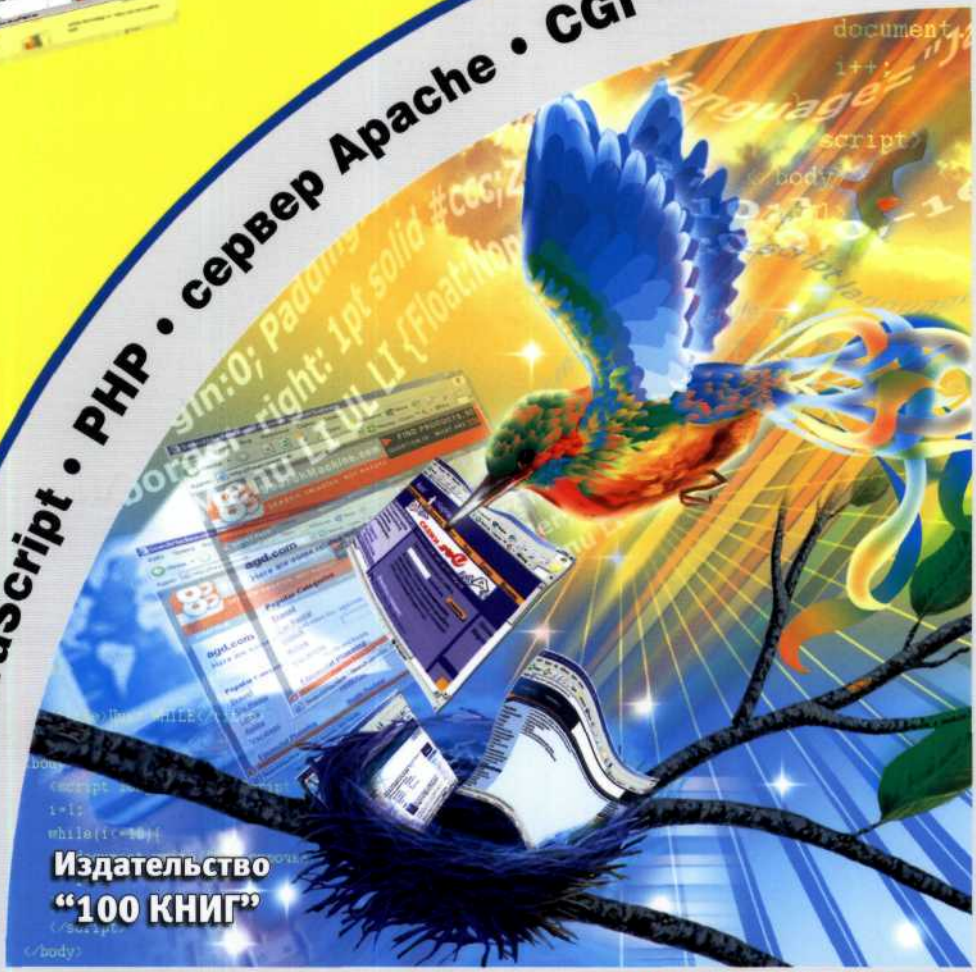
Web-мастеринг

Б. Артанов

без посторонней помощи

HTML • JavaScript • PHP • сервер Apache • CGI • Front Page



Издательство
"100 КНИГ"

All you need to be happy

Б. Артанов

WEB-мастеринг

без посторонней помощи

Учебное пособие

УДК 004.738.5(075.4)
ББК 32.973.202я78-1
А86

Серия: «Без посторонней помощи»

Артанов, Борис.

А86 Web-мастеринг без посторонней помощи : учеб. пособие / Б. Артанов. — М. : 100 книг, 2006. — 336 с. : ил. — (Серия: «Без посторонней помощи»). —
ISBN 5-89392-147-X.

Агентство CIP РГБ

Вы убежденный практик и желаете быстро стать Web-мастером? Тогда эта книга – для вас. Она научит вас технике Web-мастеринга. Начав с создания простейшего Web-сайта, вы шаг за шагом построите собственный Web-сайт, обладающий всеми качествами профессионального Интернет-портала.

Как украсить Web-страницы привлекательной графикой, сделать их интерактивными, дружественными к пользователю, как генерировать Web-страницы «на лету», как заставить Web-сервер решать задачи по запросам пользователя? Ответы на все эти вопросы вы найдете в этой книге.

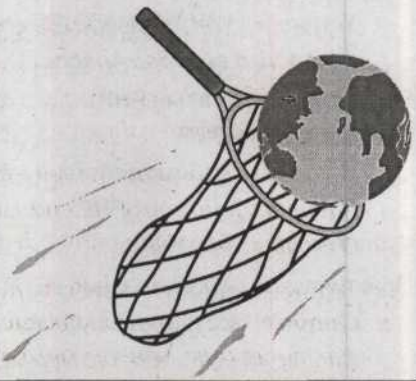
После прочтения данного самоучителя языки HTML, JavaScript и PHP станут для вас простым и понятным инструментом «строителя» Web-сайта. Также вы научитесь размещать свои Web-сайты в Интернете и обеспечивать приток большого количества посетителей, что крайне важно для сайтов коммерческого или делового назначения.

Краткое содержание

Содержание

глава 1

Основы создания Web-страниц



Современный мир – это мир коммуникаций. Никого не удивляет возможность говорить по телефону с человеком, находящимся на другом конце Земли, причем зачастую дешевле, чем с соседним городом, ежедневно получать сообщения от людей, находящихся на разных концах нашей планеты, в любой момент узнавать свежие новости от всех крупнейших новостных агентств мира. Но так было далеко не всегда, всемирная компьютерная сеть Интернет, доступная ныне в самых дальних уголках земного шара и позволяющая оперативно обмениваться информацией людям, вне зависимости от расстояний их разделяющих, появилась всего лишь несколько десятков лет назад.

Как же появился Интернет? Какие основные этапы прошел он на своем пути? На эти вопросы мы дадим ответ в этой главе. Кроме того, мы рассмотрим современное строение сети Интернет, основные ее возможности. Познакомимся поближе с самой интересной частью Интернета – Всемирной паутиной, или просто World Wide Web.

Развитие мировых информационных коммуникаций

Развитие человеческого общества можно проследить по средствам, которые они используют для общения. Самый древний способ общения – прямой контакт. Этот способ подходит, когда все люди, с которыми вы имеете дело, живут в непосредственной близости от вас. Так и было на заре человечества, когда людей было мало, они селились отдельными общинами, никак не связанными друг с другом.

Но время шло, человечество разрасталось, и все чаще возникала необходимость деловых, военных и дружеских контактов между отдельными общинами. Рядовой охотник мог позволить себе просто сходить в соседнее поселение поболтать или поменять пару шкур на новый наконечник для копья. А человек важный, например вождь племени, так поступить не мог, и ему, чтобы поддерживать отношения с вождем другого племени, приходилось прибегать к услугам посредника – гонца. Вождь передавал гонцу послание, тот его запоминал и затем пересказывал его другому.

У такой схемы, разумеется, была масса недостатков. Гонец мог случайно или умышленно исказить суть послания, его могли просто убить по дороге. Да и позволить себе услуги гонца, которого нужно кормить, содержать, следить за тем, чтобы его не склонили к предательству, могли немногие.

Если расстояния между стоянками соседних племен были невелики, для передачи сообщений от одного селения к другому могли использоваться такие средства, как почтовые барабаны или сигнальные костры.

Когда письменность получила повсеместное распространение, общины переросли в первые государства. Целые материки пересекли регулярные торговые пути, и было положено начало почтовому сообщению между людьми. Сначала письма передавались с торговыми караванами, а затем появились и специальные почтовые службы. Таким образом, люди могли общаться, даже находясь на расстоянии многих сотен километров друг от друга. Конечно, оперативность такого общения оставляла желать лучшего, письмо могло странствовать по дорогам и морям не один месяц, но и жизнь тогда была неторопливая. Решения принимались после длительных размышлений, и редко что внезапно менялось в мире.

Так продолжалось очень долго, вплоть до наступления нового времени, эпохи больших скоростей, быстро набирающего обороты прогресса и все нарастающей конкуренции во всех сферах жизни. Старушка почта явно не поспевала за временем, но ей на подмогу пришли дети прогресса – телеграф, а затем и телефон (Рис. 1.1).

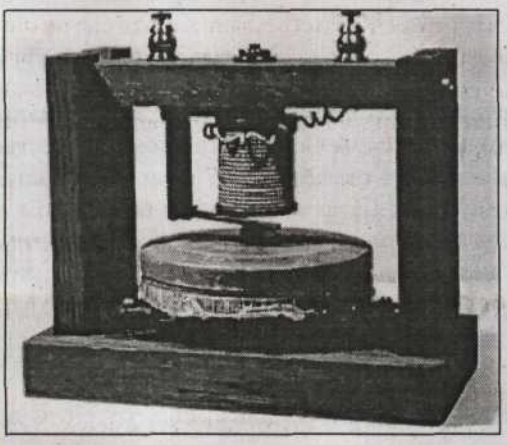


Рис. 1.1. *Самый первый телефонный аппарат*

Значимой вехой в развитии мировых коммуникаций явилась прокладка Атлантического телефонного кабеля, соединившего между собой два континента: Америку и Евразию. Это позволило людям, находящимся на разных концах Земли, общаться, решать вопросы, обсуждать торговые сделки напрямую. Паутина проводов опутала планету.

В середине двадцатого века появились первые компьютеры (Рис. 1.2). Вначале они были медленными, занимали очень много места и стоили астрономических денег. Но постепено они становились все быстрее и меньше, их количество тоже быстро увеличивалось. И по мере того, как с помощью компьютеров выполнялось все больше дел, все острее вставала проблема обмена информацией между отдельными компьютерами. Рано или поздно должна была начаться следующая эпоха информационных коммуникаций – цифровая.

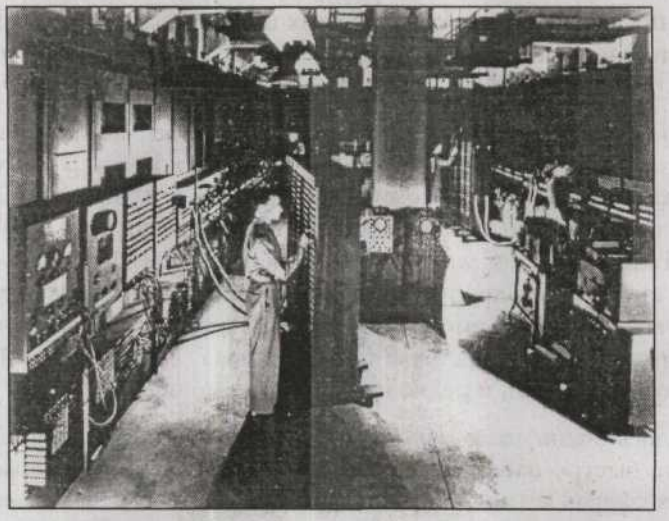


***Рис. 1.2.** ENIAC, первый электронный компьютер*

Зарождение сети Интернет

Событием, подтолкнувшим развитие цифровых коммуникаций и всемирной сети Интернет, стал запуск Советским Союзом первого искусственного спутника Земли (Рис. 1.3), состоявшийся 4 октября 1957 года. Был разгар холодной войны, и военные чины Соединенных Штатов Америки были этим событием очень напуганы, поскольку вслед за первым спутником, который тихо пищал «бип-бип» в окружающее пространство, могли последовать и другие, куда менее безобидные. США срочно нужно было делать ответный шаг.

Таким шагом стало создание в 1958 году Агентства Перспективных Разработок (Advanced Research Projects Agency – ARPA). Разумеется, это агентство создавалось отнюдь не для разработки компьютерных сетей, а для развития военных технологий.

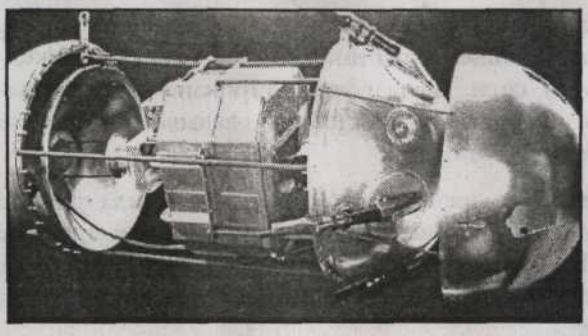


Рис. 1.3. Спутник, подтолкнувший создание сети Интернет

В 1962 году отделение компьютерных разработок в ARPA возглавляет доктор Ликлайдер (J.C.R. Licklider), Рис. 1.4. С собой он приносит в агентство идею «межгалактической сети» (intergalactic network), позволяющей людям, находящимся в любой точке земного шара получать доступ к программам и данным, располагающимся в любом другом месте Земли.

Рис. 1.4. Доктор Ликлайдер

Эта идея очень скоро завоевала свое право на жизнь. Дело в том, что исследованиями и разработками для агентства занимались многие предприятия и институты, находящиеся в разных частях США, и, для того чтобы они могли согласованно заниматься решением одних и тех же задач, необходимо было обеспечить эффективное координирование их работы, обмен рабочими данными и наработками. При этом, система должна была сохранить свою работоспособность даже в случае выхода ее части из строя, например в результате атомной бомбардировки.

Решением проблемы должна была стать сеть, объединяющая компьютеры, вне зависимости от их типа, по всем Соединенным Штатам, и прообразом ее стала «межгалактическая сеть» Ликлайдера. Сеть, пока еще не имевшая названия, предусматривала объединение между собой основных компьютеров исследовательских лабораторий, называемых также узлами сети. Узлы, в свою очередь, могли

объединять под своим контролем компьютеры более низкого ранга. При этом с каждого компьютера общей сети должен был быть возможен доступ к любому другому компьютеру сети. Была сформулирована и основная концепция сети, кажущаяся сейчас очевидной: сеть должна связывать между собой не компьютеры, а людей, за ними работающих.

И вот, в 1969 году, сеть, названная ARPANET, получила свое первое физическое воплощение – были соединены между собой четыре компьютера, находящиеся в разных частях США. На Рис. 1.5 представлена копия сделанной от руки схемы самой первой сети ARPANET – и кто бы мог тогда подумать, во что все это вскорости превратиться!

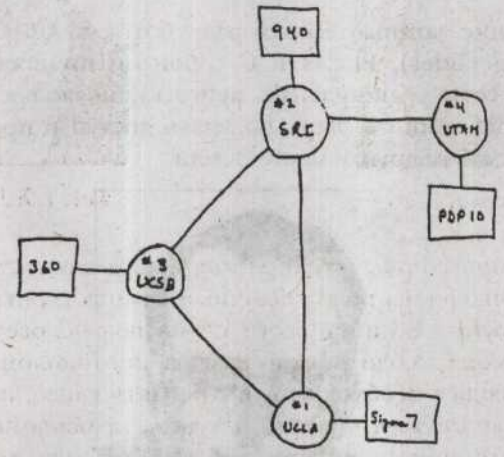


Рис. 1.5. Схема ARPANET в момент запуска

Скорость передачи данных оставляла желать лучшего, но, тем не менее, сеть заработала. Начало было положено, хотя никто не мог тогда и предположить, во что в результате выльется проект, первоначально служивший попыткой решить некоторые проблемы военной машины США.

Этапы развития сети Интернет

Уже в начале 1971 года сеть состояла из 14 узлов и скорость ее роста все увеличивалась. Появлялись все новые сетевые технологии: через сеть начали обмениваться сообщениями электронной почты, ее изобрел Рэй Томлинсон (Ray Tomlinson). Был разработан механизм передачи файлов между компьютерами сети – FTP (File Transfer Protocol – протокол передачи файлов).

Появление протокола TCP/IP

Вскоре, кроме ARPANET, появилось и несколько других компьютерных сетей. Перед разработчиками встала проблема обмена информацией с ними. Для ее реше-

ния Боб Канн (Bob Kahn), Рис. 1.6 и Винтон Сёрф (Vint Cerf), Рис. 1.7. разработали универсальный механизм для межсетевых соединений. Его назвали TCP (Transmission Control Protocol – протокол управления передачей данных).

Рис. 1.6. Боб Канн

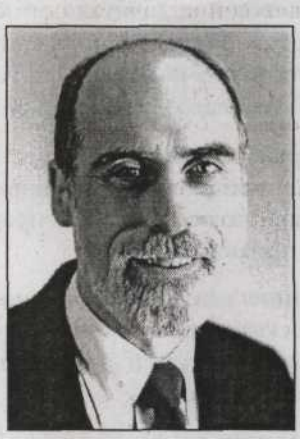


Рис. 1.7. Винтон Сёрф

Протоколами, в компьютерных технологиях, называют стандарты поведения, которых должны придерживаться различные компьютерные системы при взаимодействии друг с другом. Если провести аналогию с человеческим общением, то язык, на котором говорят люди, можно назвать протоколом общения. Чтобы понимать друг друга, людям необходимо употреблять слова, известные всем участникам беседы, и пользоваться общими для всех правилами языка. Так же дело обстоит и с компьютерной техникой: чтобы «общаться» друг с другом, два устройства должны использовать один и тот же общий протокол или, другими словами, «говорить на одном языке».

В 1983 году протокол TCP/IP (приставка IP расшифровывается как Internet Protocol, протокол Интернет) был утвержден в качестве официального стандарта для передачи данных в сети Интернет. В это же время сеть ARPANET разделяется на две части: публичную, за которой остается название ARPANET, и закрытую, названную MILNET (от Military Network – Военная Сеть), предназначенную для военных нужд. Военные, по сути, отпускают Интернет на волю, оставив себе небольшой кусочек. Примерно с этого момента начинается коммерциализация Интернета. Доступ в него начинают предоставлять частные компании, их можно назвать первыми провайдерами Интернета.

Система DNS

В этом же году, чтобы упростить ориентирование во все более разрастающемся Интернете, была разработана система DNS (Domain Name System – система именования доменов сети). Дело в том, что каждому компьютеру или компьютерной сети, подключенной к Интернету, назначается уникальная последовательность цифр, называемая IP-адресом IP-адрес состоит из четырех чисел, от 0 до 255 каждое, например 198.105.232.001. Зная IP-адрес, пользователь одного компьютера с

легкостью находит другой компьютер в Интернете, и может к нему подключиться, если у него есть на это соответствующие права. Все просто, когда вам нужно получать доступ к одному-двум компьютерам, но если их количество переваливает за десяток или даже за сотню, а, тем более, если вам необходимо сообщать определенный IP-адрес многим людям, ситуация становится поистине кошмарной. Избежать этого помогает система имен DNS. Она позволяет заменять цифровые IP-адреса на благозвучные буквенные, например: «**microsoft.com**» или «**yandex.ru**».

Как же работает DNS? Все пространство Интернета делится на несколько групп, называемых «доменными зонами». Эти зоны называются доменами первого уровня. Разделение по зонам может проводиться как по географическому, так и по тематическому признаку.

Географическая доменная зона определяет расположение компьютера в том или ином государстве. Вот несколько примеров географических доменов первого уровня: **ru** – Россия, **fr** – Франция, **uk** – Великобритания, **jp** – Япония, **su** – бывший Советский Союз.

Тематические доменные зоны группируют компьютеры по информации, содержащейся на них, либо по типу организаций, ими владеющих, вне зависимости от их географического расположения. Два компьютера, зарегистрированные в одной тематической доменной зоне, могут находиться в противоположных концах земного шара. Вот примеры тематических доменных зон: **com** – коммерческое предприятие, **net** – что-то связанное с сетевыми технологиями, **edu** – образовательное учреждение, **info** – информационный проект, **gov** – государственное учреждение, **biz** – бизнес-проект, **mil** – военная организация.

Несмотря на обилие доменных зон, далеко не все из них пользуются большой популярностью. Основная часть компьютеров в Интернете зарегистрирована в доменных зонах **com** и **net**. Некоторые доменные зоны используются и вовсе не по прямому назначению. Например, островное государство Тувалу стало обладателем географической доменной зоны **tv**, которую сейчас облюбовали организации, так или иначе связанные с телевидением: телеканалы, производители бытовой техники, киноделы, рекламщики и прочие...

Каждая доменная зона делится на поддомены, или домены второго уровня, и каждому из этих поддоменов присваивается свое имя, например совпадающее с названием организации, владеющей доменом. Это имя приписывается к имени домена верхнего уровня слева, в виде суффикса, и отделяется точкой.

Например, в имени **microsoft.com** строка **com** означает доменную зону, а суффикс **microsoft** – имя домена второго уровня. Как нетрудно догадаться, по этому адресу находится сеть, принадлежащая корпорации Microsoft. Однако сеть корпорации Microsoft весьма велика, поэтому каждый домен второго уровня, в свою очередь, может делиться еще на несколько подподдоменов, или доменов третьего уровня. Это записывается так – **mail.microsoft.com**. В этом примере **mail** – это суффикс домена третьего уровня. Такое деление может продолжаться до бесконечности, но обычно ограничивается доменами третьего-четвертого уровня.

Общее руководство и контроль над доменными зонами, осуществляет организация ICANN (The Internet Corporation for Assigned Names and Number – Интернет-ассоциация по выдаче имен и чисел). Она передает полномочия на выдачу адресов в той или иной доменной зоне другим организациям и следит за соблюдением основных правил. Организации, уполномоченные выдавать доменные адреса в той или иной доменной зоне, торгуют доменными адресами второго уровня. То есть, если кто-то хочет, чтобы у его компьютера в Интернет был адрес **vasya-pupkin.com**, он должен обратиться к организации, выдающей доменные имена в зоне **com**. Затем попросить зарегистрировать в ней домен второго уровня vasya-pupkin, предоставить IP-адрес своего компьютера в Сети и, разумеется, уплатить некоторую сумму денег. В результате, компьютер Васи в Интернете можно будет отыскать не только по малопонятному набору цифр IP-адреса, но и по звучному текстовому адресу. При желании, одному IP-адресу можно сопоставить даже несколько доменных имен, например **vasya-pupkin.com** и **vasiliy.ru**. Адреса в Российской доменной зоне **ru** выдает организации РосНИИРОС, Российский НИИ развития общественных сетей.

Интернет – международная компьютерная сеть

Тем временем, компьютеры становились все быстрее, меньше и дешевле, вскоре уже достаточно небольшие организации могли позволить себе приобрести в пользование полноценный компьютер. Переломным моментом, как для компьютерной индустрии, так и для развития Интернета стало появление и стремительное распространение персональных компьютеров (Рис. 1.8), которые тоже вскоре получили возможность подключаться к компьютерным сетям, в том числе и к ARPANET. В результате, количество пользователей Сети начало увеличиваться лавинообразно. К Сети начали подключаться и компьютеры из других стран, она становилась международной. Примерно к этому времени относится и рождение термина «Интернет» (Internet – International Network, Международная Сеть).

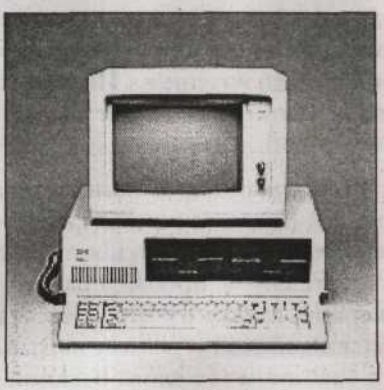


Рис. 1.8. Один из первых персональных компьютеров IBM PC

В 1990 году ARPANET официально была закрыта, так как с успехом выполнила свою миссию. За двадцать лет существования сети количество компьютеров в ней увеличилась с 14 до трехсот тысяч. К 1990 году к Интернету подключились такие

страны, как Аргентина, Австрия, Бельгия, Бразилия, Чили, Греция, Индия, Ирландия, Южная Корея, Испания и Швейцария.

Можно сказать, что именно в это время было закончено формирование сети Интернет в том виде, в каком мы видим ее сейчас. И, хотя с тех пор Интернет очень сильно разросся, скорости передачи данных многократно выросли, но внутреннее устройство Сети изменений почти не претерпело. Это стало одной из причин того, что в 90-х годах была разработана международная сеть нового поколения, Интернет-2. Целью создания этой сети был обход основных ограничений, которые накладывают на развитие Интернета технологии и стандарты, положенные в его основу. На данный момент, Интернет-2 охватывает собой только небольшое количество крупнейших научных исследовательских центров и учебных заведений США. Но вполне возможно, что уже через несколько лет Интернет-2 сможет на равных конкурировать по популярности со своим прародителем.

Строение сети Интернет

Современный Интернет представляет собой сложнейшую систему из тысяч компьютерных сетей, объединенных между собой. Состоит эта система из двух основных элементов: узлов сети Интернет и соединяющих их информационных магистралей. Узлом Интернета называют любое устройство, имеющее свой IP-адрес и подключенное к Сети.

Несмотря на кажущуюся мешанину межкомпьютерных соединений и отсутствие централизованного руководства, Интернет имеет определенную иерархическую структуру.

В самом низу иерархии находится многочисленная армия конечных пользователей. Часто не имеющие даже постоянного IP-адреса подключаются к Интернету по низкоскоростным каналам. Тем не менее, пользователи являются одними из основных потребителей услуг Сети и главными «спонсорами» коммерческой части Интернета. Причем на одного «физического» пользователя, т. е. реального человека, пользующегося услугами Сети, может приходиться несколько пользователей «логических», т. е. различных подключений к Интернету. Так, кроме компьютера, возможность подключения к Интернету может иметь мобильный телефон, карманный компьютер, бытовая техника, автомобиль и даже кондиционер.

Конечные пользователи подключаются к компьютерам Интернет-провайдера, или, как их еще называют, ISP (Internet Service Provider – провайдер Интернет). ISP – это организация, основная деятельность которой связана с предоставлением услуг Интернета пользователям. У провайдера есть своя компьютерная сеть, размеры которой могут варьироваться от сотен десятков узлов в нескольких городах до многих тысяч, раскиданных по целому континенту. Эта сеть называется магистральной сетью, или бэкбоном (от слова backbone – стержень, магистраль). Сети отдельных провайдеров соединяются между собой и другими сетями. Среди ISP есть «монстры», которые обеспечивают соединение между собой сетей различных стран и континентов, являясь своего рода «провайдерами для провайдеров». Весь этот конгломерат компьютерных сетей и образует то, что называется Интернетом (Рис. 1.9).

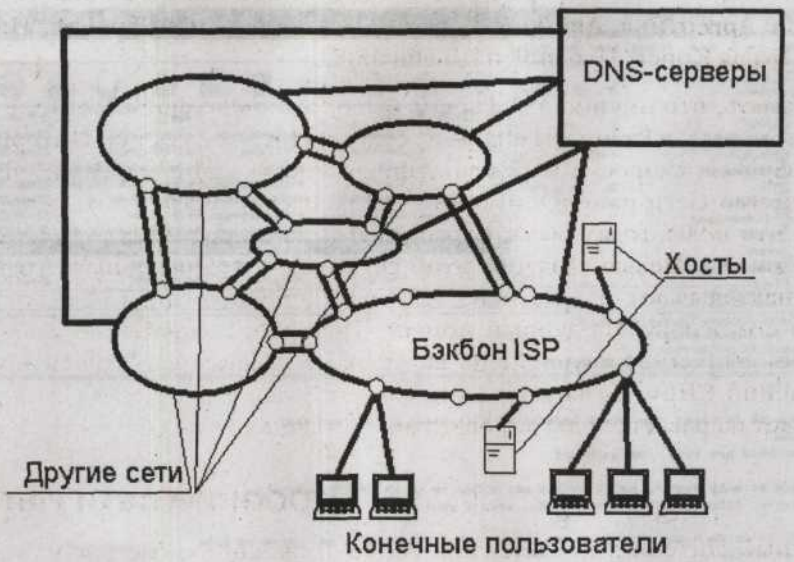


Рис. 1.9. Примерная структура сети Интернет

Особняком стоят DNS-серверы – компьютеры, отвечающие за функционирование системы DNS.

Для подключения конечных пользователей к ISP служат так называемые «точки доступа» – компьютеры или специальные устройства, содержащие оборудование для подключения «извне».

Подключившись к точке доступа провайдера, пользователь становится частью магистральной сети провайдера и, соответственно, получает доступ к ее ресурсам, а также к ресурсам сетей, соединенных с бэкбоном провайдера, т. е. ко всему Интернету.

Кроме конечных пользователей, к сети провайдеров подключаются различного рода серверы, или «хосты» (от слова host – хозяин). Это узлы сети, на которых работает программное обеспечение, обеспечивающее практически все услуги, предоставляемые сетью Интернет.

Основные возможности Интернета

На что же способен современный Интернет?

Электронная почта

Самой популярной, и одной из самых старейших его возможностей, является электронная почта (Рис. 1.10), или e-mail (от electronic mail – электронное письмо), явившаяся прямой наследницей почты бумажной и во многом потеснившая ее. Электронное письмо может содержать тексты, графику и любое другое содержимое, представимое в виде компьютерных файлов.

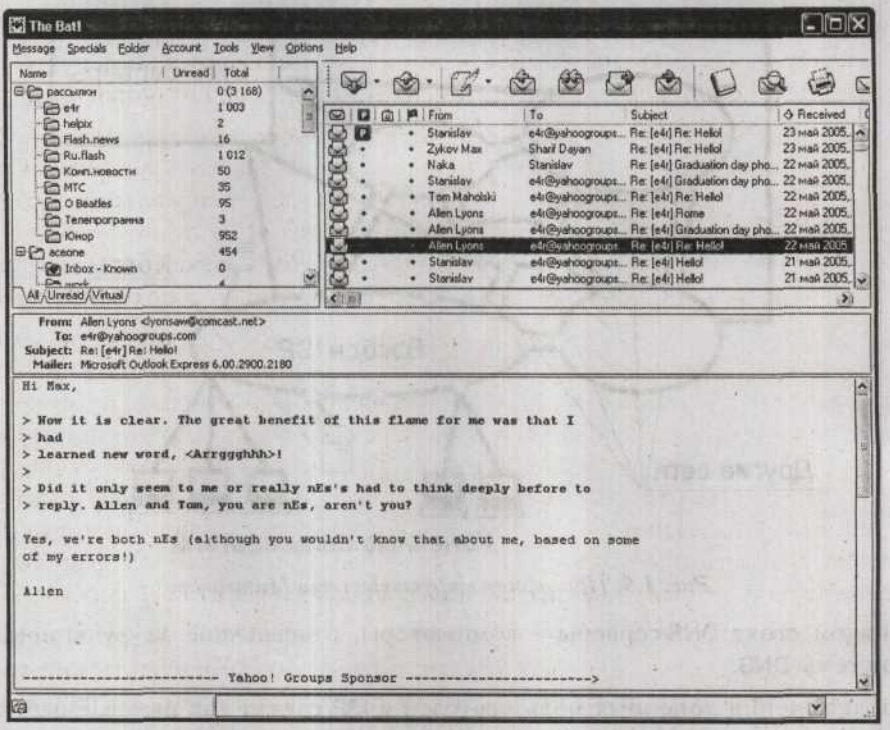


Рис. 1.10. Программа для работы с электронной почтой

Адреса электронной почты выглядят следующим образом: **bulbulator@mail.ru**. Значок «@», называется «коммерческое at», но обычно его называют просто «собакой». Справа от «собаки» в почтовом адресе идет Интернет-адрес компьютера, который занимается отсылкой, приемом и, возможно, первичной обработкой вашей почты. Этот компьютер называется почтовым сервером. Слева от «@» располагается собственно название почтового ящика на этом сервере. Один почтовый сервер может обслуживать до нескольких миллионов почтовых адресов.

Главным преимуществом электронного письма перед его бумажным аналогом является скорость доставки. Тогда как обычная почта может пересылать письмо в пределах одного города в течение нескольких дней, электронное послание за считанные секунды облетает всю Землю. Хотя случаются казусы, когда e-mail может идти до адресата несколько дней и даже лет. Однако ведь и с обычной почтой такое частенько случается, только вот электронное письмо вы всегда можете перепослать заново, чего не скажешь о письме бумажном.

Вторым преимуществом электронных писем является дешевизна доставки. У классической почтовой службы большие накладные расходы: письма нужно собрать, отсортировать по точкам назначения, доставить и, наконец, вручить адресату. Электронное письмо существует лишь в виде электромагнитных импульсов, и себестоимость его пересылки стремится к нулю.

Низкая себестоимость отправки электронных писем привела к огромной популярности такой услуги, как почтовые рассылки. Людям, подписавшимся на такую рассылку, могут приходить свежие анекдоты, новости, статьи определенно тематики и т. д.

Но, как известно, недостатки часто являются продолжениями достоинств, и простота отправки электронных писем породила такую напасть как спам. Спамом называют несанкционированные почтовые рассылки, отправляемые одновременно на миллионы почтовых адресов. Многим людям, активно пользующимся электронной почтой, приходится ежедневно разгребать завалы из сотен писем с назойливой рекламой. Со спамом пытаются бороться, но пока не очень успешно.

Относительным недостатком электронной почты является простота ее перехвата и прочтения посторонними лицами. Достаточно сказать, что электронное письмо проходит через Интернет к адресату письма в совершенно открытом виде и ознакомиться с его содержанием может любой член сети, через чей компьютер это письмо проходит. Хотя и обычное письмо можно вскрыть и прочитать, но это требует определенных материальных затрат. Разумеется, электронные письма можно шифровать, существует несколько надежных систем шифрования, на взлом шифров которых может уйти время, превышающее время жизни Вселенной. Однако, уже сам факт того, что вы шифруете свои письма, может привлечь излишний интерес к вашей персоне.

И еще одной проблемой, связанной с электронной почтой, является ее неуниверсальность. То есть, если у человека, которому вы хотите написать письмо, нет компьютера или доступа к Интернету, то вы просто не сможете воспользоваться услугой e-mail. Хотя некоторые фирмы и предлагают услуги по распечатке на принтере и доставке электронной почты по любому адресу, но в таком случае проще воспользоваться услугами почты традиционной.

Группы новостей

Во многом похожа на электронную почту технология групп новостей (newsgroups). С той лишь разницей, что группы новостей изначально предназначены для общения между собой сразу многих людей. Любое сообщение, размещенное в группе одним из ее участников, смогут увидеть и все остальные подписчики группы. Обычно группы новостей делятся по темам, которые в них обсуждаются. Поскольку технология новостных групп появилась в академической среде американских университетов, она располагает к вдумчивому и неторопливому обсуждению вопросов, связанных с тематикой группы.

Поскольку в новостных группах участвует множество людей, обычно создаются специальные правила, которые обязаны соблюдать все подписчики. За соблюдением правил следит администрация группы – так называемый модератор. За нарушение правил группы модератор может, обычно несколько раз предупредив нарушителя, отключить подписчика от группы. В большинстве групп нарушением является отклонение от темы группы, нецензурная лексика, оскорбление других участников группы и вопиющая безграмотность. Поэтому новостные группы являются очень удобным местом для плодотворного общения по вопросам, которые вас действительно интересуют.

Системы общения в реальном времени

Логическим продолжением системы новостных групп стала программа IRC (Internet Relay Chat – Разговоры, транслируемые через Интернет), см. Рис. 1.11. В отличие от групп новостей общение в этой системе проходит в реальном времени – все участники одновременно находятся у компьютеров и участвуют в разговоре. С одной стороны, быстрая реакция собеседника позволяет представить себя участником живой беседы, но с другой, – отсутствие времени на обдумывание, часто вынуждает собеседников на излишне скоропалительные ответы.

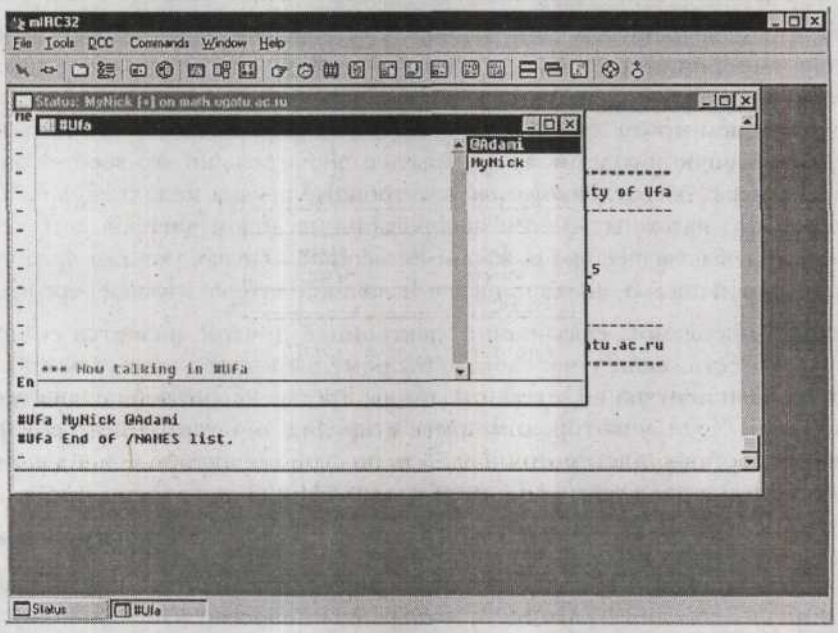


Рис. 1.11. Программа для работы с IRC

Как и новостные группы, IRC делится на тематические разделы, называемые каналами. Например, канал, посвященный обсуждению книг, называется **#books**.

Еще одним способом общения через Интернет являются системы мгновенного обмена сообщениями, или IMS (Instant Messaging System – система мгновенного обмена сообщениями), их также частенько называют интернет-пейджерами. Первой системой такого рода, остающейся и сейчас самой популярной, является ICQ (от I Seek You – я ищу тебя), см Рис. 1.12. Достаточно популярны еще такие системы, как Odigo, AOL Instant Messenger, который используют преимущественно клиенты крупнейшего американского провайдера интернет-услуг AOL (America On-Line – Америка на связи), и MSN Messenger, входящий в комплект последних версий операционной системы Windows.

По сути своей работы все системы мгновенного обмена сообщениями одинаковы – на компьютер устанавливается программка, позволяющая с компьютера, подключенного к Интернету, отправлять другим пользователям этой программы сообщения, которые мгновенно отображаются у них на экране компьютера.

Рис. 1.12. Программа ICQ

Хотя первоначально такие системы создавались исключительно для необременительного, ни к чему не обязывающего общения, оперативность, которую они предоставляют, побудила многих людей использовать интернет-пейджеры в сугубо деловых целях – для принятия заказов через Интернет, управления подчиненными и даже для проведения экстренных совещаний.

Популярность IMS систем настолько высока, что некоторые операторы мобильной связи даже предлагают услугу общения в ICQ посредством SMS.

Следующая возможность, которую предоставляет сеть Интернет, на первый взгляд несколько парадоксальна. Интернет начинал развиваться на основе кабелей телефонной связи, по которым вместо голоса начали передавать потоки нулей и единиц. Да и сейчас подавляющее большинство конечных пользователей, по крайней мере в России, подключаются к Интернету через телефонный кабель. И, в то же время, сейчас все больше набирает популярность услуга Интернет-телефонии, то есть голосовое общение посредством Интернета. И выглядит немного странно, когда человек, чтобы поговорить по телефону, вместо того, чтобы просто набрать номер, подключает компьютер к телефонной розетке и выходит в Интернет и уже после этих манипуляций начинает разговор.

В чем же дело? А в том, что передавая и получая информацию с соседней улицы и с другого континента, вы платите одинаково, чего нельзя сказать о телефонных разговорах. Разговор по телефону в пределах одного города чаще всего бесплатен или стоит копейки, но если вам необходимо пообщаться с Австралией, то вас, возможно, ждут финансовые проблемы. До последнего времени Интернет-телефония была доступна, в основном, корпоративным клиентам, пользующихся скоростным выделенным каналом в Интернет, но сравнительно недавно вышла

программа под названием Skype (Рис. 1.13), позволяющая совершенно бесплатно, не считая стоимости доступа в Интернет, общаться через Сеть даже обладателям медленных телефонных модемов. Причем ее популярность оказалась настолько высока, что в некоторых странах ее пытались запретить, чтобы телефонные компании не лишились своих прибылей.

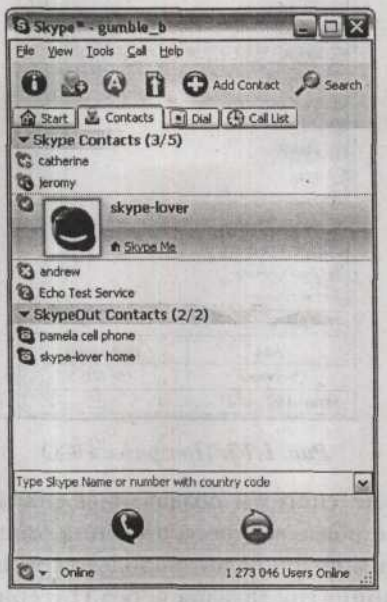


Рис. 1.13. Skype

Сервисы по передаче файлов в сети Интернет

Интернет позволяет пересылать файлы с одного компьютера, подключенного к Сети, на другой. Для этого существует масса способов, например файлы можно пересылать в письмах электронной почты, но существует механизм, разработанный специально для передачи файлов, это упоминавшийся ранее протокол FTP. Компьютер, к которому вы подключились по протоколу FTP, называемый еще FTP-сервером, предстает перед вами как иерархическая структура папок и файлов, как еще один жесткий диск на вашем компьютере (Рис. 1.14). В зависимости от уровня доступа, который предоставляет вам FTP-сервер, вы можете копировать файлы с него на свой компьютер, копировать свои файлы в одну из папок сервера, переименовывать их и даже удалять.

FTP-сервер можно использовать как удаленное хранилище вашей информации, доступ к которому вы можете получить с любого компьютера, имеющего выход в Интернет.

На публичных, то есть доступных для всеобщего посещения, FTP-серверах доступно множество разнообразнейшей информации: книги, музыка, программы, фильмы, графика и т. д.

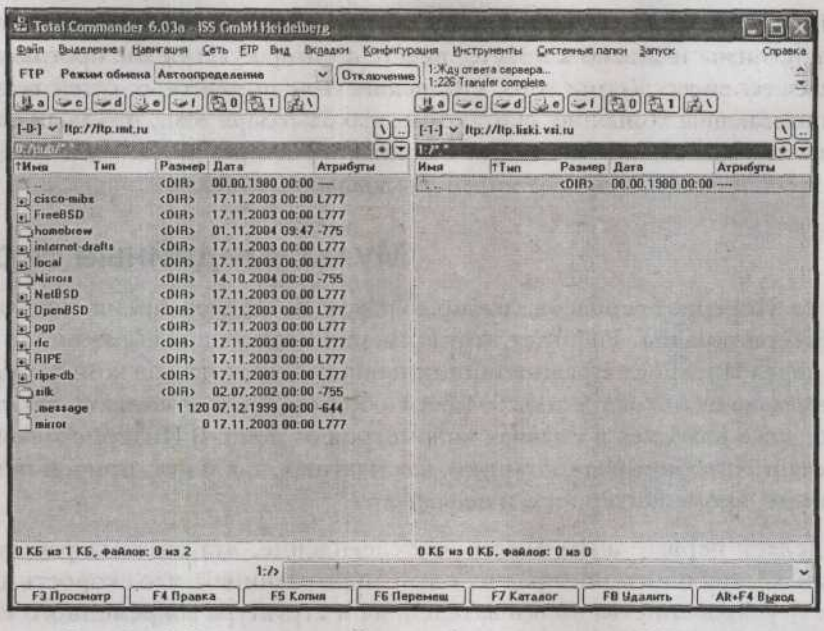


Рис. 1.14. Файловый менеджер, подключившийся к FTP-серверу

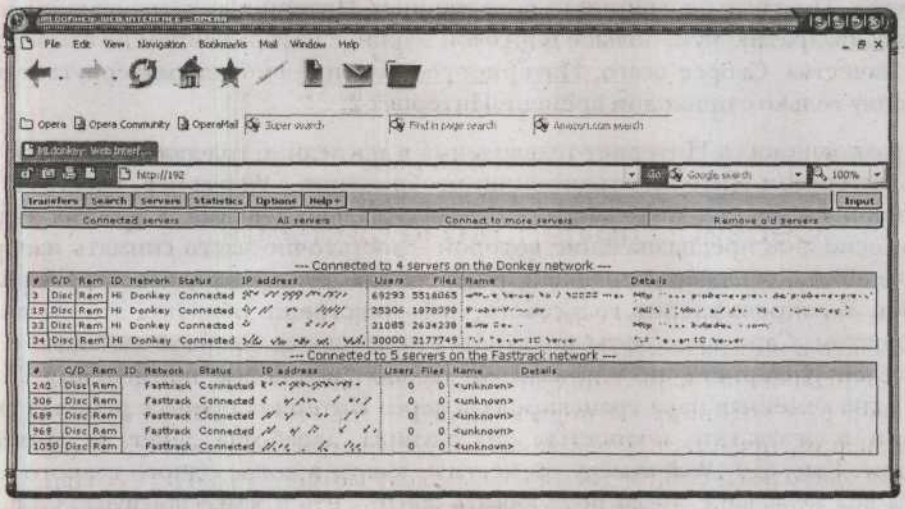


Рис. 1.15. Программа для работы с пиринговыми сетями

Еще одним способом обмена файлами через Интернет являются пиринговые сети. Слово это произошло от их концепции: peer-to-peer (равный с равным), см. Рис. 1.15. Суть пиринговых сетей заключается в следующем: человек, желающий участвовать в такой сети, отдает в общий доступ некоторое количество файлов со своего жесткого диска, в обмен он получает возможность скачивать файлы с других компьютеров сети. Таким образом, получается, что все участники пиринговой сети равны в правах, хотя у всех и разные возможности. Самой первой пиринговой сетью была сеть под названием Napster, сейчас известны такие сети, как Gnutella, KaZaa, e-Donkey и

другие. Недостатком этих сетей является то, что файлы на компьютерах пользователей не проверены и можно запросто, вместе с интересующей вас программой, скачать и свежий вирус. Кроме того, с большинством пиринговых сетей ведут войну звукозаписывающие компании, считающие, что благодаря тому, что люди скачивают пиратские музыкальные композиции с компьютеров друг друга, сами звукозаписывающие компании недополучают миллионы долларов прибыли.

