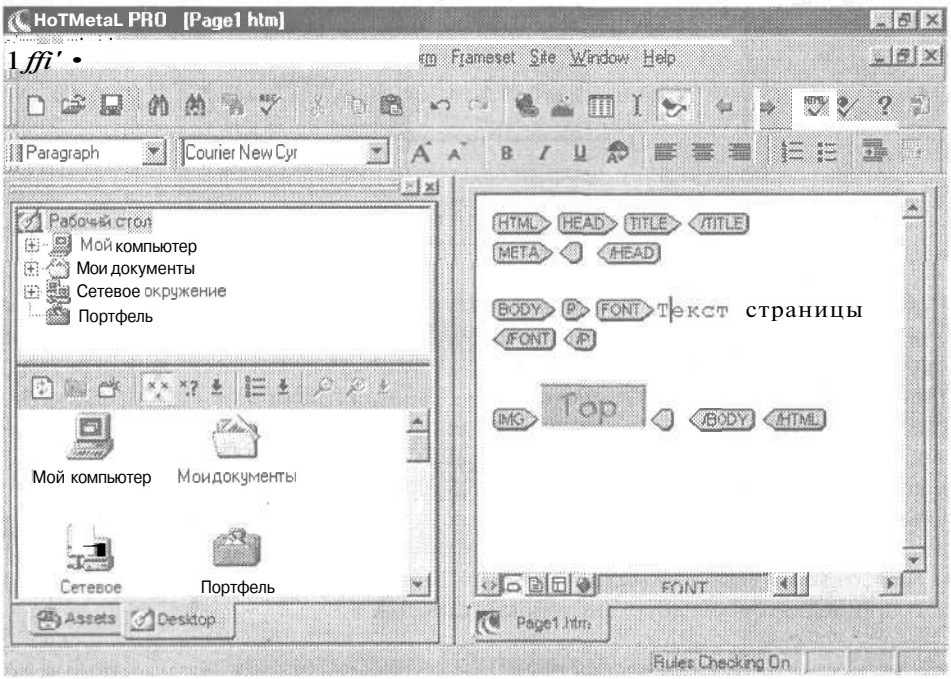


Рис. 8.1. Редактор гипертекста HoTMetaL PRO 5.0

Recource Manager. Если вы хотите, чтобы для просмотра гипертекста было больше места, отключите левую панель этой кнопкой.

Previouus Document и Next Document (предыдущий документ и следующий документ). Переход к окнам открытых в редакторе документов. В отличие от броузеров, которые тоже имеют такие кнопки, в данном случае не осуществляется переход по ряду просмотренных страниц.

Spelling (проверка орфографии). После щелчка на этой кнопке открывается окно, показанное на рис. 8.2. Проверка начинается от места расположения курсора и выполняется для английских слов. В данном случае встретилось слово «toolbar», которого не оказалось в словаре. Программа предлагает список похожих слов для замены. В окне имеются кнопки, позволяющие добавить новое слово в словарь (Add to Dictionary), заменить слово (Replace), перейти к следующему слову (Next), установить режим пропуска для данного слова (Set Restriction).

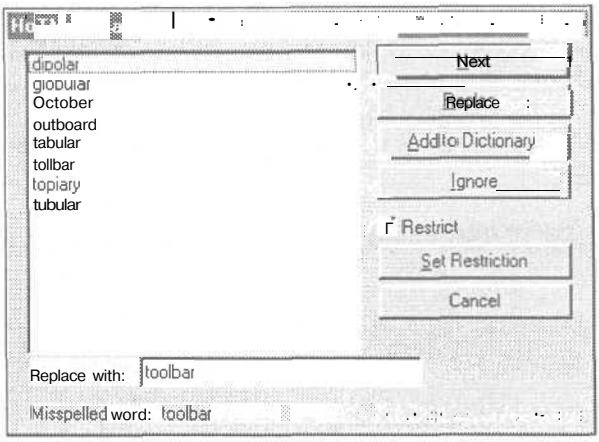


Рис. 8.2. Окно проверки орфографии

Чтобы воспользоваться встроенным тезаурусом, необходимо выделить одно и; слов документа, а затем выбрать команду Tools ▸ Thesaurus. Окно, которое должно открываться вслед за перечисленными действиями, показано на рис. 8.3. В данном случае было выделено слово «assign». Работая с этим окном, мы тоже можем заменить слово, подобрав подходящий синоним. Кроме этого, список позволяет просмотреть и выбрать (кроме синонимов) антонимы (Antonyms), близкие по смыслу слова (Related Words) и слова, отдаленные по смыслу (Contrasted Words). Это очень полезный инструмент для создания страниц на английском языке. Кроме того, мне кажется, что такая сервисная возможность выходит за рамки подготовки гипертекста. Тезаурус можно использовать и при изучении английского языка, и при подготовке любых английских текстов.

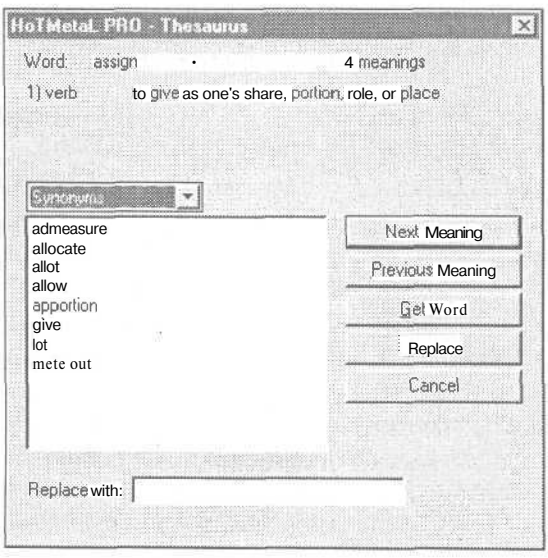


Рис. 8.3. Окно тезауруса

HoTMetaL PRO можно использовать и как обычный текстовый редактор. Создать текстовый документ можно двумя путями. Во-первых, можно обойтись минимальным количеством элементов, позволяющих набирать произвольный текст, — самый простой, но и самый малоэффективный способ. В этом случае редактор имеет примерно такие же возможности, как Блокнот в Windows. Во-вторых, можно использовать знание HTML и задействовать все необходимые элементы. То есть текстовый документ должен создаваться как обычная Web-страница, но с перспективой вывода на печать. Вы уже знаете, что возможности для форматирования HTML-документа велики: выбор шрифтов, использование таблиц, рисунков и т. д. Чтобы распечатать только полезную информацию, необходимо изменить режим просмотра. Форматирование страниц выполняется автоматически, применительно к каждому устройству вывода.

Я предвижу возможные возражения читателя: зачем использовать программу не по назначению, когда существует великое множество текстовых редакторов? Действительно, современные редакторы позволяют выполнять некоторые очень удобные специальные операции, необходимые для разработки текстовых документов: создание оглавления, индекса, распечатка конвертов и т. д. В этом смысле они незаменимы. Сравнение гипертекстового и «обычного» редактора надо проводить в другой плоскости.

Возьмем, к примеру, популярный редактор Word. Все, кто с ним работал, прекрасно знают о форматировании текста при помощи стилей. Если вы захотите понять, что представляет собой тот или иной стиль, вам надо будет раскрыть несколько окон с описанием стиля и внимательно изучить их содержимое. Параметров много: и тех, которые вы используете, и тех, которые вам безразличны. В самом документе подобная информация скрыта от пользователя. Все, кто работал с этим редактором, знают о его «своеволии»: часто программа сама, не спрашивая разрешения, проводит ряд операций по форматированию текста, и пользователь не всегда может отследить эти «правки» и понять причины их возникновения. Или, наоборот, текст не форматируется там, где это необходимо. Возникают сложности и с переносом документов на другой компьютер.

Гипертекстовый документ создается совершенно по другому принципу. Здесь тоже можно применить понятие стиля: это совместное использование элементов форматирования FONT, B, I, U и им подобных. В данном случае пользователь может увидеть все параметры оформления и область их действия прямо в документе. Изменить их может только он сам, путем редактирования *текста* документа. Дает ли это какое-нибудь преимущество? Думаю, что в некоторых случаях работать с документом в формате гипертекста намного удобнее и легче, чем с таким же документом в формате традиционного редактора для Windows.

HTML Source (источник HTML). Любой редактор гипертекста должен иметь два основных режима просмотра документа: показ всех данных, включая теги, и показ только той информации, которую увидит клиент. Данная кнопка позволяет редактировать гипертекст в его традиционном виде.

Tags On. В этом режиме тоже видны теги, но они показаны в виде красивых фигур (см. рис. 8.1).

WYSIWYG view. Эта аббревиатура означает: «что видишь, то и получаешь». Это стандартный режим редактирования документа в современном редакторе.

WYSIWYG Frames view. Редактирование документа с фреймами.

Browse (просмотр). Открытие документа в броузере, используемом по умолчанию.

Insert Element (вставить элемент). Кнопка раскрывает окно, показанное на рис. 8.4. Оно позволяет выбрать и вставить элемент в Web-страницу. Особенность режима заключается в том, что программа отбирает для показа в окне только те элементы, которые позволительно вставить внутри текущего элемента.

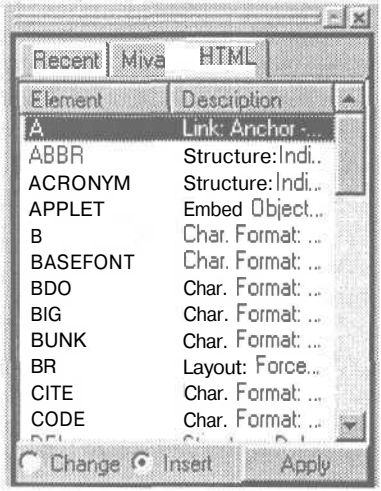


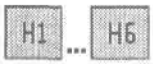
Рис. 8.4. Окно для вставки нового элемента

Insert Element Window (вставить окно элементов). Эта кнопка открывает или закрывает окно, показанное на рис. 8.4.

Remove Tags (удалить теги). Удаление начального и конечного тегов текущего элемента. Курсор должен быть установлен внутри элемента. Содержимое элемента (например, текст) удалено не будет. Элементы можно удалять двумя способами. Если выделить оба тега (со всем содержимым) и нажать клавишу Del, то элемент будет удален целиком. Если поставить курсор после начального тега и нажать клавишу Backspace, будут удалены только теги, а содержимое останется в документе.

Check HTML (проверить HTML). Программа тестирует гипертекстовый документ и сообщает о найденных ошибках в его разметке.

Check Accessibility (проверить доступность). Программа проверяет документ на предмет доступности просмотра для всех пользователей и выдает рекомендации. Например, если броузер по какой-то причине не может воспроизвести изображение, пользователь должен иметь возможность воспользоваться информацией атрибута alt. Иными словами, разработчику надо позаботиться о том, чтобы такие атрибуты существовали и содержали полезный текст.



Insert Heading (вставить заголовок). Вставка элемента заголовка H1...H6. На рис. 8.5 показан пример заголовка второго уровня и контекстное меню этого элемента (элемент FONT полезно добавлять там, где нужен русский текст). Меню позволяет использовать буфер обмена для копирования, вырезки и вставки элемента. Кроме того, команда Insert Element (вставить элемент) открывает окно для вставки нового элемента внутри текущего (см. рис. 8.4). Команда Select Element (выбрать элемент) позволяет выделить текущий элемент. Команда Element Attributes (атрибуты элемента) обеспечивает определение и редактирование атрибутов текущего элемента в режиме диалога. Окно для работы с атрибутами показано на рис. 8.6. Разумеется, для каждого элемента доступен свой набор атрибутов.

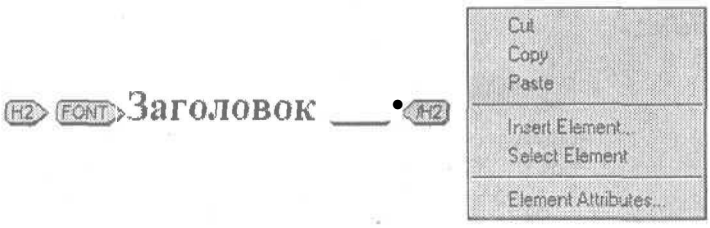


Рис. 8.5, Элемент заголовка и связанное с ним контекстное меню

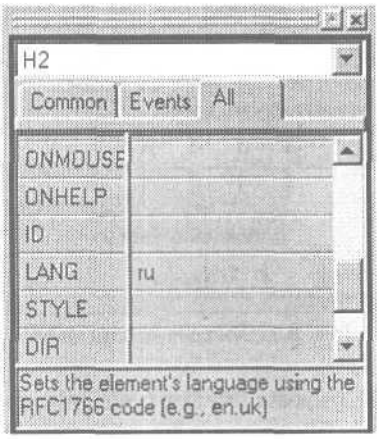


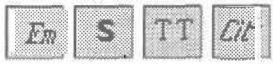
Рис. 8.6. Окно для задания атрибутов элемента H2

Attribute Inspector. Эта кнопка открывает и закрывает окно атрибутов.

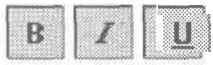
Insert Paragraph (вставить абзац). Вставка элемента абзаца P.

Insert Line Break (вставить разрыв строки). Вставка элемента принудительного разрыва строки BR.

Insert Horisontal Rule (вставить горизонтальную линию). Вставка элемента горизонтальной линии HR.



Emphasis, Strong, TeleType, Citation (выразительность, усиление, телетайп, цитата). Инструменты для создания элементов содержания. Эти элементы рассмотрены в разделе «Элементы содержания» главы 3. Рисунки на кнопках дают представление о том, как будет выглядеть текст внутри элемента.



Italic, Bold, Underline (курсив, полужирный, подчеркнутый). Ввод в документ элементов форматирования текста (см. раздел «Форматирование текста» главы 3).

Text Color (цвет текста). Эта кнопка позволяет создать новый элемент FONT в месте расположения курсора. После создания элемента открывается окно с палитрой, показанное на рис. 8.7. Когда цвет выбран, для элемента определяется атрибут цвета color.

Рис. 8.7. Окно для выбора цвета

Increase Text Size, Decrease Text Size (увеличить размер шрифта, уменьшить размер шрифта). Так же, как и предыдущий, этот инструмент создает новый элемент FONT, в котором устанавливается новое значение атрибута size.

Align Left, Align Center, Align Right (выравнивание влево, выравнивание по центру, выравнивание вправо). Кнопки, хорошо знакомые всем, кто работал с текстовыми редакторами. В данном случае они позволяют установить соответствующее значение атрибута align.

Indent (отступ). Создание отступа слева для выбранного элемента.

Special Characters (спецсимволы). Открытие палитры (рис. 8.8) для ввода спецсимволов. В данном случае используются буквы различных алфавитов. Спецсимволы рассматриваются в разделе «Использование спецсимволов» главы 2.

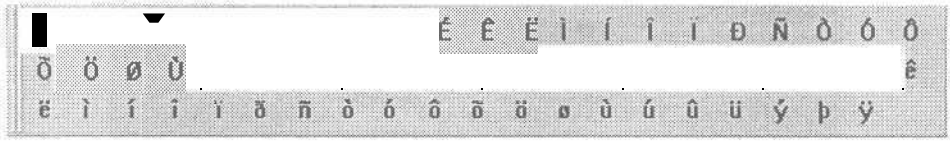


Рис. 8.8. Палитра для ввода спецсимволов (в данном случае букв)

С помощью команды Insert ▸ Symbols можно открыть палитру, показанную на рис. 8.9. Она тоже позволяет вставлять спецсимволы.

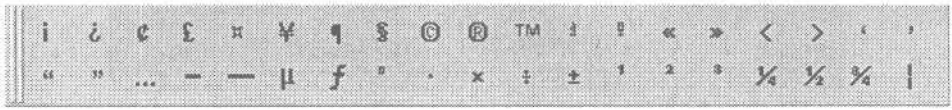


Рис. 8.9. Палитра для ввода спецсимволов

Billeted List (ненумерованный список). Создание заготовки для ненумерованного списка. Первоначально создаются два элемента: UL и LI. Если курсор находится внутри элемента LI, нажатие клавиши Enter приводит к созданию нового пункта списка. Если курсор находится внутри элемента UL, то нажав клавишу Enter, можно создать еще один список.

Numbered List (нумерованный список). Создание заготовки для нумерованного списка.

Insert Definition List (вставить список с определениями). Создание заготовки для списка с определениями, то есть элементов DL и DT.

Insert Definition Type (вставить тип определения). Создание элемента DT для списка с определениями.

Insert Definition Data (вставить данные определения). Создание элемента DD для списка с определениями.

Insert Preformatted (вставить заранее отформатированный текст). Создание элемента PRE для выделения текста, который не должен форматироваться броузером.

Block Quote (цитата). Создание элемента BLOCKQUOTE для форматирования цитаты.

Address (адрес). Создание элемента ADDRESS для форматирования адреса.

Insert Comment (вставить комментарий). Создание элемента COMMENT для ввода комментария.

Insert Image (вставить изображение). Создание элемента IMG для размещения ссылки на графический файл. Определение атрибутов изображения происходит с помощью окна диалога, показанного на рис. 8.10.

Если адрес графического файла хорошо известен пользователю, то он может сразу ввести его в поле Image File. Другой способ — открыть окно для определения URL, щелкнув на кнопке Choose. Работая в окне диалога, пользователь может определить по отдельности каждый элемент URL.

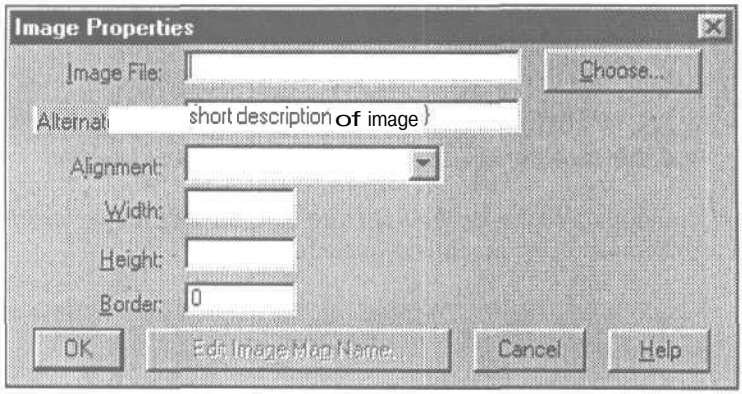


Рис. 8.10. Окно для определения атрибутов изображения

Insert Link (вставить гиперссылку). Создание элемента A и определение его атрибутов. Прежде всего, пользователю предлагается определить адрес ссылки.

Insert Bookmark (вставить закладку). Создание элемента A и определение его атрибута name (это и есть закладка). Такой способ особенно удобен для создания ссылок в пределах одного документа, но он используется также при разработке больших Web-страниц, в которых существует несколько точек перехода. В любом случае созданный элемент будет служить конечной точкой перехода. Необходимо учитывать, что такая ссылка будет работать, если внутри элемента A существует текст. То есть после закрытия окна для определения атрибута надо ввести текст-подсказку в документ.

Insert Link to Bookmark (вставить ссылку на закладку). Создание элемента A и определение его атрибута href. Эта кнопка, как правило, используется в паре с предыдущей. После вставки закладки (с помощью кнопки Insert Bookmark), можно создать гиперссылку, приводящую в указанное закладкой место документа. Данная кнопка позволяет сделать это автоматически: если элемент A, являющийся целью, был выделен, то можно перейти в ту часть страницы, откуда должен быть выполнен переход, и щелкнуть на этой кнопке. Элемент гиперссылки будет создан сразу же, но затем разработчик должен ввести текст, который будет служить подсказкой и сделает элемент видимым при просмотре страницы в броузере.

Insert Table (вставить таблицу). Определение параметров таблицы и вставка соответствующих тегов и атрибутов в документ. Пример окна диалога для создания таблицы показан на рис. 8.11. Прочитавшим раздел «Таблицы» главы 3 легко понять, какие именно параметры требуют определения. При использовании данных величин таблица будет выглядеть так, как показано на рис. 8.12. Обратите внимание, что видимыми являются только внешние теги таблицы TABLE. Сама же таблица прорисовывается в естественном виде. Тем самым облегчается размещение данных в ячейках. Внутрь каждой ячейки автоматически вставляется неразрывный пробел. Это необходимо, чтобы рамка таблицы была правильно показана при отсутствии данных.

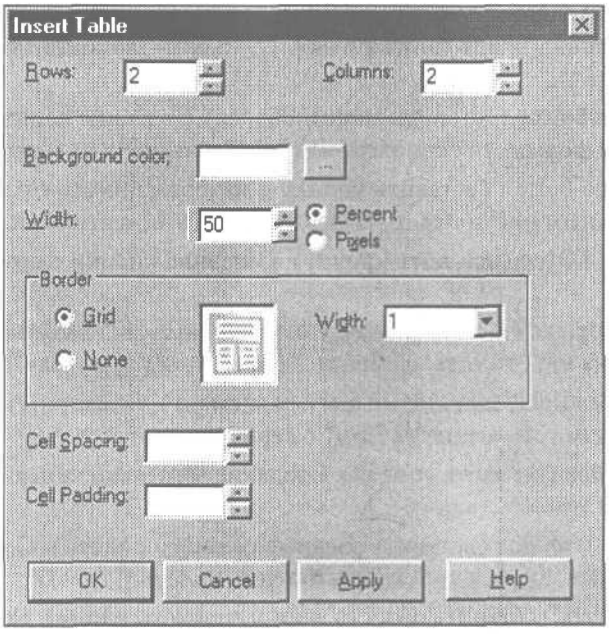


Рис 8.11. Окно для выбора параметров таблицы

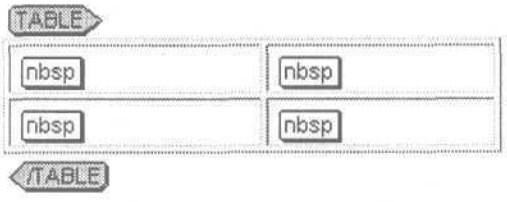


Рис. 8.12. Пример заготовки таблицы

Ниже приведен ряд кнопок, необходимых для создания форм. Они находятся на панели инструментов Forms.

Insert Form (вставить форму). Создание в документе элемента FORM (см. раздел «Элементы форм» главы 4).

Insert Text Box (вставить строку ввода текста). Создание в документе элемента INPUT **с атрибутом** `type="text"`.

Insert Password Entry (вставить строку ввода пароля). Создание в документе элемента INPUT с атрибутом `type="password"`.

Insert Check Box (вставить флажок). Создание флажка, то есть элемента INPUT **с атрибутом** `type="checkbox"`.

Insert Option Button (вставить переключатель). Создание переключателя (который обычно входит в группу), то есть элемента INPUT с атрибутом `type="radio"`.


Insert Submit Button (вставить кнопку подтверждения). Создание кнопки для подтверждения ввода данных в форму, то есть элемента INPUT с атрибутом type="submit".


Insert Reset Button (вставить кнопку сброса). Создание кнопки для удаления данных из формы, то есть элемента INPUT с атрибутом type="reset".


Insert Image Button (вставить кнопку с изображением). Создание кнопки с произвольным рисунком, то есть элемента INPUT с атрибутом type="image".


Insert Push Button (вставить кнопку). Создание кнопки с произвольной надписью.


Insert File Upload (выбор файла). Создание поля для выбора имени файла в режиме диалога, то есть элемента INPUT с атрибутом type="file".


Insert Hidden Input (вставить скрытый элемент). Создание скрытого элемента формы, то есть элемента INPUT с атрибутом type="hidden".


Insert List Box (вставить список). Создание заготовки списка, состоящего из трех строк: <SELECT size=3>.


Insert Dropdown Box (вставить раскрывающийся список). Создание заготовки раскрывающегося списка, состоящего из одной строки: <SELECT size=1>.


Insert Text Area (вставить область многострочного текста). Создание области с полосами горизонтальной и вертикальной прокрутки, позволяющей вводить произвольный текст (элемента TEXTAREA с атрибутами rows и cols).


Insert Java Applet (вставить Java-апплет). Активизация диалога для выбора файла класса. После указания имени файла создается элемент APPLET и автоматически определяются значения его атрибутов.


Insert ActiveX Control (вставить элемент управления ActiveX). Активизация диалога для выбора элемента ActiveX. Затем создается элемент OBJECT и автоматически определяются значения его атрибутов.

Главное меню редактора содержит команды, которые, в общем, повторяют инструменты, вынесенные на основную панель. Но существуют оригинальные и полезные дополнения. Так, например, команда Format ▸ CSS Styles позволяет открыть редактор каскадных таблиц стилей и выбрать необходимые параметры для текущего элемента. На рис. 8.13 показано окно этого редактора, настроенного на элемент TD. В области предварительного просмотра в правой части окна показывается образец элемента (в данном случае ячейки таблицы). Таким образом, можно сразу увидеть, как влияют выбранные настройки на внешний вид элемента.

Отдельно следует упомянуть об использовании кириллицы в HoTMetaL. Если пользователь вводит не латинские символы, то редактор автоматически преобразует введенные буквы в спецсимволы. Разумеется, разработчики редактора не предусматривали возможность использования кириллицы. Просто русские буквы занимают в кодовой таблице место «экзотических» букв из разных языков. В языке может использоваться латинский алфавит и некоторые его «расширения». При вводе русского текста символы прорисовываются обычным образом, но исходный код записывается по-особому: буква А обозначается как À, буква Б — как Á и т. д. Для броузеров с установленным русским шрифтом это не

имеет значения: кириллица в любом случае будет воспроизведена верно. Для броузеров, в которых русский шрифт не установлен, это тоже не имеет значения, так как кириллица в них в любом случае прорисована не будет. Главное отличие такого способа набора текста заключается в том, что исходный код будет трудно читать. Поэтому, если вы используете HoTMetaL в работе, но создаете документы с кириллицей, набирайте русский текст в другой программе, например, в режиме просмотра источника в броузере.

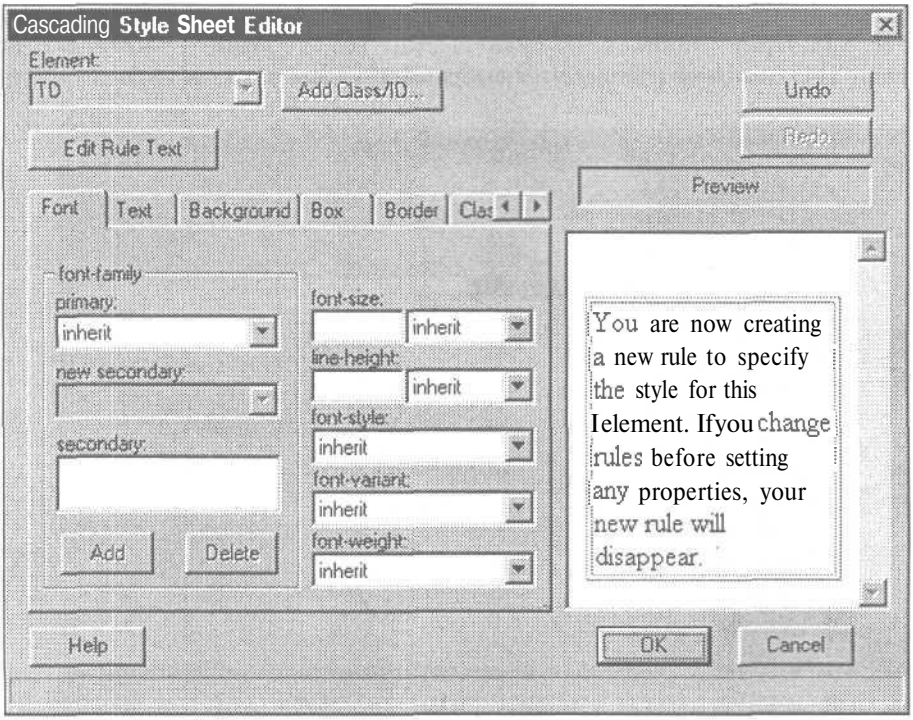


Рис. 8.13. Окно для задания параметров каскадных таблиц стилей

ПРИМЕЧАНИЕ

На Web-странице Spec.htm прилагаемой дискеты продемонстрирован результат ввода русских букв в редакторе HoTMetaL.