Мультимедийные сервисы

Одним из Интернет-сервисов, набирающих в последнее время популярность, стало Интернет-радио. Работает это примерно следующим образом: вы подключаетесь через Интернет к радиостанции и она передает на ваш компьютер сигнал из студии в виде потока данных. Таким образом, можно слушать любимую fm-станцию, даже находясь в тысячах километров от дома. В Интернете сейчас доступны сотни Интернет-радиостанций, как платных, так и нет, причем некоторые из них нигде, кроме Интернета, и не вещают.

Сейчас делает первые шаги Интернет-телевидение, эта система работает примерно так же, как и интернет-радио, с той лишь разницей, что скорость доступа к Интернету у большинства пользователей, да и структура современного Интернета не позволяют передавать в реальном времени видеоизображение хорошего качества. Поэтому большинство современных Интернет-телетрансляций выглядят как квадратик, чуть больше почтовой марки, с изображением не самого высокого качества. Скорее всего, Интернет-телевидение сможет развернуться в полную силу только с приходом времени Интернет-2.

Как разновидность Интернет-телевидения в последние год-два пользуются большим интересом Интернет-трансляции изображения с Web-камер, расставленных в различных местах. Web-камера – это максимально дешевая и простая видеокамера, основное предназначение которой – достаточно часто снимать изображение человека, сидящего перед ней, и пересылать его другому человеку через Интернет. В сопровождении голосового, либо текстового общения это называется видеочатом. Сами видеочаты пользуются не слишком большой любовью у пользователей Интернета, но Web-камерам находят все новые применения. Например, одна семейная пара транслировала через Интернет процесс ремонта своего дома и в результате полностью его окупила, заработав денег на Интернет-рекламе. Еще одну Web-камеру разместили как-то в холле одного старого замка, чтобы все желающие могли попробовать обнаружить в замке призрака. Самым же распространенным вариантом является размещение камеры просто на домашнем компьютере, чтобы все могли наблюдать, как человек живет своей жизнью.

Системы распределенных вычислений

Достаточно перспективной областью применения Интернета является сфера распределенных вычислений. Дело в том, что современные компьютеры большую часть времени работают на 1-2% своей вычислительной мощности, в то время как для некоторых расчетов, как жизненно необходимых человечеству, так

и достаточно бесполезных, не хватает мощности самых быстрых суперкомпьютеров или просто не хватает денег у ученых. Поэтому, в свое время, родилась идея разделить одно большое вычисление на много маленьких частей и считать каждую часть на отдельном компьютере, а после получить общий результат. Как это происходит? Обычно необходимо установить на свой компьютер небольшую программу – вычислительный клиент, который будет периодически выходить в Интернет, брать очередной кусочек задачи для расчетов, после чего будет использовать незадействованные ресурсы компьютера для вычислений, а результаты будет отправлять обратно. Сейчас действует масса проектов распределенных вычислений: одни ищут внеземные цивилизации, кто-то взламывает шифры, некоторые пытаются спасти человечество от рака. Люди, участвующие в таких вычислениях, разбиваются на команды, соревнуются друг с другом в количестве просчитанной информации, иногда победителям даже раздают призы. Некоторые интернет-аналитики полагают, что, возможно, в ближайшем будущем, распределенные вычисления через Интернет будут поставлены на коммерческую основу, и за компьютерное время будут платить.

World Wide Web

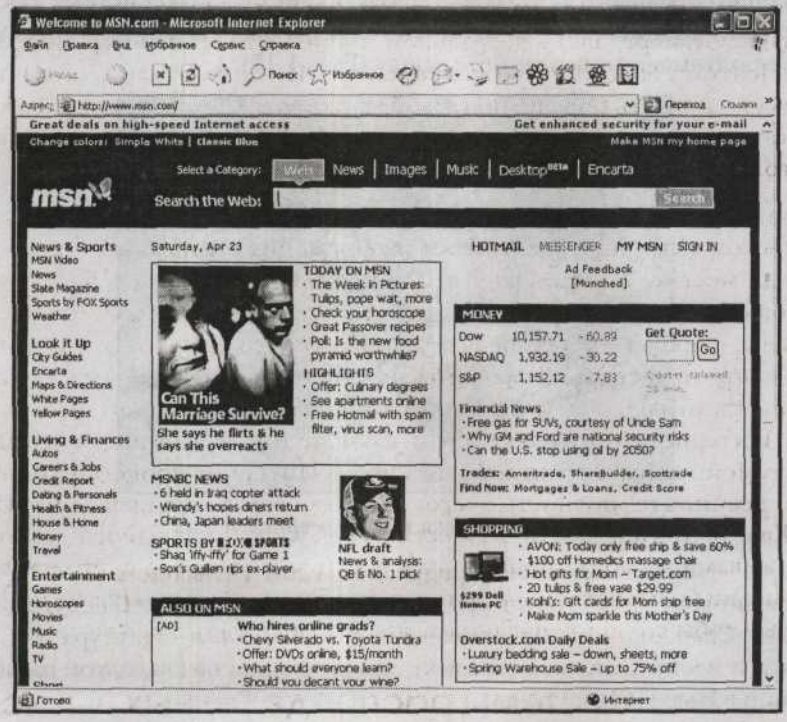


Рис. 1.16. Пример гипертекста в Web

И, наконец, одной из самых интересных возможностей Интернета является доступ к гипертекстовым документам, находящимся на компьютерах Сети. Гипертекст – это обычный текстовый документ, содержащий ссылки на другие доку-

менты по ссылкам можно переходить от одного документа к другому, возвращаться назад, к предыдущему тексту и т. д. Кроме того, гипертекстовые документы могут содержать иллюстрации, анимационные вставки, небольшие программные модули и т.д. Подавляющее большинство текстовой информации в Интернете представлено именно гипертекстами. Технология гипертекста является основой для одной из самых значимых частей Интернета, называемой World Wide Web (Всемирная паутина), WWW, или просто Web (Рис. 1.16).

История развития Всемирной паутины

Первым предложил использовать технологии гипертекста для удобства доступа к информации, научный советник президента США Рузвельта Ванневар Буш (Vannevar Bush). В 1945 году он опубликовал свою статью, «As We May Think» (Как мы можем мыслить), в которой описал устройство гипотетической машины «Memex», которая позволяла хранить текстовую и графическую информацию так, что любая ее часть могла быть связана с любой другой ее частью.

В 60-х годах, сотрудник исследовательского центра PARC Xerox, Дуглас Энгельбарт (Doug Engelbart) разработал и создал первую действующую систему для работы с гипертекстами, специально для этой системы им был создан прообраз манипулятора, далее получившего название «мышь» (Рис. 1.17).

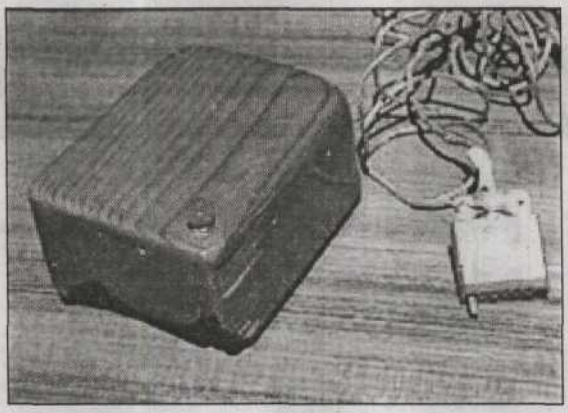


Рис. 1.17. *Первая компьютерная мышь*

А само слово «гипертекст» было придумано Тедом Нельсоном (Ted Nelson), оно впервые прозвучало в 1965 году, в его докладе под названием «A File Structure for the Complex, the Changing, and the Indeterminate» (Файловая структура для сложных, меняющихся и неопределенных данных), прозвучавшем на двадцатой национальной конференции в Нью-Йорке.

В 1986 году Международная организация по стандартизации ISO одобрила стандарт ISO-8879 языка гипертекстовой разметки SGML (Standard Generalized Markup Language – Стандартный обобщенный язык разметки). Основанный на языке гипертекстовой разметки GML (Generalized Markup Language – Обобщенный язык разметки), он позволил отказаться от конкретных способов представления информа-

ции. Он дал возможность сосредоточить усилия на продумывании структуры документов с помощью правил определения собственных команд форматирования документа, называемых тегами, их атрибутов и синтаксиса использования. Для создания конкретных прикладных наборов тегов было введено понятие «SGML-приложение». SGML оказался очень мощным и универсальным языком, но он требовал точного описания всех нюансов создаваемого синтаксиса документа и подробных правил формирования тегов.

Классическое определение гипертекста Тед Нельсон сформулировал в 1987 году, он сказал, что гипертекст – это «форма письма, которое ветвится или осуществляется по запросу». Иначе говоря, это «нелинейное письмо», которое «больше, чем текст».

В том же 1987 году была проведена первая конференция, посвященная технологиям гипертекста, Hypertext'87; конференция состоялась в США, вызвала много шума и породила большой энтузиазм у достаточно широкого круга людей. В это же время Бил Аткинсон, известный созданием первого графического растрового редактора MacPaint, дал компьютерному миру первую популярную гипертекстовую систему HyperCard. Эта программа позволяла относительно легко создавать графические гипертекстовые приложения.

Таким образом, к концу 80-х годов XX века технология гипертекста прочно вошла в мир компьютерной техники. Оставался один шаг до появления WWW, и этот шаг вскоре был сделан.

Рождение World Wide Web

Родоначальником Всемирной паутины, объединившим возможности гипертекстовых документов с потенциалом Интернета, считается Тим Бернерс-Ли (Tim Berners-Lee), Рис. 1.18. В 1989 году, работая в Европейской лаборатории физики элементарных частиц в Женеве (CERN), он вместе со своим коллегой Робертом Каиллау (Robert Cailliau), предложил и разработал концепцию распределенной информационной системы на основе гипертекста. Целью создания этой системы было упрощение обмена информацией между различными группами физиков. Разрабатывая стандарты будущей всемирной паутины, Тим Бернерс-Ли исходил из следующих соображений:

- Широко разбросанное по миру сообщество исследователей (прежде всего ученых в области физики высоких энергий), нуждается в быстром и эффективном обмене экспериментальной информацией в промежутках между конференциями.
- Эти ученые работают с самым широким спектром всевозможного оборудования, от терминалов ввода-вывода до персональных компьютеров и рабочих станций.
- Невозможно централизовать или как-либо регулировать информационные потоки. Каждый исследователь делится информацией с любым, кто мог бы помочь в данный момент. Невозможно предсказать, кому какая

информация необходима в данное время. Нужно, чтобы каждый исследователь мог представлять свою информацию в нужном формате и соотносить ее с уже существующей информацией.

Рис. 1.18. Тим Бернерс-Ли

Таким образом, необходима была система, которая могла бы работать с любым видом интерфейса, графическим или текстовым. Система, которая была бы распределенной, нецентрализованной и подстраивающейся под пользователя. Система, которая вместо попыток искать стандарты на уровне оборудования или программного обеспечения, делала бы это на уровне данных.

В результате, вскоре была разработана система гипертекстовой разметки данных, названная языком HTML (Hypertext Markup Language – язык разметки гипертекста). Язык HTML являлся, по сути, значительно упрощенной и урезанной версией языка SGML. За это многие сторонники «академического» гипертекста ругали и продолжают ругать как сам HTML, так всю систему WWW, но, тем не менее, WWW и HTML завоевали такую популярность, которая другим гипертекстовым системам и не снилась.

Также был разработан механизм доступа к документам HTML, он был назван протоколом HTTP (Hypertext Transfer Protocol – протокол передачи гипертекста). Для доступа к HTML документам в Интернете через протокол HTTP используется специальная программа, называемая Web-браузером (от английского «browser» – просмотр).

История развития WWW

В конце 1990 года был создан первый Web-браузер, названный WWW, сначала для компьютеров NEXT, а затем и для других компьютерных платформ. С этого же момента начинается распространение Всемирной паутины по CERN и по всему Интернету в целом. Всемирная паутина стала мощным стимулом для развития Интернета, позволяя получать доступ к информации в виде, удобном для понимания и близком человеческому восприятию. Как признавались сами разработчики WWW, такое взрывообразное развитие их идей явилось для них самих полной неожиданностью.

Полные графики, удобные в использовании и красиво выглядящие HTML документы явились благодатной почвой для развития Интернет-коммерции, в Сеть пошли потоки частных инвестиций и появилась масса новых пользователей, которых Интернет раньше отпугивал своей «академичностью» и непонятностью. Менее чем за 30 лет эксперимент, затеянный агентством ARPA, стал неотъемлемой частью человеческой культуры.

Важным этапом для коммерческого развития WWW стала разработка в 1994 году протокола защищенного доступа в WWW и лицензирование Web-браузера Mosaic, (Рис. 1.19), давшее зеленый свет разработке коммерческих Web-браузеров. На основе Mosaic, были впоследствии разработаны такие Web-браузеры, как Netscape Navigator (Рис. 1.20) и самый ныне популярный Web-браузер Microsoft Internet Explorer (Рис. 1.21).

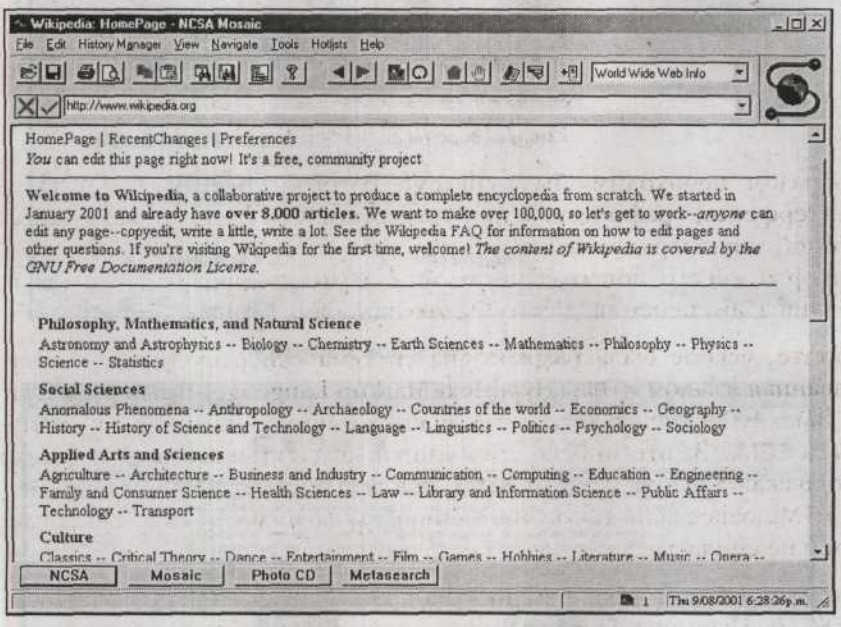


Рис. 1.19. Web-браузер Mosaic

Как обычно бывает в системах с высокими темпами роста, в только начинающей свой путь Всемирной паутине царил некоторый хаос. Разные производители выпускали браузеры, частично не совместимые друг с другом, поскольку каждый стремился добавить в язык HTML свои теги форматирования, которые не поддерживали конкуренты. В результате, часть HTML документов могли просматривать только одни браузеры, а часть – другие.

Недовольные этой неразберихой руководители CERN, совместно с Массачусетским Технологическим Институтом (MIT – Massachusetts Institute of Technology) сформировали в декабре 1994 года консорциум WWW (W3C), быстро взявший под свой контроль работу практически над всеми стандартами важнейших технологий Сети. Надо отметить, что формально W3C выпускает только рекомендации, и некоторые компании их игнорируют, но в целом рекомендации W3C признаются всем рынком в качестве стандартов.

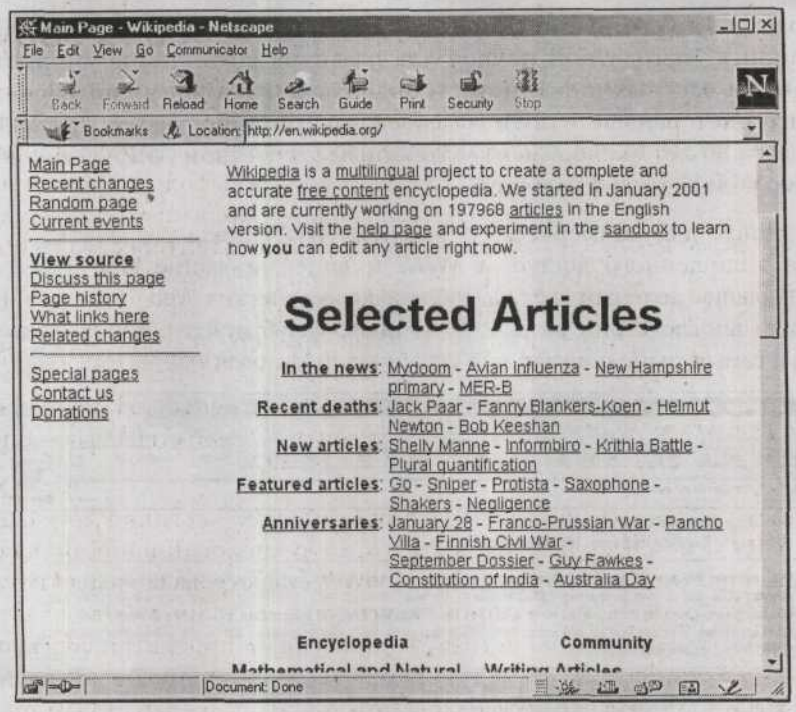


Рис. 1.20. Netscape Navigator

Рис. 1.21. Internet Explorer

Войны браузеров

Одним из самых ярких событий времен становления WWW, стали так называемые «браузерные войны». В 1996 году самым популярным Web-браузером был Netscape Navigator 2.0, являвшийся прямым наследником браузера Mosaic. Он поддерживал много дополнительных тегов, не предусмотренных спецификациями HTML, но позволяющих качественно оформлять внешний вид гипертекстовых документов. Но как раз в это время рынком Web-браузеров заинтересовалась компания Microsoft, ее лидер Билл Гейтс (Bill Gates), провозгласил: «HTML стал нашим типом данных». И вскоре, практически одновременно с выходом третьей версии Netscape Navigator, Microsoft выпустила свой браузер, Internet Explorer 3.0, также основанный на браузере Mosaic, причем Microsoft включила в свой браузер поддержку всех дополнительных тегов, что были у конкурентов. И, к тому же, у браузера Internet Explorer было серьезное конкурентное преимущество – он был бесплатным.

Вскоре вышли и следующие версии обоих браузеров, частично друг с другом несовместимые, даже поддержка одних и тех же технологий в них осуществлялась различным образом. Некоторое время в WWW существовала такая ситуация, что для комфортной работы с Всемирной паутиной необходимо было являться пользователем обоих браузеров и часть HTML-документов просматривать в Internet Explorer, а часть в Netscape Navigator. Но через довольно непродолжительное время чаша весов склонилась в сторону Internet Explorer, этот браузер стал стандартом де-факто, и продолжает им оставаться по сей день, несмотря на то, что разработчики браузера не соблюдают часть рекомендаций W3C. По статистике, более 90% пользователей WWW пользуется именно им, хотя его доля в последнее время стала понемногу уменьшаться.

Устройство Всемирной паутины

За неполные 15 лет своего существования Всемирная паутина очень сильно разрослась. Настолько, что говоря об Интернете, люди часто подразумевают именно WWW, несмотря на то, что Web является лишь одной из многочисленных услуг, предоставляемых Всемирной сетью. Чтобы получить доступ к WWW, даже не нужен персональный компьютер. Ресурсы WWW можно просматривать, хотя обычно и в несколько урезанном виде, с мобильных телефонов, карманных компьютеров, игровых приставок и даже при помощи бытовой техники, например некоторых телевизоров, микроволновых печей и холодильников.

Такие Web-технологии, как HTML и HTTP, широко используются и отдельно от Интернета. Ярким примером этого являются, например, терминалы для проверки наличия свободных мест в поездах, стоящие на многих железнодорожных вокзалах.

На что же похожа современная система WWW? Больше всего – на всемирный «Информаторий», идея которого еще в середине XX века, зародилась в умах многих писателей-фантастов. О такой системе упоминали в своих книгах братья Стругацкие, Роберт Хайнлайн и многие другие писатели, как известные, так и не очень. По

сути, можно представить Web как некую, невероятных размеров, базу данных, обращаясь к которой пользователь может получить интересующую его информацию или, наоборот, передать свою информацию на тот или иной узел Web.

Основой WWW являются Web-серверы. В простейшем случае Web-сервером называется компьютер, подключенный к Интернету и настроенный таким образом, чтобы Интернет-пользователи, подключающиеся к этому компьютеру с помощью Web-браузера, могли просматривать HTML-файлы, на нем хранящиеся. На Web-сервер устанавливают специальное программное обеспечение, называемое тоже Web-сервером, которое обеспечивает работу этой системы. Чтобы избежать путаницы, компьютер, работающий Web-сервером, мы будем называть просто сервером, а Web-сервером будем именовать программу. Отдельный HTML-документ на сервере называется страницей, а набор страниц, связанных друг с другом ссылками, объединенных общей тематикой и одним доменным именем, называется сайтом. На одном сервере могут одновременно работать сотни небольших сайтов, но бывает и наоборот: работу одного единственного сайта обеспечивают десятки и сотни серверов. Все зависит от размеров и посещаемости сайта.

Какие бывают Web-сайты?

Для каких целей создают сайты? Если сказать в двух словах, то – для любых. Просто перечисление сфер человеческой деятельности и видов услуг, нашедших свое отражение в WWW, может занять не один десяток страниц, поэтому ограничимся кратким рассмотрением наиболее распространенных видов сайтов и Web-сервисов.

Рис. 1.22. Web-представительство фирмы

- Web-представительства фирм (Рис. 1.22). Сейчас для практически любой компании, вне зависимости от области ее деятельности, наличие собственного Web-сайта является правилом хорошего тона. Обычно такие сайты называют «визитками». На них размещают информацию об основных направлениях своей деятельности, делятся основными новостями компании, рассказывают о производимых товарах и новинках.

- Промо-сайты. Сайты, создаваемые исключительно в рекламных целях. Для продвижения какого-нибудь товара, раскрутки новой торговой марки, рекламы нового фильма и т. д. Обычно отличаются минимальной информативностью и максимальной зрелищностью.

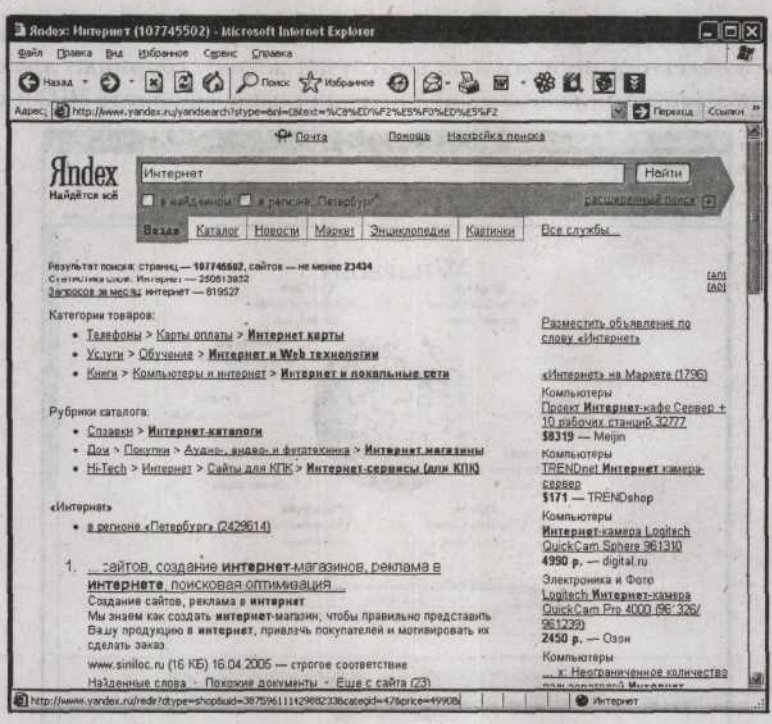


Рис. 1.23. Поисковая система Яндекс

- Поисковые системы (Рис. 1.23). Хорошо известна поговорка, что «в Интернете есть все». Она, если и не полностью справедлива, то достаточно близка к истине. Информации в Интернете чрезвычайно много. Это порождает проблему поиска в этом океане информации действительно нужной. Специально для этого были созданы поисковые системы, или просто «поисковики». Основой любой поисковой системы являются специальные поисковые «боты». Это программы, которые методично обследуют Web-сайты Интернета, переползая по ссылкам с одного на другой, и составляют специальную «выжимку» содержания сайтов, называемую «индексом». Когда пользователь обращается к поисковой системе с запросом на поиск определенного слова или группы слов, «поисковик», на основе индексов, выдает пользователю

список Web-страниц, содержащих нужные слова. Положение страницы в списке определяется по ряду признаков: частоте употребления этого слова в документе, частоте посещений сайта, количестве ссылок, которые ведут на сайт с других Web-сайтов, и т. д.

- Порталы. Существует род сайтов, объединяющих в себе массу разнородной информации и услуг. Например, от службы знакомств до почтовой системы и энциклопедии. Сделано это для того, чтобы у пользователя не возникало необходимости обращаться к другим сайтам, а вместо этого он находился постоянно в пределах портала, принося владельцам доход от просмотренных им рекламных сообщений.

- Файловые хранилища. WWW заполнена всевозможными сборниками файлов: коллекции программ на все случаи жизни, сборники драйверов для компьютерного «железа», библиотеки музыки в формате MP3 и т. д.

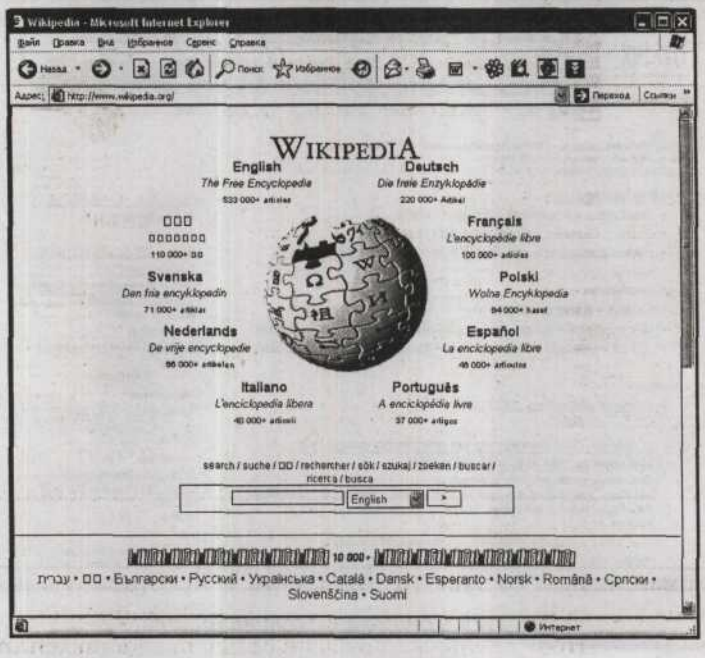


Рис. 1.24. Энциклопедия Wikipedia

- Архивы, библиотеки, энциклопедии, словари. Оправдывая звание всемирного Информатория, Web содержит огромные залежи различных документов, частично сгруппированных на специальных сайтах. Часть этих архивов, в основном научных, доступна только за плату, но значительная часть информации находится в свободном доступе. Отдельного упоминания заслуживают Web-энциклопедии, а точнее одна из них, носящая название Wikipedia (Рис. 1.24). Уникальность ее заключается в том, что в отличие от других энциклопедий, составлением которых занимаются специально нанятые люди, содержимое Wikipedia пополняют сами пользователи Сети, причем весьма успешно.

- Электронные средства массовой информации (Рис. 1.25). С распространением Web, многие издательства стали создавать Web-представительства своих журналов и газет, размещать в WWW новости, дайджесты новых номеров, а иногда и полностью тексты публикаций. Через некоторое время начали появляться и полностью «Электронные» СМИ, вообще не выпускающиеся в бумажном виде. Преимущество Web-технологий для средств массовой информации очевидно – оперативность. В то время, как бумажные журналы и газеты выходят с определенной периодичностью: раз в месяц, раз в неделю, каждый день, информацию на сайте можно обновлять хоть каждые несколько минут. Кроме того, в Интернет-изданиях, в отличие от бумажных, практически нет проблем с размерами публикаций.

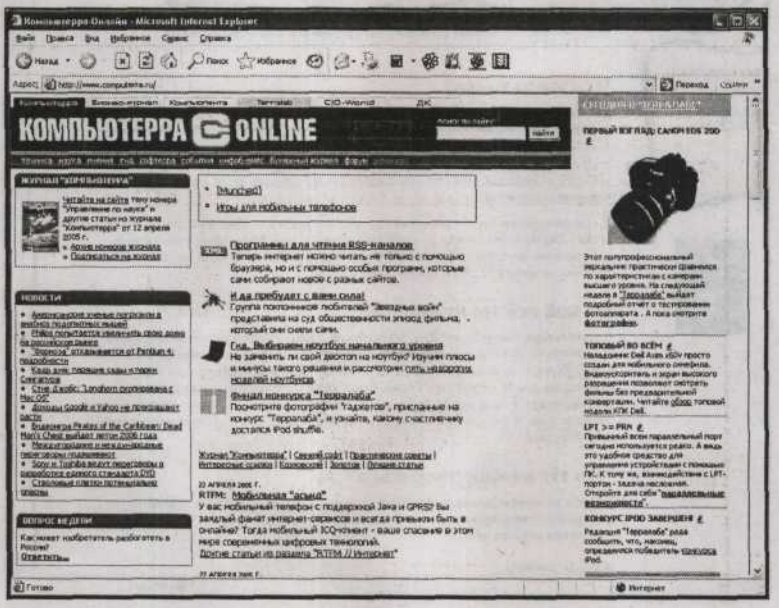


Рис. 1.25. Web-версия журнала

- Информационные сервисы. Доступность и простота обновления информации на Web-сайтах привели к появлению во Всемирной паутине широкого спектра сайтов, специализирующихся на предоставлении часто изменяющейся информации. Например, прогнозов погоды, программ телепередач, расписаний сеансов в кино и театрах и т.д.

- Техническая поддержка. Многие компании, особенно работающие в сфере высоких технологий, предоставляют своим пользователям ряд дополнительных услуг через свои Web-сайты. Большинство компаний сотовой связи, например, предоставляют своим клиентам возможность проверить баланс телефона, просмотреть список последних операций, изменить список подключенных услуг, через персональную страничку пользователя на Web-сайте компании.

- Домашние страницы (Рис. 1.26). Эта категория сайтов возможно одна из самых многочисленных. Многие пользователи Web считают своим долгом оставить свой след в мире WWW и создают сайты, посвященные своему жизненному пути, своим увлечениям и интересам. Как правило, являются психологическим портретом автора домашней страницы.

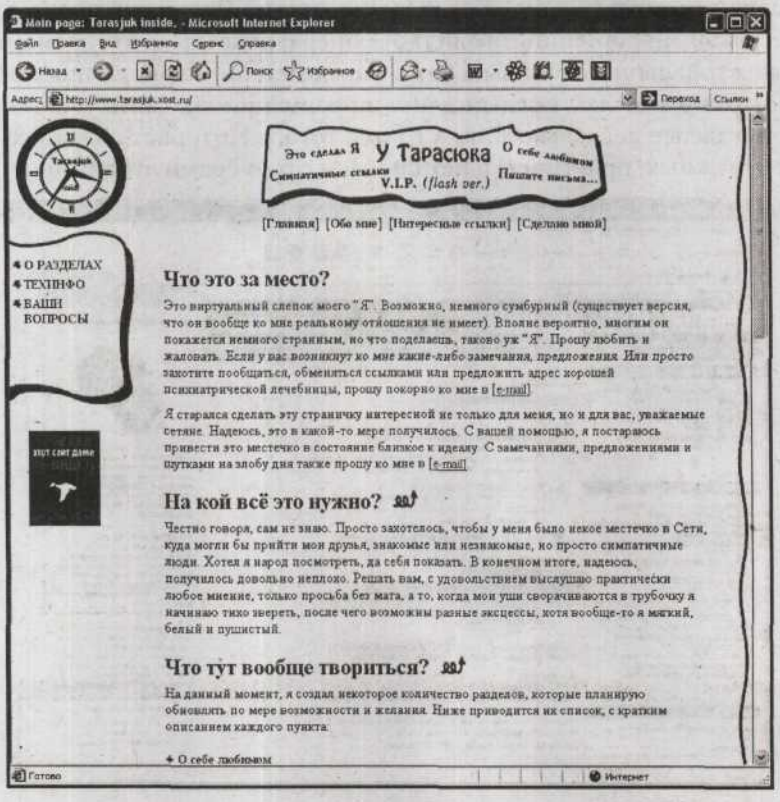


Рис. 1.26. Типичная домашняя страница

- Фанатские ресурсы. Как и в реальном мире, в Web постоянно возникают сообщества, посвященные культовым личностям, фильмам, сериалам или просто кумирам молодежи. Как правило, такие сайты быстро наполняются совершенно исчерпывающей информацией о предмете поклонения, разнообразными мифами и сопутствующими деталями.
- Интернет-магазины (Рис. 1.27). Основой любой коммерческой деятельности является торговля. WWW не является исключением. Торгуют в Интернет-магазинах всем, чем угодно: от бытовой техники до хорошего настроения. Большой популярностью пользуются так называемые «онлайн-аукционы», самым известным из которых, является аукцион e-Buy.
- Форумы (Рис. 1.28). Являются некоторым подобием новостных групп, сделанным для WWW. Один из пользователей форума может задать вопрос или поместить сообщение по какой-то теме, остальные пользователи мо-

гут отвечать на этот вопрос и комментировать его. При необходимости можно просмотреть всю историю обсуждения того или иного вопроса.

Рис. 1.27. Интернет-магазин

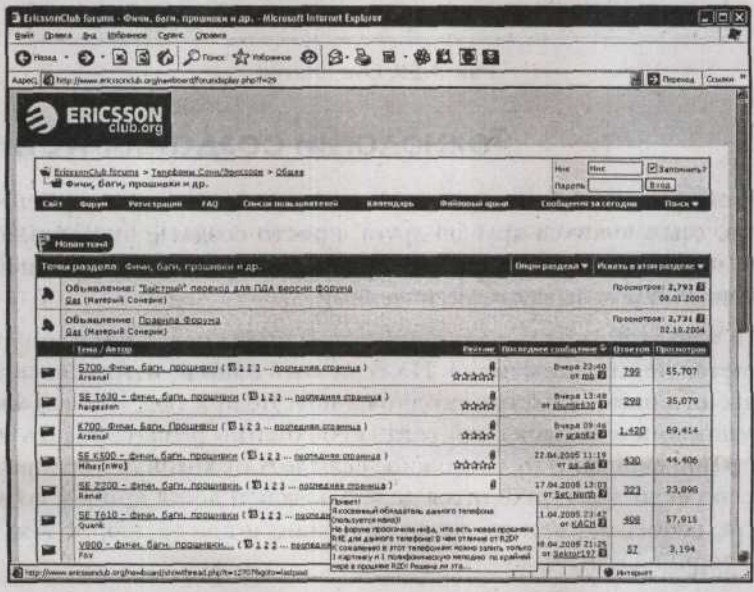


Рис. 1.28. Форум

- LiveJournal (Живой журнал, или просто ЖЖ), Рис. 1.29. Средство, созданное специально для размещения в Web личных дневников пользователей Сети. Члены сообщества LiveJournal и ему подобных, могут писать свои дневники, читать дневники других участников этой системы, а также добавлять свои комментарии к чужим дневникам.

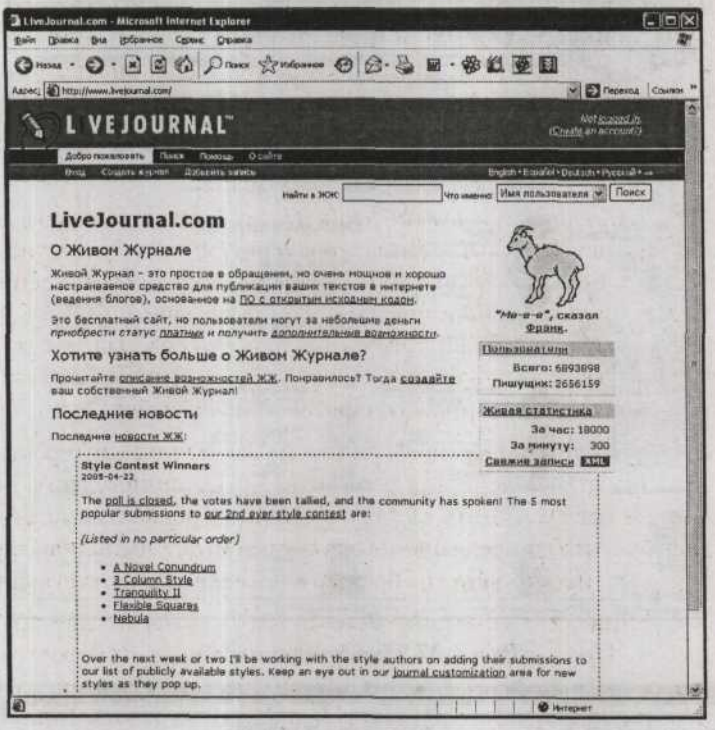


Рис. 1.29. LiveJournal

Технологии создания Web-сайтов

Web-сайт в том виде, как было описано выше, т.е. состоящий просто из набора HTML файлов, ссылающихся друг на друга, просто создать, он неприхотлив в обслуживании, надежен и не требует от сервера большой вычислительной мощности. Такие Web-сайты называют статическими.

Статический Web-сайт всем хорош, пока его размеры невелики и информация на нем обновляется не слишком часто. Но если хоть одно из этих условий не выполняется, то все его плюсы быстро сходят на нет. Дело в том, что для внесения каких-либо изменений в статический сайт, необходимо редактировать непосредственно сами HTML файлы. Это несложно, если вам нужно изменить пару страниц раз в месяц, но если это приходится делать каждый день, причем со всеми страницами сайта, особенно если их число переваливает за сотню, положение становится катастрофическим.

Кроме того, современные Web-сайты все чаще делаются на основе некоторой базы данных, содержащей элементы его наполнения, и эти элементы необходимо показывать пользователю тогда, когда они ему требуются, и в том порядке, в котором ему нужно. Ярким примером таких систем являются Интернет-магазины. Где-то в недрах сервера хранится база данных, в которой содержится информация обо всех товарах, которые продаются в этом магазине: названия товаров, цены, фотографии, различная дополнительная информация. Допустим, что этот магазин торгует часами. Пользователь может пожелать просмотреть все мужские часы определенных марок, обладающие нужными ему функциями и стоящие не дороже определенной суммы.

Третий недостаток статических Web-сайтов виден из названия – они статичны, т. е. лишены движения и, в какой-то мере, скучны.

Преодолеть все эти недостатки статических Web-сайтов можно, создавая так называемые «динамические» Web-сайты. Динамический Web-сайт в ответ на запросы Web-браузера пользователя может не просто выдавать заранее подготовленные страницы HTML, а создавать страницы «динамически», на основе записей в базе данных. Такой подход к созданию сайта позволяет также отделить его содержимое от внешнего вида, т. е. появляется возможность быстро менять оформление сайта или даже предлагать пользователям несколько вариантов оформления на выбор.

Второй возможностью динамических Web-сайтов является их «интерактивность», т. е. способность реагировать на действия пользователя. Например, сайт, торгующий экранами для ванн, может предлагать своим посетителям подобрать желательный декор и цвет экрана, выбрав их из предлагаемого ассортимента. Очень популярны игры, играть в которые можно, просто запустив браузер и посетив соответствующий Web-сайт.

Технологии создания динамических Web-сайтов делятся на две группы: серверные и клиентские. Под серверными технологиями подразумевают специальные программы, которые выполняются под руководством Web-сервера и заняты обработкой запросов Web-браузера. Чаще всего эти программы пишутся на специальных языках программирования, называемых языками сценариев. Самыми популярными языками сценариев являются PERL (Practical Extraction and Report Language – Практичный язык для создания выборок и отчетов) и PHP (расшифровывается как самоповторяющая аббревиатура PHP: Hypertext Preprocessor – PHP: препроцессор гипертекста). Главное требование к языкам программирования динамических Web-сайтов – это совместимость со стандартом CGI (Common Gateway Interface – общий шлюзовой интерфейс), который обеспечивает работу в «одной упряжке» Web-браузера, Web-сервера и программ, генерирующих содержимое сайта.

Еще одной достаточно популярной серверной технологией является ASP (Active Server Pages – активные серверные страницы), разрабатываемая фирмой Microsoft.

Серверные технологии создания сайтов не предъявляют никаких особенных требований к Web-браузеру пользователя, браузер получает только итоговый результат обработки данных уже в конечном виде. Но на сервер ложится значительная вычислительная нагрузка, с которой он может просто не справиться при наплыве пользователей. А, кроме того, обычно линии, соединяющие сервер и Web-браузер пользователя, не слишком скоростные, сигнал может пробежать через весь земной шар, прежде чем достигнуть компьютера. Поэтому время реак-

ции на действия пользователя может быть достаточно большим, что далеко не всегда приемлемо. Решить эту проблему, призваны «клиентские» технологии.

«Клиентом» в терминологии Всемирной паутины, называют Web-браузер конечного пользователя, а клиентскими технологиями, соответственно, технологии, с ним связанные. Суть этих технологий в том, чтобы переложить часть или всю работу по динамическому формированию страниц на Web-браузер. Есть две основные технологии этого типа: JavaScript и Flash. Технология JavaScript – это достаточно простой язык программирования, позволяющий манипулировать содержимым HTML страниц, перемещать отдельные объекты по окну Web-браузера и производить другие относительно несложные действия. Технология Flash – это система создания графических приложений, ориентированных на Web. С ее помощью создаются красочные мультфильмы, музыкальные клипы, заставки, интерактивные сайты и полноценные игры. И все это можно сделать частью Web-страницы. Главный недостаток Flash в том, что сложная Flash-анимация может полностью загрузить даже самый современный компьютер.

Достоинства клиентских технологий ясны – нет необходимости загружать Web-сервер и перекачивать с него лишние объемы информации, но, как часто бывает, недостатки являются продолжениями достоинств. Дополнительные вычисления нагружают компьютер пользователя, особенно если он не слишком быстр, кроме того, сам браузер должен уметь работать с этими клиентскими технологиями. Причем, если с поддержкой Flash все обстоит просто, достаточно скачать модуль для работы с ним с сайта производителя этой системы (фирмы Macromedia), то при использовании JavaScript это не так. Браузер должен поддерживать соответствующие сценарии JavaScript, причем написанные в разных версиях языка, которые могут несколько отличаться друг от друга.

Серверные и клиентские технологии создания динамических сайтов не соперничают, а успешно дополняют друг друга при создании действительно современных динамических сайтов.

Заключение

Теперь вы знаете, как появился Интернет, познакомились с основными особенностями его строения и главными технологиями, положенными в его основу. Перед вами открылся весь спектр возможностей, которые предоставляют различные Интернет-сервисы. Вы узнали, что такое WWW, как устроена Всемирная паутина и с помощью каких технологий создают Web-сайты. Вероятно, после знакомства с таким богатством возможностей и количеством технологий вам покажется, что создавать Web-сайты самостоятельно очень сложно, но на самом деле это не так. Разумеется, сразу вы не сможете сделать большой динамический сайт, но сделать небольшой статический сайт из нескольких страниц вам будет по силам и без особой подготовки. Как его сделать, вы узнаете из следующей главы. А набив руку в создании простых страниц, вы сможете перейти и к созданию более сложных сайтов и достаточно скоро вы сможете создавать по-настоящему профессиональные Web-сайты. Продолжайте читать книгу!

глава 2

Первые Web-страницы своими руками

Итак, вы уже познакомились с основными возможностями сети Интернет и практически не ограниченными просторами World Wide Web. Вам, безусловно, не терпится внести свой вклад в многообразный мир Всемирной паутины. Вся оставшаяся часть книги призвана помочь вам в этом благом начинании. Вам, безусловно, хотелось бы сразу создать большой, красивый и динамичный сайт, но все великое начинается с малого. В этой главе мы познакомимся с основами создания статических сайтов, с инструментами их разработки, и вы создадите свой первый статический сайт. Он будет небольшим, но вполне работоспособным.

Файлы HTML «изнутри»

Любой статический сайт состоит из одного или нескольких файлов формата HTML, называемых еще «страницами». Как уже было рассказано в первой главе, HTML – это язык гипертекстовой разметки документов. Основой языка HTML являются теги, специальные команды, заключаемые в угловые скобки, например **<BODY>**, **<A HREF=>** или **<B>**. С помощью тегов описывается структура документа, его внешний вид и ссылки на другие документы. Код простого HTML-документа приведен в Листинг 2.1.

Листинг 2.1. Простейший HTML документ

```
<HTML>
<HEAD>
<TITLE> Это простейший документ HTML</TITLE>
</HEAD>
<BODY>
Содержимое документа HTML
</BODY>
</HTML>
```

Если вы откроете документ с этим кодом в Web-браузере, то увидите результат, показанный на Рис. 2.1.

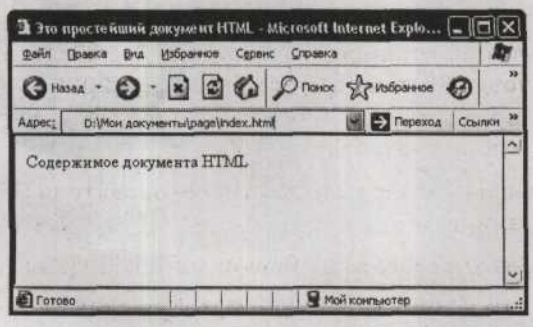


***Рис. 2.1.** Простейший HTML документ*

Есть два основных способа создания HTML-документов. Первый – написание кода HTML вручную. Для этого вы должны изучить все основные теги HTML; хорошо знать какое действие выполняет каждый из них и хорошо представлять, как будет выглядеть итоговый результат после того или иного изменения кода. С одной стороны, такое «прямое» кодирование достаточно трудоемко и требует постоянно держать в памяти массу информации, но с другой – позволяет полностью контролировать все тонкости HTML-кода и в некоторых случаях добиваться результата, недостижимого другими способами. Еще одним плюсом такого способа разработки является то, что писать код HTML вы можете абсолютно в любом текстовом редакторе, например в Блокноте Windows. Хотя и существуют различные программы, облегчающие «ручное» кодирование HTML.

Другой способ – «визуальное» составление страниц в специальном HTML-редакторе. Создавать HTML-документы таким способом можно, просто вставляя на страницу нужные элементы оформления, ссылки на другие документы, изображения, тексты и размещая их так, как необходимо. Такой способ работы часто называют WYSISWYG (What You See is What You Get – что вы видите, то и получаете). Такая методика составления HTML документов проста в освоении, позволяет быстро создавать достаточно сложные гипертекстовые документы и, кроме того, наглядна. Но недостатком такого способа работы является то, что HTML-редактор, как правило, не позволяет

реализовать все тонкости языка HTML. Поэтому визуальные HTML редакторы предусматривают и работу непосредственно с кодом страницы.

Одним из самых популярных визуальных HTML–редакторов является программа Microsoft FrontPage 2003, и в большей части этой книги рассматривается создание Web-сайтов именно с помощью этой программы.

Установка программы Microsoft FrontPage 2003

Начнем с установки программы Microsoft FrontPage 2003 на ваш компьютер.

Чтобы установить эту программу:

- Вставьте диск с программой Microsoft FrontPage в **CD**-привод компьютера. Запустится программа автозапуска диска.
- В зависимости от поставки способы запуска системы установки FrontPage могут быть различны. Руководствуясь подсказками системы автозапуска, запустите систему установки программы FrontPage. Вы должны увидеть диалог **Установка Microsoft Office FrontPage 2003** (Microsoft Office FrontPage 2003 setup) (Рис. 2.2). Через некоторое время вы увидите диалог ввода сведений о пользователе (Рис. 2.3).
- Щелкните мышью на поле ввода **Имя пользователя** (User name) и введите свои имя и фамилию.
- Щелкните мышью на поле ввода **Инициалы** (Initials) и введите свои инициалы.
- Щелкните мышью на поле ввода **Организация** (Organization) и введите название своей организации.

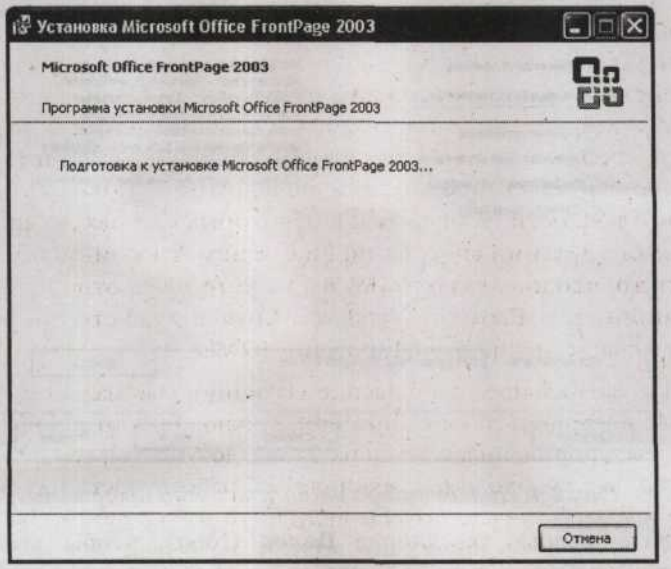


Рис. 2.2. Диалог ***Установка Microsoft Office FrontPage 2003*** *(Microsoft Office FrontPage 2003 setup)*

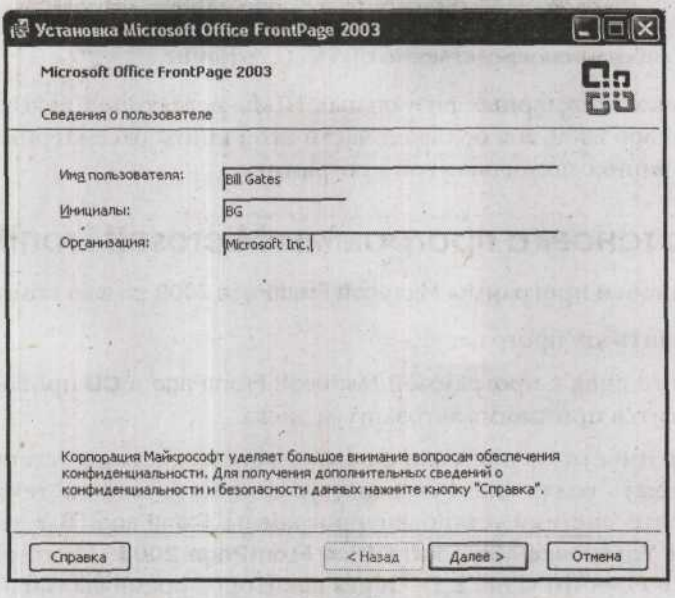


Рис. 2.3. Диалог ввода сведений о пользователе

- Щелкните мышью на кнопке **Далее** (Next), чтобы продолжить работу. Появится диалог выбора типа установки программы (Рис. 2.4).

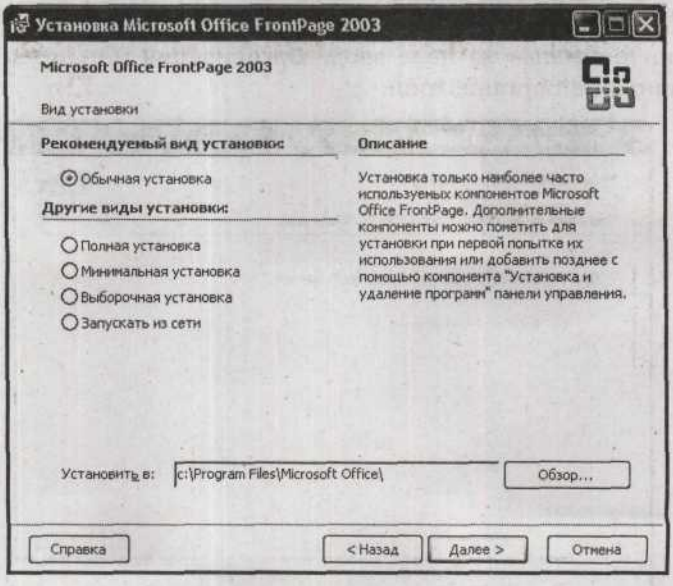


Рис. 2.4. Диалог выбора типа установки программы

- Щелкните мышью на кнопке **Далее** (Next), чтобы выбрать установку Microsoft FrontPage с наиболее часто используемыми компонентами. Откроется диалог общих сведений о выбранных настройках установки (Рис. 2.5).

Рис. 2.5. Диалог общих сведений о выбранных настройках установки

- Щелкните мышью на кнопке **Установить** (Install), чтобы начать установку программы Microsoft FrontPage на ваш компьютер. Появится диалог хода установки программы (Рис. 2.6), в графическом виде отображающий процесс установки Microsoft FrontPage.

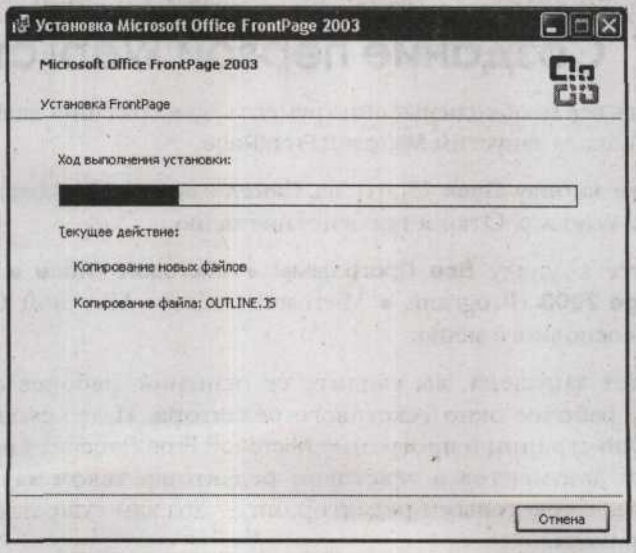


Рис. 2.6. Диалог хода установки программы

- Когда будут установлены все необходимые файлы и произведены требуемые настройки, появится диалог завершения установки программы (Рис. 2.7).

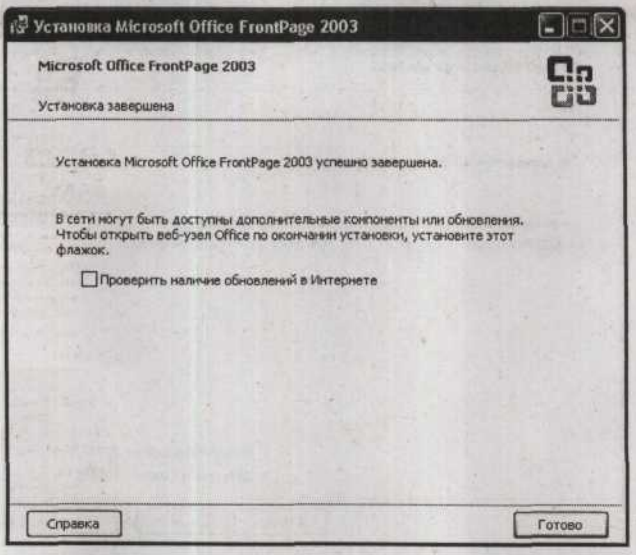


Рис. 2.7. Диалог завершения установки программы

- Щелкните мышью на кнопке **Готово** (Finish), чтобы завершить процесс установки программы.

Теперь программа Microsoft FrontPage успешно установлена.

Создание первой Web-страницы

Сейчас у вас есть все необходимые инструменты для создания вашей первой Web-страницы. Для начала запустим Microsoft FrontPage.

- Нажмите кнопку **Пуск** (Start) на **Панели задач** (Taskbar) операционной системы Windows. Откроется основное меню.
- Выберите команду **Все Программы ◆ Microsoft Office ◆ Microsoft Office FrontPage 2003** (Programs ◆ Microsoft Office ◆ Microsoft Office FrontPage 2003) из основного меню.

Программа будет запущена, вы увидите ее основное рабочее окно (Рис. 2.8). Оно похоже на рабочее окно текстового редактора. И это сходство не случайно. Создание Web-страниц в программе Microsoft FrontPage во многом похоже на создание новых документов в текстовом редакторе, таком как Microsoft Word. Если вы знакомы с текстовыми редакторами, – это вам существенно поможет в освоении программы.

Давайте создадим вашу первую Web-страницу.

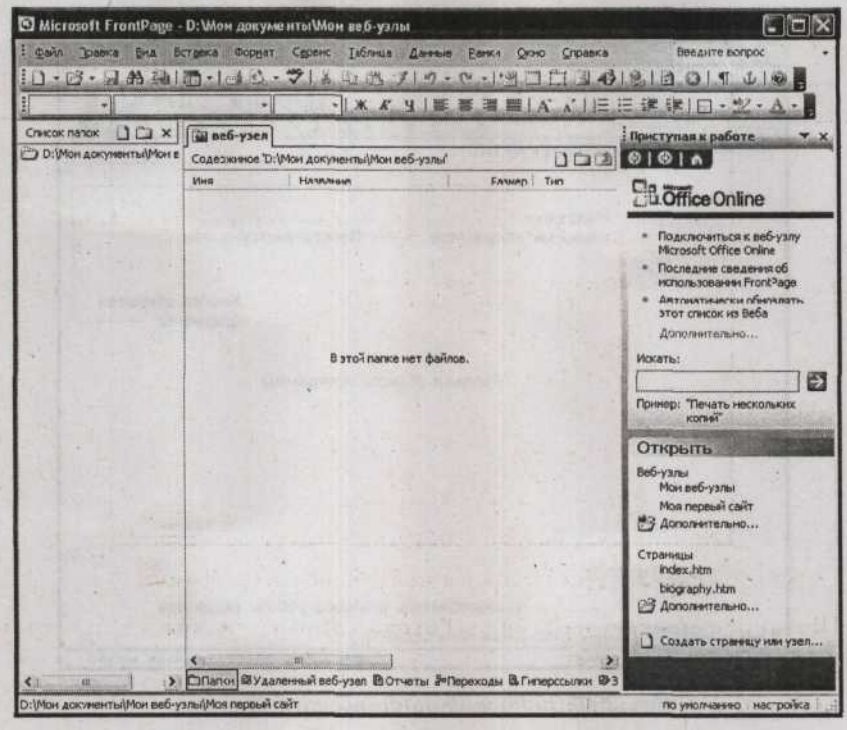


Рис. 2.8. Основное рабочее окно программы Microsoft FrontPage

Создание Web-страницы

Создайте новый HTML-документ:

- Выберите пункт меню **Файл ♦ Создать** (File ♦ New). В правой части окна программы откроется панель **Создание** (Create) (Рис. 2.9).
- Щелкните мышью на строке **Пустая страница** (Empty Page) группы **Создать страницу** (Create new page), панели **Создание** (Create). Программа FrontPage откроет для редактирования новый HTML-документ (Рис. 2.10).

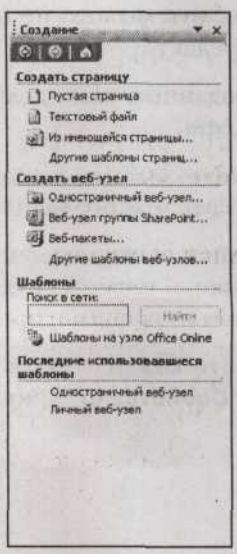


*Рис. 2.9. Панель **Создание** (Create)*

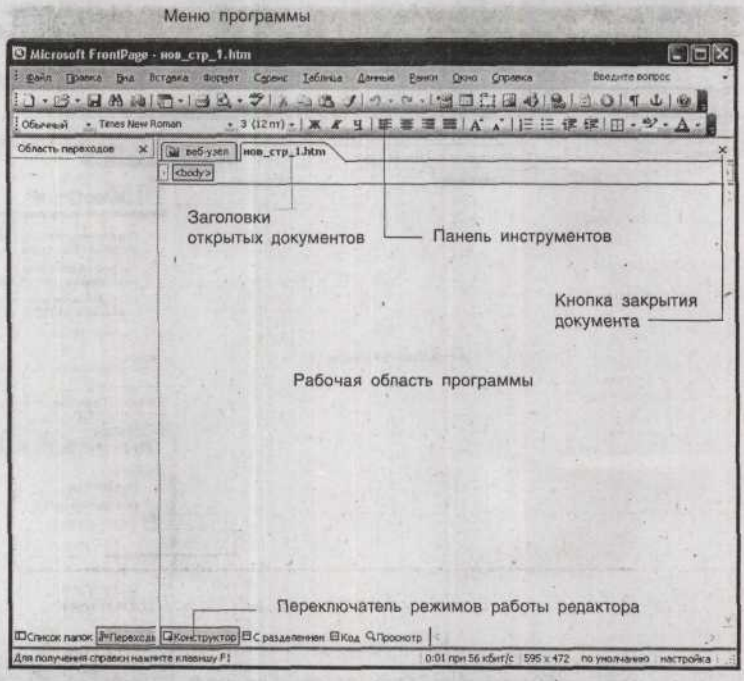


Рис. 2.10. Редактирование нового документа

Вся правая часть рабочего окна FrontPage отведена под макет создаваемой страницы.

Непосредственно над рабочей областью программы располагается область заголовков документов. В ней отображаются названия всех документов, открытых для редактирования, а также некоторых вспомогательных модулей программы FrontPage. Щелкнув мышью на названии документа в области заголовков вы перейдете к его редактированию.

Только что созданный вами документ, пока вы не дали ему название, озаглавлен как **нов_стр_1.htm.**

- Щелкните мышью в области макета страницы и введите с клавиатуры текст: «Это моя первая Web-страница».

Чтобы созданный вами документ везде корректно определялся как написанный на русском языке, а также для удобства просмотра исходного кода страницы, что понадобится нам чуть позже, проделайте следующее:

- Выберите команду меню **Файл ◆ Свойства** (File ◆ Properties). Откроется диалог **Свойства страницы** (Page properties), Рис. 2.11.

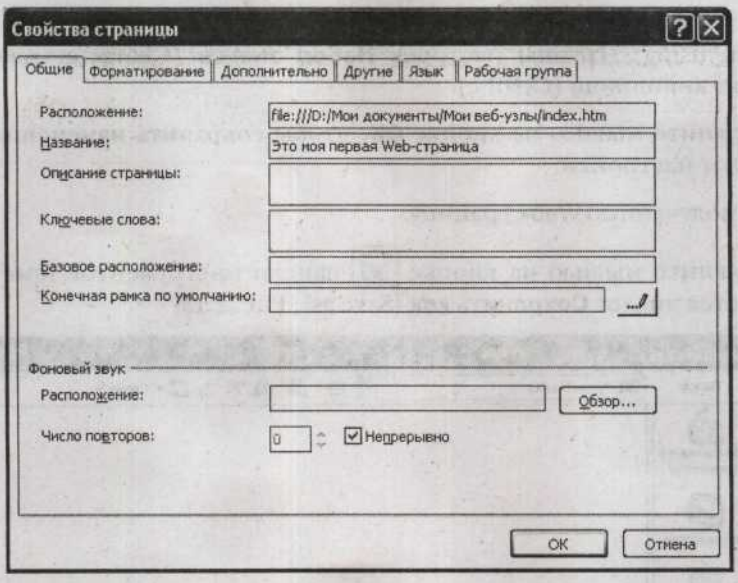


Рис. 2.11. Диалог ***Свойства страницы*** *(Page Properties)*

- Щелкните мышью на вкладке **Язык** (Language), откроется диалог настройки языка страницы (Рис. 2.12).

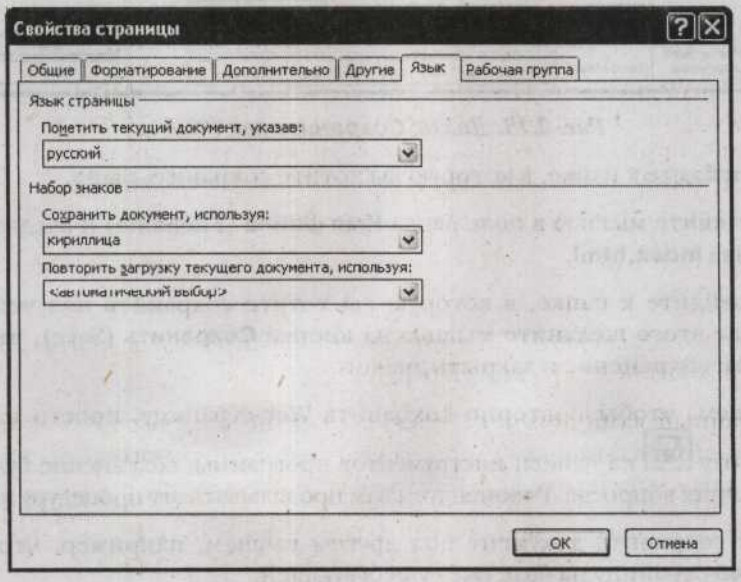


Рис. 2.12. Вкладка ***Язык*** *(Language), диалога* ***Свойства Страницы*** *(Page properties)*

- В раскрывающемся списке **Пометить текущий документ, указав:** (Mark current document as), группы настроек **Язык страницы** (Page Language), выберите пункт **русский** (russian).

- В раскрывающемся списке **Сохранить документ, используя** (Save document using), группы настроек **Набор знаков** (Character set), выберите пункт **кириллица** (Cyrillic).
- Щелкните мышью на кнопке **OK**, чтобы сохранить изменения и закрыть диалог настройки.

Сохраните полученную Web-страницу:

- Щелкните мышью на кнопке  панели инструментов программы. Откроется диалог **Сохранить как** (Save as), Рис. 2.13.

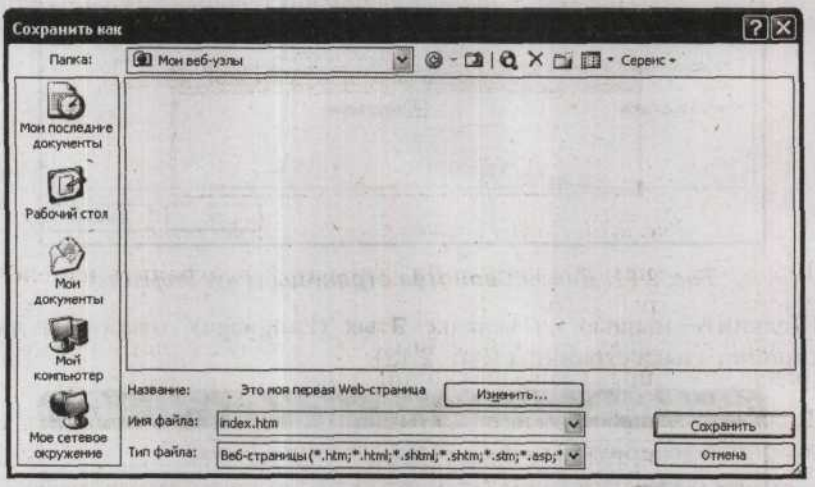


Рис. 2.13. Диалог ***Сохранить как*** *(Save as)*

- Перейдите к папке, в которую вы хотите сохранить файл.
- Щелкните мышью в поле ввода **Имя файла** (Filename) и введите название файла **index.html**.
- Перейдите к папке, в которую вы хотите сохранить полученный файл, после этого щелкните мышью на кнопке **Сохранить** (Save), чтобы произвести сохранение и закрыть диалог.

В дальнейшем, чтобы повторно сохранить Web-страницу, просто нажмите мышью на кнопку  на панели инструментов программы, сохранение будет произведено без лишних вопросов. Рекомендуем вам проделывать эту процедуру почаще.

Вы можете сохранить документ под другим именем, например, чтобы создать еще одну Web-страницу на базе уже существующей.

- Выберите команду меню **Файл ♦ Сохранить как** (File ♦ Save as). Откроется диалог сохранения файла (Рис. 2.13).
- Перейдите к папке, в которую вы хотите сохранить файл.

- Щелкните мышью в поле ввода **Имя файла** (Filename) и введите новое название файла.
- Перейдите к папке, в которую вы хотите сохранить полученный файл, после этого щелкните мышью на кнопке **Сохранить** (Save), чтобы произвести сохранение и закрыть диалог.

Закончив работу с документом, вы можете закрыть его. Для этого:

- Щелкните мышью на кнопке закрытия документа, в области заголовков документов (Рис. 2.8).

Если документ перед закрытием не был сохранен, появится диалог запроса о сохранении документа (Рис. 2.14).

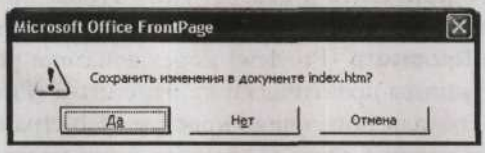


Рис. 2.14. Диалог запроса о сохранении документа

- Щелкните мышью на кнопке **Да** (Yes), чтобы сохранить изменения в документе и закрыть его.
- Щелкните мышью на кнопке **Нет** (No), чтобы закрыть документ, не сохраняя внесенные в него изменения.
- Щелкните мышью на кнопке **Отмена** (Cancel), чтобы отменить закрытие документа и вернуться к его редактированию.

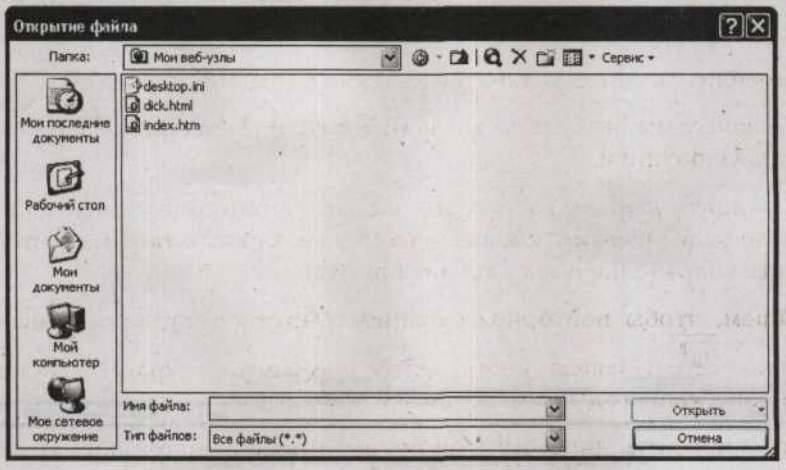


Рис. 2.15. Диалог ***Открытие файла*** *(Open file)*

Чтобы снова открыть для редактирования закрытый документ:

- Выберите команду меню **Файл ♦ Открыть** (File ♦ Open). На экране появится диалог **Открытие файла** (Open file), Рис. 2.15.

- Перейдите к папке, в которой находится нужный вам файл и щелкните мышью на его названии, файл выделится цветом.
- Щелкните мышью на кнопке **Открыть** (Open), чтобы открыть файл для редактирования и закрыть диалог.
- Итак, вы создали свою самую первую Web-страницу. Теперь можно просмотреть полученный результат.

Просмотр созданных Web-страниц в браузере

В программу FrontPage встроен свой собственный Web-браузер и при создании Web-сайтов вы можете прибегать к его услугам. Чтобы просмотреть только что сделанную вами страницу во встроенном браузере программы FrontPage, щелкните мышью на кнопке **Просмотр** (Preview) переключателя режимов работы редактора. Внешний вид страницы практически не изменится (Рис. 2.16), это и неудивительно, поскольку вы создали пока очень простую Web-страницу. В более сложных случаях изображения в режиме создания макета и просмотра результата могут несколько отличаться друг от друга.

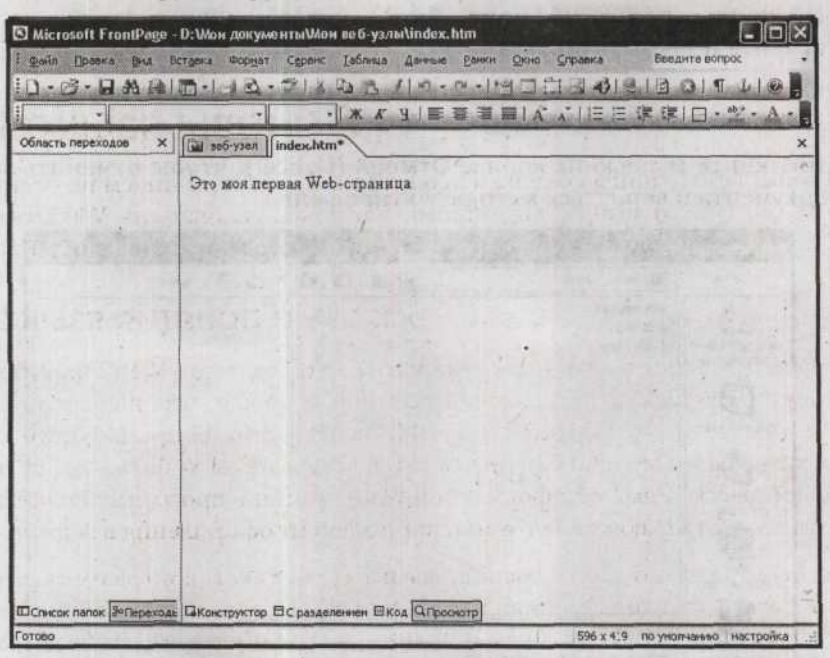


Рис. 2.16. Просмотр созданной Web-страницы во встроенном браузере программы FrontPage

Разумеется, созданную HTML-страницу можно просмотреть и в любом Web-браузере, установленном на вашем компьютере. Программа FrontPage предусматривает удобный способ открытия Web-страниц в браузере Internet Explorer.

- Выберите команду меню **Файл ♦ Просмотреть в обозревателе ♦ Microsoft Internet Explorer** (File ♦ Preview in web-browser ♦ Microsoft Internet Explorer). Текущая HTML-страница откроется в Web-браузере Internet Explorer (Рис. 2.17).

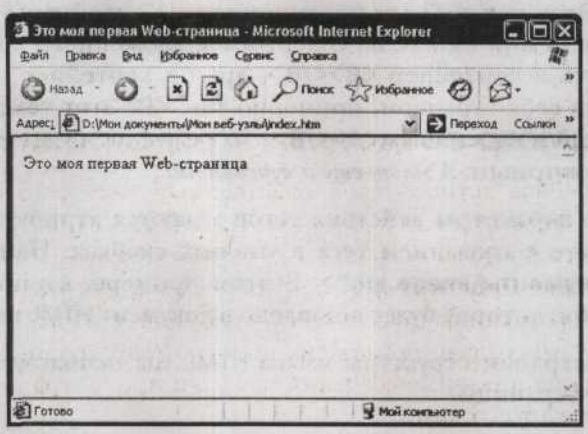


Рис. 2.17. Просмотр результатов работы в браузере Internet Explorer

Основы языка HTML

Ваша первая Web-страница создана. Но внешний вид ее пока ничем не отличается от обычного текстового файла, созданного, например, в Блокноте Windows. Давайте разберемся, какие возможности по оформлению текстов предоставляет HTML.

Базовые понятия языка HTML

Как мы уже говорили выше, несмотря на то, что создавать Web-страницы средствами редактора WYSIWYG несравненно быстрей и проще, чем напрямую вводя код HTML, но, тем не менее, знание азов HTML необходимо для разработки серьезных Web-проектов. Поэтому, сначала мы вкратце рассмотрим основы языка HTML, а в дальнейшем, рассказывая об оформлении Web-страниц в программе FrontPage, будем приводить теги, отвечающие за тот или иной элемент оформления в коде страницы.

Как уже было сказано ранее, основу языка гипертекстовой разметки HTML, составляют теги – текстовые команды, заключенные в угловые скобки. Теги бывают одиночными и парными. Одиночные теги, как и следует из их названия, используются поодиночке, обозначая самостоятельный элемент оформления, Например, тег **
** означает разрыв строки, а с помощью тега **<IMG>** в HTML-документ можно вставить изображение. Теги можно записывать как прописными буквами, например так: **<TITLE>**, так и строчными, например **<title>**.

Парные теги образуются двумя тегами, открывающим и закрывающим, которые отличаются друг от друга только значком «/» перед закрывающим тегом. Парные теги влияют на оформление всего кода HTML, находящегося между ними. Парные

теги еще называют «контейнером», а код, находящийся между ними, «содержимым контейнера». Примером использования контейнера будет следующий код: **<B>Этот текст будет полужирным </B>**. Весь текст, помещающийся в контейнер **<B></B>** оформляется полужирным шрифтом и в Web-браузере будет выглядеть примерно так: «**Этот текст будет полужирным**». Контейнеры могут вкладываться друг в друга, действуя вместе на содержимое вложенного контейнера. Например, если вложить в контейнер **<B></B>**, другой контейнер, **<I></I>**, выделяющий текст внутри себя курсивом, примерно так: **<B>Этот текст будет полужирным <I>А этот еще и курсивным</I></B>**, то результат будет следующим: «**Этот текст будет полужирным. *А этот еще и курсивным***».

Дополнительные параметры действия тегов задаются атрибутами. Атрибуты записываются вместе с названием тега в угловых скобках. Например, так: **<IMG SRC="c:\мои документы\image.jpg">**. В этом примере, атрибут **SRC** указывает адрес изображения, которое будет вставлено в документ HTML на место тега **IMG**.

В качестве иллюстрации структуры языка HTML мы используем только что созданную вами Web-страницу.

- Чтобы просмотреть код Web-страницы, щелкните мышью на кнопке **Код** (Code) переключателя режимов работы редактора (Рис. 2.18).

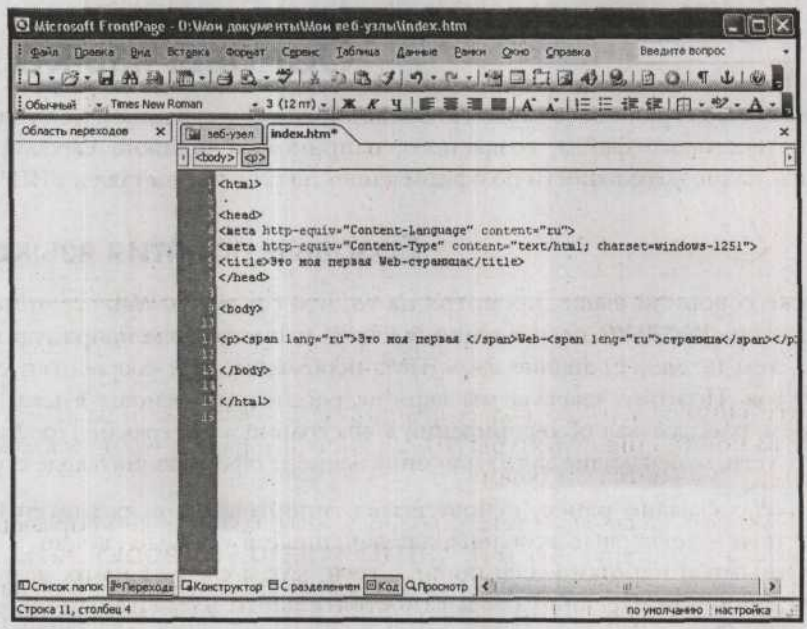


Рис. 2.18. Программа FrontPage в режиме работы с кодом страницы

Макет страницы в рабочей области программы FrontPage будет заменен на его HTML-код. При желании можно создавать Web-страницы непосредственно в этом режиме, напрямую вводя весь код.

Для понимания принципов работы HTML удобнее будет одновременно видеть как код страницы, так и ее макет, чтобы можно было заметить, какие теги как влияют на оформление документа, и наоборот – как изменение документа влияет на код страницы.

- Щелкните мышью на кнопке **С разделением** (Divide) переключателя режимов работы редактора.

Рабочая область FrontPage разделится на две части, в нижней будет макет страницы, а в верхней части будет размещаться HTML-код вашего документа (Рис. 2.19).

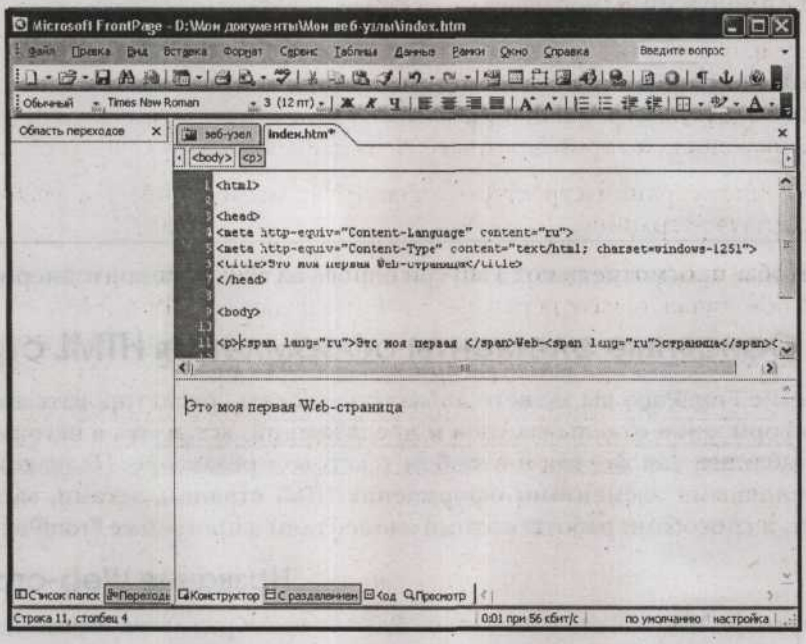


Рис. 2.19. *Одновременная работа с макетом страницы и HTML-кодом в программе FrontPage*

Посмотрите внимательно на код созданной вами Web-страницы (Листинг 2.2). Для удобства понимания структуры HTML в листинге разные уровни вложенности контейнеров отмечаются отступами.

В языке HTML есть несколько тегов, которые отвечают не за оформление документа, а за его структуру. Это теги **<HTML>**, **<HEAD>** и **<BODY>**. Весь документ HTML заключается в один большой контейнер **<HTML></HTML>**. Эта пара тегов сообщает Web-браузеру о том, что документ написан на языке HTML, а не на каком-либо другом языке разметки. Этот контейнер в свою очередь содержит еще два контейнера. Первый, **<HEAD></HEAD>**, называется заголовком, и в нем содержится разнообразная информация о содержимом всего документа: язык, на котором написаны тексты, версия языка HTML, на которую ориентировались создатели документа, основные ключевые слова для поисковых «ботов», общий заголовок HTML-документа и прочее. Второй контейнер, **<BODY></BODY>** включает в себя все основное содержимое документа, его «тело».

Листинг 2.2. Код вашей первой Web-страницы

```
<html>
<head>
<meta http-equiv="Content-Language" content="ru">
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
<title>Это моя первая Web-страница</title>
</head>
<body>
<p><span lang="ru">Это моя первая </span>Web-<span
lang="ru">страница</span></p>
</body>
</html>
```

Все остальные теги размещаются внутри одного из этих двух контейнеров.

Основные элементы оформления HTML страниц

В программе FrontPage вы можете добавлять, удалять, редактировать тексты, изменять оформление отдельных слов и предложений, вставлять в него заголовки, списки, таблицы, так же, как и в любом текстовом редакторе. Познакомимся теперь с основными элементами оформления HTML-страниц, тегами, за это отвечающими, и способами работы с этими элементами в программе FrontPage.

Название Web-страницы

При просмотре Web-сайтов в заголовке браузера отображается название текущей страницы. За название Web-страницы отвечает парный тег **<TITLE>**, размещаемый в заголовке HTML-документа. Название страницы размещается между открывающим и закрывающим тегами **<TITLE>**. Например, так: **<TITLE>Это заголовок Web-страницы</TITLE>**. Это единственный тег в заголовочной части HTML-документа, действие которого отображается непосредственно в браузере.

Чтобы изменить заголовок Web-страницы в программе FrontPage:

- Выберите команду меню **Файл ♦ Свойства** (File ♦ Properties). Откроется диалог **Свойства страницы** (Page properties), Рис. 2.20.
- Щелкните мышью на поле ввода **Название** (Title) и введите другое название страницы, например: «Хороший заголовок – половина успеха».
- Щелкните мышью на кнопке ОК, чтобы принять изменения и закрыть диалог.

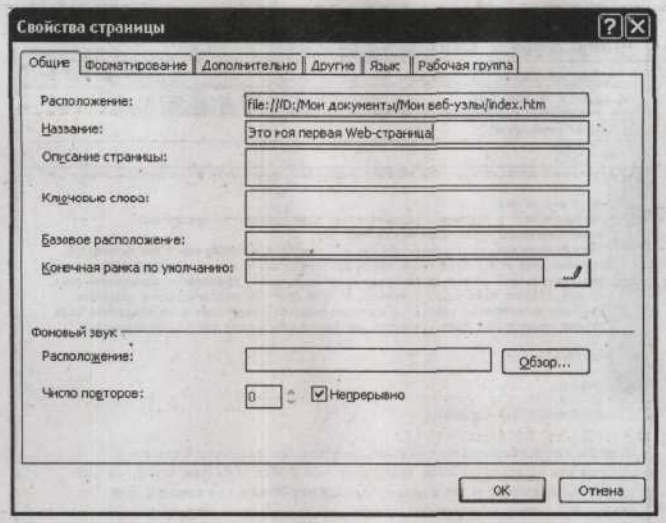


Рис. 2.20. Диалог ***Свойства страницы*** *(Page Properties)*

Теперь вы изменили название Web-страницы. Посмотрев HTML-код документа, вы можете убедиться, что текст между тегами **<TITLE>** тоже изменился. Мы рекомендуем вам, чтобы лучше познакомиться с языком HTML, после каждого изменения Web-страницы смотреть на изменения, которые произошли с ее кодом, и самостоятельно пробовать вносить изменения в HTML-теги.

Форматирование абзацев текста

Удалите все содержимое текущей Web-страницы.

- Щелкните мышью в области макета страницы.
- Выполните пункт меню **Правка ◆ Выделить все** (Edit ◆ Select All).
- Нажмите клавишу Delete. Все содержимое Web-страницы будет удалено.

Чтобы начать новый абзац текста, просто нажмите клавишу Enter. В языке HTML абзац заключается в контейнер парным тегом **<P>** (Рис. 2.21).

Содержимое абзаца может быть выровнено относительно окна просмотра (Рис. 2.22). В HTML выравнивание абзаца задается атрибутом **ALIGN** тега **<P>**. Он может принимать четыре значения. **LEFT** – выравнивание по левому краю окна, **RIGHT** – по правому, **CENTER** – по центру и **JUSTIFY** – выравнивание текста по ширине окна. Задать выравнивание текущего абзаца в программе FrontPage можно, щелкнув мышью на соответствующей кнопке в панели инструментов. Щелкнув мышью на кнопке , вы выровняете текст абзаца по левому краю окна. Щелкнув на кнопке – по правому. Кнопка отвечает за выравнивание текста по центру, а за выравнивание по ширине текста.

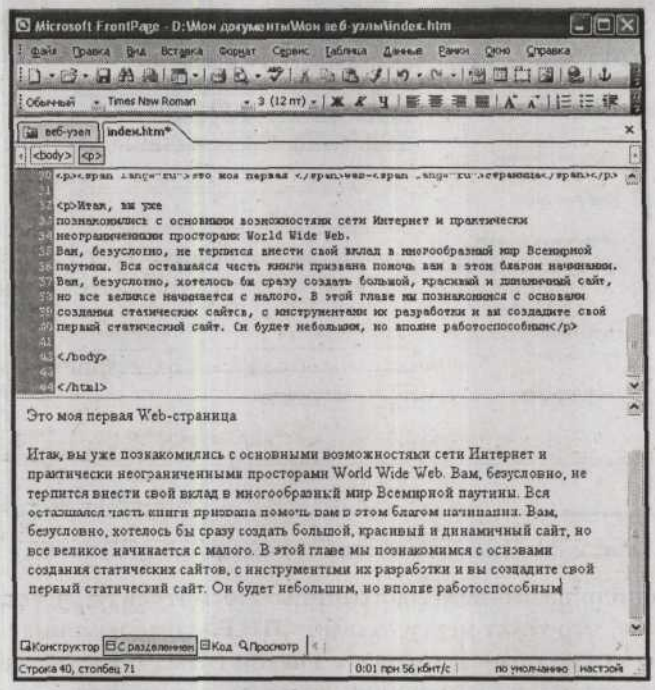


Рис. 2.21. Оформление абзаца в HTML-документе

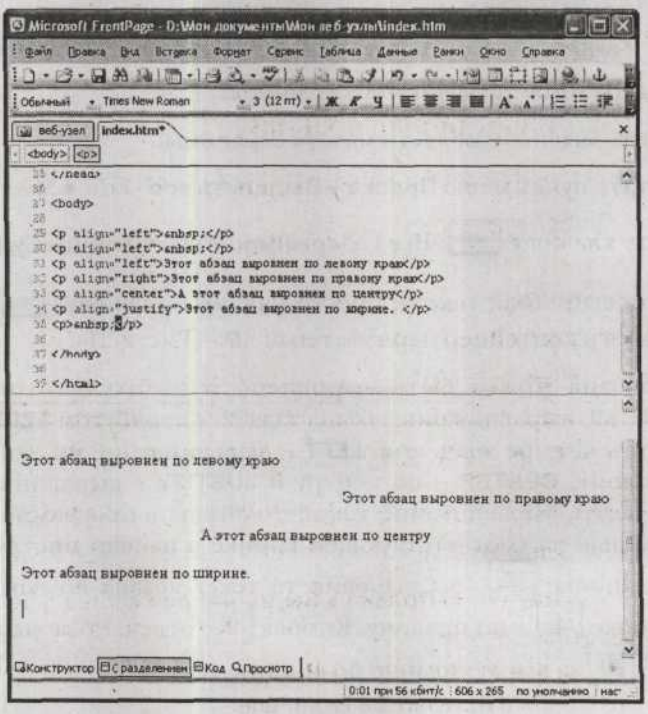


Рис. 2.22. Выравнивание абзацев текста в HTML

Заголовки

Основой структурирования текста и разбиения его на отдельные смысловые фрагменты, в HTML являются заголовки (Рис. 2.23). Язык HTML поддерживает шесть уровней заголовков. Они создаются с помощью контейнеров **<H1></H2>**, **<H2></H2>** и до **<H6></H6>**, где цифра, стоящая после буквы «H» означает уровень заголовка.