Microsoft Word 2000

Последовательность Microsoft во внедрении средств работы с Интернетом в офисные приложения создает удобства, в первую очередь, для российских пользователей. В отличие от англоязычных гипертекстовых редакторов, русифицированные программы не преобразуют кириллицу в спецсимволы. Приложения MS Office 2000 (Access, Excel, Power Point, Word) могут использоваться для генерации Web-

страниц. Источниками информации служат файлы в формате этих программ. Естественно, что на вид страниц влияет специализация каждого приложения. MS Word 2000, являясь текстовым редактором, обеспечивает создание гипертекстовых документов традиционного формата и общего назначения.

Основным инструментом для создания HTML-документов в Word является мастер Web-страниц. Он запускается при создании нового файла. Для запуска мастера надо выбрать команду Файл ▸ Создать, и в открывшемся окне Создание документа перейти на вкладку Web-страницы. На ней надо щелкнуть на значке Мастер Web-страниц. Сама программа находится в файле Webpage.wiz. Первый шаг мастера показан на рис. 8.14. Предполагается, что будет создаваться Web-сайт — набор связанных страниц. Пользователю предлагается выбрать папку для новых HTML-документов.

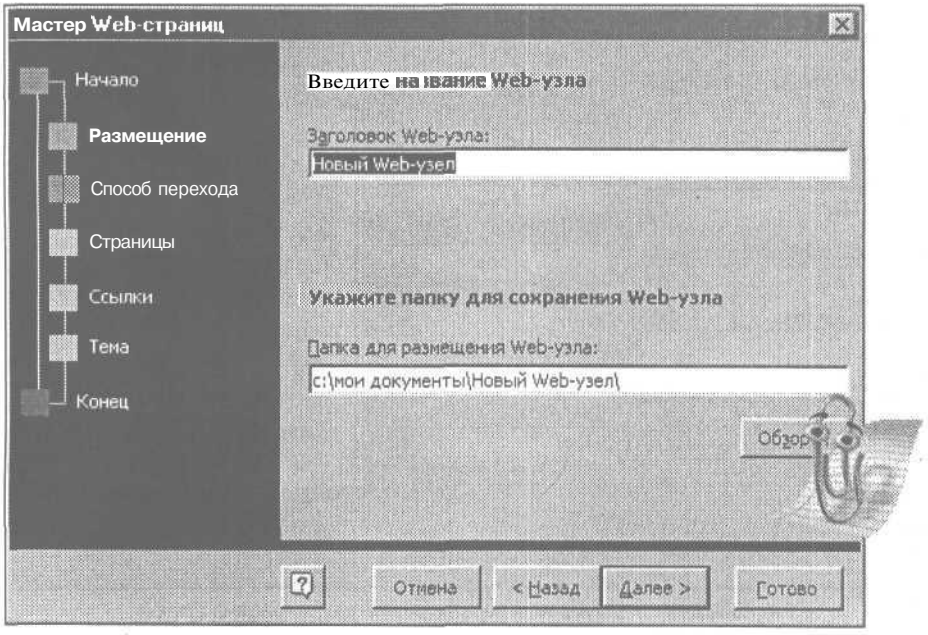


Рис. 8.14. Мастер Web-страниц

На втором шаге мастера открывается окно, показанное на рис. 8.15. Здесь пользователю предлагается выбрать способ организации ссылок для перехода по страницам узла. Мастер предлагает два стандартных способа, которые уже обсуждались в разделе «Стиль и традиции» главы 6: вертикально расположенные ссылки в левой части страницы и горизонтальное меню в верхней части страницы.

Третий шаг мастера (рис. 8.16) позволяет добавить в Web-сайт новые страницы. Для этого используется кнопка Страница шаблонов, открывающая доступ к списку заготовок страниц. Шаблон Текст с оглавлением позволяет создать страницу, показанную на рис. 8.17. После добавления страниц структуру Web-сайта можно считать законченной (четвертый и пятый шаги мастера можно пропустить, ничего не меняя в документах).

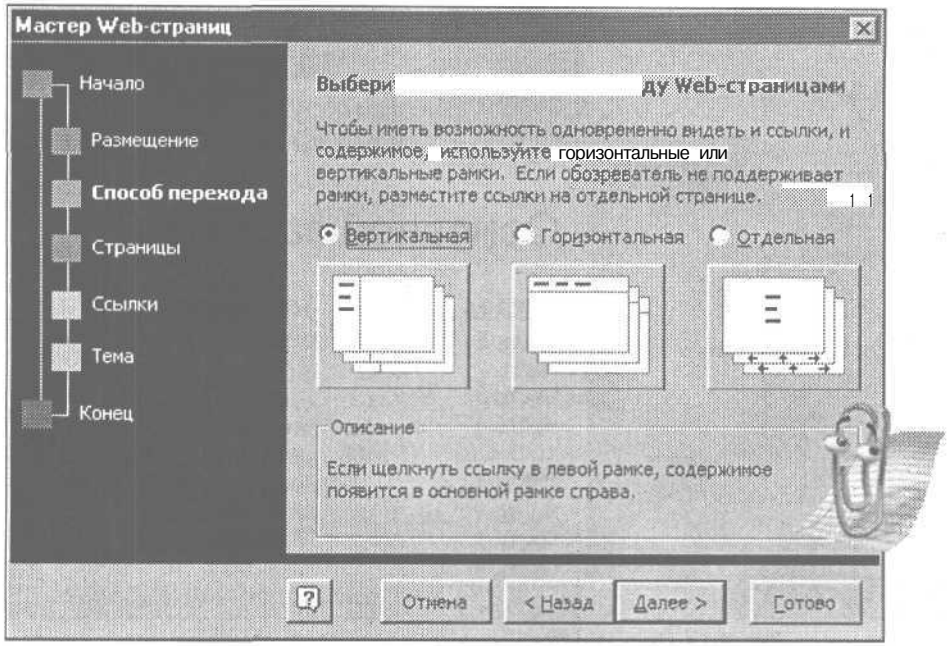


Рис. 8.15. Окно диалога для второго шага мастера Web-страниц

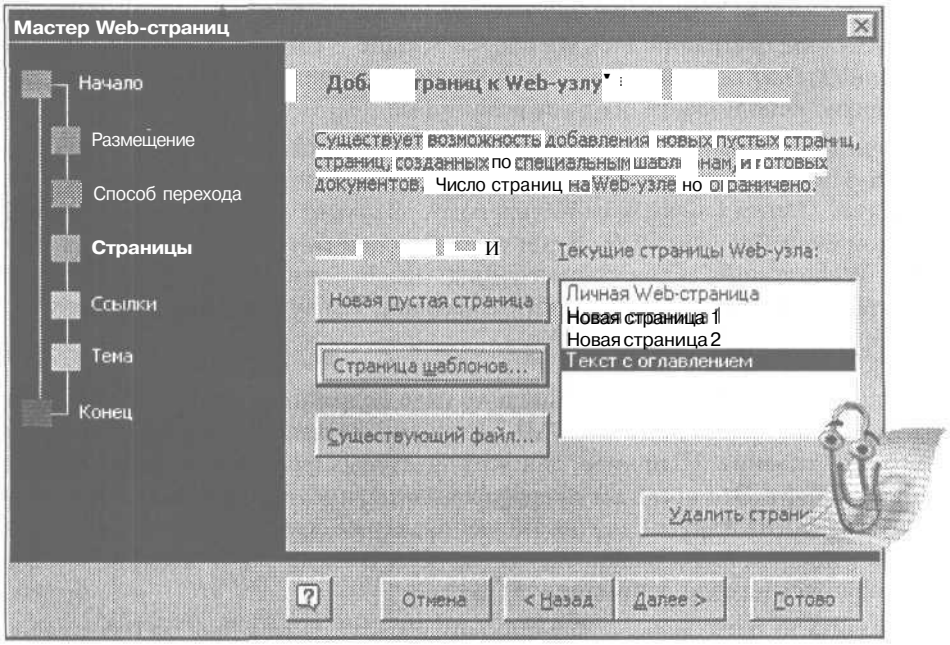


Рис. 8.16. Окно диалога для третьего шага мастера Web-страниц

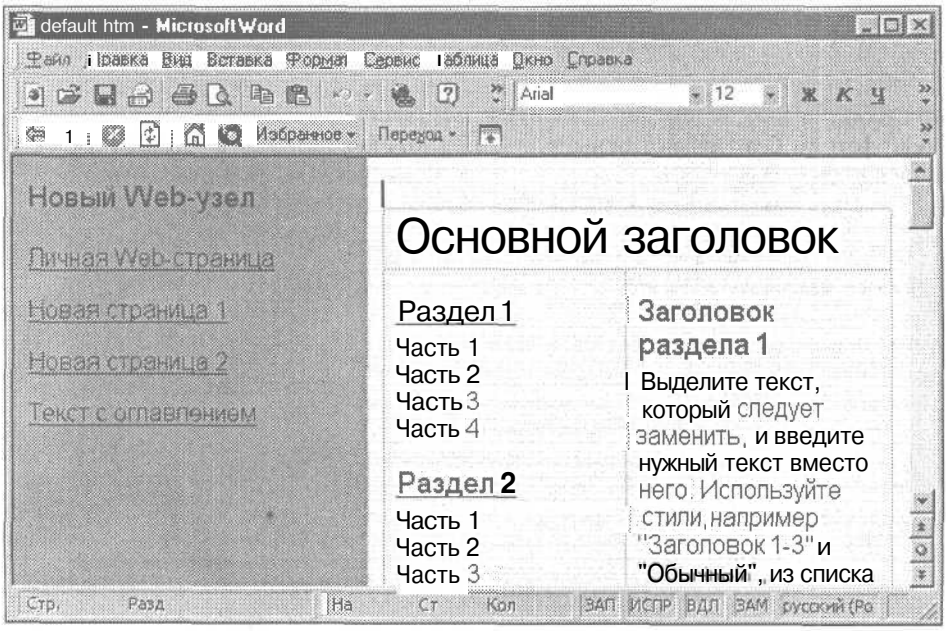


Рис. 8.17. Окно редактора Word 2000 с заготовкой Web-сайта

Остается заполнить страницы полезной информацией, добавить или удалить некоторые элементы. Мастер удобен тем, что позволяет быстро создать несколько взаимосвязанных страниц. Разумеется, настоящие Web-сайты, особенно те, которые принадлежат организациям, постоянно подвергаются модификациям. Для этого можно использовать и другие редакторы.

Полученный документ сохраняется обычным образом, и ему дается расширение htm. Иногда в созданном таким способом документе возникает проблема со шрифтами. В этом случае можно перевести документ в режим просмотра кода командой Вид ▸ Источник HTML. Будет запущен специальный компонент для просмотра Web-страниц в режиме источника. Необходимо убедиться, что внутри элемента META присутствует параметр, задающий «нашу» кодовую страницу:

```
charset=windows-1251
```

Если указана другая кодовая страница, то надо исправить этот фрагмент и сохранить файл. Просматривая код HTML, можно убедиться, как влияет на документ способ его создания. Например, для всех текстовых фрагментов определен конкретный шрифт. Это помогает преобразовывать гипертекстовые страницы в документы Word, но в других случаях может оказаться помехой. Точно также может оказаться лишней информация элементов META. В таких случаях код надо упростить, полагаясь на свое понимание HTML.

На упомянутой выше вкладке Web-страницы имеется значок Простая Web-страница, шаблону которого соответствует файл html.dot. Он позволяет создать пустой файл, и пользователь должен заполнить его информацией от начала и до конца.

Наконец, существует возможность преобразовать в Web-страницу обычный документ Word. Конечно, преобразование ограничивается спецификацией HTML, так как возможности форматирования в Word намного шире. Но, как бы то ни было, таблицы, рисунки, списки и другие фрагменты конвертируются в гипертекстовый формат без труда. Для выполнения этой операции предусмотрена команда Файл ▸ Сохранить как.

В Word не предусмотрены специальные инструменты для форматирования Web-страниц. Использовать необходимо те, которые предназначены для обычного редактирования. Многие кнопки позволяют создавать адекватные эффекты и в гипертексте. Это, в первую очередь, инструменты форматирования текста: выделение полужирным и курсивом, изменение размера и цвета букв, создания эффекта подчеркивания, верхнего и нижнего индексов. Удается создавать и абзацные отступы.

Во время редактирования HTML-файла становится доступной панель Web. Она позволяет редактору выполнять основные функции броузера: передвигаться «вперед» и «назад» по страницам, повторять загрузку текущей страницы, выполнять переход на домашнюю страницу (home page), использовать ссылки из папки Избранное.

Microsoft FrontPage Express

Один из самых простых и доступных редакторов гипертекста — MS FrontPage Express. Он входит в состав Windows, и любой пользователь может быстро установить его на своем компьютере. Бывает, что для этой программы автоматически не создается ярлык. Тогда надо найти файл fpxpress.exe и создать ярлык или команду меню. Полный путь к файлу может быть, к примеру, таким:

```
C:\Program Files\Microsoft FrontPage Express\BIN\FPXPRESS.EXE
```

Начало работы с гипертекстовым редактором заключается в создании пустой страницы. Эта программа похожа на обычный редактор, и вы можете создавать в нем документы так же, как и обычные текстовые. Я рекомендую ввести на новой странице какой-нибудь русский текст, сохранить файл, закрыть программу, а затем открыть файл снова. И тут сразу возникнет проблема: русские буквы отобразятся неправильно (рис. 8.18).

Чтобы решить эту проблему, нам придется изменить код страницы вручную. Используем команду Вид ▸ HTML. Откроется окно редактора, в котором структура документа будет видна в своем первозданном виде. Удобно, что элементы разметки выделяются разными цветами, так что читать текст в режиме *источника* достаточно удобно. Текст документа для нашего примера представлен в листинге 8.1.

Листинг 8.1. Заготовка Web-страницы

```
<html>
<head>
<meta http-equiv="Content-Type"
```

продолжение ⇙

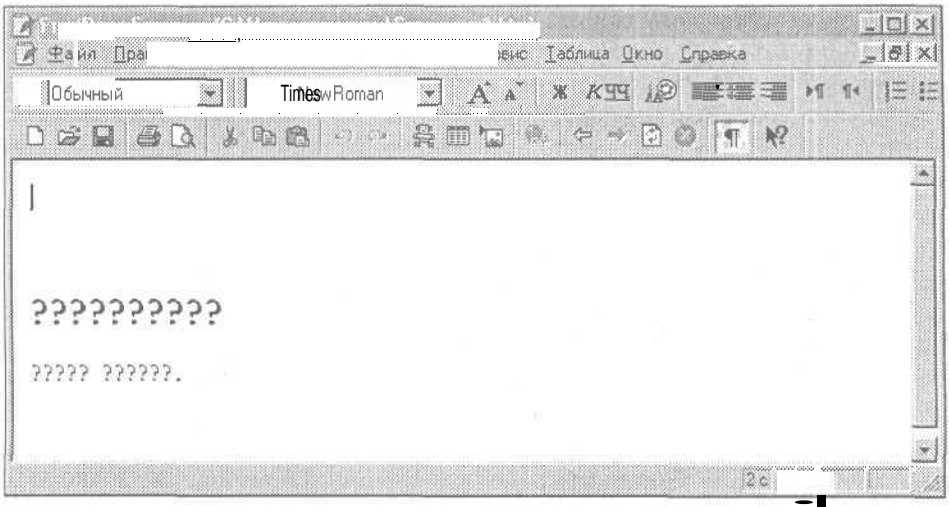


Рис. 8.18. FrontPage Express: проблема с кодовой страницей

Листинг 8.1 *(продолжение)*

```
content="text/html; charset=iso-8859-1">
<meta name="GENERATOR" content="Microsoft FrontPage Express 2.0">
<title>????????</title>
</head>
<body bgcolor="#FFFFFF">
<p> </p>
<h2>??????????</h2>
<p>????? ??????.</p>
</body>
</html>
```

Первый же взгляд на листинг позволяет найти причину ошибки. Кодовая страница указана неправильно. Вместо ISO-8859-1 необходимо указать страницу Windows-1251. Затем надо заменить знаки вопроса русским текстом и сохранить файл. В результате мы получим новый вариант кода HTML, приведенный в листинге 8.2. Только после этого с документом снова можно работать в FrontPage Express. Если бы мы сразу ввели много текста, то его пришлось бы набирать потом заново.

Листинг 8.2. Исправленный текст Web-страницы

```
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=windows-1251">
<meta name="GENERATOR" content="Microsoft FrontPage Express 2.0">
<title>Страница 1</title>
</head>
<body bgcolor="#FFFFFF">
```

```
<p> </p>
<h2>Заголовок</h2>
<p>Текст абзаца.</p>
</body>
</html>
```

Итак, у нас опять есть несколько путей для совершенствования страницы. В каком бы редакторе мы ни работали, приходится выбирать различные режимы просмотра или применять текстовый редактор для правки кода. Мой опыт показывает, что при создании страниц всегда приходится использовать разные способы редактирования.

FrontPage Express работает по такому же принципу, как и другие редакторы гипертекста: для создания большинства элементов HTML предусмотрена специальная кнопка или команда меню. Создадим для примера фон документа. Выбираем команду Формат ▸ Фон и устанавливаем флажок Фоновое изображение. Затем в поле ввода указываем имя графического файла типа GIF или JPG. Если путь к файлу не указан, то подразумевается, что файл находится в той же папке, что и страница. После закрытия окна диалога фон появляется в документе.

Для вставки рисунка надо установить курсор в нужной строке и щелкнуть на кнопке Вставить изображение. В поле ввода печатаем имя графического файла.

Строку заголовка можно выровнять при помощи кнопки Центрировать. Текст на Web-странице вводится и форматируется так же, как в обычном текстовом редакторе. Можно выбрать шрифт и его размер, цвет символов или выравнивание.

Если нужно оснастить Web-страницу возможностью отправки сообщения по электронной почте, надо набрать в нужном месте страницы текст, который служил бы подсказкой, (например, «Электронная почта»). Затем надо выделить этот текст и щелкнуть на кнопке Создать или изменить ссылку. Из списка необходимо выбрать тип гиперссылки mailto:. В строке Адрес (URL) надо ввести адрес электронной почты. В результате первоначально введенная фраза превращается в гиперссылку: она меняет цвет и обретает подчеркнутое начертание. Если пользователь во время просмотра страницы щелкнет на ней, будет запущена программа электронной почты, которая позволит составить и отправить сообщение.

Microsoft FrontPage 2000

В отличие от предыдущего редактора, FrontPage 2000 — достаточно «серьезная» программа. С ее помощью можно создавать как отдельные страницы, так и крупные сайты[1].

В разделе, посвященном редактору HoTMetaL PRO, были приведены описания многих инструментов для конструирования страниц. Данный редактор построен по такому же принципу: множество полезных инструментов сосредоточено на главных панелях и в меню Insert. Видимо, нет необходимости перечислять их еще раз.

[1] Подробное описание этой программы вы найдете в книгах издательства «Питер»: «Microsoft Frontpage 2000: учебный курс» и «Эффективная работа с Microsoft Frontpage 2000».

Редактор FrontPage входит в состав MS Office, поэтому он унаследовал от офисных программ инструментарий для работы с документами. При создании страницы можно пользоваться кнопками, которые выполняют одинаковые функции во всех офисных программах. В то же время есть различия. Обратите внимание, что список для выбора шрифта (рис. 8.19) содержит пункт default font (шрифт по умолчанию). В обычных текстовых документах все шрифты должны быть явно указаны, а на Web-странице это не обязательно. Имеется в виду, что шрифт определяется настройками программы просмотра.

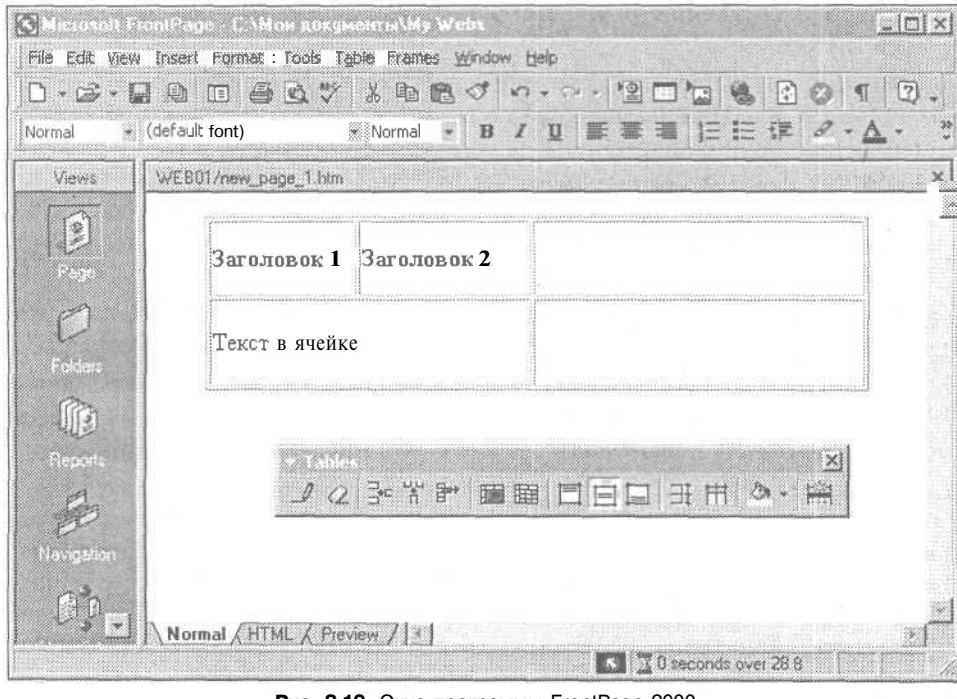


Рис. 8.19. Окно программы FrontPage 2000

Окно редактора имеет три вкладки, позволяющие менять режим просмотра документа: Normal (обычный), HTML и Preview (предварительный просмотр). Вкладка HTML позволяет редактировать текст в режиме источника. Что касается обычного и предварительного режимов просмотра, то они похожи, хотя и имеют определенные отличия. Например, если в некоторых ячейках таблицы *отсутствуют* данные, то в режиме Normal рамка таблицы будет показана полностью, а в режиме Preview рамка будет показана так, как ее увидит человек, использующий обычный броузер: часть линий будет отсутствовать.

Для создания таблицы предназначена специальная панель инструментов (рис. 8.19), а контуры таблицы можно нарисовать мышью так же, как и в редакторе Word. Если пользователь создаст таблицу нерегулярной структуры (с объединенными ячейками), то это будет автоматически зафиксировано в коде HTML с помощью атрибутов colspan и rowspan.

Если использовать подходы, обсуждаемые в главе 6, то напрашивается мысль, что при создании новой страницы удобно нарисовать таблицу, которая определит положение различных элементов, и заполнить ее информацией (текстом, рисунками, гиперссылками). Менять размеры, высоту строк и ширину столбцов можно с помощью мыши. Когда документ будет завершен, таблицу можно будет скрыть при помощи атрибута `border=0`.

Панель инструментов Tables содержит ряд интересных кнопок.

Draw Table (нарисовать таблицу). Карандаш, при помощи которого создается контур таблицы. Не удивляйтесь, если в процессе рисования пунктирная линия, обозначающая деталь рамки, будет «скакать» с места на место. Не все конфигурации таблицы доступны, и программа в некоторых случаях будет вас поправлять.

Eraser (ластик). Чтобы удалить часть рамки, надо «потереть» ее так, чтобы она изменила цвет (обычно на красный).

Insert Rows, Insert Columns (вставить строки, вставить столбцы). Добавление новых строк (выше текущей) или новых столбцов (слева от текущего). Курсор достаточно установить в одну из ячеек.

Delete Cells (удалить ячейки). Вначале удаляемые ячейки необходимо выделить. Если требуется удалить строку (или столбец), надо поместить указатель мыши слева от строки (или над столбцом), чтобы указатель принял вид черной стрелки, и щелкнуть один раз.

Merge Cells (объединить ячейки). Вначале необходимо выделить несколько ячеек таблицы, а затем щелкнуть на этой кнопке.

Split Cells (разделить ячейки). Можно поместить курсор в одну ячейку (если надо разделить ее), а можно выделить строку или столбец. В любом случае требуется установить параметры в окне диалога (рис. 8.20). Если выбрана одна ячейка, деление можно выполнить по вертикали (Split into columns) или горизонтали (Split into rows). Количество новых столбцов определяется числовым значением. Если предварительно выделена группа ячеек, то программа сама определит, какие параметры разделения будут доступны.

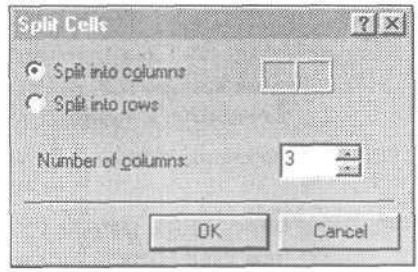


Рис. 8.20. Окно для разделения ячейки

Align Top, Align Vertically, Align Bottom. Три очень полезные кнопки для выравнивания данных в ячейках по вертикали: по верхнему краю, по середине, по нижнему краю соответственно.

Distribute Rows Evenly. С помощью этой кнопки можно задать одинаковую высоту для нескольких строк. Программа выбирает некоторую среднюю величину, если выделенные строки имеют разную высоту.

Distribute Columns Evenly. Аналогичная операция, но для выравнивания ширины столбцов.

Fill Color (заполнить цветом). Выбор цветного фона для ячеек таблицы.

AutoFit. Эта кнопка воздействует сразу на всю таблицу. Размеры ячеек уменьшаются, насколько это возможно: так, чтобы все данные были видны. Размеры пустых сокращаются до минимума.

Редактор позволяет создавать различные страницы, в том числе с фреймами. Для любого нового документа используется шаблон, поэтому после выбора команды File ▸ New ▸ Page, надо выбрать один вариант из многих. На вкладке Frames Pages есть шаблон Nested Hierarchy, который позволяет создать три фрейма (рис. 8.21). Полученная страница является *документом раскладки,* и для редактирования его кода предусмотрена вкладка Frames Page HTML.

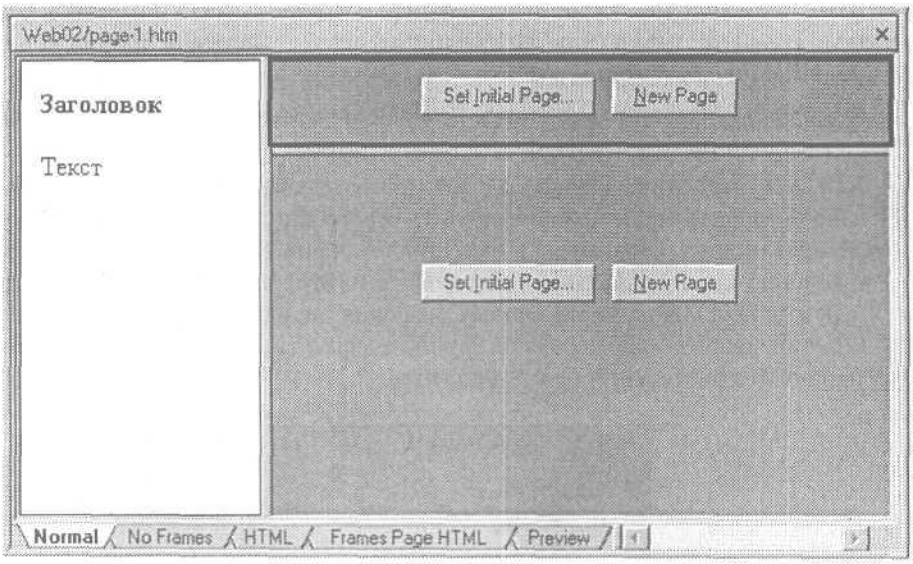


Рис. 8.21. Окно просмотра страницы с фреймами

Документы содержания должен определить разработчик. Для создания новых страниц предусмотрены кнопки New Page (каждая страница в своем фрейме), а если страница уже существует, то нужно указать ее местоположение при помощи кнопки Set Initial Page. На рис. 8.21 видно, что одна из страниц уже выбрана. После того как все страницы определены, с ними можно работать в режиме редактирования, но в пределах фрейма. Это может показаться не очень удобным, так как пространство документа ограничено, но в этом случае вы работаете с реальным маке-

том: если фрейм узкий, то и не следует размещать на нем слишком большие объекты.