Заголовки первого уровня обычно содержат название всего документа. Заголовки второго уровня – названия основных разделов, третьего – названия подразделов и т. д.

Правильно расставленные заголовки позволяют проще ориентироваться в документе и легко находить нужную информацию.

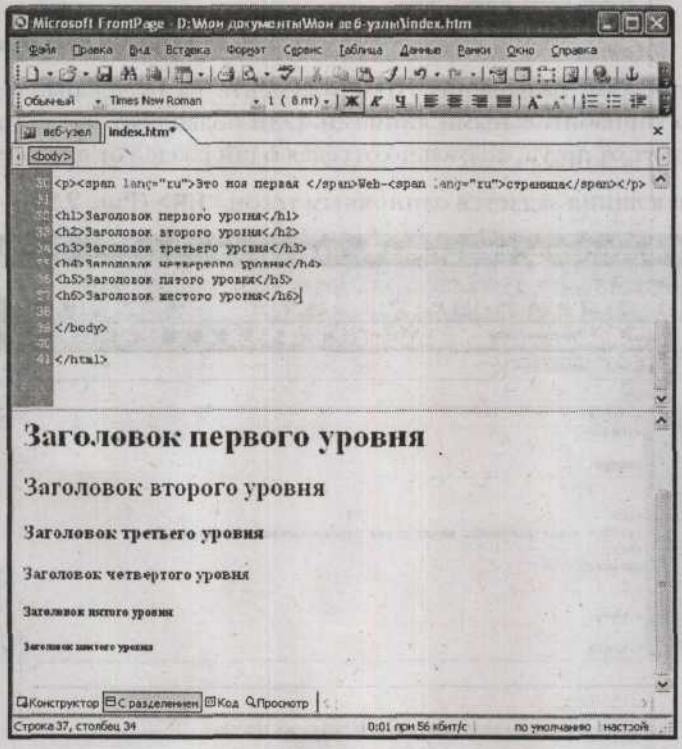


Рис. 2.23. Заголовки в HTML

Удалите все содержимое текущей Web-страницы.

- Щелкните мышью в области макета страницы.
- Выполните пункт меню **Правка ◆ Выделить все** (Edit ◆ Select All).
- Нажмите клавишу Delete. Все содержимое Web-страницы будет удалено.

Чтобы создать заголовок в программе FrontPage:

- Щелкните мышью на выпадающем меню **Стиль** (Style) – Обычный, панели инструментов.
- В открывшемся меню щелкните мышью на одной из строк, подписанных **Заголовок 1**, **... Заголовок 6** (Header 1 ... Header 6), в зависимости от того, заголовок какого уровня вы хотите создать.
- Приступайте к вводу текста заголовка с клавиатуры.
- Чтобы закончить ввод заголовка, нажмите клавишу Enter. Вы начнете новый абзац обычного текста.

Как и обычные абзацы, заголовки можно выравнивать относительно окна просмотра кнопками , , и .

Горизонтальные линии

Еще одним способом разделения HTML-документа на отдельные части является разбиение его горизонтальными линиями. Они позволяют четко отделять разделы страницы друг от друга, визуально отсекая один раздел от другого.

Горизонтальная линия задается одиночным тегом **<HR>** (Рис. 2.24).

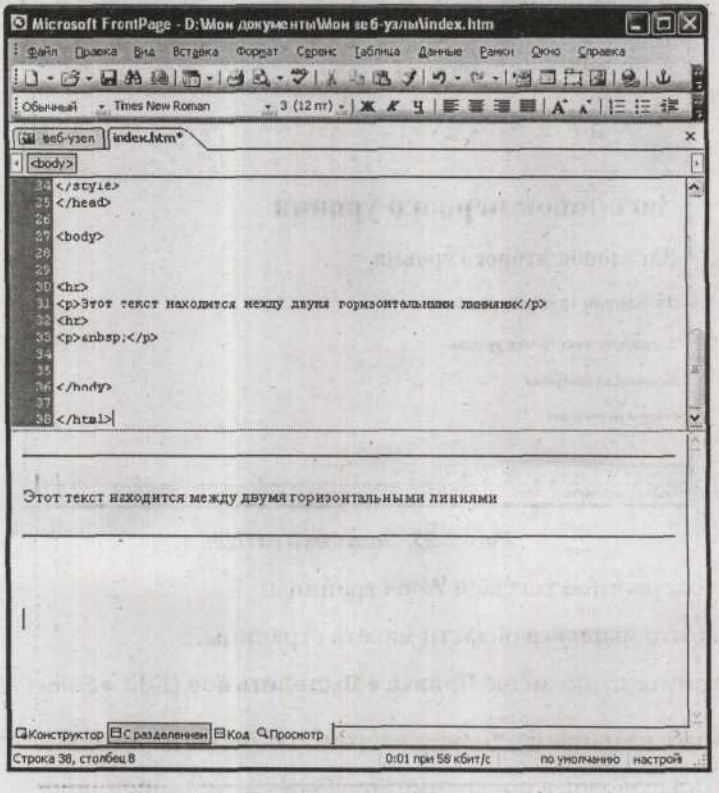


Рис. 2.24. Горизонтальные линии в HTML

Удалите все содержимое текущей Web-страницы.

- Щелкните мышью в области макета страницы.
- Выполните пункт меню **Правка ♦ Выделить все** (Edit ♦ Select All).
- Нажмите клавишу Delete. Все содержимое Web-страницы будет удалено.

Чтобы в программе FrontPage вставить на Web-страницу горизонтальную линию:

- Щелкните мышью в месте страницы, в которое вы хотите вставить линию.
- Выберите пункт меню **Вставка ♦ Горизонтальная линия** (Insert ♦ Horizontal Line). Линия будет вставлена.

Чтобы удалить горизонтальную линию со страницы:

- Щелкните мышью на линии, которую хотите удалить. Линия будет выделена черным цветом.
- Нажмите на клавишу Delete. Линия будет удалена.

Списки

Одним из часто используемых способов упорядочивания текста являются списки. Они бывают нумерованными, маркированными, в виде меню и в виде списка определений.

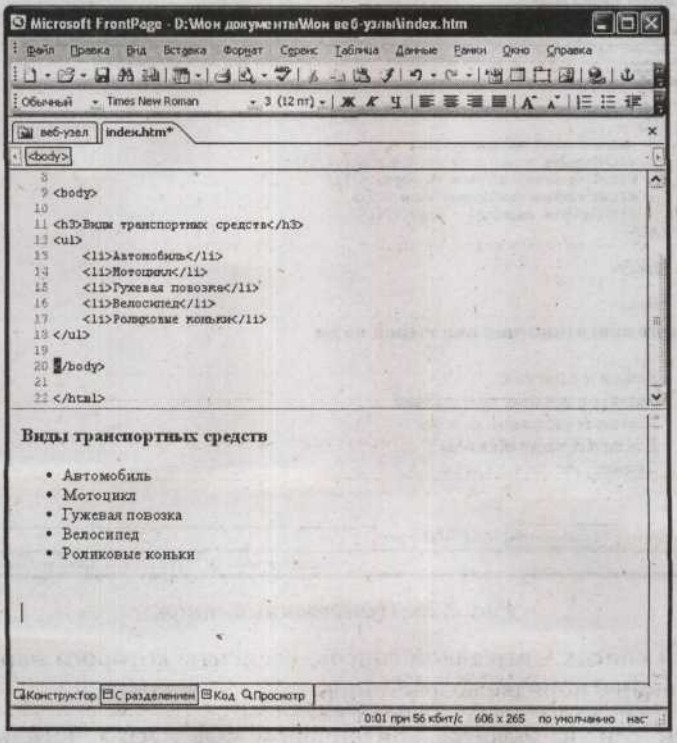


Рис. 2.25. Маркированный список

Для оформления ряда похожих элементов лучше всего подходят маркированные списки (Рис. 2.25). Каждый элемент такого списка выделяется специальным значком, маркером.

Чтобы закончить создание списка:

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строчке, подписанной **Обычный** (Normal)

Либо

- Дважды нажмите клавишу Enter.

Для оформления различных рецептов, пошаговых инструкций и различных перечислений, в которых порядок элементов имеет значение, целесообразно использовать нумерованные списки (Рис. 2.26).

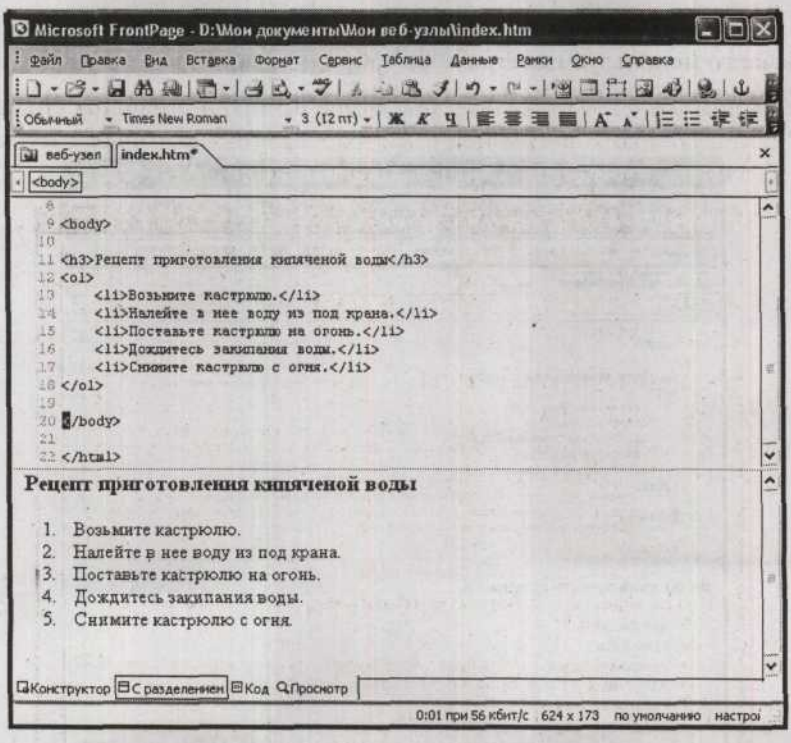


Рис. 2.26. Нумерованный список

Нумерованный список – это такой список, элементы которого маркируются числами либо буквами в порядке возрастания.

Нумерованный список задается контейнером **<OL></OL>**. Каждый отдельный элемент списка содержится в контейнере **<LI></LI>**.

Чтобы создать нумерованный список:

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строчке, подписанной **Нумерованный список** (Ordered list)
- Можете приступать к вводу содержимого списка.

В процессе создания нового списка нажатием клавиши Enter вы будете создавать новый его элемент.

Чтобы закончить создание списка:

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строчке, подписанной **Обычный** (Normal).

Либо

- Дважды нажмите клавишу Enter.

Для составления списков терминов или понятий предназначены списки определений (Рис. 2.27). Это такие списки, каждый элемент которых состоит, в свою очередь, из двух подэлементов: термина и его описания.

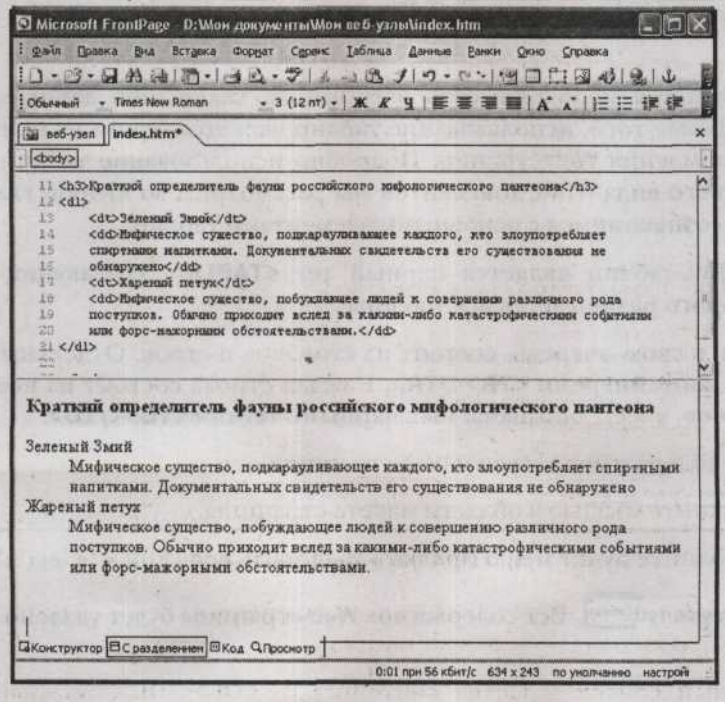


Рис. 2.27. Список определений

Список определений задается контейнером **<DL></DL>**. Термины задаются контейнерами **<DT></DT>**, а их описания контейнерами **<DD></DD>**.

Чтобы создать список определений:

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строчке, подписанной **Определенный термин** (Determined term).
- Можете приступать к вводу содержимого списка.

В процессе создания нового списка нажатием клавиши Enter вы будете создавать новый его элемент. Термины и их описания будут чередоваться.

Чтобы закончить создание списка:

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строчке, подписанной **Обычный** (Normal).

Либо

- Дважды нажмите клавишу Enter.

Таблицы

Часто встречается необходимость располагать какие-либо данные в таблицах (Рис. 2.28). Более того, использование таблиц является одним из популярнейших средств оформления Web-страниц. Подробно использование таблиц для оформления внешнего вида HTML-документов мы рассмотрим во второй главе, а сейчас лишь кратко ознакомимся с основными элементами таблиц.

Основой HTML-таблиц является парный тег **<TABLE>**, создающий контейнер, внутри которого размещается вся таблица.

Вся таблица, в свою очередь, состоит из столбцов и строк. Отдельные строки определяются контейнерами **<TR></TR>**. Каждая строка состоит из нескольких, по числу столбцов, ячеек, обозначаемых парными тегами **<TD></TD>**.

Удалите все содержимое текущей Web-страницы.

- Щелкните мышью в области макета страницы.
- Выполните пункт меню **Правка ♦ Выделить все** (Edit ♦ Select All).

Нажмите клавишу Delete. Все содержимое Web-страницы будет удалено.

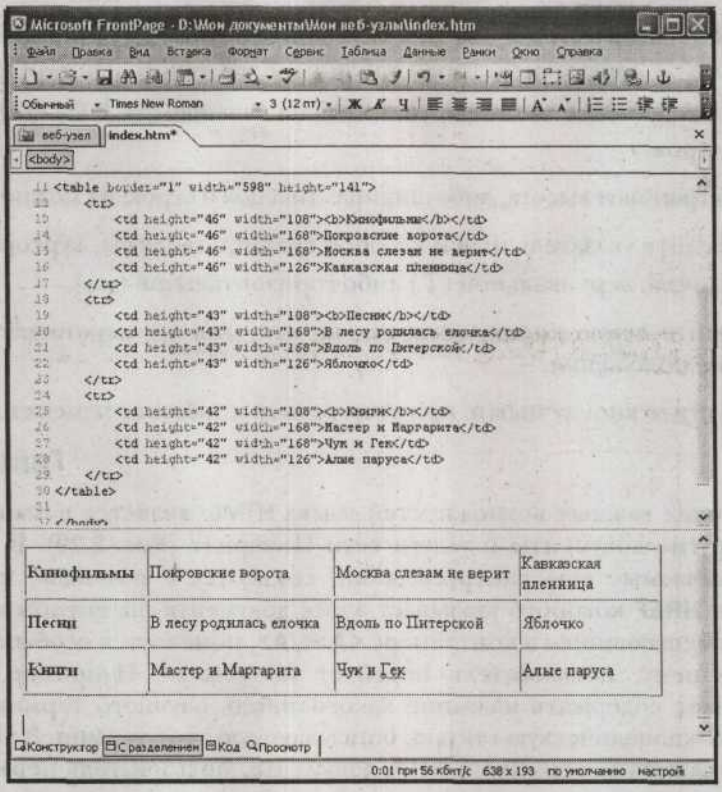


Рис. 2.28. Таблица

Теперь создадим новую таблицу в программе FrontPage.

- Щелкните мышью на кнопке **Добавить таблицу** (New Table) ( ) панели инструментов программы. Появится меню выбора конфигурации таблицы.

В меню выбора конфигурации таблицы в схематическом виде представлена создаваемая вами таблица, каждый квадратик меню соответствует одной ячейке.

- Перемещая курсор мыши, добейтесь, чтобы в меню темным цветом было выделено столько строк и столбцов, сколько вам необходимо в создаваемой таблице.
- Выделив нужное количество квадратиков в меню, щелкните левой кнопкой мыши. Таблица будет создана.

Щелкнув мышью внутри любой ячейки созданной таблицы, вы получите возможность ввести туда текст.

При редактировании таблицы, нажимая клавишу Tab, вы будете перемещать текстовый курсор последовательно по ячейкам таблицы. С каждым нажатием клавиши Tab курсор будет смещаться на ячейку вправо, пока не достигнет пра-

вой границы таблицы. После этого, нажатие на клавишу Tab приведет к переходу в левую ячейку следующей строки таблицы. Если текстовый курсор находился в правой нижней ячейке таблицы, то, нажав на клавишу Tab, вы создадите в таблице новую строку.

Если вас не устраивают высота, либо ширина столбцов и строк, их можно изменить.

- Подведите указатель мыши к любой границе таблицы, курсор примет вид стрелочки, вертикальной (↕) либо горизонтальной (↔).
- Нажмите левую кнопку мыши и, не отпуская ее, перетащите границу в другое положение.
- Отпустите кнопку мыши, при этом размеры таблицы изменятся.

Гиперссылки

Одной из самых важных возможностей языка HTML, является возможность ссылаться на другие документы и услуги сети Интернет (Рис. 2.29). В языке HTML ссылки, называемые еще гиперссылками, создаются с помощью парного тега **<A>**, атрибут **HREF** которого указывает адрес документа, на который ведет ссылка. Текст, помещающийся в контейнере **<A></A>**, помечается особым образом, и, щелкнув по нему, пользователь перейдет по ссылке. Например, контейнер **<A></A>** может содержать название какого-нибудь научного термина, а ссылка вести на энциклопедическую статью, описывающую этот термин. В таком случае, щелкнув по названию термина в HTML-документе, пользователь перейдет к документу, содержащему описание этого термина. Содержимое контейнера **<A></A>** называется элементом привязки.

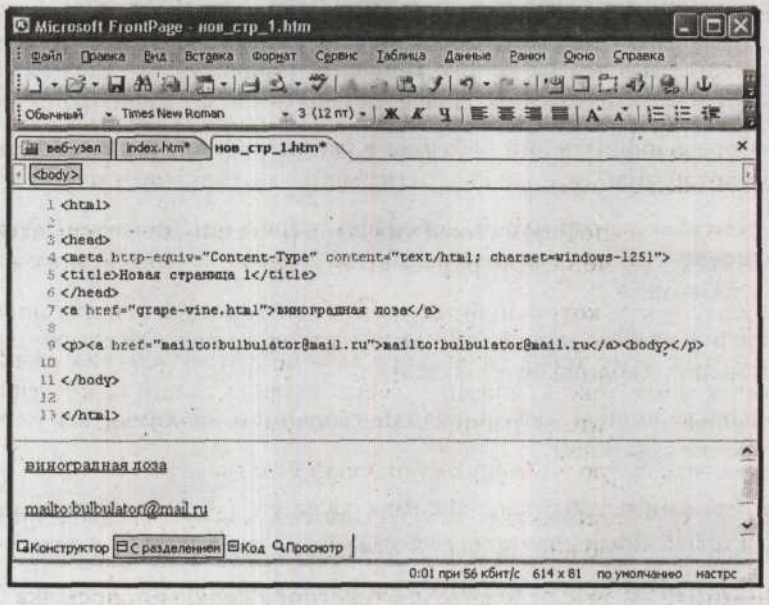


Рис. 2.29. Ссылка в HTML-документе

Ссылка может вести на другой документ, на определенную часть документа или вызывать некоторое действие, например создание письма электронной почты на определенный адрес или скачивание на компьютер пользователя какого-нибудь файла.

Удалите все содержимое текущей Web-страницы.

- Щелкните мышью в области макета страницы.
- Выполните пункт меню **Правка ♦ Выделить все** (Edit ♦ Select All).
- Нажмите клавишу Delete. Все содержимое Web-страницы будет удалено.

Теперь создадим гиперссылку в программе FrontPage:

- Выберите команду меню **Вставка ♦ Гиперссылка** (Insert ♦ Hyperlink).

Либо

- Нажмите кнопку на панели инструментов программы.

Появится диалог **Добавление гиперссылки** (Add hyperlink), Рис. 2.30. С помощью этого диалога можно создать несколько типов гиперссылок, но мы пока ограничимся созданием ссылки на новый документ, который будет создан после создания ссылки на него. В следующей главе мы рассмотрим работу с механизмом гиперссылок более подробно.

- Нажмите кнопку **новым документом** (new document) на панели кнопок **Связать с** (Link with); диалог примет вид, представленный на Рис. 2.31.
- Щелкните мышью в поле ввода **Текст** (Text) и введите с клавиатуры слово «ссылка». Этот текст будет элементом привязки создаваемой ссылки.
- Щелкните мышью в поле ввода **Имя нового документа** (Document name) и введите название **link.htm**. Это будет название нового документа, на который создается ссылка.
- Щелкните мышью на кнопке **OK**. Будет создана гиперссылка в текущем HTML-документе, а также будет создан и открыт для редактирования новый HTML-документ.
- Еще одним способом создания гиперссылки является использование контекстного меню программы FrontPage.
- Введите текст, который будет элементом привязки гиперссылки.
- Выделите этот текст. Для этого переместите указатель мыши в начало выделяемого текста, нажмите левую клавишу мыши и, не отпуская ее, переместите указатель мыши к концу выделяемого текста.
- Отпустите левую клавишу мыши, текст будет выделен.
- После этого переместите указатель мыши в область выделенного текста и нажмите правую кнопку мыши. Появится контекстное меню.
- Щелкните мышью на пункте контекстного меню **Гиперссылка** (hyperlink).

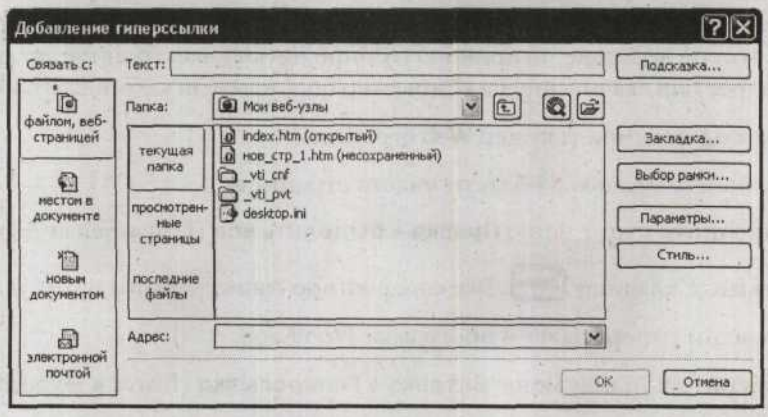


Рис. 2.30. Диалог ***Добавление гиперссылки*** *(Add hyperlink)*

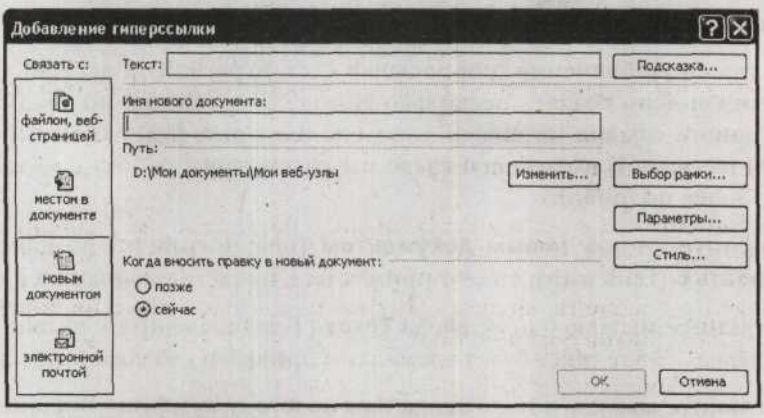


Рис. 2.31. Раздел ***новым документом*** *(new document), диалога* ***Добавление гиперссылки*** *(Add hyperlink)*

Появится диалог **Добавление гиперссылки** (Add hyperlink), представленный на Рис. 2.30.

- Нажмите кнопку **новым документом** (new document) панели кнопок **Связать с** (Link with), диалог примет вид, показанный на Рис. 2.32.

Как видно из приведенной иллюстрации, в отличие от предыдущих случаев поле **Текст** (Text) диалога будет заполнено текстом, взятым из области выделения.

- Щелкните мышью в поле ввода **Имя нового документа** (Document name) и введите название «**link.htm**». Это будет название нового документа, на который создается ссылка.

- **Щелкните** мышью на кнопке **OK**. Будет создана гиперссылка в текущем HTML-документе, а также будет создан и открыт для редактирования новый HTML-документ.

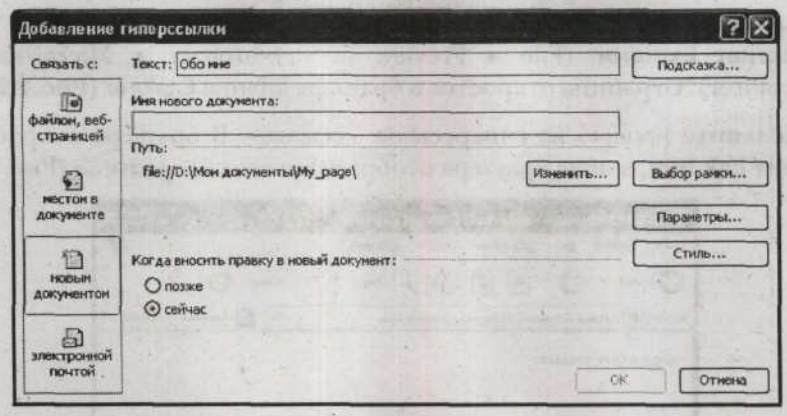


Рис. 2.32. Раздел ***новым документом*** *(new document), диалога* ***Добавление гиперссылки*** *(Add hyperlink) в случае использования контекстного меню*

- Итак, нами создана гиперссылка и документ, на который она ссылается.
- Щелкните мышью в рабочей области созданного документа и напечатайте текст: «**сюда ведет ссылка**».
- Сохраните документ, щелкнув мышью на кнопке  панели инструментов программы.
- Закройте документ, щелкнув мышью на кнопке закрытия документа в области заголовков документов.

Теперь просмотрим полученный результат в Web-браузере.

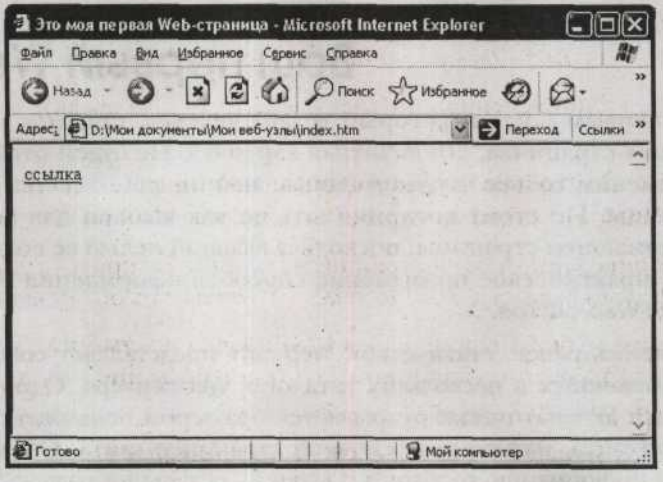


Рис. 2.33. Гиперссылка в Web-браузере Microsoft Internet Explorer

- Сделайте активным документ **index.html**, щелкнув мышью на его названии в области заголовков программы FrontPage.

- Выберите команду меню **Файл ♦ Просмотреть в обозревателе ♦ Microsoft Internet Explorer** (File ♦ Preview in web-browser ♦ Microsoft Internet Explorer). Страница откроется в браузере Internet Explorer (Рис. 2.33).
- Щелкните мышью на гиперссылке «ссылка». В браузере откроется документ **link.htm**, в окне браузера отобразится его содержимое (Рис. 2.34).

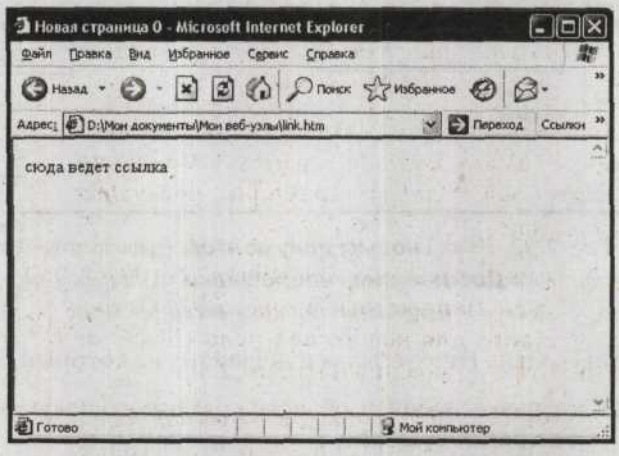


Рис. 2.34. *Результат перехода по гиперссылке в браузере Microsoft Internet Explorer*

Теперь вы знакомы со всеми основными элементами оформления Web-страниц. Вам теперь по силам создать полноценный Web-сайт, состоящий из нескольких связанных друг с другом страниц, с системой заголовков, списками и простыми таблицами. Созданию вашего первого Web-сайта посвящен следующий раздел этой главы.

Ваш первый Web-сайт

Как правило, первым сайтом, который делает человек, приходя в мир WWW, является домашняя страничка, его визитная карточка. Не будем отступать от этого правила и применим только что полученные знания для создания несложной домашней страницы. Не стоит воспринимать ее как шаблон для создания вашей собственной домашней страницы, поскольку главной целью ее создания является не она сама, а практическое применение способов оформления HTML-страниц и основ строения Web-сайтов.

Как уже говорилось ранее, статический Web-сайт представляет собой набор HTML-файлов, расположенных в нескольких каталогах Web-сервера. Один из этих HTML-файлов, который автоматически открывается браузером пользователя при посещении такого Web-сайта, называется основной страницей сайта. Обычно на ней размещают общую информацию о сайте и ссылки на остальные разделы сайта. Страницы, связанные с основной страницей сайта, называют второстепенными.

Чтобы сделать HTML-документ основной страницей сайта, ему достаточно присвоить название **index.html**.

Создание нового Web-сайта

Создадим новый Web-сайт в программе FrontPage.

- Выберите команду меню **Файл ♦ Создать** (File♦ New).
- Выберите пункт меню **Файл ♦ Создать** (File ♦ New). В правой части окна программы откроется панель **Создание** (Create), представленная на Рис. 2.35.
- Щелкните мышью на пункте **Одностраничный веб-узел** (Single Page Web-site) панели **Создание** (Create). Откроется диалог **Шаблоны веб-узлов** (Web-site templates) (Рис. 2.36.)
- Щелкните мышью на кнопке **Обзор** (Browse), группы настроек **Параметры** (Properties). Откроется диалог **Место для нового веб-узла** (Place for new Web-site), представленный на Рис. 2.37.
- Выберите или создайте папку, в которой будут размещаться файлы Web-сайта, и щелкните мышью на кнопке **Открыть** (Open), чтобы принять выбор и закрыть диалог **Место для нового веб-узла** (Place for new Web-site).
- Щелкните мышью на ярлыке **Одностраничный веб-узел** (Single page Web-site).

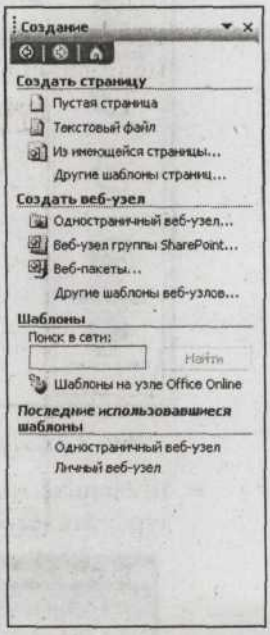


Рис. 2.35. Панель ***Создание*** *(Create)*

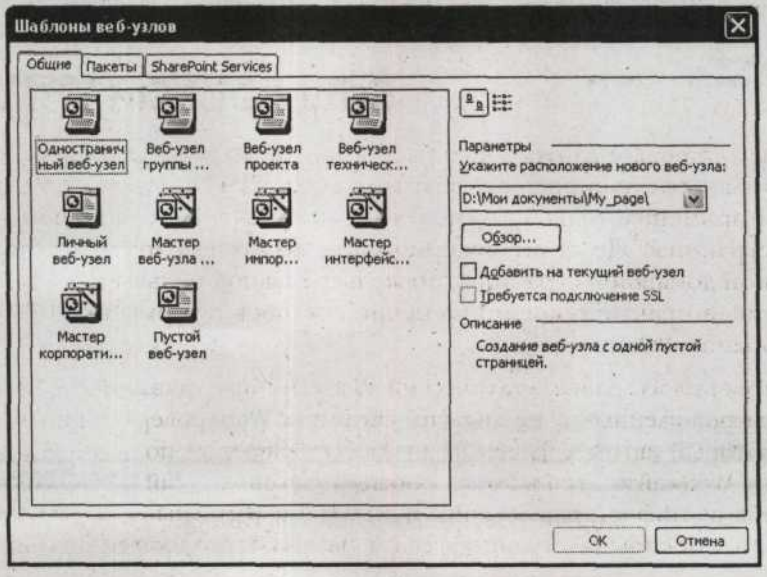


Рис. 2.36. Диалог ***Шаблоны веб-узлов*** *(Web-site templates)*

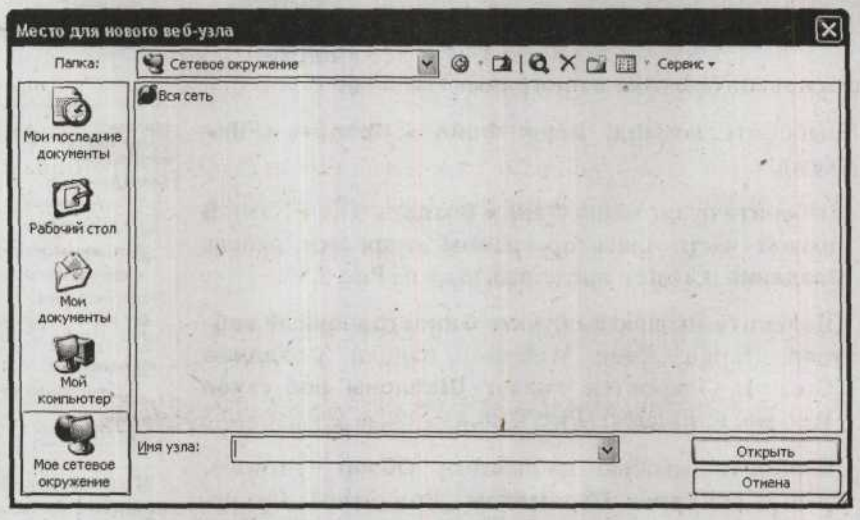


Рис. 2.37. Диалог ***Место для нового веб-узла*** *(Place for new Web-site)*

- Щелкните мышью на кнопке **OK**. Диалог закроется, будет создана структура каталогов Web-сайта и его главная страница (Рис. 2.38).

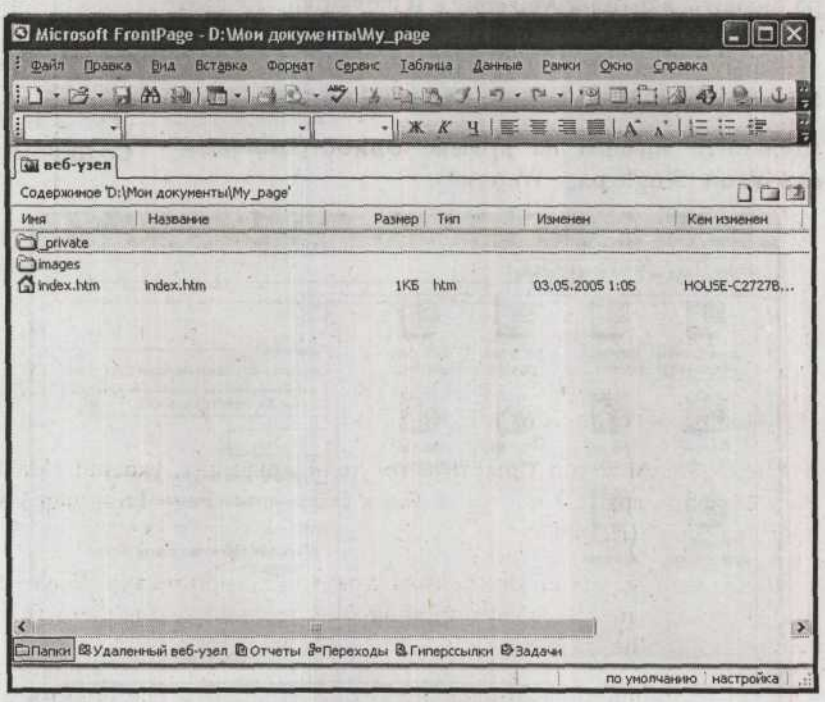


Рис. 2.38. Одностраничный Web-узел

Создание главной страницы

Приступим к созданию содержимого главной страницы вашего первого Web-сайта.

Дважды щелкните левой кнопкой мыши на имени главной страницы сайта – **index.htm**. Страница откроется в режиме редактирования.

Чтобы созданный вами документ везде корректно определялся как написанный на русском языке, проделайте следующее:

- Выберите команду меню **Файл ◆ Свойства** (File ◆ Properties). Откроется диалог **Свойства страницы** (Page properties), представленный на Рис. 2.39.

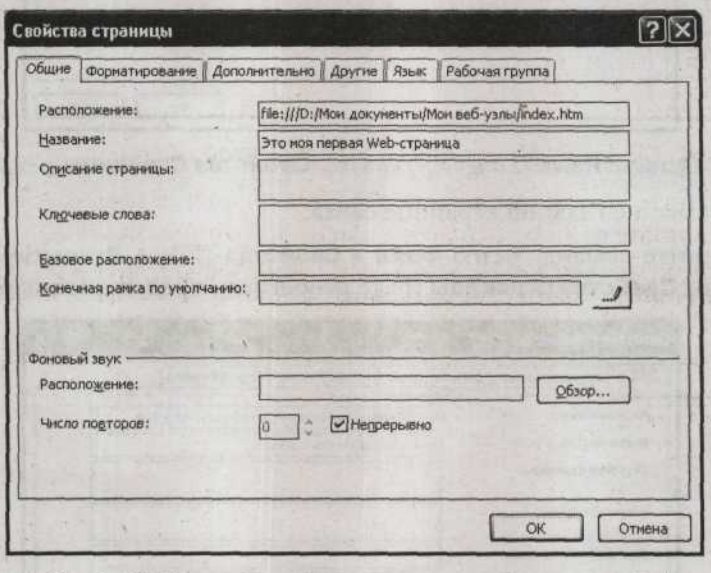


Рис. 2.39. Диалог ***Свойства страницы*** *(Page properties)*

- Щелкните мышью на вкладке **Язык** (Language), откроется диалог настройки языка страницы (см. Рис. 2.40).
- В выпадающем меню **Пометить текущий документ, указав:** (Mark current document as) группы настроек **Язык страницы** (Page Language) выберите пункт **русский** (russian).
- В выпадающем меню **Сохранить документ, используя** (Save document using) группы настроек **Набор знаков** (Character set) выберите пункт **кириллица** (Cyrillic).
- Щелкните мышью на кнопке **OK**, чтобы сохранить изменения и закрыть диалог настройки.

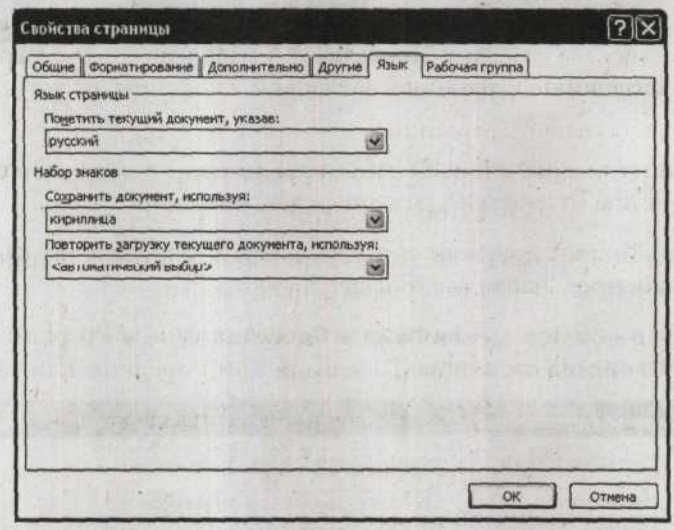


Рис. 2.40. Вкладка ***Язык*** *(Language) диалога* ***Свойства Страницы*** *(Page properties)*

Назначьте название главной странице сайта:

- Выберите команду меню **Файл ♦ Свойства** (File ♦ Properties). Откроется диалог **Свойства страницы** (Page properties), представленный на Рис. 2.41.

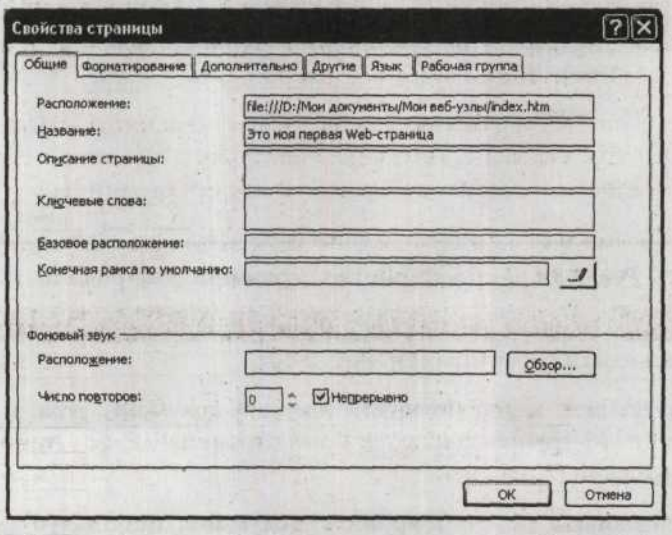


Рис. 2.41. Диалог ***Свойства страницы*** *(Page Properties)*

- Щелкните мышью на поле ввода **Название** (Title) и введите название: «**Главная страница**».
- Щелкните мышью на кнопке **OK**, чтобы принять изменения и закрыть диалог.

Главная страница вашего первого Web-сайта будет состоять из одного заголовка, некоторого количества текста и нескольких ссылок на другие страницы сайта. Приступим к наполнению страницы:

Создайте заголовок главной страницы.

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строчке, подписанной **Заголовок 1**, чтобы создать заголовок первого уровня.
- Щелкните мышью в области макета страницы и введите текст заголовка: «Моя домашняя страница»
- Нажмите клавишу Enter, чтобы завершить ввод заголовка.

Пришел черед основного текста страницы.

Введите с клавиатуры следующий текст:

Домашняя страница является визитной карточкой человека в WWW, слепком его «Я», поэтому я постараюсь, по возможности полно отразить свой внутренний мир на этих нескольких страничках. Чтобы познакомиться со мною поближе, побродите по разделам сайта, ссылки на которые представлены чуть ниже.

Теперь отделите горизонтальной линией раздел текста от раздела ссылок:

- Выберите пункт меню **Вставка ◆ Горизонтальная линия** (Insert ◆ Horizontal line). Линия будет вставлена.

Теперь необходимо вставить ссылки на другие страницы сайта и небольшие комментарии к этим ссылкам. Поскольку остальные страницы еще не созданы, вместе с созданием ссылок мы будем создавать и сами страницы.

- Нажмите кнопку на панели инструментов. Появится диалог **Добавление гиперссылки** (Add hyperlink), представленный на Рис. 2.42.

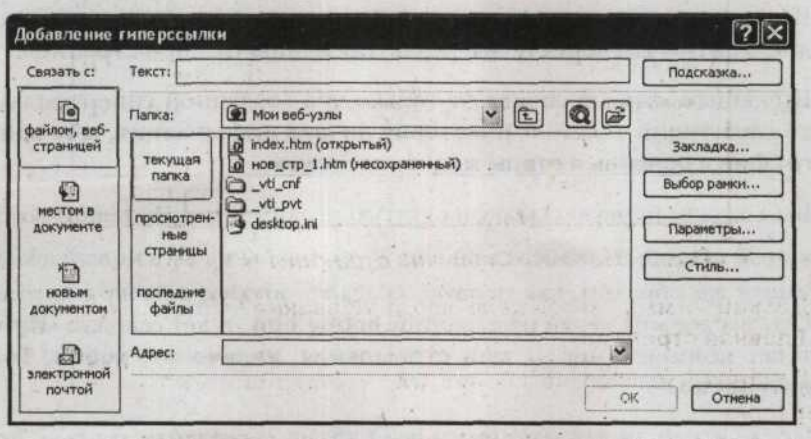


Рис. 2.42. Диалог ***Добавление гиперссылки*** *(Add hyperlink)*

- Нажмите кнопку **новым документом** (new document) панели кнопок **Связать с** (Link with), диалог примет вид, указанный на Рис. 2.43.

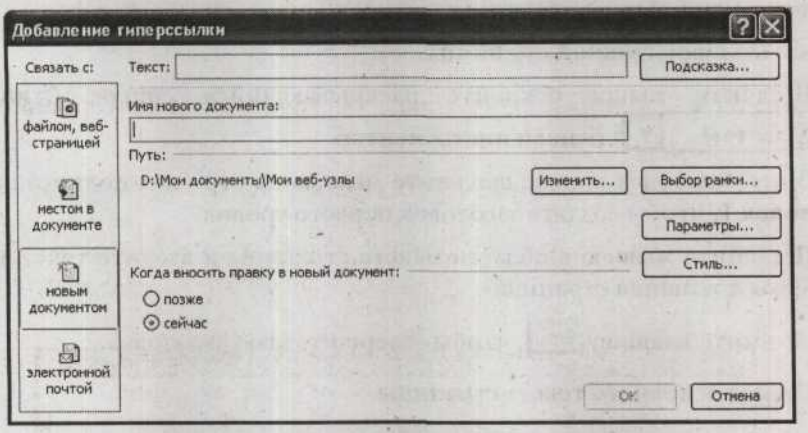


Рис. 2.43. Раздел ***новым документом*** *(new document), диалога* ***Добавление гиперссылки*** *(Add hyperlink)*

- Щелкните мышью в поле ввода **Текст** (Text) и введите с клавиатуры «Обо мне». Этот текст будет элементом привязки создаваемой ссылки.
- Щелкните мышью в поле ввода **Имя нового документа** (Document name) и введите название **about.htm**. Это будет название нового документа, на который создается ссылка.
- Щелкните мышью на кнопке **OK**. Будет создана гиперссылка в текущем HTML-документе, и будет создан и открыт для редактирования новый HTML-документ под названием **about.htm**.

К редактированию файла **about.htm** мы вернемся чуть позже, а сейчас завершим создание главной страницы сайта.

- Щелкните мышью на названии **index.htm** в области заголовков открытых документов. Вы вернетесь к редактированию главной страницы.
- Щелкните мышью справа от только что созданной гиперссылки и введите следующий текст: «**- Некоторая личная информация, моя краткая биография и основные этапы жизненного пути.**»
- Вы создали первую ссылку на другую страницу и небольшой комментарий к этой ссылке. Нажмите клавишу Enter, чтобы начать новый абзац и точно таким же образом, как первую, создайте вторую ссылку с комментарием. Ссылка должна вести на страницу **hobby.htm**, текст ссылки: «**Интересы**», а текст комментария: «**- мои стремления, увлечения, хобби. То, что мне интересно.**»
- После этого, опять начните новый абзац, и создайте третью, последнюю на этой странице, ссылку. Эта ссылка будет вести на страницу **links.htm**,

текст ее: «**Интересные ссылки**», а комментарий к ней: «**- ссылки на сайты, которые показались мне интересными.**»

- И завершите создание главной страницы еще одной горизонтальной линией.
- Выберите пункт меню **Вставка ◆ Горизонтальная линия** (Insert ◆ Horizontal line). Линия будет вставлена.

Полученная в результате страница должна выглядеть, как на Рис. 2.44.

Не забудьте сохранить главную страницу, щелкнув мышью на кнопке панели инструментов программы, и приступайте к редактированию страницы сайта «**Обо мне**».

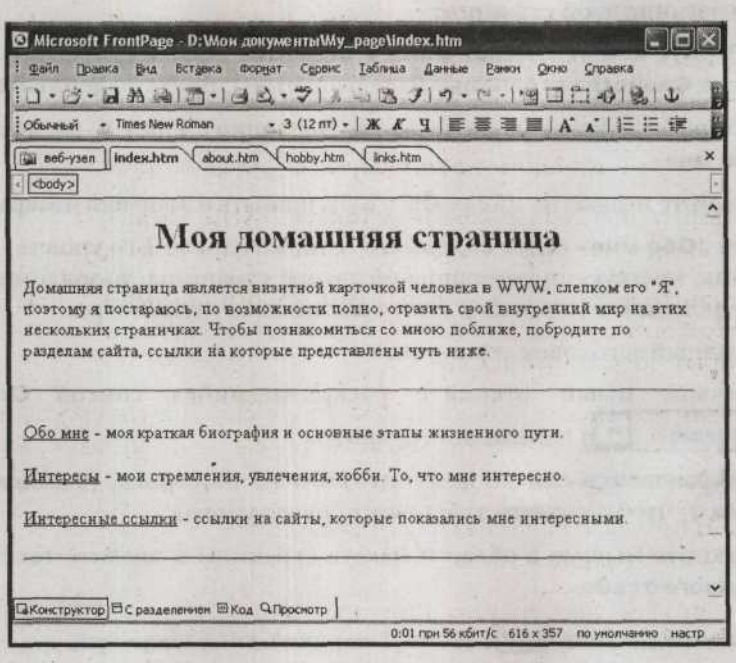


Рис. 2.44. Главная страница сайта

Создание раздела сайта «Обо мне»

Щелкните мышью на названии страницы **about.htm** в области заголовков документов. Файл **about.htm**, содержащий раздел «**Обо мне**», станет активным и доступным для редактирования.

Чтобы созданный вами документ везде корректно определялся как написанный на русском языке, проделайте следующее:

- Выберите команду меню **Файл ◆ Свойства** (File ◆ Properties). Откроется диалог **Свойства ◆ страницы** (Page properties), представленный на Рис. 2.39.

- Щелкните мышью на вкладке **Язык** (Language), откроется диалог настройки языка страницы (Рис. 2.40).
- В выпадающем меню **Пометить текущий документ, указав:** (Mark current document as) группы настроек **Язык страницы** (Page Language) выберите пункт **русский** (russian).
- В выпадающем меню **Сохранить документ, используя** (Save document using) группы настроек **Набор знаков** (Character set) выберите пункт **кириллица** (Cyrillic).
- Щелкните мышью на кнопке **OK**, чтобы сохранить изменения и закрыть диалог настройки.

Назначьте название этой странице:

- Выберите команду меню **Файл ♦ Свойства** (File ♦ Properties). Откроется диалог **Свойства страницы** (Page properties), представленный на Рис. 2.41.
- Щелкните мышью на поле ввода **Название** (Title) и введите название: «**Обо мне**».
- Щелкните мышью на кнопке **OK**, чтобы принять изменения и закрыть диалог.

Раздел сайта «**Обо мне**» будет содержать один заголовок 1-го уровня, два заголовка 2-го уровня, краткую информацию об авторе страницы, упорядоченную в виде списка определений, и несколько слов о жизненном пути автора.

Создайте главный заголовок страницы.

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строке, подписанной **Заголовок 1**, чтобы создать заголовок первого уровня.
- Щелкните мышью в области макета страницы и введите текст заголовка: «**Немного о себе**».
- Нажмите клавишу Enter, чтобы завершить ввод заголовка.

Создайте подзаголовок.

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строке, подписанной **Заголовок 2**, чтобы создать заголовок второго уровня.
- Щелкните мышью в области макета страницы и введите текст заголовка: «**Личная информация**»
- Нажмите клавишу Enter, чтобы завершить ввод заголовка.

Теперь создайте список определений, в который будет сведена информация личного характера. В качестве терминов будут различные личные параметры, а в качестве описания терминов – описание этих параметров

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строке, подписанной **Определенный термин** (Determined term).
- Введите первый личный параметр: «**Фамилия Имя Отчество**»
- Нажмите клавишу Enter, чтобы перейти к написанию значения этого параметра. Введите текст: «**Трамвай Иванович Вагонов**».
- Нажмите клавишу Enter, чтобы перейти ко второму элементу списка.

Введите остальные элементы списка. Второй элемент: «**Дата рождения**», его значение – «**21 мартобря 2902 года**». Третий элемент: «**Образование**», его значение – «**Три класса церковно-приходской школы**». Четвертый элемент: «**Иностранные языки**», его значение – «**Зулу, хинди, суахили, немного наречие браминов**». Пятый и последний элемент: «**Семейное положение**», его значение – «**Холост**».

Введя все данные, закончите создание списка

- Дважды нажмите клавишу Enter.

Создайте следующий подзаголовок

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строке, подписанной **Заголовок 2**, чтобы создать заголовок второго уровня.
- Щелкните мышью в области макета страницы и введите текст заголовка: «**Биография**».
- Нажмите клавишу Enter, чтобы завершить ввод заголовка.

Завершите создание раздела краткой биографией.

- Введите текст биографии: «**Родился. Вырос. Пошел в школу. Закончил, пошел работать. Этим занимаюсь и по сей день.**»

Вот вы и создали вторую страницу сайта, она должна выглядеть так, как показано на Рис. 2.45.

Не забудьте сохранить проделанную работу, щелкнув мышью на кнопке панели инструментов программы, и приступайте к редактированию страницы сайта «**Интересы**».

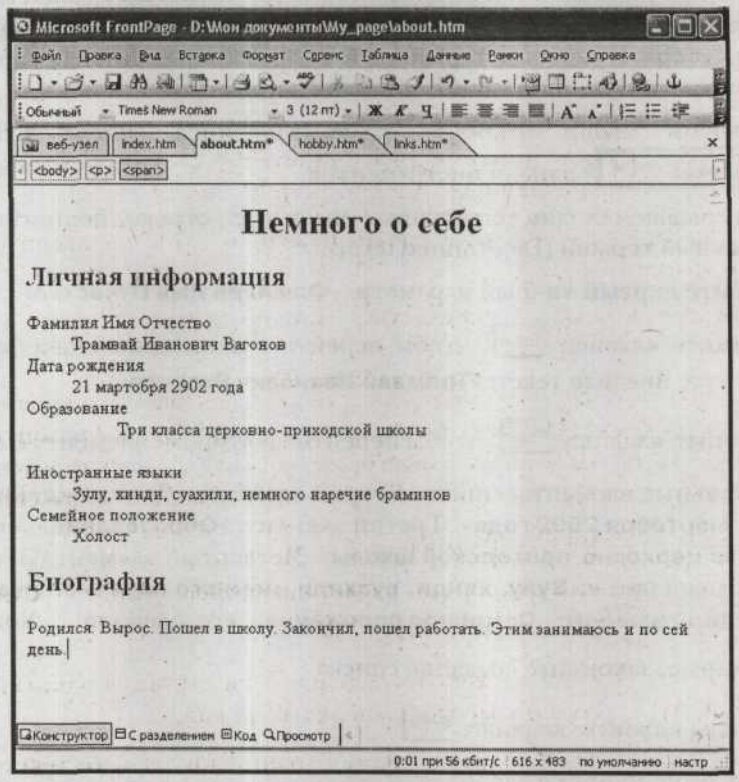


Рис. 2.45. Раздел домашней страницы «Обо мне»

Создание раздела сайта «Интересы»

Щелкните мышью на названии страницы **hobby.htm** в области заголовков документов. Файл **hobby.htm**, содержащий раздел «**Интересы**», станет активным для редактирования.

Чтобы созданный вами документ везде корректно определялся как написанный на русском языке, проделайте следующее:

- Выберите команду меню **Файл ♦ Свойства** (File ♦ Properties). Откроется диалог **Свойства страницы** (Page properties), Рис. 2.39.
- Щелкните мышью на вкладке **Язык** (Language), откроется диалог настройки языка страницы (Рис. 2.40).
- В раскрывающемся списке **Пометить текущий документ, указав:** (Mark current document as) группы настроек **Язык страницы** (Page Language) выберите пункт **русский** (russian).
- В раскрывающемся списке **Сохранить документ, используя** (Save document using) группы настроек **Набор знаков** (Character set) выберите пункт **кириллица** (Cyrillic).

- Щелкните мышью на кнопке **OK**, чтобы сохранить изменения и закрыть диалог настройки.

Назначьте название этой странице:

- Выберите команду меню **Файл ♦ Свойства** (File ♦ Properties). Откроется диалог **Свойства страницы** (Page properties), Рис. 2.41.
- Щелкните мышью на поле ввода **Название** (Title) и введите название: «**Мои интересы**».
- Щелкните мышью на кнопке **OK**, чтобы принять изменения и закрыть диалог.

Раздел сайта «Интересы» будет содержать один заголовок 1-го уровня, два заголовка 2-го уровня и краткое перечисление интересов автора, разбитых по двум категориями: «музыка» и «спорт». Категория «музыка» будет упорядочена в виде маркированного списка, а категория «спорт» в виде нумерованного списка.

Создайте главный заголовок страницы.

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строке, подписанной **Заголовок 1**, чтобы создать заголовок первого уровня.
- Щелкните мышью в области макета страницы и введите текст заголовка: «**Мои пристрастия**».
- Нажмите клавишу Enter, чтобы завершить ввод заголовка.
- Добавьте небольшое описание содержимого этой страницы.
- Введите текст: «**Здесь приведены мои основные интересы, разделенные на несколько категорий.**»

Создайте подзаголовок.

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строке, подписанной **Заголовок 2**, чтобы создать заголовок второго уровня.
- Щелкните мышью в области макета страницы и введите текст заголовка: «**Музыка**».
- Нажмите клавишу Enter, чтобы завершить ввод заголовка.

Теперь создайте маркированный список, в котором будут перечислены все музыкальные направления, которыми увлекается автор страницы.

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся меню щелкните мышью на строке, подписанной **Маркированный список** (Unordered list).
- Введите первый пункт списка, «**Rock'n'Roll**».
- Нажмите клавишу Enter, чтобы перейти к вводу второго элемента списка.

Введите остальные элементы списка. Пункты такие: «**Art-Rock**», «**Acid Jazz**», «**Drum'n'Bass**», «**Trip-Hop**», «**New Age**».

Введя все данные, закончите создание списка.

- Дважды нажмите клавишу Enter.

Создайте следующий подзаголовок.

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строке, подписанной **Заголовок 2**, чтобы создать заголовок второго уровня.
- Щелкните мышью в области макета страницы и введите текст заголовка: «**Спорт**».
- Нажмите клавишу Enter, чтобы завершить ввод заголовка.

Теперь создайте нумерованный список, в котором будут перечислены в порядке убывания значимости виды спорта.

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строке, подписанной **Нумерованный список** (Ordered list).
- Введите первый пункт списка «**Роликовые коньки**».
- Нажмите клавишу Enter, чтобы перейти к вводу второго элемента списка.

Введите остальные элементы списка. Пункты такие: «**Ледовые коньки**», «**Лыжи**», «**Плавание**», «**Рыбалка**».

Введя все данные, закончите создание списка.

- Дважды нажмите клавишу Enter.

Еще один раздел успешно создан, полученная страница должна выглядеть так, как показано на Рис. 2.46.

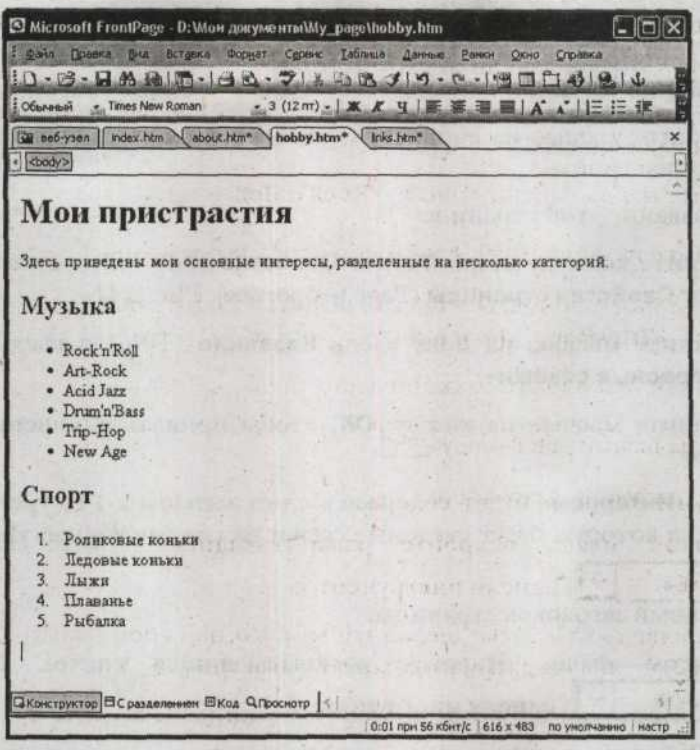


Рис. 2.46. Раздел домашней страницы «Интересы»

Не забудьте сохранить проделанную работу, щелкнув мышью на кнопке  панели инструментов программы, и приступайте к редактированию последней страницы сайта «**Интересные ссылки**».

Создание раздела сайта «Интересные ссылки»

Щелкните мышью на названии страницы **links.htm** в области заголовков документов. Файл **links.htm**, содержащий раздел «**Интересные ссылки**», станет активным для редактирования.

Чтобы созданный вами документ везде корректно определялся как написанный на русском языке, проделайте следующее:

- Выберите команду меню **Файл ♦ Свойства** (File ♦ Properties). Откроется диалог **Свойства страницы** (Page properties), Рис. 2.39.
- Щелкните мышью на вкладке **Язык** (Language), откроется диалог настройки языка страницы (Рис. 2.40).
- В раскрывающемся меню **Пометить текущий документ, указав:** (Mark current document as) из группы настроек **Язык страницы** (Page Language) выберите пункт **русский** (russian).

- В раскрывающемся списке **Сохранить документ, используя** (Save document using) группы настроек **Набор знаков** (Character set) выберите пункт **кириллица** (Cyrillic).
- Щелкните мышью на кнопке **OK**, чтобы сохранить изменения и закрыть диалог настройки.

Назначьте название этой странице:

- Выберите команду меню **Файл ♦ Свойства** (File ♦ Properties). Откроется диалог **Свойства страницы** (Page properties), Рис. 2.41.
- Щелкните мышью на поле ввода **Название** (Title) и введите название: «**Интересные ссылки**».
- Щелкните мышью на кнопке **OK**, чтобы принять изменения и закрыть диалог.

Раздел сайта «**Интересы**» будет содержать один заголовок 1-го уровня и небольшую таблицу, в которую будут сведены ссылки на разные Web-ресурсы и их краткое описание.

Создайте главный заголовок страницы.

- Щелчком мыши откройте раскрывающийся список **Стиль** (Style) (Обычный) панели инструментов.
- В открывшемся списке щелкните мышью на строке, подписанной **Заголовок 1**, чтобы создать заголовок первого уровня.
- Щелкните мышью в области макета страницы и введите текст заголовка: «**Интересные ссылки**».
- Нажмите клавишу Enter, чтобы завершить ввод заголовка.
- Добавьте небольшое описание содержимого этой страницы.
- Введите текст: «**В этом разделе собраны ссылки на различные Web-страницы, которые мне показались интересными и полезными.**»

Теперь создайте таблицу, в которой будет содержаться описание Интернет-ссылок. Таблица будет состоять из трех колонок. В первой колонке будут идти сами ссылки, во второй колонке – краткие описания страниц, а в третьей – различные комментарии.

- Щелкните мышью на кнопке **Добавить таблицу** (New Table) ( ) панели инструментов программы. Появится меню выбора конфигурации таблицы.

В меню выбора конфигурации таблицы в схематическом виде представлена создаваемая вами таблица, каждый квадратик меню соответствует одной ячейке.

- Перемещая курсор мыши, добейтесь, чтобы в меню темным цветом было выделено три столбца и две строки.

- Щелкните левой кнопкой мыши. Будет создана таблица из трех столбцов и двух строк.
- Теперь заполните таблицу содержимым.
- Щелкните мышью в левой верхней ячейке таблицы. Текстовый курсор переместится в эту ячейку.
- Введите заголовок первого столбца таблицы: «**Ссылки**».
- Перемещая нажатиями клавиши Tab, текстовый курсор по ячейкам первой строки, введите остальные два заголовка таблицы: «**Краткое описание**» и «**Комментарии**».
- Нажмите еще раз клавишу Tab текстовый курсор переместится в первую ячейку второй строки.
- Вставьте гиперссылку на Интернет-сайт.
- Выберите команду меню **Вставка ♦ Гиперссылка** (Insert ♦ Hyperlink).
- Появится диалог **Добавление гиперссылки** (Add hyperlink), Рис. 2.47. До этого вы создавали гиперссылки вместе с созданием самих документов, на которые ссылались. Теперь вам предстоит создать несколько ссылок на документы, которые не только существуют, но и располагаются вне вашего компьютера. Создадим первую такую ссылку.

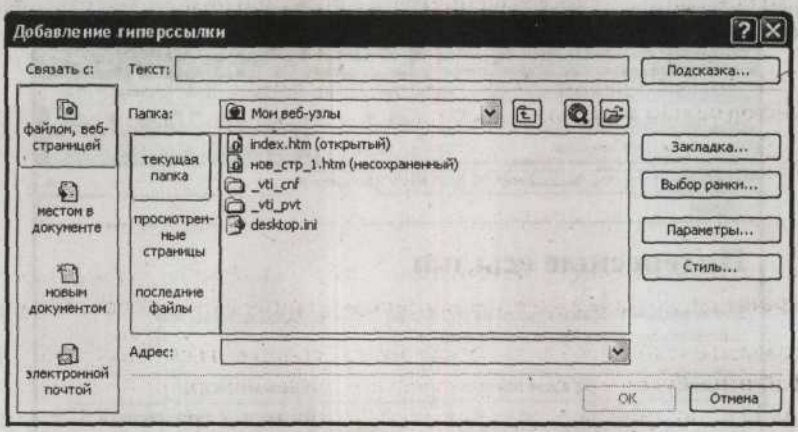


Рис. 2.47. Диалог ***Добавление гиперссылки*** *(Add hyperlink)*

- Щелкните мышью в поле ввода **Адрес** (Address). Введите в поле адрес сайта: **http://www.yandex.ru**.
- Щелкните мышью в поле ввода **Текст** (Text) и введите в него текст элемента привязки ссылки, «**www.yandex.ru**».
- Щелкните мышью на кнопке **OK**, чтобы закрыть диалог и создать гиперссылку.

- Заполните остальные два столбца строки таблицы.
- Нажмите клавишу Tab, текстовый курсор перейдет на вторую ячейку второй строки.
- Введите текст описания ссылки: «**Самая популярная российская поисковая система.**»
- Еще раз нажмите клавишу Tab, текстовый курсор перейдет на третью ячейку второй строки.
- Введите текст комментария: «**Может и не самая лучшая в мире, но отнюдь не самая плохая поисковая система.**»
- Снова нажмите клавишу Tab. В таблице будет создана третья строка. Текстовый курсор перейдет на первую ячейку третьей строки.

Аналогично первой гиперссылке введите в таблицу ссылки на оставшиеся сайты. Первый: адрес – **http://www.mult.ru**, текст элемента привязки – «**www.mult.ru**», краткое описание – «**Мультипликационная студия Олега Куваева**», комментарий – «**Студия создателя Масяни.**». И второй: адрес – **http://www.microsoft.com**, текст элемента привязки – «**www.microsoft.com**», краткое описание – «**Сайт корпорации Microsoft.**», комментарий – «**А это создатели Windows!**».

Вот вы закончили создание и последней страницы вашего первого Web-сайта. Полученная страница должна выглядеть, как на Рис. 2.48.

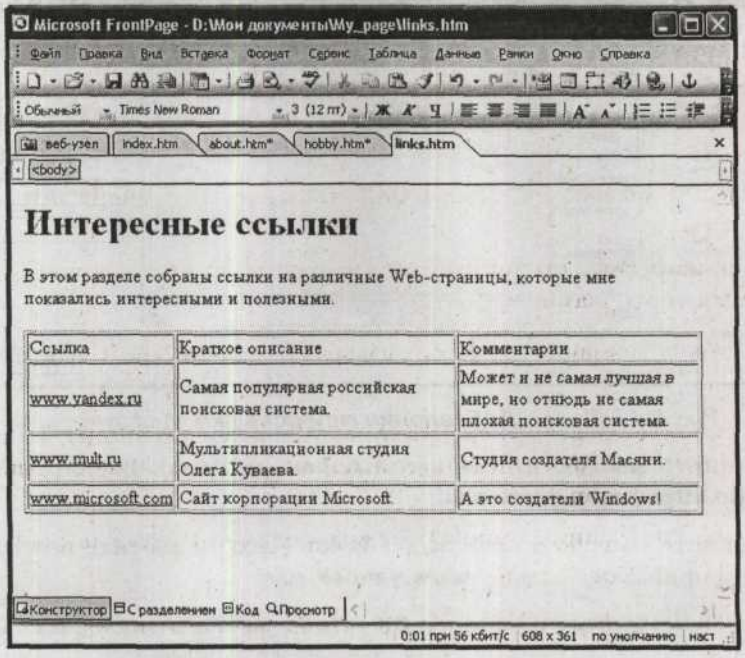


Рис. 2.48. Раздел домашней страницы «Интересные ссылки»

Не забудьте сохранить проделанную работу, щелкнув мышью на кнопке  панели инструментов программы.

Теперь осталось добавить несколько финальных штрихов – и ваш Web-сайт готов.

Добавление навигационного меню на второстепенные страницы Web-сайта

Некоторым недостатком структуры только что созданного вами сайта является то, что вы можете открывать второстепенные страницы сайта, только находясь на главной странице. Эту оплошность можно исправить, добавив на второстепенные страницы навигационное меню, в котором будут ссылки на все остальные разделы сайта.

Чтобы навигационное меню было отделено от основной части раздела, отделите его горизонтальной линией.

- Щелкните мышью в нижней части макета страницы.
- Выберите пункт меню **Вставка ◆ Горизонтальная линия** (Insert ◆ Horizontal line). Линия будет вставлена.
- Теперь последовательно вставьте гиперссылки на все разделы сайта.
- Выберите команду меню **Вставка ◆ Гиперссылка** (Insert ◆ Hyperlink).

Появится диалог **Добавление гиперссылки** (Add hyperlink), представленный на Рис. 2.47.

- Щелкните мышью на названии файла **index.html** в списке файлов **Папка** (Folder).
  В поле ввода **Адрес** (address) подставится название файла, а в поле ввода **Текст** (Text) автоматически подставится название главной страницы сайта.
- Щелкните мышью на кнопке **OK**, чтобы создать гиперссылку и закрыть диалог.

Чтобы ссылки на разные страницы сайта не сливались друг с другом, отделим их вертикальными черточками

- Нажмите клавишу Пробел, затем введите значок «|» и еще раз нажмите клавишу Пробел.

Таким же образом добавьте гиперссылки на остальные страницы, тоже разделяя их вертикальными черточками. Напоминаем названия файлов остальных разделов: **about.htm**, **hobby.htm**, **links.htm**.

В результате макет страницы приобретет вид, показанный на Рис. 2.49.

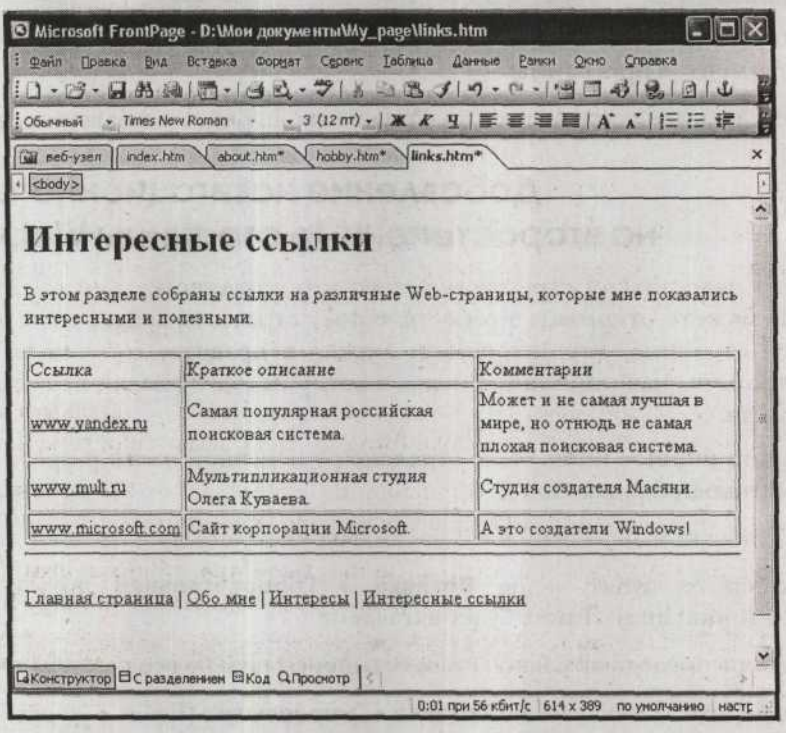


Рис. 2.49. Раздел сайта с добавленным навигационным меню

Скопируйте полученное навигационное меню в буфер обмена Windows.

- Переместите указатель мыши так, чтобы он находился чуть левее горизонтальной линии над навигационным меню.
- Нажмите левую кнопку мыши и, не отпуская ее, перемещайте указатель мыши так, чтобы горизонтальная линия и навигационное меню оказались выделенными.
- Отпустите левую кнопку мыши, навигационное меню выделено.
- Выберите команду меню **Правка ◆ Копировать** (Edit ◆ Copy). Меню будет скопировано в буфер обмена.

Теперь последовательно вставьте навигационное меню на все второстепенные страницы сайта.

- Сделайте активной страницу «Интересы», щелкнув мышью на названии файла страницы **hobby.htm** в области заголовков документов.
- Щелкните мышью в нижней части макета страницы.
- Выберите команду меню **Правка ◆ Вставить** (Edit ◆ Insert). Меню будет вставлено на страницу.

- Сделайте активной страницу «Обо мне», щелкнув мышью на названии файла страницы **about.htm** в области заголовков документов.
- Щелкните мышью в нижней части макета страницы.
- Выберите команду меню **Правка ♦ Вставить** (Edit ♦ Insert). Меню будет вставлено на страницу.

Теперь навигационное меню есть на всех второстепенных страницах. Поскольку в меню стоят ссылки на все страницы сайта, то получается, что в меню каждого раздела одна гиперссылка ссылается на сам этот раздел. Эту несуразность желательно исправить. Можно просто удалить лишние ссылки вместе с элементами привязки, но лучше удалить ссылку, оставив текст ссылки.

- Сделайте активной страницу «Интересные ссылки», щелкнув мышью на названии файла страницы **links.htm** в области заголовков документов.
- Щелкните правой кнопкой мыши на гиперссылке **Интересные ссылки** навигационного меню. Откроется контекстное меню.
- Щелкните мышью на строке меню **Свойства гиперссылки** (Hyperlink properties). Откроется диалог **Изменение гиперссылки** (Change hyperlink).

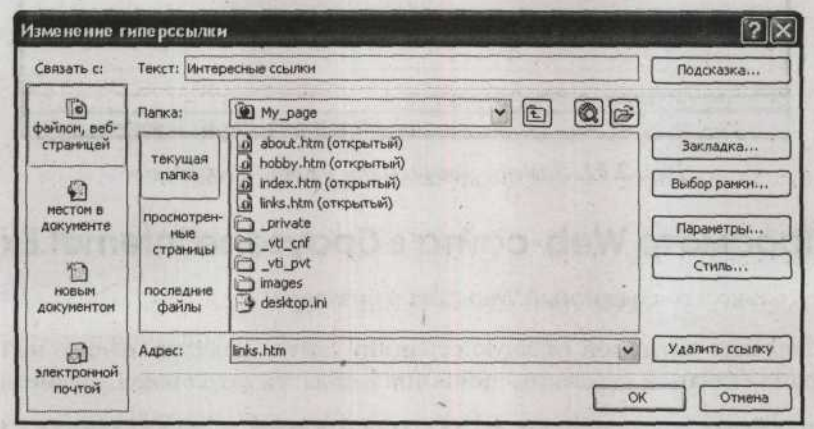


Рис. 2.50. Диалог ***Изменение гиперссылки*** *(Change hyperlink)*

- Щелкните мышью на кнопке **Удалить ссылку** (Delete link), чтобы удалить ссылку и закрыть диалог. В результате ссылка заменится на простой текст (Рис. 2.51).

Проделайте то же самое на других второстепенных страницах. Удалите из файла **about.htm** ссылку на раздел «**Обо мне**», а из файла **hobby.htm** ссылку на раздел «**Интересы**».

После этого ваш Web-сайт будет полностью готов к использованию. Сохраните все изменения на страницах. Чтобы сохранить все измененные страницы разом:

- Выберите пункт меню **Файл ♦ Сохранить все** (File ♦ Save all), все измененные страницы будут сохранены.

Теперь можете приступать к просмотру полученного Web-сайта.

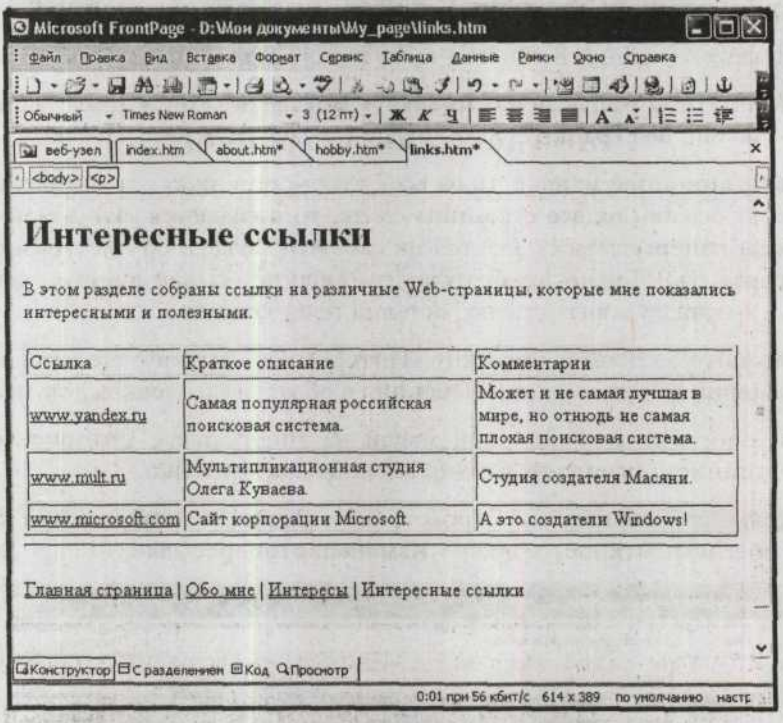


Рис. 2.51. Замена гиперссылки на обычный текст

Просмотр Web-сайта в браузере Internet Explorer

Откроем только что созданный Web-сайт в браузере.

- Сделайте активной главную страницу сайта, щелкнув мышью на названии файла главной страницы **index.htm** в области заголовков документов.
- Выберите команду меню **Файл ♦ Просмотреть в обозревателе ♦ Microsoft Internet Explorer** (File ♦ Preview in web-browser ♦ Microsoft Internet Explorer). Главная страница сайта откроется в Web-браузере Internet Explorer (Рис. 2.52).

Щелкая мышью на ссылках, вы можете перейти к другим страницам сайта. Перейдите к разделу сайта «**Интересные ссылки**».

- Щелкните мышью на гиперссылке **Интересные ссылки**. Браузер откроет раздел сайта «**Интересные ссылки**» (Рис. 2.53).

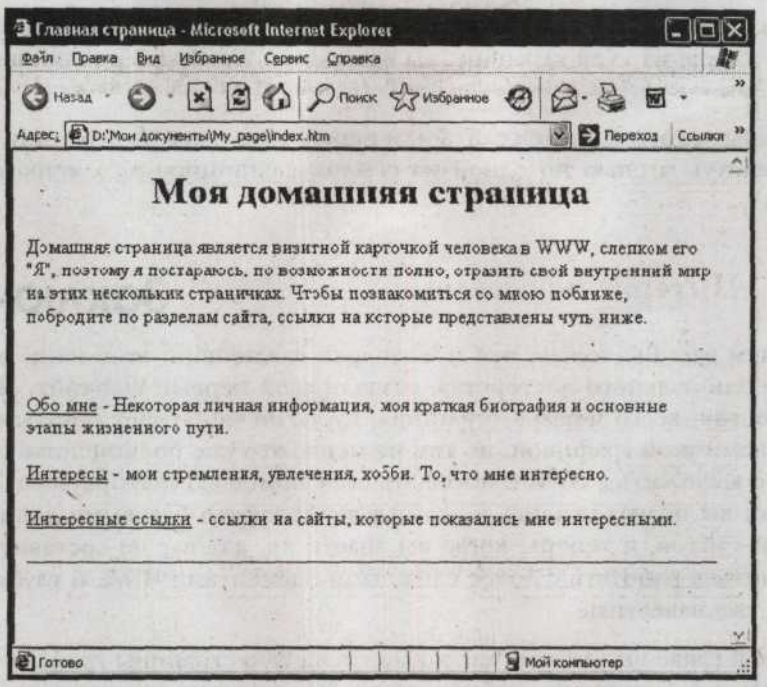


Рис. 2.52. Главная страница сайта в Web-браузере Microsoft Internet Explorer

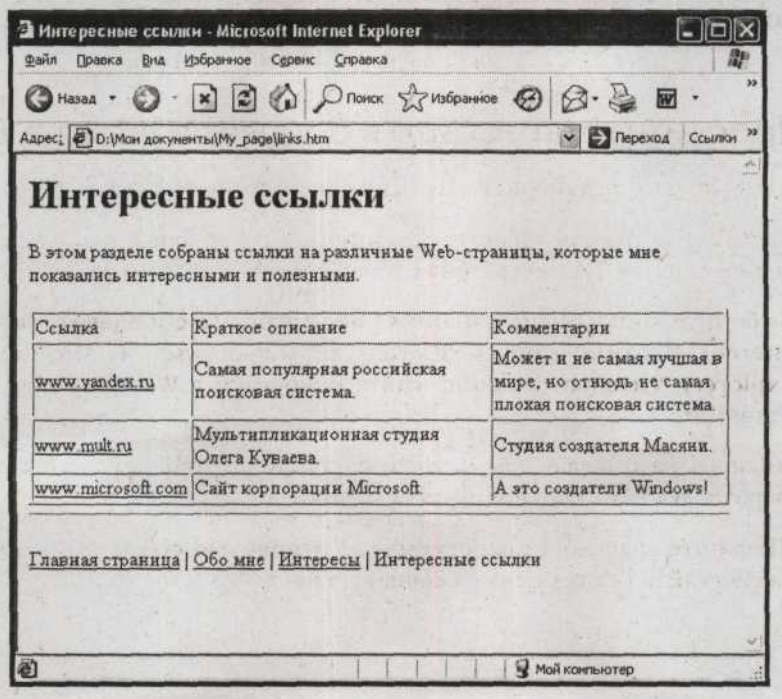


Рис. 2.53. Страница «Интересные ссылки» сайта

Если ваш компьютер подключен к сети Интернет, то, щелкнув по одной из гиперссылок в таблице на этой странице, вы откроете в Web-браузере описанные в этой таблице сайты.

Чтобы открыть другой раздел сайта или перейти на главную страницу, вам достаточно щелкнуть мышью по одной из ссылок навигационного меню внизу Web-страницы.

Заключение

Поздравляем вас. Вы только что преодолели важнейший этап на пути к вершинам профессионального мастерства, создали свой первый Web-сайт. Пусть он не слишком велик, всего четыре страницы, пусть он не блещет богатым оформлением и динамичной графикой, но тем не менее это уже полноценный сайт, который можно выложить в WWW и посещать его с помощью Web-браузера. Элементы, с которыми вы познакомились в этой главе, являются базовыми для всех статических Web-сайтов, и теперь, когда вы знаете их, для вас не составит большого труда научиться работать с более сложными элементами HTML и глубже изучить элементы, уже известные.

В следующей главе вы узнаете, как добавить на Web-страницы графику и звук, познакомитесь с некоторыми особенностями оформления текстов и многими другими вещами, которые позволят вашим Web-сайтам стать богаче и выразительнее, засиять всеми цветами радуги.

глава 3

Приемы, без которых нельзя обойтись

Вы познакомились с основными элементами Web-страниц и уже создали свой первый Web-сайт. Пришло время делать следующий шаг в освоении высот Web-мастеринга. В этой главе вы узнаете о том, как оживить ваши страницы, добавив на них графические элементы, звуки и видео. Вы познакомитесь с основами табличной верстки Web-страниц, что позволит вам создавать страницы довольно сложного строения. Кроме того, в этой главе будет рассказано об основных элементах оформления текстов Web-страниц, что позволит сделать их более наглядными и удобочитаемыми.

Графика на Web-страницах

Язык **HTML** предоставляет возможность использовать в оформлении Web-страниц графические элементы (Рис. 3.1). Это позволяет сделать страницы более наглядными, красочными и яркими. Ведь, как известно, одна хорошая иллюстрация может стоить нескольких страниц текста. Но, с другой стороны, чрезмерно увлекаться графикой не следует, все хорошо в меру.

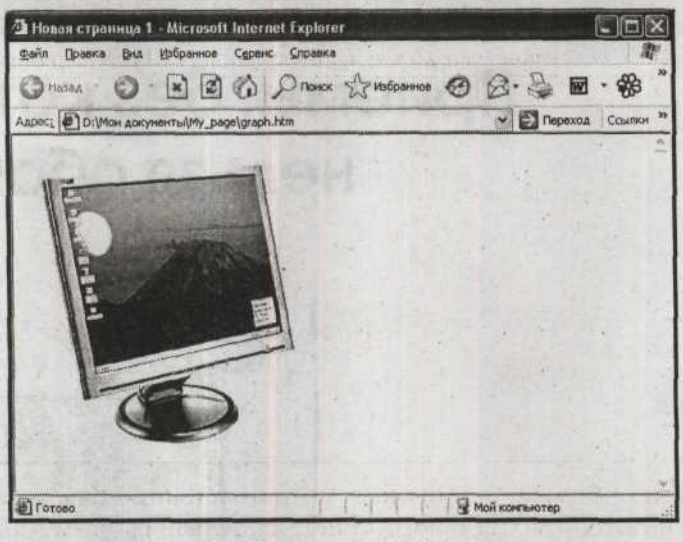


Рис. 3.1. Графика на Web-странице

Особенности графических файлов, используемых на Web-страницах

На Web-страницах можно размещать изображения из файлов форматов **GIF**, **JPEG** и **PNG**. Эти файлы обычно имеют расширения **.gif**, **.jpg** и **.png**. О принципиальных особенностях этих форматов и о том, как создавать такие файлы подробно будет рассказано в следующей главе. Сейчас же, мы лишь слегка коснемся общих моментов, важных для понимания материала данной главы.

Изображения в файлах всех перечисленных форматов хранится в так называемом «растровом виде». Растровые изображения представляют собой множество точек разного цвета. Поскольку точки очень маленькие и близкорасположенные, то создается впечатление, будто это одна большая картинка. Ширина и высота растрового изображения определяются количеством точек в нем по горизонтали и по вертикали.

Размещение графических элементов на Web-страницах

В языке HTML картинки на Web-страницу добавляются с помощью одиночного тега **<IMG>** и его атрибута **SRC**, с помощью которого задается адрес, по которому располагается картинка. Например, тег **<IMG SRC="HTTP://WWW.MY_SITE.COM/FACE.JPG">** добавит на страницу картинку **FACE.JPG**, находящуюся по адресу **HTTP://WWW.MY_SITE.COM/** (Листинг 3.1).