Важным достоинством FrontPage 2000 является возможность создания каскадных таблиц стилей на основе шаблонов. После выбора команды File ▸ New ▸ Page надо открыть вкладку Style Sheets и выбрать подходящий шаблон. Будет создан файл формата CSS, в котором появятся стили для основных элементов.

Одновременно на экран будет выведена кнопка Style (стиль), с помощью которой можно отредактировать таблицу стилей. В первом окне диалога надо выбрать элемент и щелкнуть на кнопке Modify, а в следующем — на кнопке Format. Теперь можно менять свойства элемента. Работать, скорее всего, придется с дополнительными окнами диалога, в зависимости от свойства. Такой подход очень удобен. Во-первых, исключаются ошибки программирования, когда для элемента неправильно выбирается свойство или значение свойства. Во-вторых, работая с элементом, вы узнаете, какими свойствами он обладает.

Напомню, что каскадная таблица стилей создается не для конкретной страницы, то есть может быть связана с любым количеством страниц. Если надо выбрать CSS для Web-страницы, используя FrontPage 2000, откройте страницу и выберите команду Format ▸ Style Sheet Links. Укажите CSS-файл, и редактор добавит на страницу элемент LINK с необходимой ссылкой. Страница сразу же будет отформатирована в соответствии с новыми стилями.

Еще одна важная особенность редактора — возможность работы с Web-сайтами. Новый сайт создается командой File ▸ New ▸ Web тоже на основе шаблонов. Можно использовать и существующий сайт (команда File ▸ Open Web). Создайте новый Web-сайт при помощи мастера Corporate Presence Wizard, который позволяет создавать образцы сайтов организации. Работа с мастером заключается в ответе на вопросы о рубриках, которые должны существовать. В конце работы мастер создает комплекс Web-страниц.

После создания образца сайта можно щелкнуть на кнопке Hyperlinks на левой панели редактора и увидеть схему сайта (рис. 8.22). Файл домашней (титульной) страницы, как и положено, называется index.htm, линии обозначают ссылки на подчиненные страницы. Теперь все зависит от вас. Щелкнув дважды на значке любой страницы, можно запустить режим редактирования, отформатировать и заполнить документ оригинальной информацией.

Netscape Composer

Этот редактор существует как составная часть броузера Netscape Communicator (см. раздел «Netscape Communicator» главы 1). Поэтому в меню Файл существуют две команды: Открыть Страницу (в броузере) и Редактировать Страницу (в редакторе). Разработчику не надо задумываться о том, в каком режиме открыть документ. Оба режима легко переключаются при помощи кнопок, выведенных на панель инструментов. Окно редактора показано на рис. 8.23.

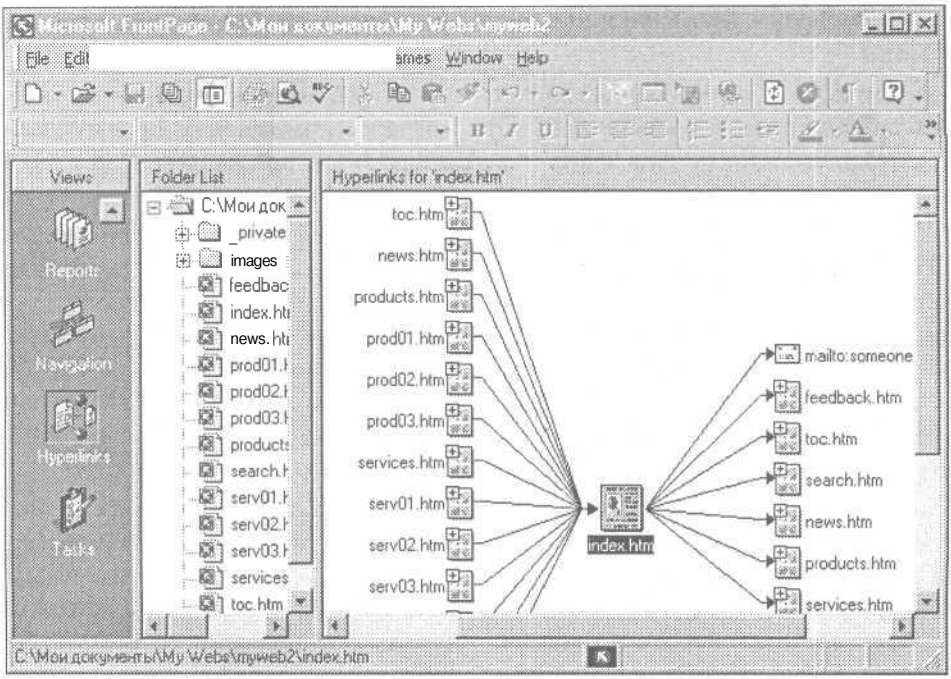


Рис. 8.22. Просмотр схемы сайта

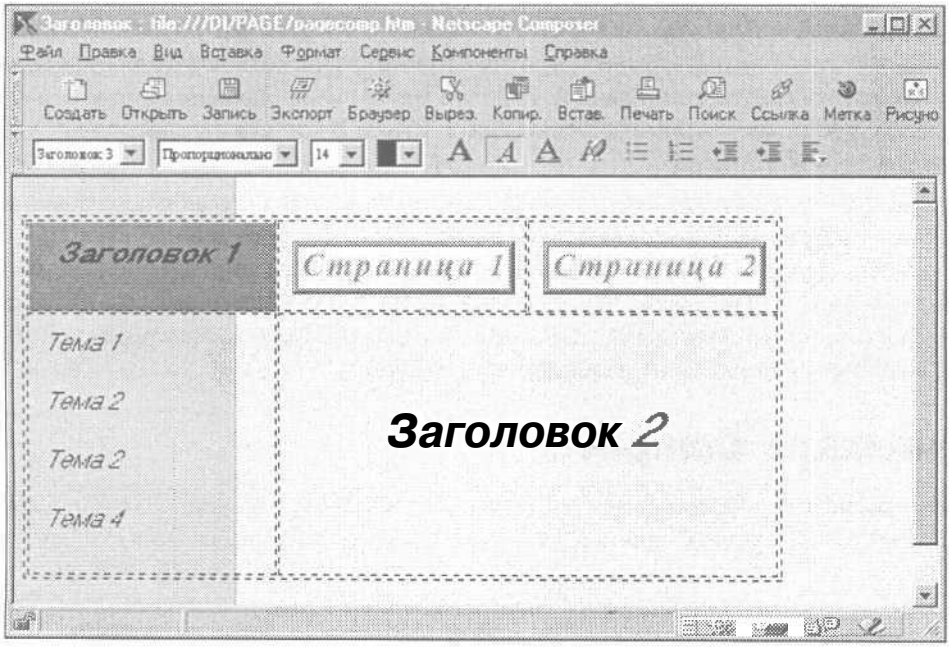


Рис. 8.23. Редактор гипертекста Netscape Composer

В Netscape Composer гипертекстовый документ представлен примерно в том же формате, что и в броузере. То есть код HTML не виден полностью. Во время редактирования документа пользователь должен выделять фрагменты или выбирать отдельные объекты для того, чтобы к ним можно было применить операции форматирования. Форматирование в большинстве случаев заключается в создании новых элементов или изменении атрибутов уже существующих. Щелкнув правой кнопкой мыши на объекте и выбрав в контекстном меню команду Свойства, можно перейти к окну свойств элемента (объекта) и изменить их, заполняя соответствующие поля. Недостатком редактора является автоматическое (во время сохранения документа) преобразование символов кода HTML в верхний регистр. Редактор весьма терпимо относится к параллельному редактированию *открытого* документа. Если в последний внесены изменения с помощью другого приложения, то Netscape Composer фиксирует этот факт и предлагает загрузить файл заново, чтобы увидеть текущие изменения.

Рассмотрим инструменты, вынесенные на основную панель.

Формат абзаца. Из этого списка можно выбрать тип элемента (заголовок, список и т. д.).

Шрифт. Выбор шрифта. Кроме обычных шрифтов, список содержит пункты Моноширинный и Пропорциональный. Это типы шрифтов, выбранных для использования в броузере. У разных пользователей они могут быть разные. Выбор моноширинного шрифта производится при помощи элемента TT.

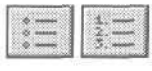
Размер шрифта. Величины в списке соответствуют стандартным размерам, но в документе указывается размер по правилам HTML, например:

```
<FONT SIZE=+2>
```

Цвет шрифта. Выбор цвета из палитры.

Традиционные кнопки для форматирования полужирным, курсивом и подчеркиванием.

Удалить все стили. Эта кнопка позволяет удалить некоторые элементы, определяющие стилевое оформление. Текст можно выделить, а можно и не выделять. Удаляется элемент, внутри которого оказался курсор. Ну, а насчет «всех» — это, конечно, преувеличение...



Маркированный Список, Нумерованный Список. Кнопки позволяют создавать или удалять элементы одноуровневых списков.

Уменьшить отступ, Увеличить отступ. В HTML не предусмотрены абзацные отступы, но, тем не менее, две последние кнопки прекрасно работают. Дело в том, что отступ создается при помощи элемента BLOCKQUOTE, который обеспечивает создание отступа. Вкладывая один элемент в другой, можно получить любое количество позиций отступа. Остроумное решение! Ниже показан пример абзаца с двойным отступом:

```
<BLOCKQUOTE>
<BLOCKQUOTE>
<P>Текст абзаца</P>
</BLOCKQUOTE>
</BLOCKQUOTE>
```

 Выравнивание. Раскрытие списка и выбор способа *вертикального* выравнивания.

Разумеется, редактор имеет и другие средства конструирования страниц. В меню спрятано много полезных команд. Но полезность редактора, на мой взгляд, заключается не только в этом. Использование инструментов Netscape Composer позволяет выявить надежно работающие элементы. Если редактор добавляет элемент BLOCKQUOTE или TT для создания определенного визуального эффекта, то можно быть уверенным, что этот эффект проявится при просмотре страницы в большинстве броузеров.

Приложение А

Шестнадцатеричные числа

dec	hex	dec	hex	dec	hex	dec	hex	dec	hex	dec	hex	dec	hex	dec	hex
0	00	32	20	64	40	96	60	128	80	160	AO	92	CO	224	EO
1	01	33	21	65	41	97	61	129	81	161	A1	193	C1	225	E1
2	02	34	22	66	42.	98	62.	130	82	162	A2	194	C2	226	E2
3	03	35	23	67	43	99	63	131	83	163	A3	195	C3	227	E3
4	04	36	24	68	44	100	64	132	84	164	A4	196	C4	228	E4
5	05	37	25	69	45	101	65	133	85	165	A5	197	C5	229	E5
6	06	38	26	70	46	102	66	134	86	166	A6	198	C6	230	E6
7	07	39	27	71	47	103	67	135	87	167	A7	199	C7	231	E7
8	08	40	28	72	48	104	68	136	88	168	A8	200	C8	232	E8
9	09	41	29	73	49	105	69	137	89	169	A9	201	C9	233	E9
1C	OA	42	2A	74	4A	106	6A	138	8A	170	AA	202	CA	234	EA
И	OB	43	2B	75	4B	107	6B	139	8B	171	AB	203	CB	235	EB
12	OC	44	2C	76	4C	108	6C	140	8C	172	AC	204	CC	236	EC
13	OD	45	2D	77	4D	109	6D	141	8D	173	AD	205	CD	237	ED
14	OE	46	2E	78	4E	110	6E	142	8E	174	AE	206	CE	238	EE
15	OF	47	2F	79	4F	111	6F	143	8F	175	AF	207	CF	239	EF
16	10	48	30	80	50	112	70	144	90	176	BO	208	DO	240	FO
17	11	49	31	81	51	113	71	145	91	177	B1	209	D1	241	F1
18	12	50	32	82	52	114	72	146	92	178	B2	210	D2	242	F2
19	13	51	33	83	53	115	73	147	93	179	B3	211	D3	243	F3
20	14	52	34	84	54	116	74	148	94	180	B4	212	D4	244	F4
21	15	53	35	85	55	117	75	149	95	181	B5	213	D5	245	F5
22	16	54	36	86	56	118	76	150	96	182	B6	214	D6	246	Г6
23	17	55	37	87	57	119	77	151	97	183	B7	215	D7	247	F7
24	18	56	38	88	58	120	78	152	98	184	B8	216	D8	248	F8
25	19	57	39	89	59	121	79	153	99	185	B9	217	D9	249	F9
26	1A	58	3A	90	5A	122.	7A	154	9A	186	BA	218	DA	250	FA
27	1B	59	3B	91	5B	12.3	7B	155	9B	187	BB	219	DB	251	FB
28	1C	60	3C:	92	5C	124	7C	156	9C	188	BC	220	DC	252	PC
29	1D	61	3D	93	5D	125	7D	157	9D	189	BD	221	DD	253	FD
30	1E	62	3E	94	5E	126	7E	158	9E	190	BE	222	DE	254	FE
31	1F	63	3F	95	5F	127	7F	159	9F	191	BF	223	DF	255	FF

Приложение Б

Свойства таблиц стилей

Ниже приведены свойства, которые можно использовать в таблицах стилей. Это
перечень определяется как CSS2 — каскадные таблицы стилей второго уровня
Жирным шрифтом выделены значения свойств, принимаемые по умолчанию. Во
свойства помимо перечисленных значений могут иметь стандартное значение inherit
которое позволяет наследовать аналогичное свойство родительского элемента.