Листинг 3.1. HTML-документ, отображающий картинку на **Web***-странице*

```
<HTML>
<HEAD>
<TITLE>Картинка</TITLE>
</HEAD>
<BODY>
<IMG SRC=HTTP://WWW.MY_SITE.COM/FACE.JPG>
</BODY>
</HTML>
```

Если картинка по каким-либо причинам не смогла или не успела загрузиться, то на ее месте будет отображаться рамка (Рис. 3.2). В этой рамке может содержаться текст, отражающий содержание рисунка. Этот текст задается с помощью атрибута **ALT**. Текст, задаваемый этим атрибутом, называется альтернативным представлением картинки. Пользователь, который не может или не хочет познакомиться с самой картинкой, сможет прочитать текст, ее описывающий.

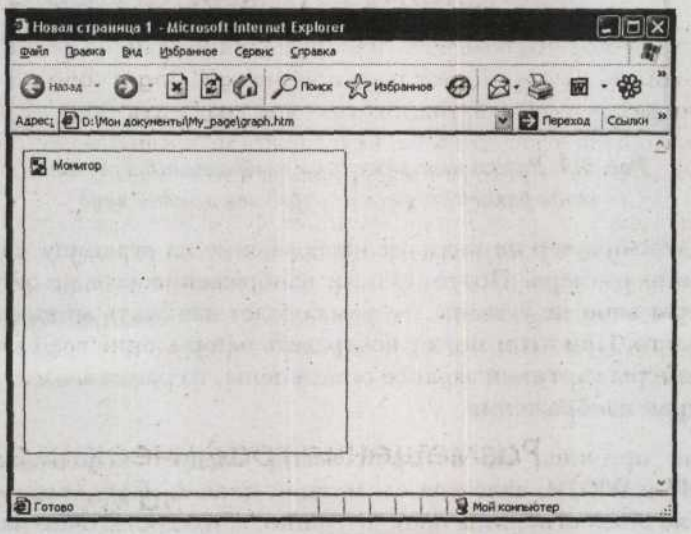


Рис. 3.2. Рамка на месте не загрузившегося изображения

Рекомендуем вам ко всем изображениям, которые вы будете размещать на Web-страницах, добавлять альтернативные описания, потому что достаточно велик процент пользователей, которые отключают отображение картинок в своих Web-браузерах для уменьшения объема загружаемой информации. Кроме того, для людей с ослабленным зрением и слепых практически единственным способом взаимодействия с WWW является использование «голосовых» браузеров, которые «озвучивают» содержимое Web-страниц. Разумеется, картинки они озвучи-

вать не могут, поэтому голосовые браузеры «читают» описания изображений, определяемые атрибутом **ALT**.

Атрибуты **HEIGHT** и **WIDTH** тега **<IMG>** позволяют указать браузеру размеры загружаемой картинки в точках или в процентах от всего размера Web-страницы (Рис. 3.3). Атрибут **HEIGHT** задает ее высоту, а **WIDTH** – ширину. Если размеры картинки с помощью этих атрибутов не задать, ничего страшного не произойдет, браузер все равно определит размеры изображения, когда будет его загружать, но, тем не менее, лучше указывать размеры картинки явно. Для этого есть несколько причин.

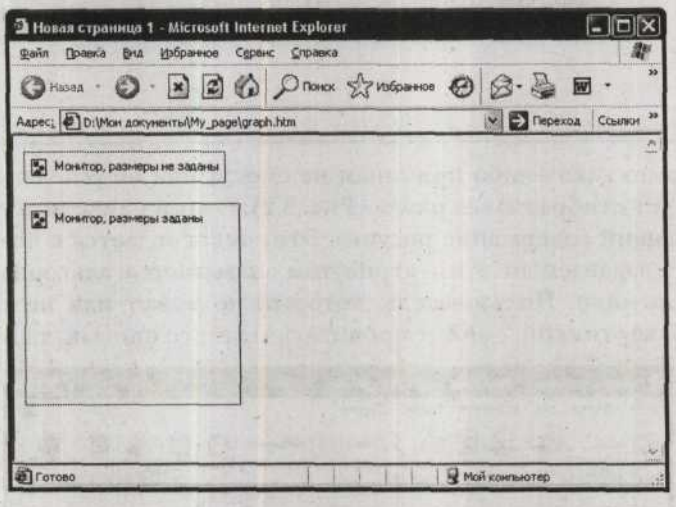


Рис. 3.3. *Рамки незагруженных изображений в случаях, когда размер рисунка не определен и задан явно*

- Пока Web-браузер не загрузит изображение на страницу, он не знает, какого оно размера. Поэтому, если изображение еще не загружено и его размеры явно не указаны, то рамка будет занимать минимально возможное место. При этом может пострадать оформление всей страницы. Если же размеры картинки заранее определены, то рамка займет место, равное размерам изображения.

- Вторая причина, по которой всегда следует использовать атрибуты **HEIGHT** и **WIDTH**, является следствием первой. Представьте такую ситуацию: загрузка страницы идет медленно, а это достаточно частое явление, и размеры изображений заранее не указаны. В этом случае, по мере загрузки картинок на страницу, рамки, окружающие место изображений, будут раздвигаться под размер картинок и все элементы оформления страницы будут перемещаться друг относительно друга. Этот процесс выглядит не слишком привлекательно. Поэтому, лучше таких случаев избегать и указывать размеры изображений.

Когда вы добавляете изображение на Web-страницу при помощи программы FrontPage, она автоматически указывает в тегах HTML их размеры, позволяя вам не задумываться об этой проблеме.

Если при помощи атрибутов **HEIGHT** и **WIDTH** указать размеры изображения, не равные его фактическим размерам, то при размещении на страницы картинка будет отмасштабирована в соответствии с указанными размерами. С использованием этой особенности тега **<IMG>** связаны некоторые приемы оформления Web-страниц, но в обычной практике рекомендуется этого избегать.

Изображения можно размещать прямо в тексте (Рис. 3.4), при этом можно выбрать, как именно будет выравниваться текст относительно картинки (Рис. 3.5). За выравнивание текста относительно изображения отвечает атрибут **ALIGN** тега **<IMG>**. Существуют три варианта выравнивания текста относительно изображения: по верхней части изображения, по его середине и по нижней части. Этим вариантам соответствуют значения атрибута **ALIGN**: **TOP**, **MIDDLE** и **BOTTOM** (Листинг 3.2).

Листинг 3.2. Способы выравнивания текста относительно изображения

```
<html>
<head>
<title>Выравнивание текста относительно изображения</title>
</head>
<body>
<p> <img border="0" src="images/monitor_small.gif"
width="100" height="94" align="top">
Текст может выравниваться по верхней части изображения</p>
<p> <img border="0" src="images/monitor_small.gif"
width="100" height="94" align="middle">
Текст может выравниваться по средней части изображения</p>
<p> <img border="0" src="images/monitor_small.gif"
width="100" height="94" align="bottom">
Текст может выравниваться и по нижней части изображения</p>
</body>
</html>
```

Также, можно сделать так, чтобы картинка не находилась в строке наряду с текстом, а находилась слева или справа от текста, который будет ее обтекать (Рис. 3.6). Обтекание текстом задается тоже атрибутом **ALIGN**. Значение этого атрибута **LEFT** соответствует расположению картинки слева от текста, а значение **RIGHT** – справа от него (Листинг 3.3).

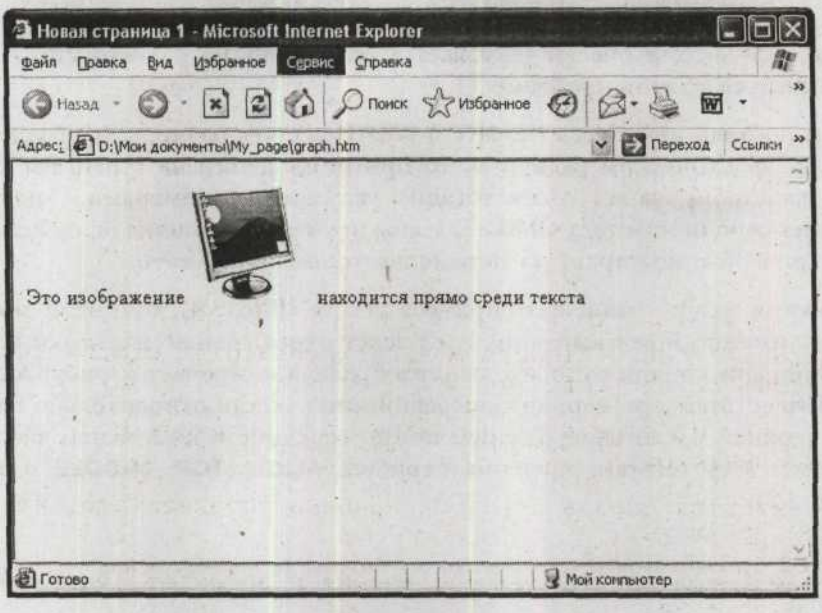


Рис. 3.4. Размещение изображения среди текста

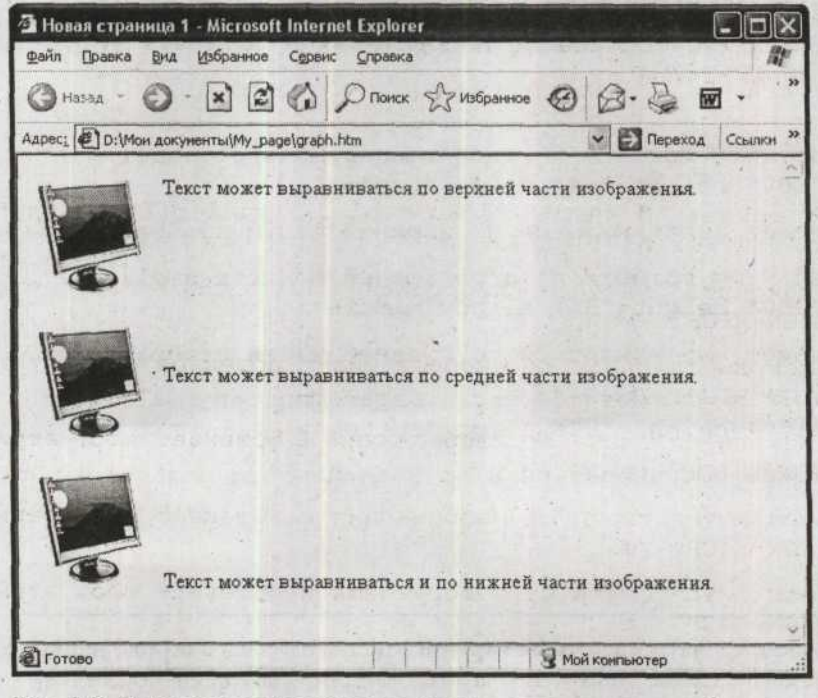


Рис. 3.5. Варианты выравнивания текста относительно изображения

Листинг 3.3. Способы выравнивания текста относительно изображения

```
<html>
<head>
<title>Выравнивание текста относительно изображения</title>
</head>
<body>
<p> 
<img border="0" src="images/monitor_small.gif" width="100"
height="94" align="left">
Текст находится справа от изображения и обтекает его, атрибут
<b>ALIGN=LEFT</b>.
Текст находится справа от изображения и обтекает его, атрибут
<b>ALIGN=LEFT</b>.
Текст находится справа от изображения и обтекает его, атрибут
<b>ALIGN=LEFT</b>.
Текст находится справа от изображения и обтекает его, атрибут
<b>ALIGN=LEFT</b>.
Текст находится справа от изображения и обтекает его, атрибут
<b>ALIGN=LEFT</b>.
</p>
<br>
<p><img border="0" src="images/monitor_small.gif" width="100"
height="94" align="right">
Текст находится слева от изображения и обтекает его, атрибут
<b> ALIGN=RIGHT</b>.
Текст находится слева от изображения и обтекает его, атрибут
<b> ALIGN=RIGHT</b>.
Текст находится слева от изображения и обтекает его, атрибут
<b> ALIGN=RIGHT</b>.
Текст находится слева от изображения и обтекает его, атрибут
<b> ALIGN=RIGHT</b>.
Текст находится слева от изображения и обтекает его, атрибут
<b> ALIGN=RIGHT</b>.
</p>
</body>
</html>
```

Изображения на странице можно окружать границей (Рис. 3.7). Толщина границы задается атрибутом **BORDER** (Листинг 3.4). При значении этого атрибута **BORDER=0**, граница отображаться не будет.

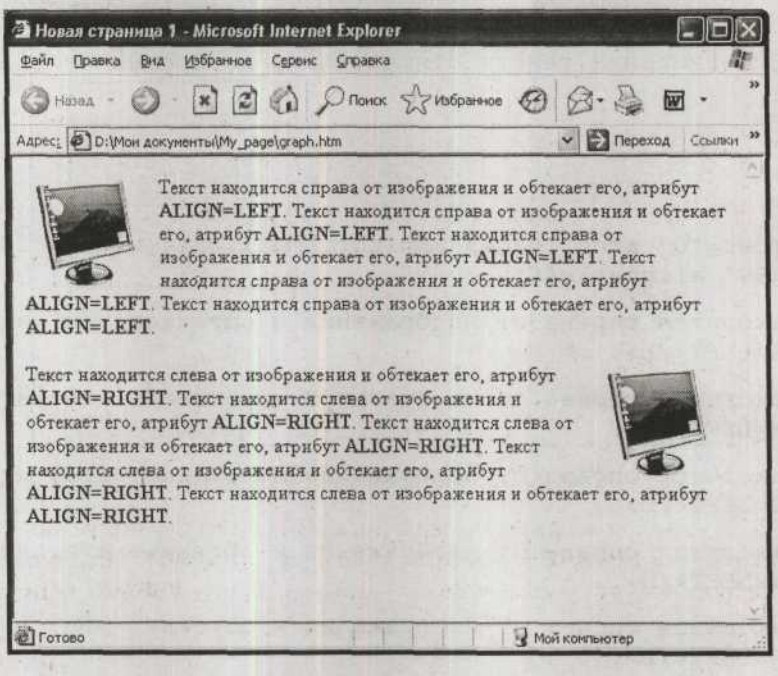


Рис. 3.6. Обтекание изображения текстом

Листинг 3.4. Отображение границ картинок

```
<html>
<head>
<title>Выравнивание текста относительно изображения</title>
</head>
<body>
<p>
<img border="0" src="images/monitor_middle.gif" width="200"
height="189">
<img border="1" src="images/monitor_middle.gif" width="200"
height="189">
<img border="5" src="images/monitor_middle.gif" width="200"
height="189"></p>
</body>
</html>
```

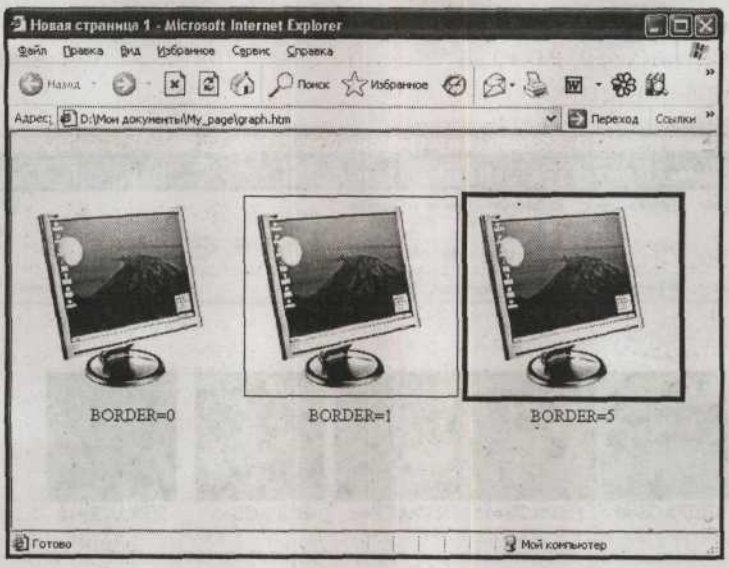


Рис. 3.7. Граница вокруг изображения

Интервал между изображением и соседними элементами страницы можно задать с помощью атрибутов **HSPACE** и **VSPACE**. С помощью **HSPACE** задается расстояние от картинки до соседних элементов по горизонтали, а **VSPACE** – по вертикали (Листинг 3.5).

Листинг 3.5. Интервал между изображением и соседними элементами страницы

```
<html>
<head>
<title></title>
</head>
<body>
<p>
<img border="0" src="images/black.gif" width="100"
height="100">
<img border="0" src="images/black.gif" width="100" height="100"
hspace="1">
<img border="0" src="images/black.gif" width="100" height="100"
hspace="3">
<img border="0" src="images/black.gif" width="100" height="100"
hspace="5">
<img border="0" src="images/black.gif" width="100" height="100"
hspace="10">
```

```
</p>
</body>
</html>
```

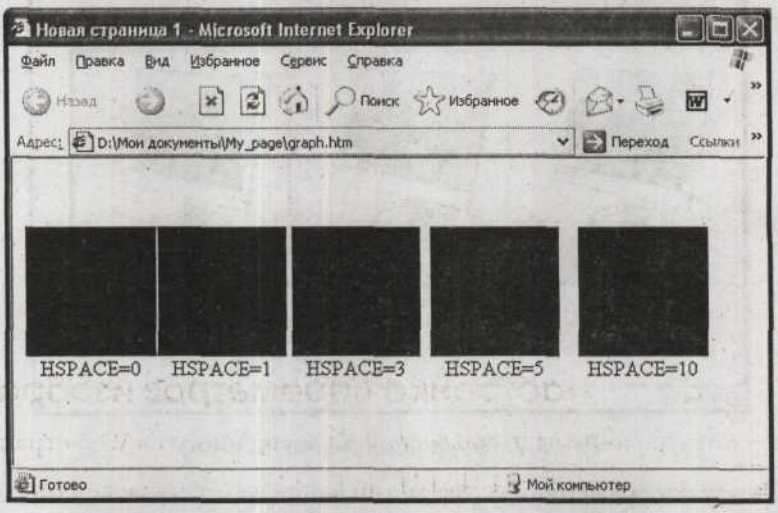


Рис. 3.8. Различный интервал между изображением и соседними объектами

Добавление графики на HTML-страницу в программе FrontPage

Теперь, когда рассмотрены основные особенности размещения графики на страницах HTML, разберемся с тонкостями работы с картинками в программе FrontPage.

Чтобы добавить изображение на текущую Web-страницу в программе FrontPage:

- Щелкните мышью в той части макета Web-страницы, куда вы хотите добавить изображение.
- Выберите команду меню **Вставка ♦ Рисунок ♦ Из файла** (Insert ♦ Image ♦ From file).

Либо

- Нажмите кнопку на панели инструментов программы.
- Появится диалог **Рисунок** (Image), Рис. 3.9.
- Перейдите к каталогу, в котором содержится графический файл, который вы хотите добавить на страницу, и щелкните мышью на его названии.
- Щелкните мышью на кнопке **Вставить** (Insert). Диалог будет закрыт, а изображение добавлено на Web-страницу.

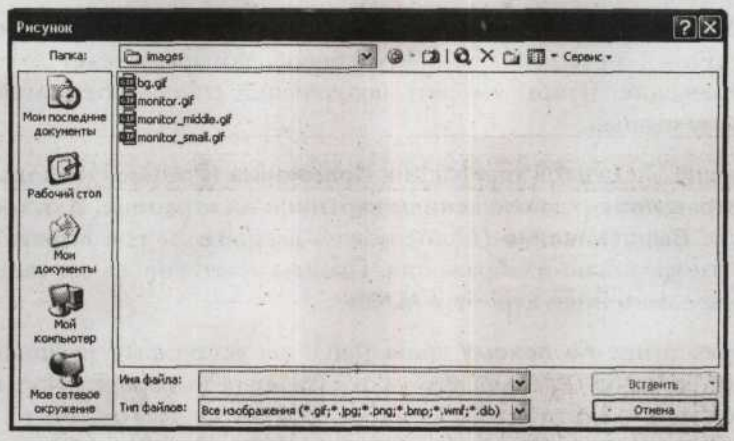


Рис. 3.9. Диалог ***Рисунок*** *(Image)*

Настройка параметров изображения

Чтобы настроить параметры изображения, размещенного на Web-странице:

- Щелкните правой кнопкой мыши на изображении. Появится контекстное меню.
- Щелкните мышью на пункте **Свойства рисунка** (Image properties) контекстного меню. Появится диалог **Свойства рисунка** (Image properties).

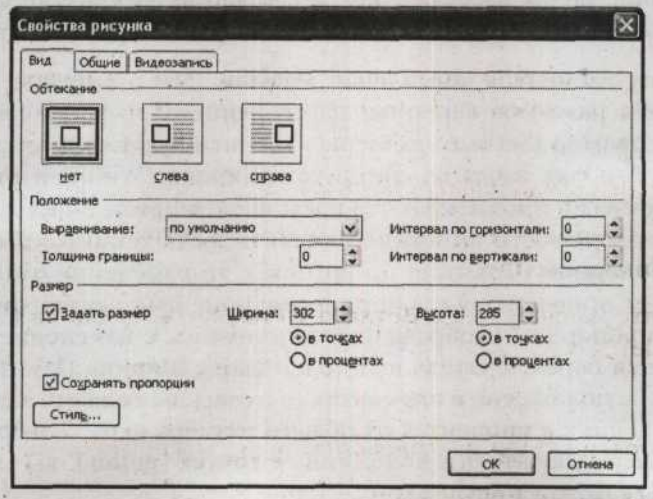


Рис. 3.10. Вкладка ***Вид*** *(Appearance) диалога* ***Свойства рисунка*** *(Image properties)*

Во вкладках этого диалога собраны все настройки изображений. Рассмотрим их по очереди.

Вкладка **Вид** (Appearance). В эту вкладку сведены все настройки, касающиеся отображения картинок на странице.

- Иконки **нет** (none), **слева** (left) и **справа** (right) группы настроек **Обтекание** (Flow), позволяют выбрать способ, которым текст будет обтекать изображение. Чтобы выбрать подходящий способ обтекания, щелкните по нему мышью.

- В группе элементов управления **Положение** (Position), находятся элементы управления расположением картинки на странице. В открывающемся списке **Выравнивание** (Align) можно выбрать способ выравнивания текста относительно изображения. Пункты этого списка соответствуют различным значениям атрибута **ALIGN**.

- Выбрав пункт **по левому краю** (left), вы установите расположение картинки слева от обтекающего ее текста, если выберете пункт **по правому краю** (right), картинка будет располагаться справа от текста. Пункты **по верхнему краю** (top), **по середине** (middle) и **по нижнему краю** (bottom) позволяют настроить выравнивание текста по верхней границе изображения, по его середине и по нижней границе соответственно. Если вы выберете пункт меню **по умолчанию** (default), то изображение будет упорядочиваться так, как задано в настройках браузера или настройках всей страницы.

- В поле ввода со счетчиком **Толщина границы** (border thickness) вы можете задать толщину рамки, окружающий картинку.

- Поле ввода со счетчиком **Интервал по горизонтали** (Horizontal space) задает расстояние между изображением и соседними элементами страницы по горизонтали, а поле **Интервал по вертикали** (Vertical space) – по вертикали.

- В группу элементов управления **Размер** (Size), сведены настройки, касающиеся размеров картинки на странице. При установленном флажке **Задать размер** (Set size) размеры картинки будут явно указываться в теге **<IMG>**. В полях ввода со счетчиком **Ширина** (Width) и **Высота** (Height) автоматически прописываются настоящие ширина и высота картинки, но при необходимости их можно изменить. Если установлен флажок **Сохранять пропорции** (Constrain proportions), то изменение одного из этих полей будет приводить к пропорциональному изменению другого поля так, чтобы пропорции изображения не менялись с изменением его размера. Установив переключатель под полем ввода **Ширина** (Width), или **Высота** (Height) в положение **в процентах** (percent) вы сможете задавать размеры изображениях в процентах от общего размера окна Web-браузера, установив же переключатель в положение **в точках** (points), вы сможете указать размеры картинки в точках.

Щелкнув мышью на ярлыке вкладки **Общие** (General) диалога **Свойства рисунка** (Image properties), вы перейдете к общим настройкам свойств рисунка (Рис. 3.11). В поле ввода **Текст** (Text) группы настроек **Альтернативные представления** (Alternate Appearances) вы можете ввести текст, который будет назначен атрибуту **ALT** тега **<IMG>**. Этот текст будет замещать изображение в том случае, если оно не будет загружено.

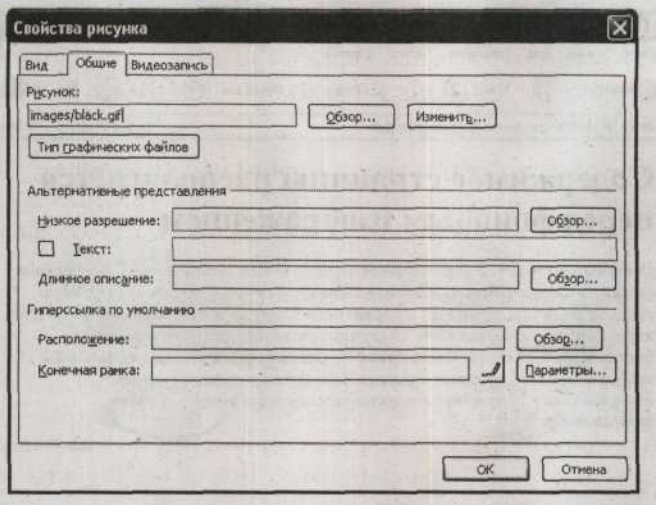


Рис. 3.11. Вкладка ***Общие*** *(General) диалога* ***Свойства рисунка*** *(Image properties)*

Сделав все необходимые настройки, щелкните мышью на кнопке **OK**, чтобы принять изменения и закрыть диалог.

Еще одним способом применения графики на Web-страницах является использование изображений в качестве фона страницы (Рис. 3.12). Фоновый рисунок можно задать Web-странице при помощи атрибута **BACKGROUND** тега **<BODY>** (Листинг 3.6).

Листинг 3.6. *Интервал между изображением и соседними элементами страницы*

```
<html>
<head>
<title></title>
</head>
<body background="images/monitor_back.gif">
<h1>
Содержимое страницы располагается перед фоновым изображением
</h1>
<p>
Содержимое страницы располагается перед фоновым изображением.
Содержимое страницы располагается перед фоновым изображением.
Содержимое страницы располагается перед фоновым изображением.
</p>
</body>
</html>
```

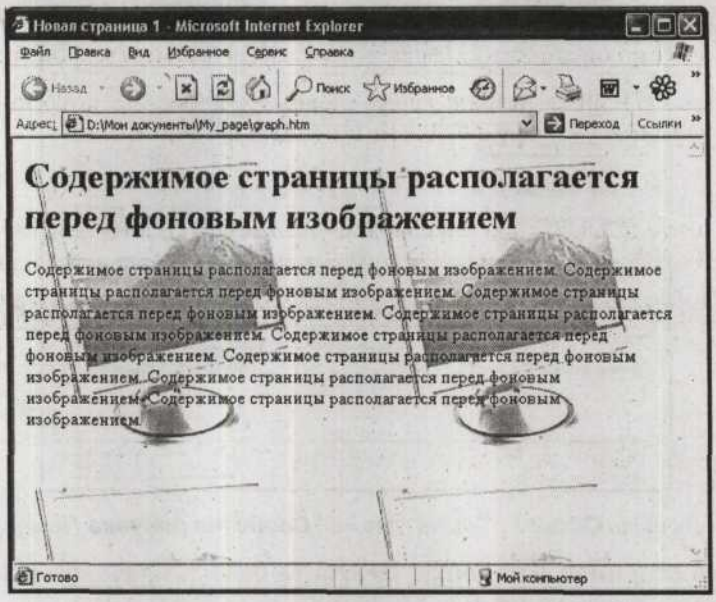


Рис. 3.12. Фоновое изображение на странице

Чтобы добавить фоновое изображение на текущую Web-страницу в программе FrontPage:

- Выберите команду меню **Формат ♦ Фон** (Format ♦ Background). Откроется вкладка **Форматирование** (Format) диалога **Свойства страницы** (Page properties), представленная на Рис. 3.13.

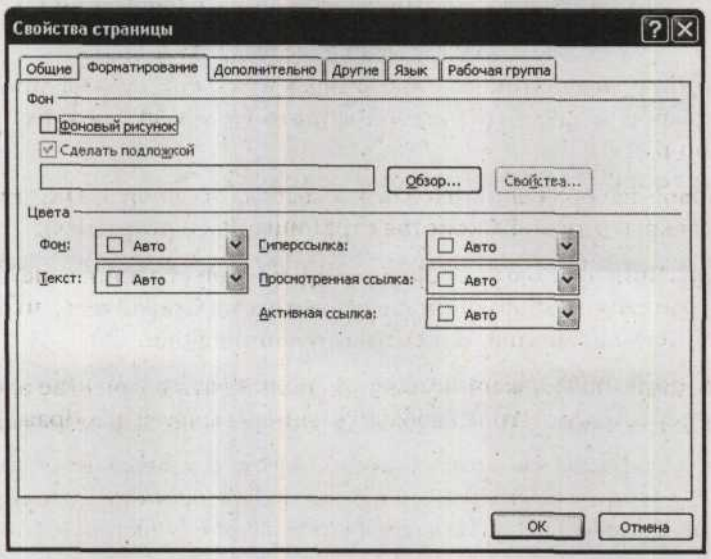


Рис. 3.13. Вкладка ***Форматирование*** *(Format) диалога* ***Свойства страницы*** *(Page properties)*

- Установите флажок **Фоновый рисунок** (Background picture). Этим вы активизируете возможность размещения картинки в качестве фона страницы.
- Щелкните мышью в поле ввода и введите адрес, по которому располагается картинка, которую вы хотите сделать фоновой.

Либо

- Щелкните мышью на кнопке **Обзор** (Browse), откроется диалог **Выбрать фоновый рисунок** (Choose background picture), представленный на Рис. 3.14.

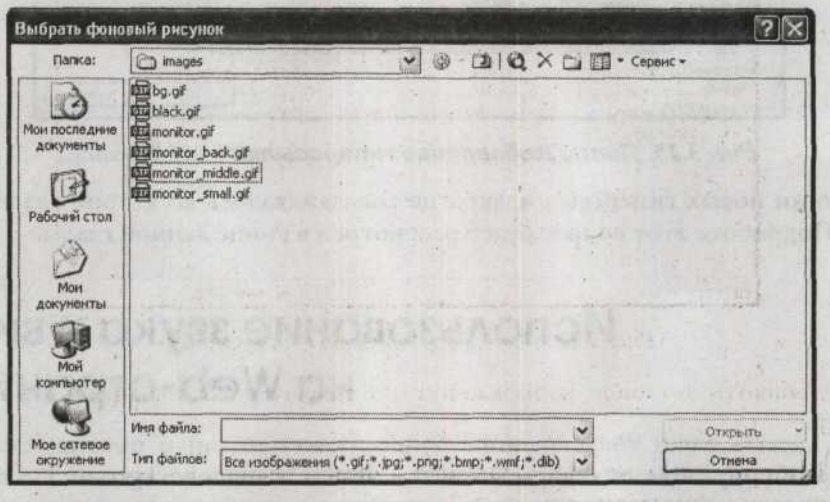


Рис. 3.14. *Диалог* ***Выбрать фоновый рисунок*** *(Choose background picture)*

- Перейдите к каталогу, в котором находится изображение, которое вы хотите сделать фоновым, и щелкните мышью на его названии.
- Щелкните мышью на кнопке **Открыть** (Open), чтобы принять сделанный выбор и закрыть диалог **Выбрать фоновый рисунок** (Choose background picture).

Выбрав фоновое изображение, щелкните мышью на кнопке **ОК**, чтобы принять изменения и закрыть диалог **Свойства страницы** (Page properties).

При использовании фоновых изображений на Web-страницах не забывайте об удобстве восприятия информации пользователем. Старайтесь, чтобы содержимое страницы легко воспринималось на выбранном фоне.

Наряду с текстами, изображения можно использовать в качестве элементов привязки для гиперссылок. Чтобы добавить гиперссылку к изображению на Web-странице:

- Щелкните правой кнопкой мыши на изображении, к которому вы хотите добавить гиперссылку. Появится контекстное меню.
- Щелкните мышью на пункте **Гиперссылка** (Hyperlink) контекстного меню. Появится диалог **Добавление гиперссылки** (Add hyperlink), Рис. 3.15.

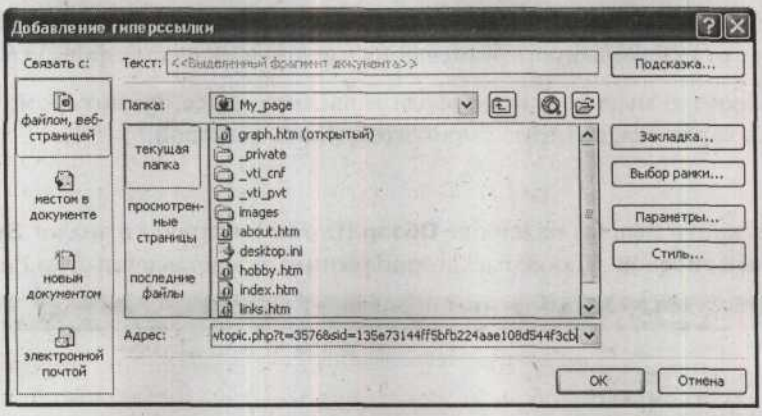


Рис. 3.15. *Диалог* ***Добавление гиперссылки*** *(Add hyperlink)*

О создании новых гиперссылок вкратце рассказывалось во второй главе данной книги. Подробнее этот вопрос будет рассмотрен в конце данной главы.

Использование звука и видео на Web-страницах

Чтобы сделать ваши Web-страницы более динамичными и привлекательными, можно разнообразить их оформление звуковыми и видеовставками. Главное, что необходимо принимать во внимание, – это то, что звук и видео обычно занимают достаточно большой объем, что негативно скажется на времени загрузки вашей Web-страницы пользователями, у которых медленное подключение к сети Интернет. Кроме того, для воспроизведения звука и видео Web-браузер пользователя должен быть оснащен специальными модулями для работы с теми или иными форматами файлов. Пользуясь некоторыми наиболее популярными форматами, вы можете быть практически уверены, что пользователь сможет насладиться красотами вашей Web-страницы в полной мере. Поддержка же менее популярных форматов браузером пользователя не гарантируется. И, хотя не составляет большого труда скачать дополнительные модули для воспроизведения того или иного типа файлов, никто не может вам обещать, что ради просмотра пары клипов с вашего сайта, пользователь будет устанавливать какие-либо модули или программы на свой компьютер.

Форматы звуковых файлов

Для размещения на Web-страницах можно использовать звуковые файлы двух разных видов. Первый вид – оцифрованный звук. В таких файлах звук записывается в виде последовательной записи уровня громкости звучания. Уровень громкости записывается несколько десятков тысяч раз в секунду. Преимуществом таких звуковых файлов является то, что в них может быть записан абсолютно любой звук – музыка, человеческая речь, различные шумы, и звучать такие файлы

будут примерно одинаково на любом компьютере, оборудованном аудиосистемой. Недостатком оцифрованного звука является большой объем, занимаемый такими файлами.

Самым простым форматом оцифрованного звука является формат WAV. Поддержка этого формата есть практически везде. Но главным недостатком этого формата является то, что данные в нем хранятся без сжатия, занимая очень много места. Файлы этого формата имеют расширение **.wav**.

Для размещения на Web-страницах большого объема звука удобнее использовать формат MP3. Этот формат характерен тем, что звуковые данные в таком файле упаковываются с учетом особенностей восприятия звука человеческим ухом. Это позволяет уменьшать размеры файла до 10 раз по сравнению в форматом WAV. Файлы формата MP3 имеют расширение **.mp3**. Также достаточно популярен для размещения звука в Интернете другой формат сжатого оцифрованного звука – Real Audio. Эти файлы имеют расширение **.ra** или **.ram.**

Другим видом звуковых компьютерных файлов являются MIDI файлы. В эти файлы записывается подобие нотных партитур музыкальных инструментов, на основе которых воспроизводят музыкальную последовательность специальные MIDI-синтезаторы, входящие в состав любой звуковой карты. Файлы такого типа занимают очень мало места. Музыкальное произведение длительностью несколько минут может занимать всего пару десятков килобайт, но звучание таких файлов обычно очень «искусственное» и в них невозможно записать звуки, для которых нет подходящих музыкальных инструментов. Файлы MIDI имеют расширение **.mid** или **.rmi**.

Самыми популярными форматами видеофайлов в сети Интернет являются формат QuickTime, файлы которого имеют расширение **.mov**, формат RealMedia, файлы которого имеют расширение **.rm**, и файлы MPEG с расширением **.mpg** или **.mpeg**.

Добавление фонового звука на Web-страницу

Звуковой файл можно назначить в качестве фонового сопровождения Web-страницы. Для этих целей можно использовать файлы WAV, MIDI и Real Audio. Звуковые файлы добавляются на Web-страницу с помощью тега **<BGSOUND>** в заголовке HTML-документа.

Добавить фоновый звук на Web-страницу в программе FrontPage можно с помощью диалога **Свойства страницы** (Page properties), Рис. 3.16. Чтобы открыть его, выберите команду меню **Файл ◆ Свойства** (File properties).

В группу элементов управления **Фоновый звук** (Background sound) собраны настройки управляющие воспроизведением фонового звука.

Выберите звуковой файл, который будет фоном вашей страницы:

Щелкните мышью в поле ввода **Расположение** (Location) и введите адрес, по которому располагается звуковой файл, который вы хотите сделать фоновым.

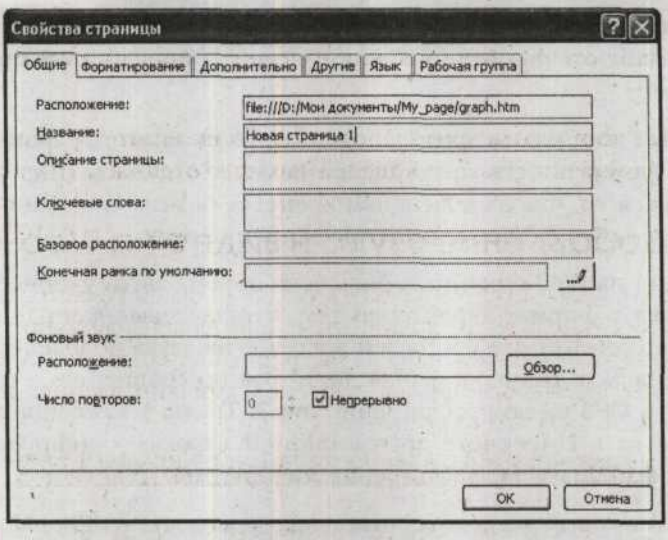


Рис. 3.16. Диалог ***Свойства страницы*** *(Page properties)*

Либо

- Щелкните мышью на кнопке **Обзор** (Browse), откроется диалог **Фоновый звук** (Background sound), Рис. 3.17.

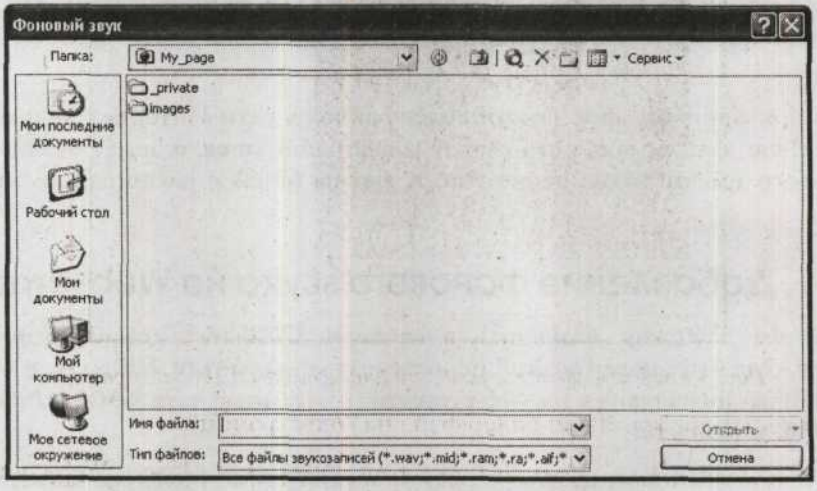


Рис. 3.17. Диалог ***Фоновый звук*** *(Background sound)*

- Перейдите к каталогу, в котором находится звуковой файл, который вы хотите сделать фоновым, и щелкните мышью на его названии.
- Щелкните мышью на кнопке **Открыть** (Open), чтобы принять сделанный выбор и закрыть диалог **Фоновый звук** (Background sound).

Если установлен флажок **Непрерывно** (Continuous), то фоновый звук будет непрерывно воспроизводиться, пока страница будет открыта. Если этот флажок

сбросить, то в поле ввода со счетчиком **Число повторов** (Repeats number) вы можете указать, сколько раз будет воспроизведен фоновый звук после открытия страницы.

После настройки всех параметров фонового звука щелкните мышью на кнопке **OK**, чтобы принять изменения и закрыть диалог **Свойства страницы** (Page properties).

Добавление звука и видео на Web-страницу

Звуковые и видео-файлы можно размещать также на самой Web-странице (Рис. 3.18). При условии, что к Web-браузеру пользователя подключены дополнительные модули по воспроизведению нужного типа файлов, перед пользователем предстанет некое подобие видео/музыкального плеера с элементами управления: перемотка, пауза/воспроизведение, регулятор громкости.

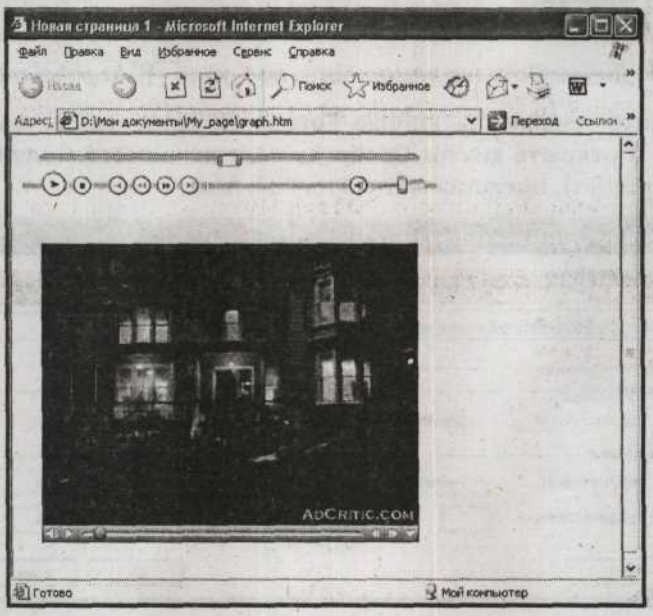


Рис. 3.18. Размещение звуковых и виде- файлов на Web-странице

Чтобы добавить звуковой или видео-файл на Web-страницу:

- Выберите пункт меню **Вставка Веб-компонент** (Insert Web-component). Откроется диалог **Вставка компонента веб-узла** (Web-component insertion), Рис. 3.19.
- В списке **Тип компонента** (Component type) щелкните мышью на пункте **Дополнительные элементы** (Additional elements).
- В списке **Выберите элемент управления** (Choose control element) щелкните мышью на пункте **Подключаемый модуль** (Plug-in module).

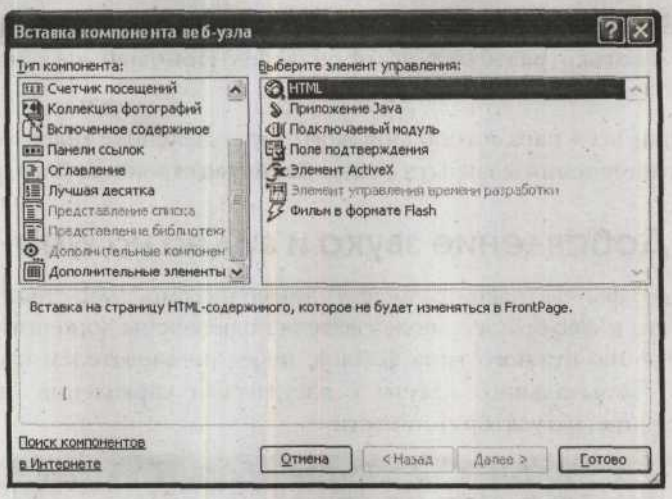


Рис. 3.19. Диалог ***Вставка компонента веб-узла*** *(Web-component insertion)*

- Щелкните мышью на кнопке **Готово** (Finish), чтобы закрыть текущий диалог и открыть диалог **Свойства подключаемого модуля** (Plug-in module properties), представленный на Рис. 3.20.

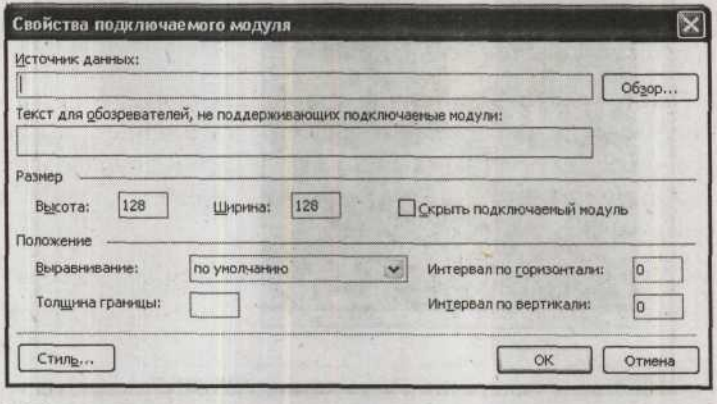


Рис. 3.20. Диалог ***Свойства подключаемого модуля*** *(Plug-in module properties)*

В появившемся диалоге настраиваются все параметры воспроизведения подключаемого файла.