Единицы измерения

В тех случаях, когда значение свойства задается числовой величиной, могут быт
использованы определенные единицы измерения.

Единицы длины:

- cm — сантиметр;
- in — дюйм (2,54 см);
- mm — миллиметр;
- pc — пика (1/6 дюйма);
- pt — пункт (1/72 дюйма);
- px — пиксел.

Единицы времени:

- ms — миллисекунда;
- s — секунда.

Величины, заданные в процентах, обозначаются знаком %.

Шрифты

font-family

Значения: названия шрифтов, которые могут быть использованы на Web-стра
нице.

Пример:

```
BODY {font-family: Times New Roman, Verdana, Tahoma}
```

font-style

Стиль шрифта.

Значения:

- **normal** — обычный;
- italic, oblique — курсив.

font-variant

Еще одно свойство для задания стиля шрифта.

Значения:

- **normal** — обычный;
- small-caps — малые прописные.

Пример:

```
H1, H2 {font-variant: small-caps}
```

font-weight

Толщина («жирность») символов шрифта.

Значения:

- **normal** — обычный (400);
- bold — полужирный (600 или 700);
- bolder — более жирный, нежели в родительском элементе;
- lighter — менее жирный, нежели в родительском элементе;
- 100, 200, 300, 400, 500, 600, 700, 800, 900 — допустимые числовые значения.

font-size

Размер шрифта.

Значения:

- xx-small, x-small, small, medium, large, x-large, xx-large — абсолютные значения;
- Npt — значение, выраженное в пунктах;
- N% — значение, выраженное в процентах;
- larger, smaller — относительные значения.

Пример:

```
<P style="font-size: 16pt ">
```

font

Выбор определенного шрифта, применительно к конкретным задачам. Вид шрифта определяется настройками программы.

В качестве значений могут использоваться величины свойств font-style, font-variant, font-weight, font-size, line-height, font-family. Другие значения:

- caption — шрифт для элементов управления (полужирный);
- icon — шрифт для пиктограмм;
- menu — шрифт для меню;
- messagebox — шрифт для окон диалога;
- smallcaption — шрифт для небольших элементов управления;
- statusbar — шрифт для строки состояния.

Пример:

```
P. icon1 {font: icon}
```

Форматирование текста

text-indent

Величина отступа первой строки абзаца.

Значения: числовые.

text-align

Выравнивание текста.

Значения:

- **left** — по левому краю;
- right — по правому краю;
- center — по центру;
- justify — по ширине.

text-decoration

Оформление текста.

Значения:

- **none** — без оформления;
- underline — подчеркивание;
- overline — черта сверху;
- line-through — перечеркивание;
- blink — мигание (не поддерживается некоторыми броузерами).

text-shadow

Создание тени для букв. Значение свойства представляет собой список величин, которые определяют характеристики тени. Данное свойство не поддерживается большинством популярных броузеров.

Примеры:

- Создание тени справа и снизу от букв:

```
H1 {text-shadow: 3px 3px }
```

- Создание тени зеленого цвета справа и снизу и задание радиуса размытия (3 пиксела) тени:

```
P {text-shadow: 2px 2px 3px green}
```

letter-spacing

Задание межбуквенного интервала.

Значения:

- **normal** – определяется броузером;
- числовые;
- auto — интервал изменяется так, чтобы текст уместился в одной строке. Не поддерживается рядом броузеров.

Пример:

```
H2 {letter-spacing: 5pt}
```

line-height

Высота строк.

Значения:

- **normal** — определяется броузером (обычно лежит в пределах 1.0-1.2);
- числовое с указанием единиц измерения — абсолютное значение;
- число — размер шрифта (значение свойства font-size) умножается на эту величину;
- процентное — по отношению к значению свойства font-size.

Примеры:

```
P { line-height: 1.2; font-size: 16pt }
P { line-height: 1.4cm }
P { line-height: 150%; font-size: 10pt }
```

word-spacing

Выбор расстояния между словами.

Значения:

- **normal** — определяется броузером;
- числовые.

text-transform

Изменение вида букв.

Значения:

- **none** — без изменения;
- capitalize — первая буква каждого слова прописная;
- uppercase — все буквы прописные;
- lowercase — все буквы строчные.

white-space

Определение вида пробелов.

Значения:

- **normal** — автоматическое форматирование текста. Например, удаляются пробелы, следующие друг за другом;
- pre — текст остается без изменения (по аналогии с элементом PRE);
- nowrap — запрет наавтоматический разрыв строк.

direction

Направление текста.

Значения:

- ltr — слева направо;
- rtl — справа налево.

Свойства списков

list-style-type

Определение вида маркеров в списке.

Значения:

- none — без маркеров;
- **disc** — круги;
- circle — окружности;
- square — квадраты;
- decimal — арабские цифры;
- lower-roman — римские цифры на основе строчных латинских букв (i, v, x...);
- upper-roman — римские цифры на основе прописных латинских букв (I, V,X...);
- lower-alpha — строчные латинские буквы (a, b,c...);
- upper-alpha — прописные латинские буквы (A, B, C...).

Пример:

```
LI {list-style-type: lower-alpha}
```

list-style-image

Определение рисунка, который будет использоваться в качестве маркера.

Значения:

- **none** — рисунок не используется;
- адрес (URL) графического файла.

Пример:

```
UL {list-style-image: url(marker.gif)}
```

list-style-position

Положение маркера относительно списка.

Значения:

- inside — маркер внутри списка (компактная форма);
- outside — маркер вне списка.

list-style

Определяет вид маркеров. В качестве значения может использоваться несколько величин свойств list-style-type, list-style-position, list-style-image (перечисленных выше).

Пример:

```
UL {list-style: circle outside}
```

Свойства таблиц

display

Создание таблиц и элементов таблиц.

Значения:

- table — таблица (аналог элемента TABLE);
- table-caption — заголовок таблицы (аналог элемента CAPTION);
- table-column — колонка таблицы (аналог элемента COL);
- table-column-group — группа колонок (аналог элемента COLGROUP);
- table-row — строка таблицы (аналог элемента TR);
- table-row-group — группа строк (аналог элемента TBODY);
- table-header-group — группа строк в начале таблицы (аналог элемента THEAD);
- table-footer-group — группа строк в конце таблицы (аналог элемента TFOOT);
- table-cell — ячейка таблицы (аналог элемента TD);

row-span

Количество строк, которые должна занять ячейка.

column-span

Количество столбцов, которые должна занять ячейка.

border-collapse

Способ прорисовки рамки.

Значения:

- **separate** — стиль броузеров (трехмерная рамка);
- collapse — стиль текстовых редакторов (плоская рамка).

border

Характеристика рамки.

Значение свойства состоит из трех элементов: толщина, тип, цвет.

Возможные типы:

- none или hidden — рамка не показана;
- dotted — рамка из точек;
- dashed — пунктирная рамка;
- solid — сплошная рамка;
- double — двойная рамка;
- groove — двугранная рамка;
- ridge — такая же, как groove, но светлая и темная грани расположены иначе;
- inset — одна грань, наклоненная внутрь;
- outset — такая же, как inset, но свет падает по-другому.

Примеры:

```
TABLE { border: 1px solid blue; }
<TABLE style="border-collapse: collapse; border: solid green;">
<TD style="border: double">
```

vertical-align

Способ (степень) вертикального выравнивания.

Значения:

- процентное;
- top — по верхнему краю;
- middle — по центру;
- bottom — по нижнему краю;
- baseline — по первой строке текста в ячейке.

table-layout

Алгоритм форматирования таблицы.

Значения:

- **auto** — стандартный;
- `fixed` — ускоренный.

Свойства границ элементов

width

Ширина объекта.

Значения:

- **auto** — определяется броузером;
- числовое;
- процентное.

min-width и max-width

Минимально возможная и максимально допустимая ширина объекта.

Значения:

- **auto** — определяется броузером;
- числовое;
- процентное.

height

Высота объекта.

Значения:

- **auto** — определяется броузером;
- числовое;
- процентное.

min-height и max-height

Минимально возможная и максимально допустимая высота объекта.

Значения:

- **auto** — определяется броузером;
- числовое;
- процентное.

position

Способ вычисления координат границ элемента.

Значения:

- **normal** — положение элемента определяется броузером;
- relative — вначале вычисляется обычное положение элемента, затем рассчитывается смещение относительно этого положения;
- absolute — координаты вычисляются относительно границ контейнера, содержащего элемент;
- fixed — координаты вычисляются относительно границ контейнера, содержащего элемент, и запрещается прокрутка.

top, bottom, right, left

Координаты верхней, нижней, правой и левой границ элемента.

Значения:

- **auto** — определяется броузером;
- числовое;
- процентное.

margin-top, margin-right, margin-bottom, margin-left

Размер пустого пространства рядом с определенной стороной элемента.

Значения:

- **auto** — определяется броузером;
- числовое;
- процентное.

margin

Величина отступов вокруг элемента.

Значения:

- **auto** — определяется броузером;
- числовое;
- процентное.

Пример:

```
P { margin: 12px 12px 12px 12px }
```

padding-top, padding-right, padding-bottom, padding-left

Величина отступа между указанной границей элемента и его содержимым.

Значения:

- auto – определяется броузером;
- числовое;
- процентное.

padding

Величина отступа между границами элемента и его содержимым.

Значения:

- auto — определяется броузером;
- числовое;
- процентное.

Пример:

```
OL { padding: 4px 4px 4px 4px }
```

border-top-width, border-right-width, border-bottom-width, border-left-width

Ширина элементов рамки.

Значения:

- thin — тонкая;
- medium — средняя;
- thick — широкая;
- числовое.

border-width

Толщина рамки.

Значения:

- thin — тонкая;
- medium —средняя;
- thick — широкая;
- числовое.

border-top-color, border-right-color, border-bottom-color, border-left-color

Цвет элементов рамки.

border-color

Цвет рамки.

border-top-style, border-right-style, border-bottom-style, border-left-style

Вид элементов рамки.

Значения:

- none или hidden — рамка не показана;
- dotted — рамка из точек;
- dashed — пунктирная рамка;
- solid — сплошная рамка;
- double — двойная рамка;
- groove — двугранная рамка;
- ridge — такая же, как groove, но светлая и темная грани расположены иначе;
- inset — одна грань, наклоненная внутрь;
- outset — такая же, как inset, но свет падает по-другому.

border-style

Вид рамки.

Значения такие же, как и у предыдущих свойств.

border-top

Одновременное задание значений свойств border-top-width, border-top-style и border-top-color.

Пример:

```
H1 { border-top: thin solid blue }
```

border-bottom

Одновременное задание значений свойств border-bottom-width, border-bottom-style и border-bottom-color.

border-left

Одновременное задание значений свойств border-left-width, border-left-style и border-left-color.

border-right

Одновременное задание значений свойств border-right-width, border-right-style и border-right-color.

border

Одновременное задание значений свойств border-width, border-style и border-color.

Пример:

```
P { border: medium double red }
```

overflow

Способ изменения размеров объекта, если его содержимое не может быть показано целиком.

Значения:

- **auto** — определяется броузером;
- visible — размер границ увеличивается в такой степени, чтобы содержимое было видимым;
- hidden — размеры границ не устанавливаются в зависимости от размеров содержимого;
- scroll — устанавливается режим прокрутки.

float

Способ обтекания элемента другими.

Значения:

- **none** — обтекания нет;
- left — слева;
- right — справа.

Пример:

```
IMG { float: left }
```

clear

Способ расположения элементов, следующих за элементом, выровненным по левому или правому краю.

Значения: **none,** left, right, both.

clip

Определение видимой части объекта.

Значения:

- **auto** — определяется броузером;
- прямоугольник, «вырезающий» часть объекта.

Пример:

```
P { clip: rect(5px, 10px, 10px, 5px); }
```

visibility

Видимость или невидимость объекта.

Значения:

- **visible** — видимый;
- hidden — невидимый.

z-index

Способ перекрытия объектов другими. Объект, имеющий большее значение индекса, располагается выше.

Значения:

- **auto** — «нулевой уровень», задаваемый по умолчанию;
- отрицательное числовое — объект располагается «ниже» нулевого уровня;
- положительное числовое — объект располагается «выше» нулевого уровня.

Свойства фона и цвета

color

Цвет элемента или его содержимого.

Примеры:

```
U { color: red }
P { color: rgb(255,120,120) }
```

background-color

Цвет фона.

Значения:

- **transparent** — прозрачный фон;
- значение цвета в стандартном формате.

Пример:

```
H1 { background-color: #9AC159 }
```

background-image

Фоновый рисунок.

Значения:

- **none** — без рисунка;
- ссылка на файл.

Пример:

```
TABLE { background-image: url(karandash.gif) }
```

background-repeat

Свойство, определяющее, надо ли повторять фоновый рисунок для заполнения пространства элемента.

Значения:

- **repeat** — заполнять все пространство элемента;
- repeat-x — заполнить только первый горизонтальный ряд;
- repeat-y — заполнить только первый вертикальный столбец;
- no-repeat — не заполнять.

Пример:

```
BODY { background-image: url(kvadrat.gif); background-repeat: repeat-y; }
```

background-attachment

Свойство, определяющее, будет ли фон прокручиваться вместе с содержимым документа.

Значения:

- **scroll** — прокрутка разрешена;
- fixed — прокрутка запрещена.

background-position

Начальное положение фонового рисунка.

Значениями свойства являются две величины в численном (или процентном) выражении, а также символьные коды:

- 0% 0% — значение по умолчанию;
- top left или left top — значение 0% 0%;
- top, top center или center top — значение 50% 0%;
- right top или top right — значение 100% 0%;
- left, left center или center left — значение 0% 50%;
- center или center center — значение 50% 50%;
- right, right center или center right — значение 100% 50%;
- bottom left или left bottom — значение 0% 100%;
- bottom, bottom center или center bottom — значение 50% 100%;
- bottom right или right bottom — значение 100% 100%.