Выберите файл, который будете размещать на Web-странице:

- Щелкните мышью в поле ввода **Источник данных** (Data source) и введите адрес, по которому располагается файл.

Либо

- Щелкните мышью на кнопке **Обзор** (Browse), откроется диалог **Выбор источника данных подключаемого модуля** (Choose data source for plug-in module), представленный на Рис. 3.21.

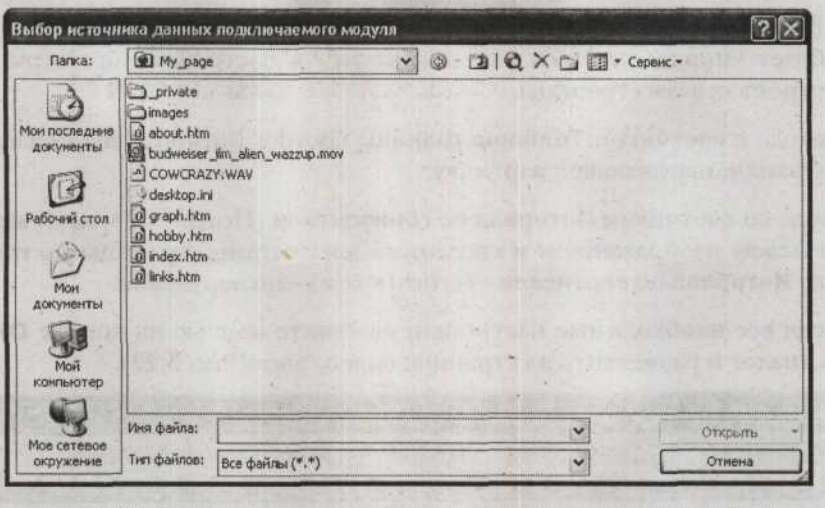


Рис. 3.21. Выбор источника данных подключаемого модуля
(Choose data source for plug-in module)

- Перейдите к каталогу, в котором находится искомый файл.
- Щелкните мышью на кнопке **Открыть** (Open), чтобы принять сделанный выбор и закрыть диалог **Выбор источника данных подключаемого модуля** (Choose data source for plug-in module).

В поле ввода **Текст для обозревателей, не поддерживающих подключаемые модули** (Text for browsers, that don't support plug-in modules) вы можете ввести текст, который будет отображаться вместо звукового или видео-файла, в том случае, если браузер пользователя не поддерживает внешние модули, необходимые для воспроизведения файла.

В полях ввода **Ширина** (Width) и **Высота** (Height) группы настроек **Размер** (Size) вы можете задать размер, который примет плеер звука или видео на вашей Web-странице.

В группе элементов управления **Положение** (Position) находятся элементы управления расположением картинки на странице. В открывающемся списке **Выравнивание** (Align) можно выбрать способ выравнивания текста относительно изображения.

- Выбрав пункт **по левому краю** (left), вы установите расположение картинки слева от обтекающего ее текста.
- Если вы выберете пункт **по правому краю** (right), картинка будет располагаться справа от текста.
- Пункты **по верхнему краю** (top), **по середине** (middle) и **по нижнему краю** (bottom) позволяют настроить выравнивание текста по верхней границе изображения, по его середине и по нижней границе, соответственно.

- Если вы выберете пункт меню **по умолчанию** (default), то изображение будет упорядочиваться так, как задано в настройках браузера, или настройках всей страницы.

В поле ввода с счетчиком **Толщина границы** (Border thickness) вы можете задать толщину рамки, окружающей картинку.

Поле ввода со счетчиком **Интервал по горизонтали** (Horizontal space), задает расстояние между изображением и соседними элементами страницы по горизонтали, а поле **Интервал по вертикали** (Vertical space) – по вертикали.

Произведя все необходимые настройки, щелкните мышью на кнопке **OK**, чтобы закрыть диалог и разместить на странице видео/звук (Рис. 3.22).

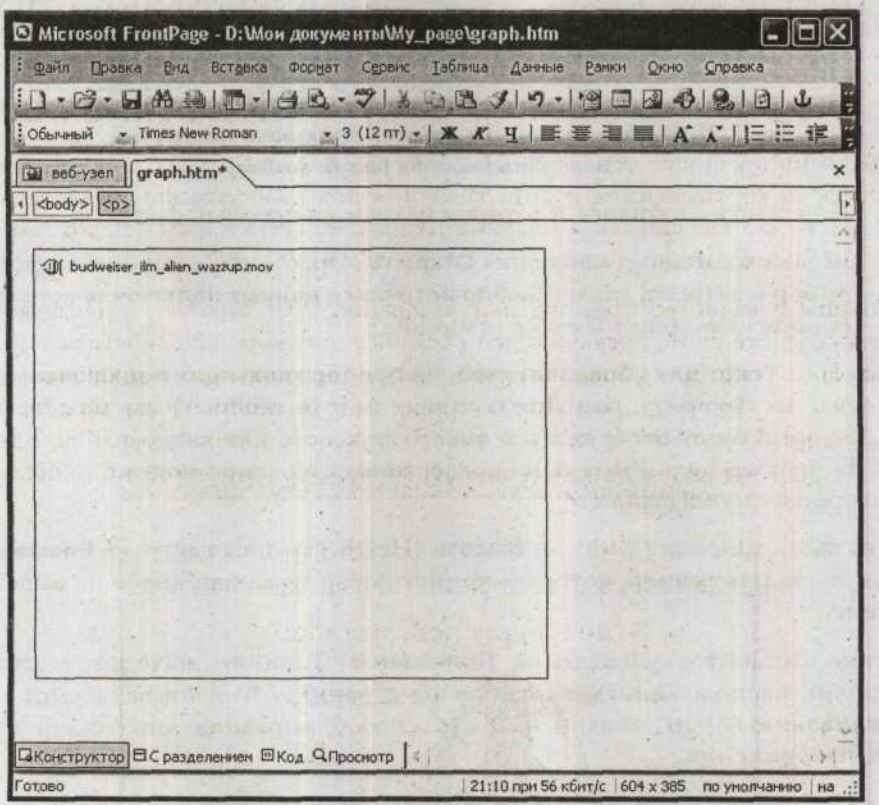


Рис. 3.22. Размещение файла видео на макете страницы

После этого, когда вы будете в очередной раз сохранять свою Web-страницу, откроется диалог **Сохранение внедренных файлов** (Save embedded files), Рис. 3.23. Щелкните в нем мышью на кнопке **OK**, чтобы сохранить все звуковые/видео файлы в каталоге вашего Web-сайта.

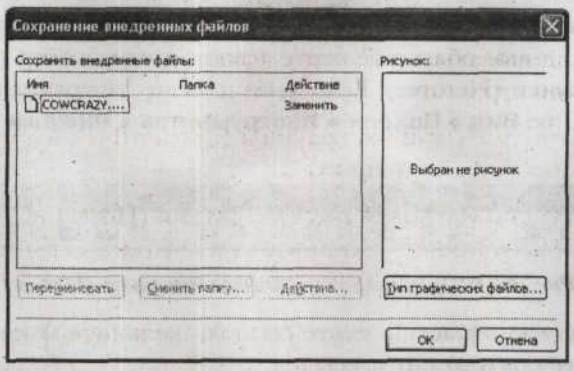


Рис. 3.23. Диалог ***Сохранение внедренных файлов*** *(Save embedded files)*

Карты ссылок

Для создания красивых и наглядных меню навигации по разделам сайта можно использовать карты ссылок. Карта ссылок – это изображение, разделенное на несколько областей; щелкая мышью на разных областях карты ссылок, пользователь будет переходить по разным ссылкам.

Рассмотрим в качестве примера использования карт ссылок следующую ситуацию: вы создаете сайт, посвященный космическому кораблю собственной конструкции. На главной странице вы размещаете рисунок корабля, щелкая на различных частях которого, посетитель страницы может перейти на разделы сайта, посвященные различным модулям корабля (Рис. 3.24).

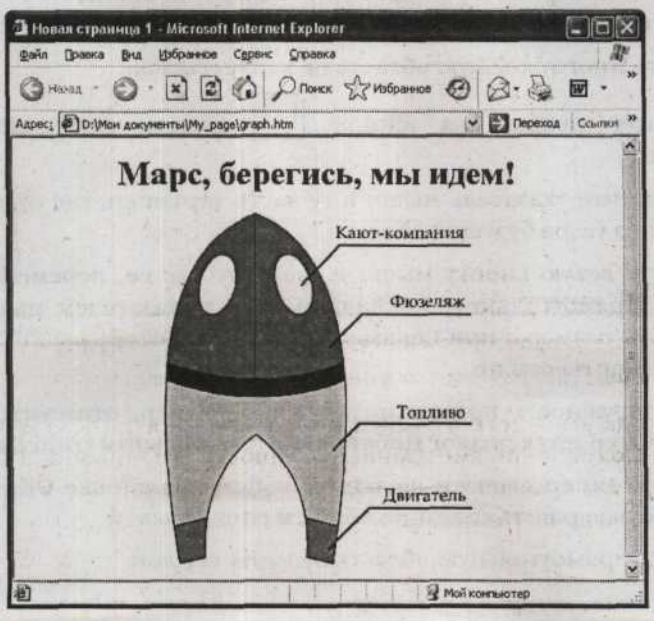


Рис. 3.24. Карта ссылок

Области на карте ссылок бывают трех различных видов: прямоугольные, круглые и многоугольные. Новые области к карте ссылок добавляются с помощью панели инструментов **Рисунки** (Pictures). Если этой панели инструментов на экране нет, выберите пункт меню **Вид ♦ Панели ♦ Инструментов ♦ Рисунки** (View ♦ Toolbars ♦ Pictures), Рис. 3.25.

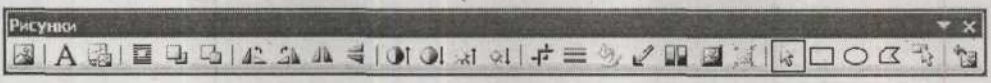


Рис. 3.25. Панель инструментов ***Рисунки*** *(Pictures)*

Чтобы добавить новую область к карте ссылок, щелкните мышью на картинке, к которой добавляете область карты ссылок.

Чтобы добавить круглую область на карту ссылок:

- Щелкните мышью на кнопке  панели инструментов **Рисунки** (Pictures).
- Переместите указатель мыши в ту часть картинки, где будет располагаться середина будущей области.
- Нажмите левую кнопку мыши и, не отпуская ее, перемещайте указатель мыши. Круг, появившийся под указателем мыши и изменяющий свои размеры при перемещении мыши, отображает размер будущей области карты ссылок.
- Когда окружность примет подходящий размер, отпустите левую кнопку мыши. Откроется диалог **Добавление гиперссылки** (Add hyperlink).
- Создайте гиперссылку и щелкните мышью на кнопке **OK**, чтобы закрыть диалог и завершить создание области гиперссылки.

Чтобы добавить многоугольную область на карту ссылок:

- Щелкните мышью на кнопке  панели инструментов **Рисунки** (Pictures).
- Переместите указатель мыши в ту часть картинки, где будет располагаться один из углов будущей области.
- Нажмите левую кнопку мыши и, не отпуская ее, перемещайте указатель мыши. Прямоугольник, появившийся под указателем мыши и изменяющий свои размеры при перемещении мыши, отображает размер будущей области карты ссылок.
- Когда окружность примет подходящий размер, отпустите левую кнопку мыши. Откроется диалог **Добавление гиперссылки** (Add hyperlink).
- Создайте гиперссылку и щелкните мышью на кнопке **OK**, чтобы закрыть диалог и завершить создание области гиперссылки.

Чтобы добавить прямоугольную область на карту ссылок:

- Щелкните мышью на кнопке  панели инструментов **Рисунки** (Pictures).
- Переместите указатель мыши в ту часть картинки, где будет располагаться один из углов будущей области.
- Щелкните мышью над точкой одного из углов будущей области.
- Переместите указатель мыши к месту следующего угла будущей области и щелкните на нем мышью. От места первого щелчка к месту второго протянется прямая.
- Последовательно щелкайте на местах всех оставшихся углов будущей карты ссылок. Образующаяся ломаная линия будет границей области.
- В месте последнего угла создаваемого многоугольника дважды щелкните мышью. Откроется диалог **Добавление гиперссылки** (Add hyperlink).
- Создайте гиперссылку и щелкните мышью на кнопке **OK**, чтобы закрыть диалог и завершить создание области гиперссылки.

Чтобы изменить размеры области карты ссылок после ее создания:

- Щелкните мышью на области карты ссылок, которую хотите изменить. Область будет окружена рядом квадратных маркеров.
- Переместите указатель мыши на один из этих маркеров и нажмите левую кнопку мыши.
- Не отпуская кнопку мыши, перемещайте указатель мыши в другую позицию. Маркер будет перемещаться вслед за указателем мыши, а вслед за ним будут передвигаться и линии, показывающие границу области.
- Когда вы будете удовлетворены результатом, отпустите левую кнопку мыши. Область карты ссылок изменена.
- По необходимости, проделайте описанную процедуру и с остальными маркерами, окружающими область карты ссылок.

Табличная верстка Web-страниц

Все HTML-страницы, которые создавались нами до сих пор, были достаточно просты: расположенный в одну колонку текст с некоторым количеством ссылок и графики. Но большинство настоящих Web-страниц имеют более сложное строение (Рис. 3.26).

Рис. 3.26. Web-страница средней степени сложности

Таблица в качестве каркаса Web-страницы

Каким же образом упорядочиваются элементы оформления на таких Web-страницах? Одним из способов является использование таблиц в качестве каркаса всей Web-страницы. На Рис. 3.27 изображено типичное устройство страницы такого сайта. В основе его лежит таблица из трех столбцов и трех строк, причем в верхней и нижней строках таблицы ячейки объединены между собой.

Верхняя строка таблицы отведена под «шапку» Web-страницы. Здесь обычно располагается логотип и название сайта.

Первая ячейка второй строки таблицы содержит блок ссылок, с навигацией по разделам сайта. Во второй ячейке второй строки размещается основное содержимое Web-страницы, ради которого она и создавалась. В третьей ячейке располагаются новости сайта и различные объявление.

В третьей строке таблицы-каркаса находится навигационный блок, дублирующий блок, находящийся в первой ячейке второй строки. Это сделано на тот случай, если длина страницы будет существенно превышать высоту окна браузера и когда посетитель пролистает страницу до конца, блок навигации из второй строки таблицы скроется с экрана.

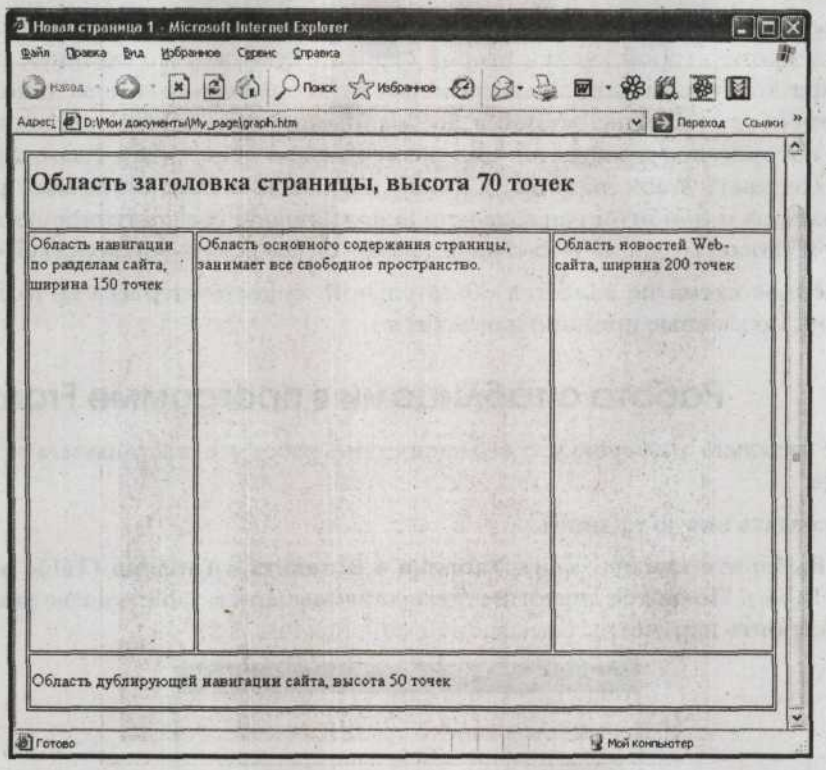


Рис. 3.27. Таблица, используемая в качестве каркаса Web-страницы

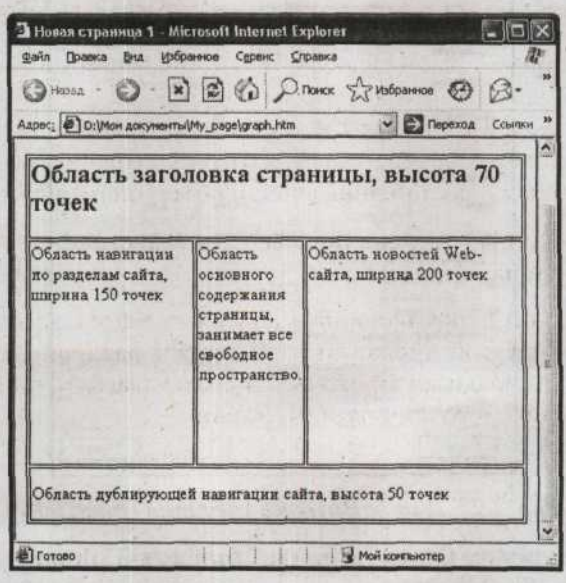


Рис. 3.28. Поведение каркаса Web-страницы при изменении размера окна браузера

Размеры всей таблицы задаются равными размерам окна браузера. Все ячейки таблицы, кроме второй ячейки второй строки имеют жестко заданные размеры. Благодаря этому, при различных размерах окна браузера, размеры дополнительных элементов страницы остаются постоянными в отличие от основного содержания, которое подстраивается под размеры окна (Рис. 3.28). Благодаря этому можно создавать Web-страницы, хорошо выглядящие при различных размерах окна браузера и при этом рационально использующие все доступное пространство. Такой способ создания Web-страниц называется «резиновой верстkой».

Приведенная схема не является обязательной, существует масса ее вариаций и различий, но главные принципы остаются.

Работа с таблицами в программе FrontPage

Давайте детально разберемся с особенностями работы с таблицами в программе FrontPage.

Чтобы создать новую таблицу:

- Выберите команду меню **Таблица ◆ Вставить ◆ Таблица** (Table ◆ Insert ◆ Table). Появится диалог **Вставка таблицы** (Insert table), позволяющий настроить параметры создаваемой таблицы Рис. 3.29.

Рис. 3.29. Диалог ***Вставка таблицы*** *(Insert table)*

В полях ввода с счетчиком **Строк** (Rows) и **Столбцов** (Columns) группы элементов настройки **Размер** (Size) задается количество строк и столбцов, которые будет содержать таблица.

- В группе элементов управления **Положение** (Position) находятся элементы управления расположением таблицы на странице и положением содержимого ячеек внутри таблицы.
- В открывающемся списке **Выравнивание** (Align) можно выбрать способ выравнивания содержимого ячеек. Впоследствии для каждой конкретной ячейки можно отдельно настроить параметры выравнивания ее содержимого.
- Открывающийся список **Обтекание** (Flow) задает способ обтекания таблицы текстом и другими элементами Web-страницы.
- Поле ввода с счетчиком **Поля ячеек** (Cell fields) определяет расстояние между границами ячеек и их содержимым.
- В поле ввода с счетчиком **Интервал ячеек** (Cells interval) задает расстояние между соседними ячейками таблицы.
- Установив флажок **Задать ширину** (Set width), вы можете явно задать ширину таблицы. Ширина задается в поле ввода под флажком **Задать ширину** (Set width). В зависимости от положения переключателя справа от поля ввода, ширину можно задавать **в точках** (points) и **в процентах** (percents). Под шириной 100% подразумевается ширина окна Web-браузера.
- Установив флажок **Задать высоту** (Set height), вы можете явно задать высоту таблицы. Высота задается в поле ввода под флажком **Задать высоту** (Set height). В зависимости от положения переключателя справа от поля ввода, высоту можно задавать **в точках** (points) и **в процентах** (percents). Под высотой 100% подразумевается высота окна Web-браузера.

В группе элементов управления **Границы** (Borders) располагаются настройки, связанные с отображением границ таблицы.

- Поле ввода с счетчиком **Размер** (Size) задает толщину границы таблицы в точках.
- Выпадающий список **Цвет** (Color) позволяет выбрать цвет, которым будет отображаться граница.
- Если установить флажок **Свернуть границу таблицы** (Cut-down table border), то граница будет отображаться в виде одиночной линии, иначе она будет принимать псевдо-объемный вид.

В группе **Фон** (Background) собраны элементы, управляющие фоновым изображением таблицы.

Выпадающий список **Цвет** (Color) позволяет выбрать цвет фона таблицы.

Если установлен флажок **Использовать фоновый рисунок** (Use background picture), то вы сможете задать таблице в качестве фона изображение.

Чтобы установить фоновое изображение для таблицы:

➢ Щелкните мышью в поле ввода и введите адрес, по которому располагается картинка, которую вы хотите сделать фоновой.

Либо

- Щелкните мышью на кнопке **Обзор** (Browse), откроется диалог **Выбрать фоновый рисунок** (Choose background picture).
- Перейдите к каталогу, в котором находится изображение, который вы хотите сделать фоновым, и щелкните мышью на его названии.
- Щелкните мышью на кнопке **Открыть** (Open), чтобы принять сделанный выбор и закрыть диалог **Выбрать фоновый рисунок** (Choose background picture).

Если установить флажок **По умолчанию для новых таблиц** (By default for new tables) группы элементов управления **По умолчанию** (By default), то все параметры, которые вы установили для этой таблицы, станут настройками по умолчанию для всех следующих таблиц, которые вы будете создавать.

Чтобы завершить процесс создания таблицы и закрыть диалог **Вставка таблица** (Insert table), щелкните мышью на кнопке **OK**. Таблица будет создана.

Впоследствии, если у вас возникнет необходимость изменить параметры уже созданной таблицы:

- Щелкните правой кнопкой мыши где-нибудь внутри таблицы. Появится контекстное меню.
- Щелкните мышью на пункте контекстного меню **Свойства таблицы** (Table properties). Откроется диалог **Свойства таблицы** (Table properties) с параметрами настройки таблицы (Рис. 3.30).

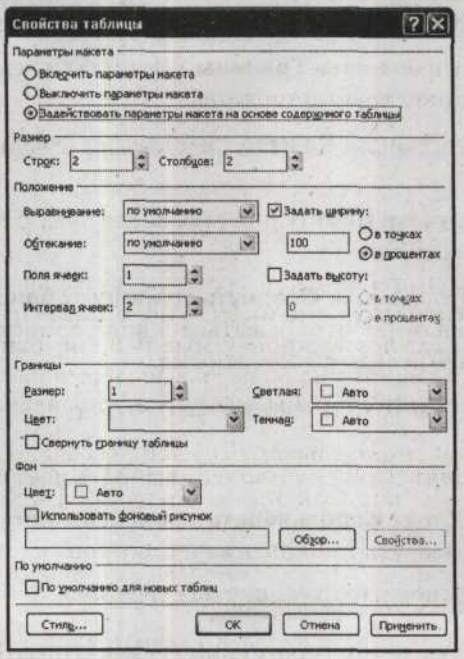


Рис. 3.30. Диалог ***Свойства таблицы*** *(Table properties)*

Настройки, содержащиеся в диалоге **Свойства таблицы** (Table properties), аналогичны настройкам в диалоге за одним исключением. В полях ввода с счетчиком **Строк** (Rows) и **Столбцов** (Columns) группы элементов настройки **Размер** (Size), в отличие от диалога **Вставка таблицы** (Insert table), нельзя задавать любое количество столбцов и строк, а можно лишь добавлять к уже существующим новые строки и столбцы.

Чтобы изменить параметры отдельной ячейки таблицы:

- Щелкните правой кнопкой мыши внутри этой ячейки. Появится контекстное меню.
- Щелкните мышью на пункте контекстного меню **Свойства ячейки** (Cell properties). Откроется диалог **Свойства ячейки** (Cell properties), представленный на Рис. 3.31.

***Рис. 3.31.** Диалог **Свойства ячейки** (Cell properties)*

Чтобы изменить параметры сразу нескольких ячеек таблицы:

- Выделите ячейки таблицы, параметры которых хотите изменить. Для этого переместите указатель мыши в положение над крайней выделяемой ячейкой, нажмите левую кнопку мыши и, не отпуская ее, переместите указатель мыши так, чтобы область выделения накрыла все ячейки, параметры которых вы хотите изменить.
- Отпустите левую кнопку мыши, выделение создано.
- Щелкните правой кнопкой мыши внутри области выделения. Появится контекстное меню.
- Щелкните мышью на пункте контекстного меню **Свойства ячейки** (Cell properties). Откроется диалог **Свойства ячейки** (Cell properties).

В группе элементов управления **Положение** (Position) находятся элементы управления расположением содержимого ячейки.

- В открывающемся списке **Выровнять по горизонтали** (Horizontal align) можно выбрать способ выравнивания содержимого ячейки, по горизонтали. Открывающийся список **Выровнять по вертикали** (Vertical align) позволяет выровнять содержимое ячейки по вертикали.
- Установив флажок **Задать ширину** (Set width), вы можете явно задать ширину ячейки. Ширина задается в поле ввода под флажком **Задать ширину** (Set width). В зависимости от положения переключателя справа от поля ввода, ширину можно задавать **в точках** (points) и **в процентах** (percents). Под шириной 100% подразумевается ширина окна Web-браузера за вычетом ширины остальных ячеек таблицы.
- Установив флажок **Задать высоту** (Set height), вы можете явно задать высоту ячейки. Высота задается в поле ввода под флажком **Задать высоту** (Set height). В зависимости от положения переключателя справа от поля ввода, высоту можно задавать **в точках** (points) и **в процентах** (percents). Под высотой 100% подразумевается высота окна Web-браузера за вычетом высоты остальных ячеек таблицы.
- В группе элементов управления **Границы** (Borders), располагаются настройки, связанные с отображением границ таблицы. Раскрывающийся список **Цвет** (Color) позволяет выбрать цвет, которым будет отображаться граница ячейки.
- В группе **Фон** собраны элементы, управляющие фоновым изображением ячейки. Раскрывающийся список **Цвет** (Color) позволяет выбрать цвет фона ячейки. Установив флажок **Использовать фоновый рисунок** (Use background picture), вы сможете задать таблице в качестве фона изображение.

Чтобы установить фоновое изображение для ячейки:

- ➢ Щелкните мышью в поле ввода и введите адрес, по которому располагается картинка, которую вы хотите сделать фоновой.

Либо

- ➢ Щелкните мышью на кнопке **Обзор** (Browse), откроется диалог **Выбрать фоновый рисунок** (Choose background picture).
- ➢ Перейдите к каталогу, в котором находится изображение, которое вы хотите сделать фоновым, и щелкните мышью на его названии.
- ➢ Щелкните мышью на кнопке **Открыть** (Open), чтобы принять сделанный выбор и закрыть диалог **Выбрать фоновый рисунок** (Choose background picture).

Установив все параметры выделенных ячеек таблицы, щелкните мышью на кнопке **OK**, чтобы принять изменения и закрыть диалог **Свойства ячейки** (Cell properties).

Чтобы объединить между собой несколько ячеек таблицы:

- Выделите ячейки таблицы, которые хотите объединить. Для этого переместите указатель мыши в положение над крайней выделяемой ячейкой, нажмите левую кнопку мыши и, не отпуская ее, переместите указатель мыши так, чтобы область выделения накрыла все ячейки, которые необходимо выделить.
- Отпустите левую кнопку мыши, выделение создано.
- Щелкните правой кнопкой мыши в пределах выделения. Появится контекстное меню.
- Щелкните мышью на пункте **Объединить ячейки** (Combine cells) контекстного меню. Ячейки будут объединены.

Чтобы разбить одну ячейку на несколько:

- Щелкните правой кнопкой мыши внутри разбиваемой ячейки. Появится контекстное меню.
- Щелкните мышью на пункте **Разбить ячейки** (Divide cells) контекстного меню. Появится диалог **Разбиение ячеек** (Divide cells), Рис. 3.32.

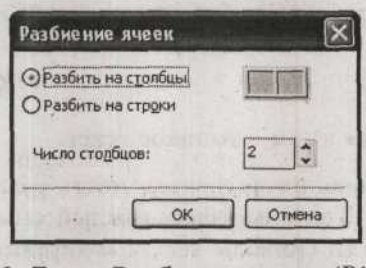


Рис. 3.32. Диалог ***Разбиение ячеек*** *(Divide cells)*

Если вы хотите разбить ячейку по горизонтали на несколько столбцов:

- Установите переключатель в положение **Разбить на столбцы** (Divide to columns).
- Впишите в поле с счетчиком **Число столбцов** (Columns number) число ячеек, на которое хотите разбить исходную ячейку.
- Щелкните мышью на кнопке **OK**, чтобы принять изменения и закрыть диалог.

Если вы хотите разбить ячейку по вертикали на несколько строк:

- Установите переключатель в положение **Разбить на строки** (Divide to rows).
- Впишите в поле с счетчиком **Число строк** (Rows number) число ячеек, на которое хотите разбить исходную ячейку.
- Щелкните мышью на кнопке **OK**, чтобы принять изменения и закрыть диалог.

Чтобы добавить новые строки к таблице:

- Щелкните правой кнопкой мыши на строке таблицы, ниже которой вы хотите добавить новую строку. Появится контекстное меню.

- Щелкните мышью на пункте **Добавить строки** (Add rows), контекстного меню. Строка будет добавлена.

Чтобы добавить новые столбцы к таблице:

- Щелкните правой кнопкой мыши на столбце таблицы, правее которого вы хотите добавить новую строку. Появится контекстное меню.
- Щелкните мышью на пункте **Добавить столбцы** (Add rows), контекстного меню. Столбец будет добавлен.

Вооружившись полученными в этом разделе знаниями, вы теперь можете создавать таблицы любой сложности и конфигурации для любых целей.

Форматирование текстов

Несмотря на то, что язык HTML предлагает обширные возможности по оформлению Web-страниц с использованием графики, звука и видео, основным содержимым большинства Web-сайтов остаются тексты. Поэтому необходимо всесторонне ознакомиться с особенностями работы с текстами в HTML.

Кодировки текста

Сначала давайте разберемся в том, что такое текст.

Любой текст – это некоторая последовательность различных символов из определенного набора знаков. В компьютерах каждый символ текста кодируется одним или двумя байтами. Если символы текста кодируются двумя байтами, то всего возможно 65535 различных знаков. Этого вполне достаточно, чтобы закодировать символы всех основных алфавитов мира и даже иероглифы. Такая кодировка текста называется Unicode (от Universal-code – универсальный код). В современном компьютерном мире идет постепенный переход к всеобщему использованию кодировки Unicode, но, тем не менее, в подавляющем числе случаев до сих пор используется система, при которой один символ текста кодируется одним байтом.

Такая система кодирования появилась, когда на компьютерах начали обрабатывать тексты и возникла необходимость упорядочивать системы кодирования символов в разных компьютерных системах. Была разработана система кодирования ASCII (American standard code for information interchange – Американский стандартный код обмена информацией). В этой системе кодирования под один символ текста отводилось семь бит, то есть всего 128 вариантов символов. Кодировка ASCII содержала прописные и строчные буквы английского алфавита, цифры, знаки препинания и некоторые дополнительные управляющие символы.

Когда компьютеры стали широко применяться и за пределами Северной Америки, возникла необходимость создавать тексты не только на английском языке. Кодировка ASCII была расширена до 8 бит, т. е. одного байта и количество вариантов символов текста увеличилось ровно в два раза, до 256 вариантов.

В стандартной 8-битной кодировке ASCII лишние 128 вариантов символов были отданы под различные дополнительные буквы алфавитов, основанных на латинице, например французского, немецкого, испанского и некоторые дополнительные символы.

Для алфавитов, не основанных на латинице, создавались дополнительные кодировки, первая половина которых совпадала с 7-битной кодировкой ASCII, а вторая содержала символы национального алфавита. Для русского языка, в свое время, было создано как минимум четыре различных кодировки символов, две из которых широко используются и поныне.

Первая из них – стандартная русская кодировка ОС Windows, называемая также 1251 кодовой страницей. Эта кодировка наиболее, распространена как среди обычных текстов, так и на Web-сайтах. Вторая кодировка – KOI8-P, эта кодировка часто используется при написании писем электронной почты и является основной русской кодировкой на компьютерах с ОС Linux, Unix и пр.

Чтобы браузер правильно отображал текст Web-страницы, он должен знать кодировку, в которой эта страница написана и правильно определить эту кодировку. Название кодировки, в которой написана Web-страница, Web-сервер может сам указать браузеру, также можно явно указать кодировку в теле HTML-страницы.

Чтобы явно задать кодировку страницы в программе FrontPage:

- Выберите пункт меню **Файл ◆ Свойства** (File ◆ Properties). Откроется диалог **Свойства страницы** (Page properties), представленный на Рис. 3.33.

Рис. 3.33. *Диалог* ***Свойства страницы*** *(Page properties)*

- Щелкните мышью на ярлыке вкладки настроек **Язык** (Language). Откроется вкладка настроек языковых параметров страницы (Рис. 3.34).

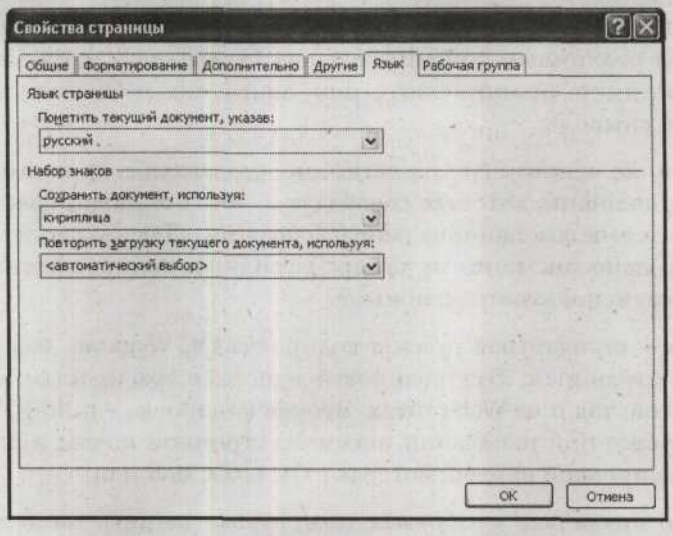


Рис. 3.34. Вкладка ***Язык*** *(Language) диалога* ***Свойства страницы*** *(Page properties)*

В выпадающем списке **Сохранить документ, используя** (Save page using) группы элементов управления **Набор знаков** (Characters set) можно выбрать кодовую страницу, в которой будет создаваться текст страницы.

- Чтобы установить кодировку страницы Windows, выберите пункт выпадающего списка **Сохранить документ, используя** (Save page using), **Кириллица** (Cyrillic).
- Чтобы установить кодировку страниц KOI8-P, выберите пункт выпадающего списка **Сохранить документ, используя** (Save page using), **Кириллица (KOI8-P)** (Cyrillic (KOI8-R).
- Чтобы установить кодировку страниц Unicode, выберите пункт выпадающего списка **Сохранить документ, используя** (Save page using), **Юникод** (Unicode).

Установив желаемую кодовую страницу, щелкните мышью на кнопке **OK**, чтобы принять настройки и закрыть диалог.

Элементы текстового оформления в документах HTML

Существует множество вариантов различного оформления текстов на Web-страницах (Рис. 3.35). Использование потенциала текстового форматирования языка HTML позволяет делать тексты более приятными визуально, оформлять их в согласии с вашими авторскими задумками, дает возможность явно подчеркивать важные слова и мысли.

Рассмотрим основные элементы текстового оформления, доступные на HTML-страницах.

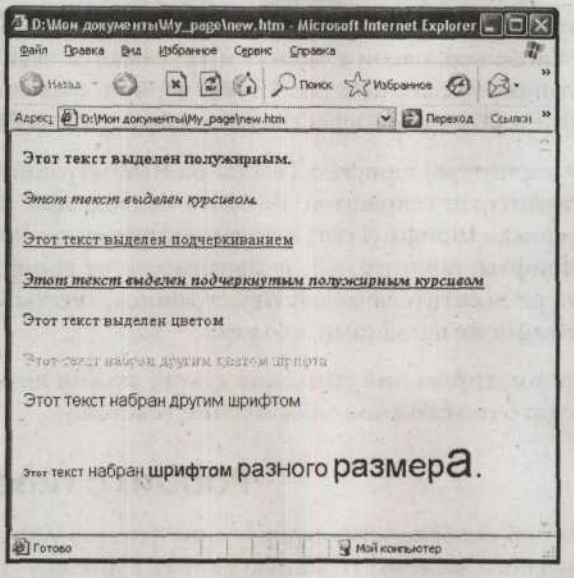


Рис. 3.35. Различные варианты оформления текста на Web-страницах

- Полужирное начертание текста. Нажав кнопку [Ж] панели инструментов программы FrontPage, вы можете вводить полужирный текст. Повторное нажатие этой кнопки вернет вас в режим ввода обычного текста.
- Курсивное начертание текста. Нажав кнопку [К] панели инструментов программы FrontPage, вы можете вводить курсивный текст. Повторное нажатие этой кнопки вернет вас в режим ввода обычного текста.
- Подчеркнутый текст. Нажав кнопку [Ч] панели инструментов программы FrontPage, вы можете вводить подчеркнутый текст. Повторное нажатие этой кнопки вернет вас в режим ввода обычного текста.
- Цветовое выделение текста. Нажав кнопку [ab] панели инструментов программы FrontPage, вы можете вводить текст, выделенный цветом. Щелкнув мышью на маленьком треугольнике справа от этой кнопки, вы откроете меню выбора цвета выделения. Чтобы выключить режим выделения цветом, выберите пункт **Авто** (Auto) меню выбора цвета выделения.
- Изменение цвета шрифта. Нажав кнопку [A] панели инструментов программы FrontPage, вы можете вводить текст шрифтом другого цвета. Щелкнув мышью на маленьком треугольнике справа от этой кнопки, вы откроете меню выбора цвета шрифта. Чтобы вернуть обычный цвет шрифта, выберите пункт **Авто** (Auto) меню выбора цвета выделения.
- Изменение размера шрифта. Выбрав размер шрифта в выпадающем списке **Размер** (Size) [3 (12 пт)] панели инструментов программы FrontPage, вы можете вводить текст разного размера. Всего в документах HTML поддержи-

вается 7 разных размеров шрифта, от самого маленького, 1, до самого большого, 7. В скобочках рядом с номером указывается эквивалент выбранного размера в типографских пунктах. Чтобы вернуть обычный размер шрифта, выберите пункт **Обычный** (Default) выпадающего списка **Размер** (Size).

- Изменение гарнитуры шрифта. Тексты на HTML-страницах можно набирать разными гарнитурами шрифтов. Выбрать подходящий шрифт можно в выпадающем списке **Шрифт** (Font) панели инструментов программы FrontPage. Выбирая шрифты, помните, что в зависимости от шрифтов, установленных на компьютере посетителя вашей Web-страницы, тексты могут отображаться не совсем такими же шрифтами, что у вас.

Разные способы форматирования символов текста можно комбинировать между собой, получая достаточно сложное оформление текстов.

Работа с гиперссылками

И, завершая рассказ об особенностях оформления текстов Web-страниц, снова коснемся самого интересного элемента HTML-документов, а именно – гиперссылок.

Адрес, на который ссылается гиперссылка, по-другому еще называется URL (Uniform Resource Locator – Унифицированный указатель информационного ресурса). Такое название говорит о том, что URL может ссылаться не только на HTML-документы, но и на любые другие ресурсы сети Интернет.

В начале текста ссылки указывается название протокола, по которому доступен ресурс. Если URL ссылается на файл, расположенный на Web-сервере, то ссылка будет начинаться с текста **http://**. Ссылки на почтовые адреса начинаются с **mailto://**. Если URL ссылается на файл, расположенный на FTP сервере, его текст будет начинаться с **ftp://** и т. д.

Например, текст URL файла **explosive.html**, находящегося в каталоге **main/bombs** Web-сайта с адресом **www.danger.com**, будет выглядеть так: **http://www.danger.com/ main.bombs /explosive.html.**

Ссылка, в которой указывается название протокола и полный путь к файлу, называется абсолютной. Если же ссылка ведет на файл, расположенный на том же компьютере, или сервере, что и Web-страница, на которой расположена гиперссылка, то возможен более короткий способ записи URL. Можно просто указать расположение файла, на который ведет ссылка, относительно текущей страницы. Такие ссылки называются относительными.

Например, если гиперссылка, размещенная на странице **index.html**, расположенной в корневом каталоге некоего Web-сайта, ведет к файлу **image.jpg**, расположенному в подкаталоге **images** этого же сайта, то адрес файла можно записать так: **images/image.jpg**.

Рассмотрим механизм создания гиперссылок в программе FrontPage.

Чтобы добавить гиперссылку к одному или нескольким элементам Web-страницы:

- Выделите этот элемент. Для этого переместите указатель мыши к одному из углов выделяемого элемента или элементов и нажмите левую кнопку мыши.

- Не отпуская кнопку, перемещайте указатель мыши так, чтобы выделенными оказались все необходимые элементы. После этого отпустите кнопку. Выделение создано.
- Щелкните мышью на пункте **Гиперссылка** (Hyperlink) контекстного меню. Появится диалог **Добавление гиперссылки** (Add hyperlink), представленный на Рис. 3.36.

С помощью диалога **Добавление гиперссылки** (Add hyperlink) можно создавать несколько видов гиперссылок.

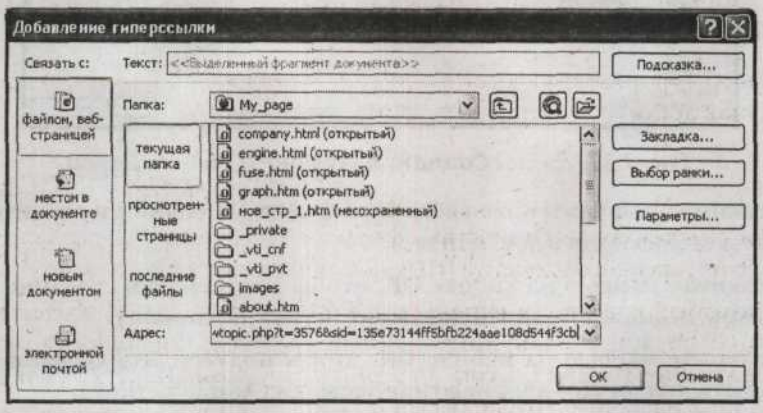


Рис. 3.36. Диалог ***Добавление гиперссылки*** *(Add hyperlink)*

Чтобы создать гиперссылку на уже существующую Web-страницу либо файл:

- Щелкните мышью на иконке **файлом, веб-страницей** (file, web-page) группы элементов управления **Связать с** (Link with).
- Щелкните мышью в поле ввода **Адрес** (Address) и введите в него URL страницы или файла, на который ссылаетесь.
- Либо перейдите к папке, в которой находится искомый файл, и щелкните мышью на его названии.
- Щелкните мышью на кнопке **OK**, чтобы подтвердить создание ссылки и закрыть диалог.

Чтобы создать гиперссылку на HTML-документ, которого еще не существует, попутно создав этот документ:

- Щелкните мышью на иконке **новым документом** (new document) группы элементов управления **Связать с** (Link with).
- Щелкните мышью в поле ввода **Имя нового документа** (New document name) и введите в него URL страницы или файла, который будет создан.

Либо

- Щелкните мышью на кнопке **Изменить** (Change), появится диалог **Создать документ** (Create document), Рис. 3.37.
- Перейдите к папке, в которой вы хотите создать новый документ.