Если указана одна величина, она интерпретируется как значение отступа по горизонтали, а значение по вертикали принимается равным 50%.

Пример:

```
BODY { background-image: url(banner.gif); background-position: top center }
```

background

Фон элемента. Одновременное задание свойств background-color, background-image, background-repeat, background-attachment и background-position.

Примеры:

```
TD { background: url(http://сервер.com/metal.jpg) }
P { background: url(sphere.gif) gray 50% repeat fixed }
```

Свойства мультимедиа

volume

Громкость звука.

Значения:

- числовое в пределах 0-100;
- процентное в пределах 0-100%;
- silent — без звука;
- x-soft — уровень звука 0 (самый тихий);
- soft — уровень звука 25;
- **medium** — уровень звука 50;
- loud — уровень звука 75;
- x-loud — уровень звука 100 (самый громкий).

speak

Режим синтезатора речи. Используется при озвучивании текстов, записанных в документе.

Значения:

- none — не использовать;
- **normal** — использовать правила произношения указанного языка;
- spell-out — озвучивать побуквенно.

speech-rate

Темп речи при работе синтезатора речи.

Значение задается как количество слов в минуту. Допускаются также определенные значения: x-slow, slow, **medium,** fast, x-fast, faster, slower.

pause-before, pause-after

Величина паузы перед воспроизведением или после воспроизведения содержимого элемента.

Значения:

- численное — в секундах или миллисекундах;
- процентное — по отношению к значению свойства speech-rate (темп речи).

pause

Одновременное задание свойств pause-before и pause-after.

Пример:

```
P.golos { pause: 20ms 40ms }
```

cue-before, cue-after

Задание звуковых файлов, воспроизводимых до и после использования элемента.

Значения:

- **none** — не использовать свойство;
- адрес (URL) файла.

Пример:

```
A.zvuk {cue-before: url(zvuk01.wav); cue-after: url(zvuk02.wav) }
```

cue

Одновременное задание свойств cue-before и cue-after.

Примеры:

- A.zvuk {cue: url(zvuk01.wav) url(zvuk02.wav) } — использованы разные звуковые эффекты;
- A.zvuk {cue: url(zvuk01.wav) } — оба звука одинаковы.

play-during

Задание файла, который определит «звуковой фон» во время воспроизведения озвученного элемента.

Значения:

- none — фоновый звук не воспроизводится;
- **auto** — в качестве звукового фона используется звук родительского элемента.
- адрес (URL) файла;
- mix — звук родительского элемента (свойство play-during) является фоновым;
- repeat — повторение фонового звукового фрагмента, если он оказался короче основного.

Пример:

```
ADDRESS {play-during: url(golos.wav) mix}
```

azimuth

Задание стереофонического эффекта, при котором направление на «источник» звука определяется в горизонтальной плоскости.

Значения задаются в градусах (deg) от -360deg до 360deg или с помощью стандартных значений:

- left-side — аналог величины 270deg;
- far-left — аналог величины 240deg;
- left — аналог величины 320deg;
- center-left — аналог величины 340deg;
- **center** — аналог величины Odeg;
- center-right — аналог величины 20deg;
- right — аналог величины 140deg;
- far-right — аналог величины 60deg;
- right-side — аналог величины 90deg;
- leftwards — дополнительный сдвиг источника на 20 градусов влево;
- rightwards — дополнительный сдвиг источника на 20 градусов вправо;
- behind — дополнительный параметр, смещающий источник звука назад (вычитание 180 градусов).

Примеры:

```
H1 { azimuth: 15deg }
H2 { azimuth: center-right}
P { azimuth: center-right behind }
```

elevation

Задание стереофонического эффекта, при котором направление на «источник» звука определяется в вертикальной плоскости.

Значения задаются в градусах (deg) от -90deg до 90deg или с помощью стандартных значений:

- below — аналог величины -90deg;
- **level** — аналог величины Odeg;
- above — аналог величины 90deg;
- higher — плюс 10 градусов к текущему углу;
- lower — минус 10 градусов от текущего угла.

Пример:

```
P { elevation: above }
```

Пользовательский интерфейс

cursor

Вид указателя мыши, расположенного над текущим элементом.

Значения:

- **auto** — вид указателя определяется броузером;
- default — вид указателя определяется операционной системой;
- crosshair — крестик;
- pointer — указующий перст;
- move — четырехглавая стрелка;
- e-, ne-, nw-, n-, se-, sw-, s-, w-resize — стрелки для перемещения границ. Приставки обозначены по аналогии с частями света (sw — юго-запад);
- text — текстовый указатель;
- wait — песочные часы;
- help — стандартный указатель со знаком вопроса;
- ссылка на нестандартный указатель (URL).

Пример:

```
A { cursor : pointer url(giper.cur) }
```

color, background-color

Существует возможность задания цвета элемента (color) и его фона (background-color) таким, каким обладает соответствующий элемент окна или элемент управления.

Значения:

- activeborder — рамка активного окна;
- activecaption — заголовок активного окна;
- appworkspace — цвет фона документа;
- background — фон рабочего стола;
- buttonface — цвет кнопки;
- buttonhighlight — светлый участок кнопки (боковая грань);
- buttontext — текст кнопки;
- captiontext — цвет текста в строке заголовка;
- graytext — недоступный текст (команда);
- highlight — выбранный элемент управления;
- highlighttext — выделенный текст;
- inactiveborder — рамка невыбранного окна;

- inactivecaption — заголовок невыбранного окна;
- inactivecaptiontext — текст в заголовке невыбранного окна;
- infobackground — фон элемента управления;
- infotext — текст элемента управления;
- menu — фон меню;
- menutext — текст меню;
- scrollbar — полоса прокрутки;
- threeddarkshadow — темная тень трехмерного элемента;
- threed face — поверхность трехмерного элемента;
- threedhighlight — выбранный трехмерный элемент;
- threedlightshadow — светлая тень трехмерного элемента;
- threedshadow — тень трехмерного элемента;
- window — фон окна;
- windowframe — рамка окна;
- windowtext — текст окна.

Пример:

```
P { color: windowtext; background-color: window }
```

Приложение В

Состав прилагаемой дискеты

На дискету записаны указанные в книге файлы Web-страниц и графические файлы. Перед началом просмотра скопируйте содержимое дискеты в одну из папок жесткого диска. Начинайте просмотр с файла Start.htm, так как он содержит ссылки на все другие страницы. Для открытия этого файла достаточно запустить Проводник Windows, открыть нужную папку и дважды щелкнуть на имени файла. Web-страницу можно открыть и в броузере, используя меню Файл или строку Адрес. Рекомендуется также создать ярлык для файла Start.htm на рабочем столе.

Все Web-страницы, использованные в качестве примеров, являются независимыми, и их можно просматривать по отдельности. Ниже приведен перечень этих страниц.

Глава 2. Синтаксис HTML 4

- Заготовка (шаблон) Web-страницы Strukt.htm, поясняющая назначение основных элементов.
- Спецсимволы — файл Spec.htm.
- Управление цветом: файл Color.htm.

Глава 3. Основные элементы HTML версии 4

- Web-страница Text.htm — способы форматирования текста.
- Web-страница Phrase.htm — примеры использования элементов содержания.
- Web-страница Style.htm — примеры стилей.
- Примеры использования классов и стилей находятся в файлах Class.htm и Formats.css.
- Примеры списков можно найти в файле List.htm.
- Примеры таблиц находятся в файле Table.htm.
- Пример страницы с фреймами включает файлы Frame.htm, Frl.htm, Fr2.htm, Fr3.htm и Fr4.htm.
- Другой пример использования фреймов — файлы Main.htm, Framel.htm, M1.htm, Right1.htm и Right2.htm.

Глава 4. Объекты и формы

- Карты — Web-страница Map.htm и графические файлы Map1.gif и Map2.gif.
- Примеры форм — Web-страница Form.htm и файлы рисунков Gif1.gif, Knopkal.gif.

Глава 5. Сценарии

- Два простых сценария использованы в файлах Simple.htm и Simple2.htm.
- Изменение цвета текста — файл Txtcolor.htm.
- Иллюстрация метода setTimeout — файл Clockl.htm.
- Использование кнопок в форме — файл Buttons.htm.
- Замена рисунков — файлы Two-pict.htm и Tb2.htm.
- Пример простейшего сценария — файл Simplest.htm.

Глава 6. Приемы разметки гипертекста

Варианты форматирования таблицы — файл Table.htm (использованы графические файлы fon01.gif, treug1.gif, dragon41.jpg, dragon42.jpg, dragon43.jpg, dragon44.jpg).

Глава 7. Создание графики

Все примеры можно увидеть на Web-странице Graphics.htm.

Приложение Г

Источники информации в Интернете по тематике книги

Сайт или фирма	Адрес	Продукт или содержание
ACME Laboratories	http://www.acme.com	Документация по HTML
Alchemy Mindworks	http://www.mindworkshop.com	Средство для работы с GIF-файлами Gif Construction Set
America Online	http://www.aol.com	Редактор AOLpress
Microsoft	http://www.microsoft.com	Пакет MS Office 2000, Броузер MS Internet Explorer и др.
Netscape	http://home.netscape.com	Броузер Netscape Communicator
North Coast Software	http://www.mv.com/biz/ncs/	Программа PhotoMorph
Фирма SoftQuad	http://www.softquad.com	Редактор HoTMetaL PRO
W3 Consortium	http://www.w3c.org	Документация по HTML, разработка новых версий HTML
Web-страница	http://www.dipart.com	Библиотека рисунков
Издательство «Питер»	http://www.piter-press.ru	Данная книга
Web-страница	http://webcenter.ru/~agonch	Страница автора книги

Алфавитный указатель

A-Z

Access, 172
AOLpress, 142
CGI, 22
CSS2, 210
Drag and Drop, 157
Dynamic HTML, 36
E-mail, 17
Eudora Pro, 17
Excel, 172
FrontPage 2000, 201
FTP, 17
GifConstruction Set, 154, 170
Gopher, 18
High Color, 157
home page, 26
HoTMetaL, 184
HTML, 14
HTTP, 17
IP-адрес, 17
ISO, 37
Java, 36
JavaScript, 36, 115
MIME, 54
Mosaic, 18, 19
MS FrontPage Express, 199
MS Image Composer, 179
MS Photo Editor, 152
MS-DOS, 184
Netscape Composer, 205
Outlook Express, 17
PhotoMorph, 169
Power Point, 172
SGML, 37
Tcl, 115
telnet, 18
True Color, 157
UNIX, 24
URL, 22, 26
VBScript, 36, 115
W3C, 35
Web-сайт, 205
Word, 172
World Wide Web, 18, 19

А-Б

аббревиатура, 64
абзац, 56
аварийная ситуация, 104
автофигура, 172
агент, 22
адрес
  базовый, 42
  Интернета, 17
  относительный, 104
  электронной почты, 17, 134
акроним, 64
альфа-канал, 145, 150
амперсанд, 47
аналоговая линия, 16
анимация, 154
антоним, 186
апплет, 22, 55, 103
атрибут
  выравнивания, 41,57
  дополнительный, 52
  кодировки, 55
  комментария к таблице, 78
  объекта, 97
  определение, 21
  размещение, 44
  события, 55
  содержания, 39
  стандартный, 53, 65
  типа данных, 39
  фона, 40, 50
  языка, 54
базовый адрес, 42
базовый путь, 103
базовый размер шрифта, 60
баннер, 96
бегущая строка, 94, 129
Блокнот, 29
броузер, 18, 22
буфер обмена, 137, 167

В

видеоролик, 55, 167, 170
визуальный эффект, 129

виртуальная книга, 87
вложение элементов, 43
вложенная таблица, 137
вложенный список, 71
вращающееся изображение, 162
время, 128
выделение текста, 63
выравнивание, 41, 97

Г-Д

гамма-коррекция, 23
гиперссылка, 26
  в виде рисунка, 134
  обратная, 75
  определение, 21
  последняя просмотренная, 40
  просмотренная, 40
  прямая, 75
  создание, 73
  структура, 75
  тин, 75
  цвет текста, 40
гипертекст, 19
глобальная переменная, 119
горизонтальная линия, 138
графика, 148
графический редактор, 144
графический файл, 98
графическое меню, 102
громкость, 224
группа
  колонок, 83
  переключателей, 108
  полей, ИЗ
группировка строк таблицы, 82
дата, 128
динамический эффект, 127
документ
  раскладки, 84, 204
  содержания, 84, 204
домашняя страница, 26
домен, 22
дополнительные атрибуты, 52

Ж—З

журнал, 32
заголовок страницы, 38, 52
загрузка, 22, 130
закладка, 32
замена рисунка, 122
запрос, 106
зачеркивание, 59
звук
  выбор файла, 92
  громкость, 92, 224
  создание, 92
  число повторений, 92
звуковое сопровождение, 53
звуковой файл, 140, 225
звуковой фон, 225

И—К

имя
  доменное, 22
  фрейма, 85, 90
  функции в сценарии, 119
индекс
  верхний, 59
  нижний, 59
интервал
  между буквами, 213
  между словами, 213
кавычки, 63
кадр, 163
карта, 75, 99
каскадная таблица стилей, 55, 65, 67, 205
кириллица, 47, 194
клавиатура, 64
класс, 66
клиент, 18
кодирование, 45
кодировка, 55
  ISO Latin 1, 45
  Unicode 2.0, 45
  Windows, 45
  КОИ-8, 45
  мнемоническая символов, 46
комментарии, 40, 93, 191
композиция, 179
компоновка
  Web-страницы, 141
  кадров движущегося изображения, 164
  сайта, 141
  связей, 142
конечный тег, 37
конструирование страниц, 132
контейнер элементов, 68
координаты границ элемента, 218
круг, 99
курсив, 59
кэш, 27, 29, 85, 151

Л—М

линия
  аналоговая, 16
  горизонтальная, 138
  связи, 16
  телефонная, 16
  цифровая, 16
логический оператор, 120
логическое выражение, 120
локальная переменная, 119
маркер списка, 215
мастер Web-страниц, 196
масштаб композиции, 181
масштабирование рисунка, 98
математическая функция, 118
межбуквенный интервал, 213

меню
графическое, 102
создание, 111
страницы, 102
метка, 42, 73, 74
мигание текста, 93
мини-Web, 142
мнемонический код, 47
многоугольник, 99
модем, 16
мозаичный рисунок, 135
моноширинный шрифт, 30
мультимедиа, 140
мягкий перенос, 61

Н-О

надпись, 108
надстройка, 94
направление текста, 214
наследование, 64, 210
начальный тег, 37
область
круг, 99
многоугольник, 99
прямоугольник, 99
обтекание, 221
объединение ячеек таблицы, 136
объект
атрибуты, 97
графический, 172
определение, 97
относительный адрес, 104
элементы, 103
объем, 174
окно броузера, 91
округление, 123
окружность, 162
оператор
логический, 120
присваивания, 118
сравнения, 120
управляющий, 120
условный, 120
цикла, 120
орфография, 185
ось рисунка, 163
относительный адрес объекта, 104
очистка формы, 108

П—Р

палитра, 148
панель инструментов, 175
папка
вложенная, 73
Избранное, 29
текущая, 73
параметр
альфа-канала, 145
идентификатор объекта, 105
ссылка, 105
строка, 105
цикла, 121
пароль, 108
переключатель, 106, 108
перекрытие объектов, 222
переменная
глобальная, 119
локальная, 119
определение, 119
создание, 119
перенос, 61
печать, 187
побуквенный вывод надписи, 129
подсветка символов, 96
подчеркивание, 59
поле ввода, 106
полоса прокрутки, 85
полужирный шрифт, 58
получение фокуса, 56
пользовательский интерфейс, 227
потеря фокуса, 56
правила вложения, 43
предварительный просмотр, 202
присваивание, 118
пробел, 214
провайдер, 142
проверка орфографии, 185
программный код, 115
прозрачность, 150, 152, 157, 174
пропорциональный шрифт, 30
протокол
telnet, 18
передачи гипертекста, 17
передачи файлов, 17
профиль, 53
прямоугольник, 99
размер
изображения, 182
объекта, 221
шрифта, 30, 60, 211
разметка гипертекста, 132
размытость, 149
разрыв строки, 57
рамка
в форме, 114
объекта, 97
страницы, 144
таблицы, 76, 78, 132
фрейма, 85
раскрывающийся список, 194
расширение, 22
редактор
гипертекста, 142, 184, 199, 205
графический, 144, 148, 151, 152, 162, 179
изображений, 152
каскадных таблиц стилей, 194
текстовый, 184

рекурсивный вызов функции, 129
рисунок
  в качестве маркера списка, 215
  вывод с чередованием строк, 149
  замена, 122
  координаты, 102
  копирование, 98
  масштабирование, 98
  мозаичный, 135
  подгонка фрагментов, 135
  размещение, 104
  фона, 40

С—Т

сайт, 17,22
свойства
  границ элементов, 217
  единицы измерения, 210
  мультимедиа, 224
  пользовательского интерфейса, 227
  списков, 214
  таблиц, 215
  таблиц стилей, 210
  фона и цвета, 222
  форматирования текста, 212
  шрифтов, 210
свойство
  azimuth, 226
  background, 224
  background-attachment, 223
  background-color, 222, 227
  background-image, 222
  background-position, 223
  background-repeat, 223
  border, 216, 221
  border-bottom, 220
  border-bottom-color, 219
  border-bottom-style, 220
  border-bottom-width, 219
  border-collapse, 216
  border-color, 219
  border-left, 220
  border-left-color, 219
  border-left-style, 220
  border-left-width, 219
  border-right, 220
  border-right-color, 219
  border-right-style, 220
  border-right-width, 219
  border-style, 220
  border-top, 220
  border-top-color, 219
  border-top-style, 220
  border-top-width, 219
  border-width, 219
  bottom, 218
  clear, 221
  clip, 221
  color, 222, 227
  column-span, 216
свойство *(продолжение)*
  cue, 225
  cue-after, 225
  cue-before, 225
  cursor, 227
  direction, 214
  display, 215
  elevation, 226
  float, 221
  font, 211
  font-family, 210
  font-size, 211
  font-style, 211
  font-variant, 211
  font-weight, 211
  height, 217
  left, 218
  letter-spacing, 213
  line-height, 213
  list-style, 215
  list-style-image, 215
  list-style-position, 215
  list-style-type, 214
  margin, 218
  margin-bottom, 218
  margin-left, 218
  margin-right, 218
  margin-top, 218
  max-height, 217
  max-width, 217
  min-height, 217
  rain-width, 217
  overflow, 221
  padding, 219
  padding-bottom, 218
  padding-left, 218
  padding-right, 218
  padding-top, 218
  pause, 225
  pause-after, 224
  pause-before, 224
  play-during, 225
  position, 218
  right, 218
  row-span, 216
  speak, 224
  speech-rate, 224
  table-layout, 217
  text-align, 212
  text-decoration, 212
  text-indent, 212
  text-shadow, 212
  text-transform, 214
  top, 218
  vertical-align, 216
  visibility, 222
  volume, 224
  white-space, 214
  width, 217
  word-spacing, 213
  z-index, 222

сглаживание шрифтов, 133
сервер, 18
сетка, 162
сеть
  глобальная, 15
  компьютерная, 15
  локальная, 15
сжатие файла, 149
синоним, 186
синтаксис
  HTML, 43
  JavaScript, 118
синтезатор речи, 224
скрипт, 22
скрытая страница, 142
скрытый элемент, 194
смена рисунков, 129
событие, 55
создание
  графики, 148
  группы колонок, 83
  группы переключателей, 108
  группы полей, 113
  заголовка таблицы, 78
  кадров движущегося изображения, 163
  кнопки, 108
  меню, 102, 111
  переключателя, 108
  переменной, 119
  полос прокрутки, 85
  поля ввода, 108
  пункта списка, 111
  сложной формы, 107
  списка, 111
  ссылки на графический файл, 98
  строки таблицы, 79
  таблицы, 76, 215
  трехмерной вращающейся фигуры, 165
  флажка, 108
  фона, 150
  формы, 106, 110, 130, 193
  фреймов, 84
  функции, 118
  элемента таблицы, 215
  элемента управления, 108
сообщение, 104
сочетания клавиш, 106
специальные символы, 46, 190
спецификация GIF89a, 149
список
  адресов, 104
  вложенный, 71
  ненумерованный, 69, 93
  нумерованный, 69
  определение, 69
  разновидности, 69
  раскрывающийся, 194
  с определениями, 71, 191
  создание, 111
спрайт, 180
ссылка
  абсолютная, 42
  относительная, 42
стандартный атрибут, 53
стереофонический эффект, 226
стиль, 38, 63
  создание, 65
  таблицы, 139
  текста, 187
  универсальный, 66
  шрифта, 211
стихотворный текст, 139
строковая функция, 118
строфа, 139
схема
  доступа, 42, 74
  сайта, 205
  связей, 141
сценарий, 22, 115
таблица
  вложенная, 137
  группировка строк, 82
  из одной ячейки, 134
  каскадная стилей, 65
  назначение, 76
  объединение ячеек, 136
  параметры, 192
  рамка, 132
  свойства, 76, 215
  создание, 76
  стилей, 64
  строки, 79
  структура, 76
тег
  конечный, 37, 44
  начальный, 37, 44
  определение, 21, 44
тезаурус, 186
текст
  выделение, 63
  мигание, 93
  свойства, 65
  сообщения, 104
  стихотворный, 139
  телетайпа, 59
  форматирование, 56
текстовая строка, 55
текстовый редактор, 184
текстовый режим, 39
текстура, 150
телетайп, 59
телефонная линия, 16
тело
  оператора, 120
  функции, 119, 120
  цикла, 121
темп речи, 224
тенденции и мода, 132
тень, 174, 212

тип
MIME, 104
гиперссылки, 75
данных, 48
объекта, 104
переменной, 119
топология связей, 141
трафарет, 180

У—Ф

угловые скобки, 47
удаленный ресурс, 101
указатель
мыши, 55, 131, 227
на файл графики, 98
универсальный ресурса, 22
унарная операция, 120
универсальный класс, 66
универсальный стиль, 66
управление формами, 129
управляющий оператор, 120
уровни заголовков, 41
условный оператор, 120
файл
графический, 98
звуковой, 140,225
мультимедиа, 140
сжатый, 149
фигурные скобки, 131
флажок, 106, 108
фокус
получение, 56
потеря, 56
фон
рисунок, 40
создание, 150
страницы, 40
текстура, 150
цвет, 40, 222
элемента, 224
фоновый рисунок, 150
форма, 28, 75, 106
очистка, 108
подтверждение ввода данных, 108
сложная, 107
создание, 106, 130, 193
управление, 129
формат
AVI, 140, 170
CMP, 158
GIF, 55, 148
HTML, 55
JPEG, 149
JPG, 55, 169
PNG, 55, 149
PostScript, 55
WAV, 140
форматирование
линии, 138
таблицы, 217
текста, 56, 187, 212
фотография, 146
фотоморфизм, 168
фрейм, 75
определение, 21
планировка, 43
создание, 84, 204
функция
вызов, 119
математическая, 118
округления, 123
определение, 118
рекурсивный вызов, 129
строковая, 118

Ц—Ч

цвет, 49
гиперссылки, 40
горизонтальной линии, 48
линии, 41
последней просмотренной гиперссылки, 40
просмотренной гиперссылки, 40
рамки, 97,219
текста, 140
тени, 134
фона, 40, 48, 134, 222
шрифта, 48, 134
элемента, 131
цикл, 120
прерывание, 121
с параметрами, 121
условие выполнения, 121
цитата, 62, 191
цифровая линия, 16
чередование строк, 149, 157
числовой массив, 120

Ш-Я

шрифт, 60
моноширинный, 30
по умолчанию, 202
полужирный, 58
пропорциональный, 30
сглаживание, 133
увеличение, 58
уменьшение, 58
электронная почта, 74, 107, 201
элемент
A, 42, 48, 73, 168
ABBR, 64
ADDRESS, 63, 168
APPLET, 103, 104
AREA, 99
B, 58
BANNER, 96

элемент *(продолжение)*
BASE, 42
BASEFONT, 60
BDO, 61
BGSOUND, 53, 92
BIG, 58
BLINK, 39, 93
BLOCKQUOTE, 62, 207
BODY, 40
BR, 57, 139
BUTTON, 113
CAPTION, 78
CENTER, 58
CITE, 63
CODE, 63
COL, 83
COLGROUP, 83
COMMENT, 93
DD, 71
DEL, 59
DFN, 62
DIR, 93
DIV, 68
DL 71
EM, 62
EMBED, 94
FIELDSET, 113
FONT, 49, 60
FORM, 107
FRAME, 48
FRAMESET, 85
HEAD, 38, 52
Hn, 41
HPn, 96
HR, 41,51, 138
HTML, 38
I, 59
IFRAME, 91, 104
IMG, 48, 98, 168
INPUT, 108
INS, 59
ISINDEX, 106
KBD, 64
LEGEND, 114
LI, 69
LINK, 52, 76
LISTING, 93
MAP, 99
MARQUEE, 94
MENU, 93
META, 39, 53, 198
NOBR, 57
NOEMBED, 94
NOFRAMES, 87
NOSCRIPT, 118
OL, 69
OPTION, 111
элемент *(продолжение)*
P, 56, 168
PARAM, 105
PLAINTEXT, 93
PRE, 58, 93, 94, 214
Q, 63
SAMP, 63
SCRIPT, 115
SELECT, 111
SMALL, 58
STRIKE, 59
STRONG, 63
STYLE, 38, 52
SUB, 59
SUP, 59
TABLE, 76, 136
TBODY, 82
TD, 80, 136
TEXTAREA, 112
TFOOT, 82
TH, 79
THEAD, 82
TITLE, 38, 52
TR, 79
TT, 59, 207
U, 59
UL, 69, 93
VAR, 64
XMP, 93
элемент управления
ActiveX, 194
меню, 106
переключатель, 106
поле ввода, 106, 107
флажок, 106
элементы
абзаца, 56
базовый адрес, 42
вложение, 38, 43
горизонтальная линия, 41
заголовок, 38, 41
меток, 42
нестандартные, 92
объектов, 103
определение, 21,44
переход на новую строку, 57
расположение, 43
служебные, 39
содержания, 62
стиль, 38
тело страницы, 40
устаревшие, 92
формы, 130
шрифт, 49
эллипс, 163
эффект
анимации, 161
вертикальных полос, 152

эффект *(продолжение)*
движения, 154
изменения текстуры, 181
имитации материала, 181
неонового света, 181
освещенности, 165
плавных переходов цвета, 152
постепенного исчезновения изображения, 170
прозрачности, 153
размытия, 181
следа от движения, 160
эффект *(продолжение)*
стереофонический, 226
тени, 165
чередования строк, 157
язык
Java, 36
JavaScript, 36, 118
VBScript, 36
программирования, 15
разметки гипертекста, 14
ярлык, 28
ячейка таблицы, 139

Алексей Гончаров

С САМОУЧИТЕЛЬ

HTML

Книга известного автора Алексея Гончарова посвящена HTML — языку гипертекстовой разметки документов, позволяющему создавать публикации в Интернете. Вы познакомитесь со спецификацией языка HTML 4, узнаете о технике подготовки данных для распространения в Интернете, научитесь применять графику на Web-страницах и решать другие задачи, стоящие пере создателями HTML-документов.

Прочитав эту книгу, вы не только научитесь самостоятельно создавать HTML-документы, но и сможете воспользоваться идеям: и готовыми решениями, которые предлагает автор.

Рекомендуем книги ИД «Питер»:

На прилагаемой дискете вы найдете файлы Web-страниц, работа с которыми рассматривается *в* книге

ISBN 5-272-00072-2

9 785272 000729

Посетите наш Web-магазин: http://www.piter-press.ru

Серия: Самоучитель

Для начинающих

ПИТЕР®
WWW.PITER-PRESS.RU