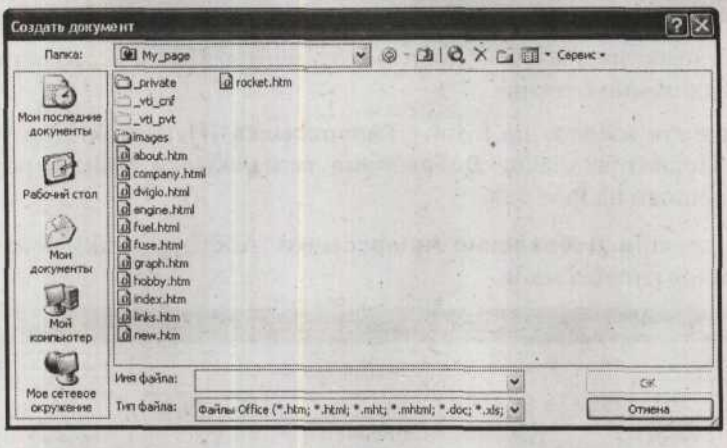


Рис. 3.37. Диалог ***Создать документ*** *(Create document)*

- Щелкните мышью в поле ввода **Имя файла** (Filename) и введите название файла создаваемого документа.
- Щелкните мышью на кнопке **OK**, чтобы подтвердить создание документа и закрыть диалог **Создать документ** (Create document).
- Щелкните мышью на кнопке **OK**, чтобы подтвердить создание ссылки и закрыть диалог **Добавление гиперссылки** (Add hyperlink).

Чтобы создать ссылку на адрес электронной почты:

- Щелкните мышью на иконке **электронной почтой** (e-mail), группы элементов управления **Связать с** (Link with).
- Щелкните мышью в поле ввода **Адрес эл.почты** (e-mail address) и введите в него почтовый адрес.
- Щелкните мышью на кнопке **OK**, чтобы подтвердить создание ссылки и закрыть диалог.

Для упрощения навигации по объемистым HTML-страницам можно использовать механизм внутренних ссылок. Суть его заключается в том, что в теле HTML-страницы размещаются специальные «якоря», на которые затем можно делать гиперссылки.

Создание внутритекстовых гиперссылок состоит из двух этапов.

- Этап первый – создание якоря в теле страницы:
  - Щелкните мышью в месте Web-страницы, на которое хотите создать ссылку.
  - Выберите пункт меню **Вставка ♦ Закладка** (Insert ♦ Bookmark). Появится диалог **Закладка** (Bookmark), представленный на Рис. 3.38.
  - Щелкните мышью в поле ввода **Имя закладки** (Bookmark name) и введите название якоря.

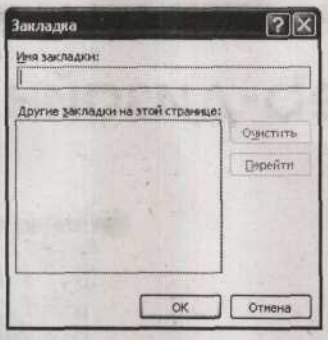


Рис. 3.38. Диалог ***Закладка*** *(Bookmark)*

- Щелкните мышью на кнопке **OK**, чтобы создать якорь и закрыть диалог.

- Этап второй – создание ссылки на якорь:
  - Выделите элемент HTML-страницы, к которому хотите добавить ссылку. Для этого переместите указатель мыши к одному из углов выделяемого элемента или элементов и нажмите левую кнопку мыши.
  - Не отпуская кнопку, перемещайте указатель мыши так, чтобы выделенными оказались все необходимые элементы. После этого отпустите кнопку. Выделение создано.
  - Щелкните мышью на пункте **Гиперссылка** (Hyperlink) контекстного меню. Появится диалог **Добавление гиперссылки** (Add hyperlink).
  - Щелкните мышью на иконке **местом в документе** (place in document) группы элементов управления **Связать с** (Link with).
  - В списке **Выберите место в документе** (Choose place in document) щелкните мышью на названии якоря, на который хотите сделать ссылку.
  - Щелкните мышью на кнопке **OK**, чтобы подтвердить создание ссылки и закрыть диалог.

Заключение

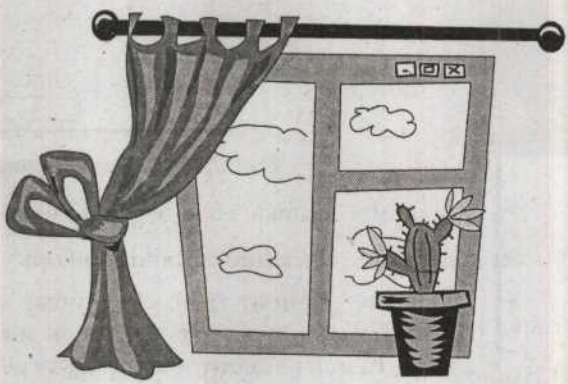
Ваш арсенал разработчика Web-страниц пополнился массой методов, которые позволят вам создавать красивые и удобные Web-сайты, на которые будет удобно заходить и которыми будет приятно пользоваться.

Теперь вы умеете добавлять на ваши Web-страницы графические изображения, видео, звук. Можете разнообразить тексты ваших страниц различными видами форматирования. Познакомились с основами табличной верстки Web-страниц и научились создавать карты ссылок. Узнали, что такое URL и как с ними работать.

Полученных вами знаний уже достаточно для создания статических Web-страниц высокой степени сложности. В следующей главе речь пойдет о подготовке графических файлов для размещения на Web-страницах, что позволит вам еще больше расширить ваши творческие горизонты и даст вам возможность создавать еще более яркие, запоминающиеся, уникальные Web-страницы.

глава 4

Web-дизайн и графика



Позади три главы книги, вы уже владеете основными приемами создания статических Web-сайтов, знаете, что такое табличная верстка и «резиновый дизайн» сайтов, умеете размещать на Web-страницах тексты, графику и прочие элементы оформления.

В этой главе основное внимание мы уделим неотъемлемой части практически любой Web-страницы – графике. Вы узнаете новые подробности о графических форматах, используемых в Web, узнаете, где лучше брать изображения и как их подготавливать к размещению на ваших страницах. И напоследок вы познакомитесь с некоторыми особенностями оптимизации Web-страниц, которые позволят вашим страницам выглядеть одинаково хорошо на компьютерах всех посетителей вашего сайта и быстро загружаться через каналы сети Интернет.

Форматы графических файлов для Web

Ранее в этой книге уже заходила речь о компьютерной графике. Теперь пришло время поговорить о ней подробнее. Компьютерная графика бывает двух видов: растровая и векторная.

Растровая графика

Растровая графика уже упоминалась в предыдущей главе. Растровое изображение состоит из совокупности маленьких точек разного цвета. Отображаемые вместе, они создают на экране монитора иллюзию изображения. Этот тип графики наиболее прост и понятен компьютеру, поскольку все изображения на его экране

формируются подобным образом. Если вы присмотритесь к экрану монитора внимательнее, то, безусловно, сможете разглядеть сетку отдельных точек, из которых он состоит. Точки на экране делятся на группы, по три точки разного цвета в каждой. Такая группа из трех точек – красного, зеленого и синего цвета, называется триадой. Одна триада отвечает за одну точку изображения на экране, называемую также пикселом. Количество пикселов, умещающееся на экране по горизонтали и по вертикали, называют разрешением экрана. Например, когда говорится, что разрешение экрана 1024x768, это значит, что в текущем режиме на экране размещается 1024 пиксела по горизонтали и 768 по вертикали.

Зажигая точки триады с разной интенсивностью, можно из трех ее цветов получать любые цвета и оттенки. Такую цветовую систему, построенную на трех основных цветах: красном, зеленом и синем, называют RGB (от Red, Green, Blue – красный, зеленый, синий). Интенсивность каждого из основных цветов может принимать 256 различных значений. От «0» – цвета нет, до «255» – максимальная интенсивность. Общее количество цветов, отображаемое в такой системе, достигает 16,7 миллиона.

Количество цветов в изображении называется глубиной цвета и измеряется в битах. Если изображение состоит из двух цветов – черного и белого, то глубина цвета будет 1 бит, каждая его точка описывается 1 битом информации – есть цвет/нет цвета, черное/белое (Рис. 4.1).

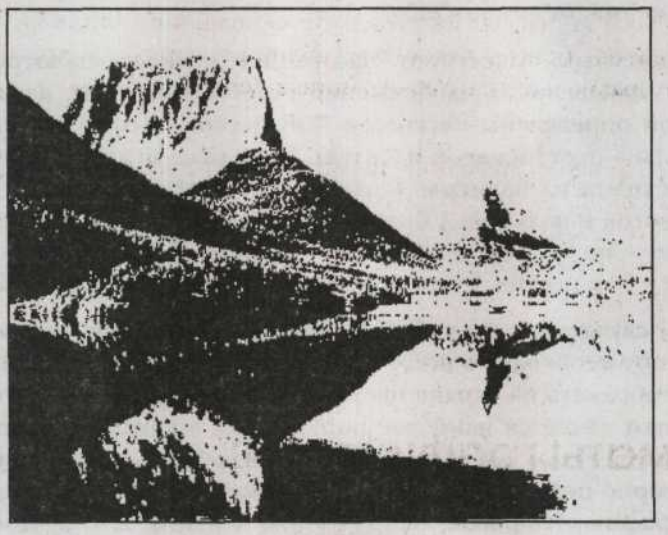


Рис. 4.1. Черно-белое изображение

Если изображение состоит из оттенков серого цвета, как черно-белая фотография, то его глубина цвета будет равняться 8 битам, или одному байту. Оттенки серого цвета получаются смешением в равных пропорциях всех основных цветов, поэтому в таком изображении, всего 256 различных оттенков (Рис. 4.2). Такое количество оттенков как раз можно записать одним байтом информации.

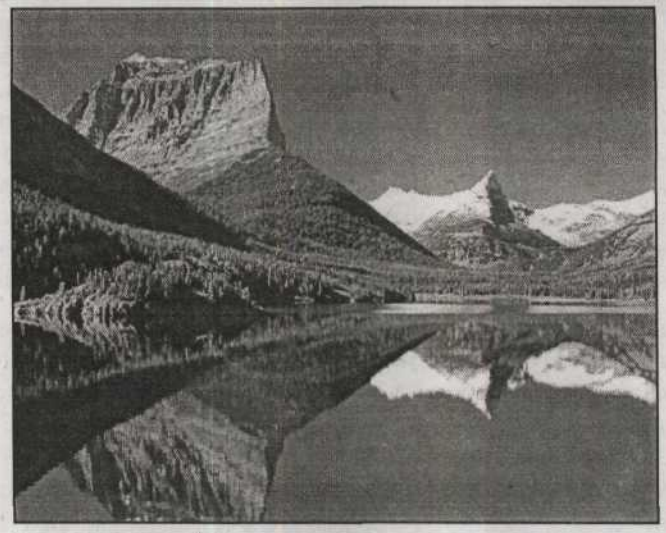


Рис. 4.2. Изображение из оттенков серого цвета

Полноцветные изображения, содержащие до 16,7 миллиона цветов, являются 24-х битными, по 8 битов на основной цвет. Существуют также и 32-х битные полноцветные изображения, в них еще один байт информации задает прозрачность точки изображения, от «0» до «255».

Кроме полноцветных, существуют еще так называемые «палитровые» цветные изображения. Файлы таких изображений содержат таблицу, называемую палитрой, в которой определены несколько RGB цветов, и само изображение может содержать только цвета из этой палитры. Цвет каждой точки изображения задается номером цвета из палитры. Глубина цвета таких изображений зависит от количества цветов в палитре. 1 битные изображения содержат не более 2-х цветов, 2-х битные – не более 4, 3-х битные – не более 8, 4-х битные – 16, 5-битные – 32-х, 6-битные – 64-х цветов, 7-битные – 128 и, наконец, 8-битные – 256 цветов.

В простейшем случае, для того чтобы отобразить на экране растровое изображение, компьютеру необходимо всего лишь прочитать графический файл и последовательно отобразить на экране цвета точек изображения. Поэтому растровый формат графики является наиболее популярным в компьютерных технологиях. Вторым преимуществом растровых форматов является то, что с их помощью можно достоверно передать любое изображение, практически без ограничений, любую фотографию, набросок, репродукцию картины и т. д. Достаточно лишь оцифровать исходное изображение, чтобы получить его электронную копию.

Но есть у растровых изображений и существенные недостатки. Первый – сложность масштабирования. Допустим, у вас есть картинка некоторого размера (Рис. 4.3). Если вы захотите увеличить ее, чтобы разглядеть отдельные детали, то вместо этого вы увидите расползшуюся груду увеличенных точек (Рис. 4.4).

Рис. 4.3. Исходное изображение

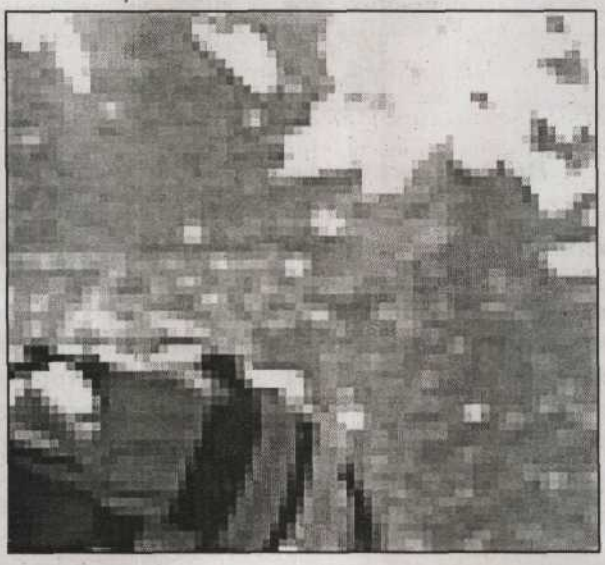


Рис. 4.4. Увеличенный фрагмент изображения

Второй недостаток – объем, который занимает изображение. Полноцветная картинка, размером 1024x768 точек будет занимать более двух мегабайт. Это несущественно, в том случае, если вы держите графику на своем компьютере, но если вы хотите сделать ее доступной через Интернет, то размер становится критически важным. Чтобы уменьшить объем, занимаемый картинкой, можно уменьшить ее размеры, но это также уменьшит и ее детальность. В некоторых случаях можно уменьшать количество цветов в картинке, например от 24-х битного цвета перей-

ти к 8-битному, – это позволит сократить объем картинки примерно в 3 раза. Но этот способ подходит отнюдь не всегда. В графических форматах, применяемых для размещения картинок в WWW, используются различные способы уменьшения объема файлов, основанные на разных принципах, что позволяет значительно их уменьшать, но только до определенной степени.

Векторная графика

Многих недостатков растровых изображений лишена так называемая векторная графика (Рис. 4.5). Файлы векторной графики, в отличие от растровой, содержат не готовую картинку, а только инструкции, как эту картинку нарисовать. Например, окружность в векторном формате будет записана диаметром, координатами центра, цветами самого круга и его границы. Также и другие геометрические фигуры. Более сложные изображения составляются из простейших геометрических фигур, собранных из линий в различных комбинациях.

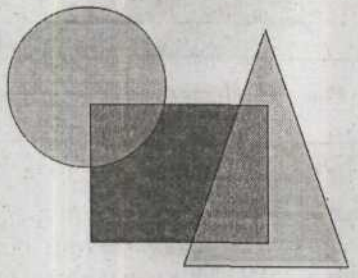


Рис. 4.5. Простейшее векторное изображение

Преимущества такого подхода очевидны: любое векторное изображение можно растягивать и сжимать во много раз без потери качества самого изображения и объем картинки не зависит от ее размеров. Еще одним плюсом векторной графики является легкость манипуляции уже созданными изображениями. Вы всегда можете изменить прочерченную линию, подправить нарисованную геометрическую фигуру, поменять цвета и т. д. Все это делает векторную графику очень удобной для рисования на компьютере. Одним из примеров векторной графики, с которым вы сталкиваетесь каждый раз, работая за компьютером, являются текстовые шрифты. Каждый символ большинства из компьютерных шрифтов представляет собой отдельный векторный рисунок, что позволяет изменять его размеры в широких пределах, менять его цвет и т. д.

Но у векторной графики есть свои специфические недостатки. Для отображения в векторном формате подходят далеко не все изображения. Лучше всего для отображения в векторном виде подходят изображения, которые можно разбить на геометрические фигуры и линии: рисунки мультипликационного типа, схемы и планы и т. д. Категорически не подходят для преобразования в векторный вид изображения фотографического типа, с множеством непредсказуемых и мелких деталей (Рис. 4.6). Кроме того, отображение векторных картинок создает существенную нагрузку на компьютер, которому нужно преобразовать инструкции из файла в картинку на экране.

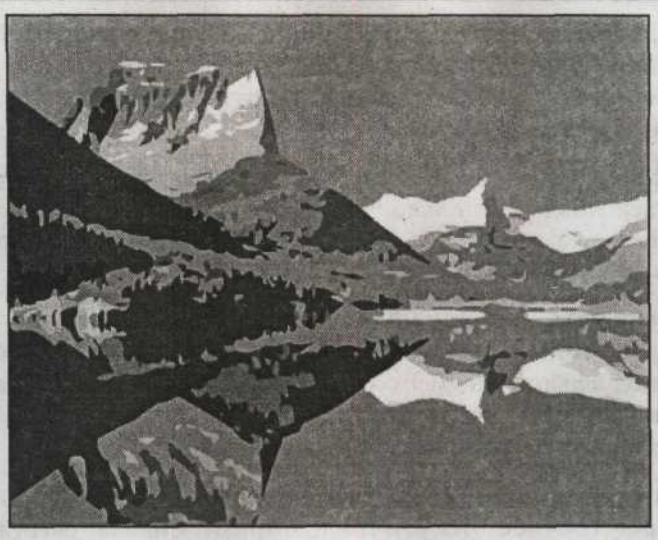


Рис. 4.6. Фотографическое изображение, преобразованное в векторный формат

Форматы графики, применяемые в Web

В связи с большей универсальностью и меньшей требовательностью к ресурсам компьютера, технологии растровых изображений получили повсеместное распространение, в том числе и в WWW. И, хотя сейчас вопрос производительности компьютеров не является столь острым и векторные технологии постепенно приходят в Web, подавляющая часть графики на Web-страницах выполняется в растровых форматах.

Для размещения картинок на Web-страница в основном используют два растровых графических формата: **GIF** и **JPEG**. Хотя оба эти формата созданы для того, чтобы служить средством хранения и переноса графики, каждый из них обладает рядом особенностей, знание которых даст вам возможность сделать правильный выбор формата для вашей графики и позволит с минимальными потерями реализовать ваш творческий замысел.

Формат GIF

Формат **GIF** (Graphic Interchange Format – формат обмена графикой) изначально создавался корпорацией CompuServe для передачи графики в своей компьютерной сети, но стал пользоваться большой популярностью и за ее пределами. Основной особенностью формата **GIF** является то, что он поддерживает только палитровые изображения не более чем с 256 цветами, поэтому его нецелесообразно использовать для размещения фотографических и других полноцветных изображений.

Чтобы изображения было проще передавать по каналам связи, графическая информация в **GIF** файлах хранится в сжатом виде. Механизмы сжатия, заложенные в этот формат, лучше всего работают с изображениями, имеющими достаточно

большие однотонные области с ровными границами. Поэтому **GIF** используют обычно для размещения рисованных элементов оформления сайтов, рисунков в мультипликационном стиле, чертежей, схем, а также других изображений с ограниченным количеством цветов.

В один файл формата **GIF** можно записать несколько изображений, которые будут показываться последовательно, через заданные промежутки времени. Эта возможность формата позволяет создавать с его помощью анимации и небольшие мультфильмы без звукового сопровождения.

Полезной особенностью формата **GIF** является возможность сделать один из цветов картинки прозрачным, при этом через участки прозрачного цвета будет видно то, что находится под изображением. То есть, в случае Web-страницы – это будет фоновое изображение или цвет. Эта возможность широко используется в тех случаях, когда фон является одним из важных элементов оформления страницы.

Для ускорения отображения **GIF**-картинок на Web-страницах в формате **GIF** предусмотрено чересстрочное отображения графики. Сначала загружаются четные строки изображения, а затем нечетные, или наоборот.

Формат JPEG

В отличие от формата **GIF**, который создавался одной фирмой, стандарт **JPEG** (Joint Photographic Experts Group – Объединенная группа экспертов по фотографии) был разработан под эгидой ISO (International Organization for Standardization – Международная организация по стандартизации) – ведущей международной организации по стандартам и CCITT (Consultative Committee for International Telephone and Telegraphy – Консультативный комитет по международной телефонии и телеграфу) организации, которая устанавливает стандарты для телефонии, радио, телевидения и т. п. Стандарт **JPEG** разрабатывался специально для хранения и передачи фотографических изображений и лучше всего он подходит именно для этих целей.

Изображения в формате **JPEG** могут содержать до 16,7 миллиона цветов, этого более чем достаточно для передачи деталей любого фотографического изображения. При преобразовании в формат **JPEG** изображения подвергаются сжатию с потерей данных. Изображение анализируется программой преобразования с точки зрения особенностей человеческого восприятия и отбрасывает детали, человеку незаметные. Это позволяет уменьшать размер фотографических изображений в десятки раз без видимых искажений картинки. Размер изображения и его качество определяются степенью сжатия, выбранной в настройках программы, создающей **JPEG** файлы.

Особенности алгоритма сжатия **JPEG** не позволяют добиться столь же впечатляющих результатов при работе с рисунками, имеющими не очень много цветов, с ровными линиями и большими участками одного цвета. На таких изображениях **JPEG** может серьезно «портить» итоговую картинку уже при сравнительно небольших степенях сжатия. В таких случаях разумнее пользоваться форматом **GIF**.

Для ускорения визуальной загрузки картинок на ваши Web-страницы, можно использовать режим прогрессивной загрузки **JPEG**-изображений. В этом режиме, в файле **JPEG**, кроме основного изображения хранятся еще несколько его копий с уменьшенным разрешением. Сначала загружается копия с самым маленьким разрешением, потом та, что чуть больше и так далее. И, хотя общий размер файла при этом немного увеличивается и время загрузки в целом возрастает, создается впечатление, что картинка загрузилась почти сразу и только потом постепенно подгрузила свои недостающие детали.

Подведем итог, в каком же формате размещать графику на Web-страницах? Если изображение содержит немного цветов, состоит из четких линий и геометрических фигур, содержит достаточно большие области одного цвета, то лучшим выбором будет формат **GIF**. Также этот формат незаменим для создания небольших анимированных вставок, рекламных баннеров и в тех случаях, когда необходима частичная прозрачность картинки.

Для размещения многоцветных изображений фотографического типа лучше всего подходит формат **JPEG**, в этой области ему нет равных.

Если вы затрудняетесь, к какому типу отнести изображение, то создайте две его версии – в форматах **GIF** и **JPEG**, и сравните их. Ту, что будет при сравнимом качестве занимать меньше месте, и используйте.

Источники изображений для Web-страниц

Перед человеком, решившим использовать графику на своих Web-страницах, непременно встанет вопрос: где же брать картинки для оформления страниц?

Существует несколько вариантов ответа. Первый и самый очевидный – создать самому. Необходимые изображения можно нарисовать, отсканировать, сфотографировать и т. д. Существует и другой способ получения картинок – использование коллекций уже готовых изображений, клипартов.

Коллекции изображений, клипарты

Клипарты существуют в виде компакт-дисков со сборниками картинок и Интернет-коллекций.

Все клипарты делятся на коммерческие и бесплатные. Целью создания коммерческих коллекций графики, является зарабатывание денег, поэтому в открытом доступе, как правило, доступны лишь уменьшенные версии изображений из таких клипартов. Чтобы получить в свое распоряжение полную версию картинки, в большом разрешении, необходимо уплатить определенную сумму. Обычно изображения для таких клипартов создают профессиональные художники и фотографы, поэтому картинки в них обычно очень качественные, но могут стоить достаточно больших денег. Кроме того, зачастую оплачивать их можно только с

помощью кредитных карт, выпущенных западными банками. При создании крупных коммерческих сайтов использование таких коллекций может быть вполне оправданным.

Бесплатные коллекций изображений выставляются на публичное обозрение совершенно свободно, но это не значит, что картинки из них можно использовать в абсолютно любых целях. Обычно, на сайтах авторов таких коллекций приводятся цели, в которых можно использовать выставляемые изображения. Как правило, против использования графики в некоммерческих целях никто не возражает, но если вы хотите использовать ее для получения выгоды, то необходимо договориться с автором. Это не только поможет вам избежать юридических проблем в будущем, но и простимулирует автора на создание новых изображений.

Особенности работы с изображениями из клипартов

Используя картинки из готовых коллекций, старайтесь не размещать их на своем Web-сайте в неизменном виде, модифицируйте их так, чтобы они максимально отвечали вашим дизайнерским замыслам. Это не только увеличит ценность оформления вашей страницы, но и сделает изображения уникальными, не такими, какими они были в клипарте. Не забывайте, что кроме вас этой коллекцией пользуются сотни и тысячи людей, поэтому вероятность, что кто-то воспользуется такой же картинкой, как и у вас, достаточно велика. Ваша Web-страница должна быть уникальной, не похожей на другие. Из этих же соображений, старайтесь не использовать изображения, находящиеся в самом начале клипарта, лучше пролистайте его на несколько страниц вперед. По статистике, наибольшей популярностью пользуются изображения, находящиеся на первой-третьей страницах коллекций.

В комплекте с многими графическими и офисными программами часто идут коллекции изображений. Проще всего использовать именно их, но прежде чем делать это, трижды подумайте, каким тиражом расходятся эти программы и сколько еще людей пользуются этими клипартами.

Постарайтесь найти несколько изображений, подходящих для ваших целей, желательно не менее трех. Расположите их в одном каталоге и просмотрите последовательно. Подумайте, какое из них больше отвечает вашим целям. Не поленитесь и проверьте, как будет смотреться каждая из картинок на вашей Web-странице, создайте несколько версий Web-страницы. Сравните их сразу же и по прошествии некоторого времени.

Поиск изображений в сети Интернет

Как лучше всего искать рисунки в Интернете? Большинство поисковых систем предусматривает поиск изображений с заданными названиями. Воспользуемся услугами двух наиболее популярных поисковых систем: Google и Яndex.

Поиск изображений с помощью поисковой системы Яndex

Чтобы найти изображения с помощью системы Яndex:

- Находясь в сети Интернет, введите в адресной строке браузера адрес поисковой системы: **http://www.yandex.ru** и нажмите клавишу Enter. Откроется Web-страница системы Яndex (Рис. 4.7).

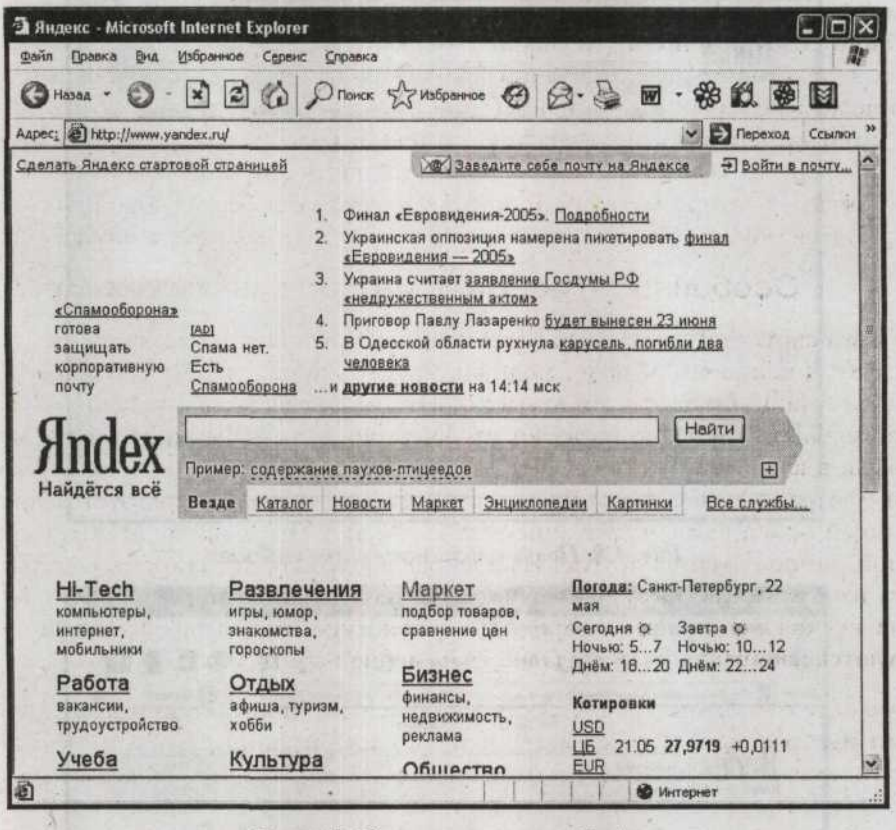


Рис. 4.7. Поисковая система Яndex

- Щелкните мышью на ссылке **Картинки** под полем ввода. Откроется страница поиска изображений (Рис. 4.8).
- Щелкните мышью в поле ввода и введите слова, которые должны присутствовать в названии картинки. Нажмите клавишу Enter. Будет выполнен поисковый запрос.
- В результатах поиска вы увидите уменьшенные копии найденных изображений и их названия. Под названиями файлов будут приведены адреса сайтов, на которых находятся найденные изображения (Рис. 4.9).

Обязательно посещайте сайты, с которых вы берете изображения, чтобы урегулировать правовые вопросы. Кроме того, вы можете обнаружить на этих сайтах немало других хороших изображений.

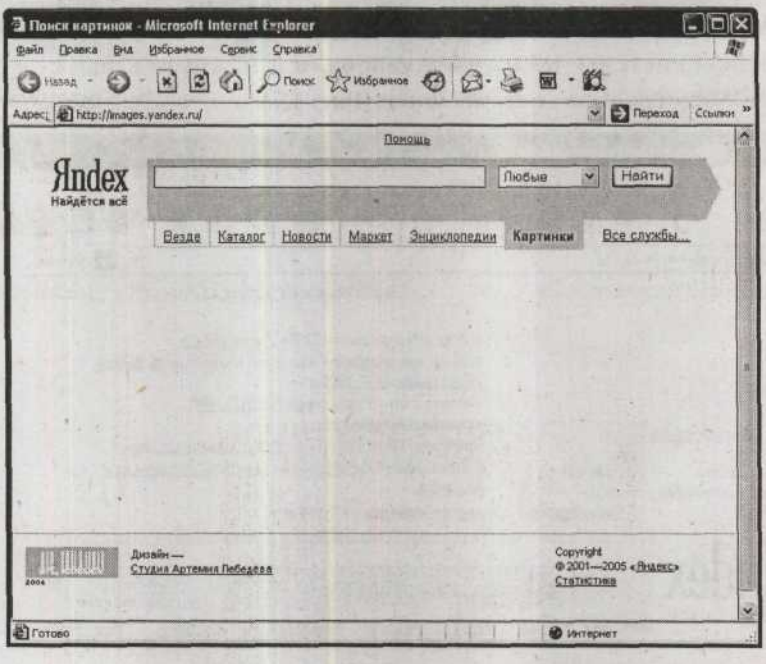


Рис. 4.8. Поиск картинок в системе Яndex

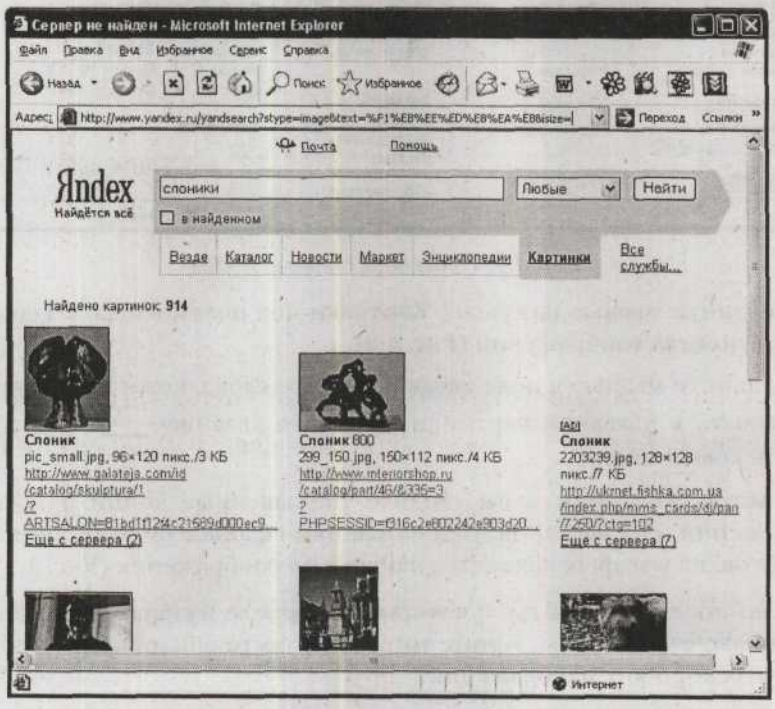


Рис. 4.9. Результаты поиска картинок в системе Яndex

- Щелкните мышью на понравившемся изображении, чтобы открыть его в полный размер в новом окне (Рис. 4.10).

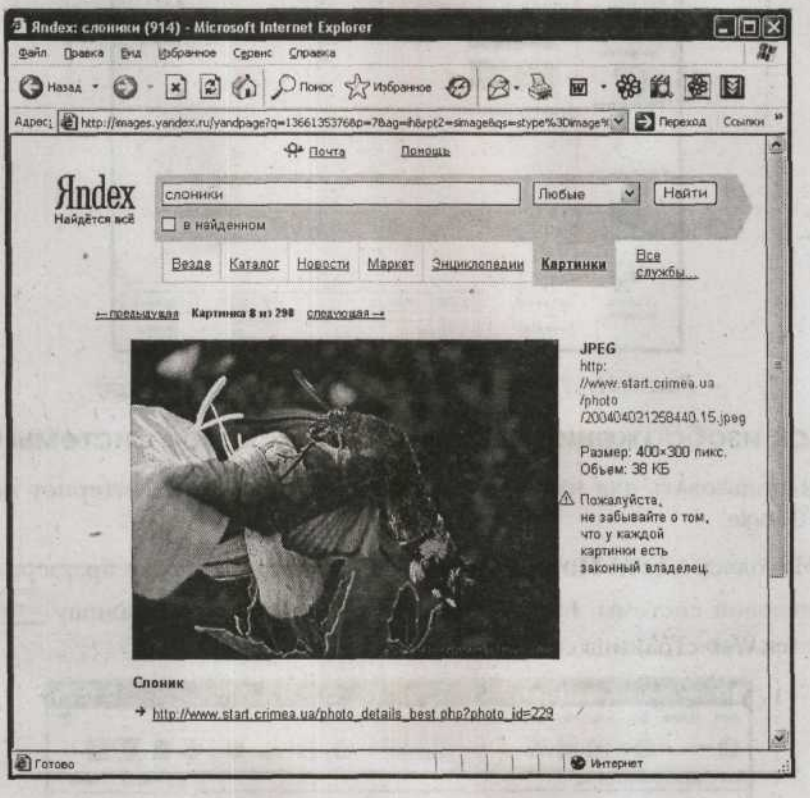


Рис. 4.10. Полная версия изображения

- Щелкните правой кнопкой мыши на полном изображении, появится контекстное меню.
- Щелкните мышью на пункте **Сохранить рисунок как** (Save image as) контекстного меню, появится диалог **Сохранение рисунка** (Save image), Рис. 4.11.
- Перейдите к каталогу, в который вы хотите сохранить изображение.
- Щелкните мышью в поле ввода **Имя файла** (File name) и введите желаемое название файла.
- Щелкните мышью на кнопке **Сохранить** (Save), чтобы сохранить изображение на ваш жесткий диск и закрыть диалог.
- Закройте окно браузера с полной копией изображения, щелкнув на кнопке закрытия окна.

Пролистывайте дальше результаты поиска и сохраняйте понравившиеся вам изображения, как описано выше.

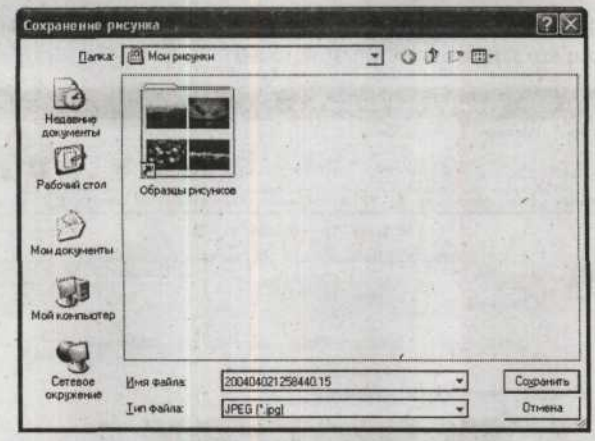


Рис. 4.11. Диалог ***Сохранение рисунка*** *(Save image)*

Поиск изображений с помощью поисковой системы Google

Чтобы использовать для нахождения изображений в сети Интернет поисковую систему Google:

- Находясь в сети Интернет, введите в адресной строке браузера адрес поисковой системы: **http://www.google.ru** и нажмите клавишу Enter. Откроется Web-страница системы Google (Рис. 4.12).

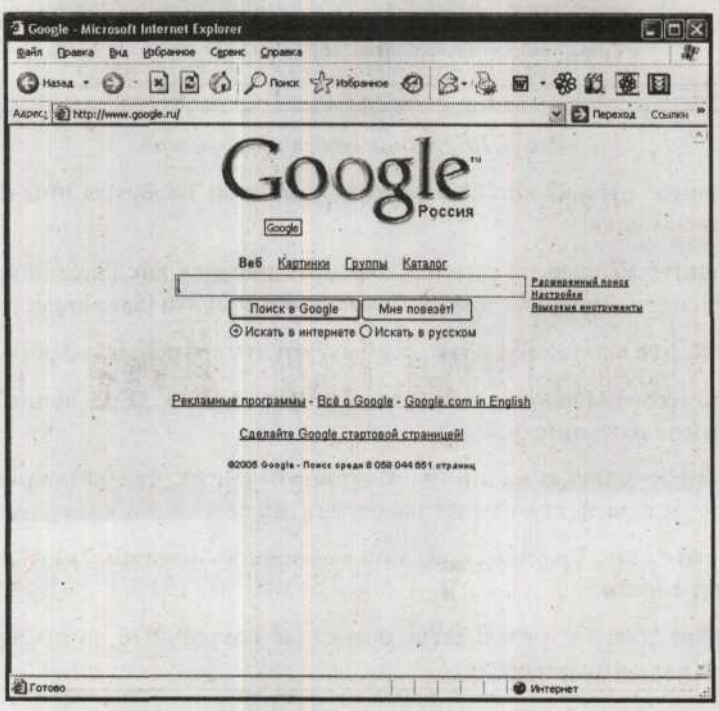


Рис. 4.12. Поисковая система Google

➢ Щелкните мышью на ссылке **Картинки**, под полем ввода. Откроется страница поиска изображений (Рис. 4.13).

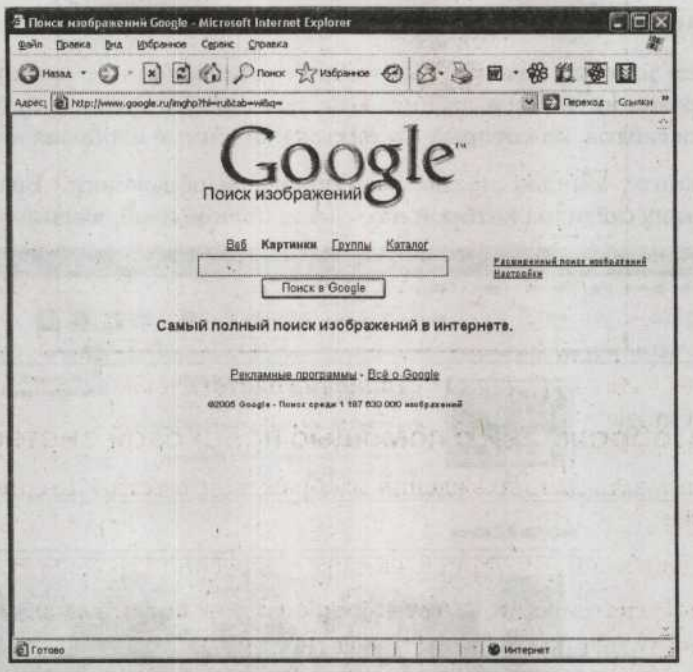


Рис. 4.13. Поиск изображений в системе Google

Рис. 4.14. Отображение результатов поиска изображений в системе Google

- Щелкните мышью в поле ввода и введите слова, которые должны присутствовать в названии картинки либо в ее описании. Нажмите клавишу Enter. Будет выполнен поисковый запрос.
- В результатах поиска вы увидите уменьшенные копии найденных изображений и их названия. Под названиями файлов будут приведены адреса сайтов, на которых находятся найденные изображения (Рис. 4.14).
- Щелкните мышью на понравившемся изображении. Браузер откроет страницу сайта, на которой находится полное изображение (Рис. 4.15).

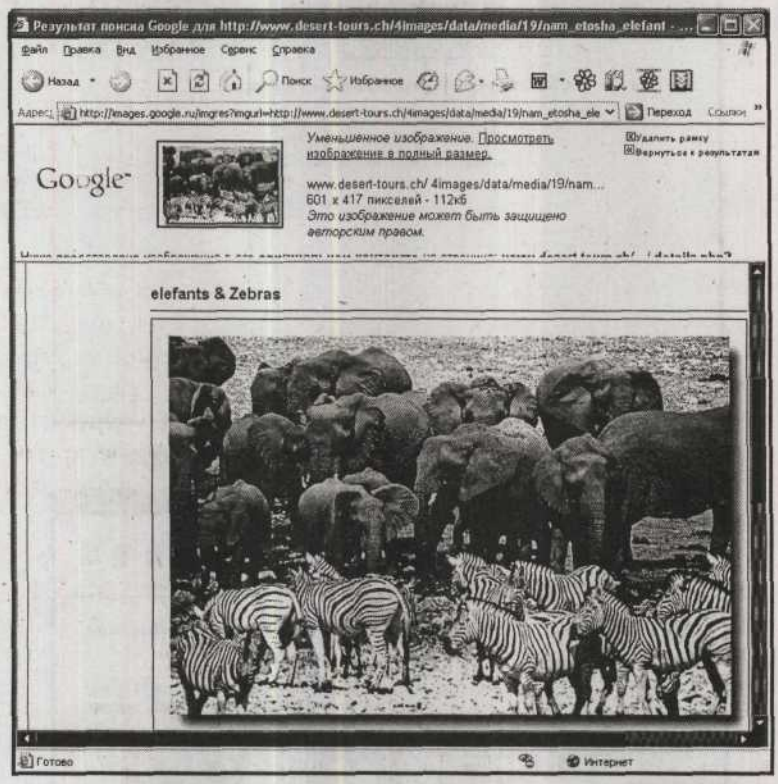


Рис. 4.15. Отображение полной версии изображения в системе Google

- Щелкните правой кнопкой мыши на полном изображении, появится контекстное меню.
- Щелкните мышью на пункте **Сохранить рисунок как** (Save image as) контекстного меню, появится диалог **Сохранение рисунка** (Save image), Рис. 4.11.
- Перейдите к каталогу, в который вы хотите сохранить изображение.
- Щелкните мышью в поле ввода **Имя файла** (File name) и введите желаемое название файла.

- Щелкните мышью на кнопке **Сохранить** (Save), чтобы сохранить изображение на ваш жесткий диск и закрыть диалог.
- Щелкните мышью на кнопке браузера **Назад** (Back), чтобы вернуться к странице с уменьшенными копиями изображений.
- Пролистывайте дальше результаты поиска и сохраняйте понравившиеся вам картинки, как было описано выше.

Обработка изображений в программе Adobe Photoshop

В процессе создания Web-страниц вам придется иметь дело с графикой из совершенно разных источников: картинками из клипартов, отсканированными изображениями, цифровыми фотографиями и т. д. О некоторых приемах подготовки этих изображений к размещению на ваших страницах будет рассказано в этом разделе.

Существует множество программ для обработки цифровых изображений; некоторыми функциями по работе с изображениями обладает и программа FrontPage, хотя и в очень небольшом количестве. Но признанным лидером по обработке растровых изображений является программа Adobe Photoshop. Она предоставляет все мыслимые и немыслимые возможности монтажа, цветовой коррекции и ретуши. Позволяет создавать различные спецэффекты и делать многое другое. Поэтому основное внимание мы уделим обработке графики именно в этой программе.

Подробную информацию по установке, настройке и использованию всего спектра функций программы Adobe Photoshop вы можете почерпнуть из книг, посвященных работе с этой замечательной программой. В этой же книге мы лишь коснемся некоторых аспектов применения Adobe Photoshop для подготовки изображений к размещению во Всемирной паутине.

Запустите программу Adobe Photoshop, выполнив команду стартового меню **Пуск ◆ Все программы ◆ Adobe Photoshop** (Start ◆ Programs ◆ Adobe Photoshop), откроется главное окно программы Adobe Photoshop (Рис. 4.16).

Большую часть основного окна программы Adobe Photoshop занимает рабочая область, в которой располагаются окна открытых изображений, плавающая панель инструментов и несколько дополнительных панелей со всевозможными настройками. Также, в главном окне, под строкой меню находится контекстная панель настроек, в которой доступны все основные настройки для активного рабочего инструмента.

Чтобы открыть какое-либо изображение в программе Adobe Photoshop:

Выберите команду меню **File ◆ Open** (Файл ◆ Открыть). Появится диалог **Open** (Открыть), Рис. 4.17.

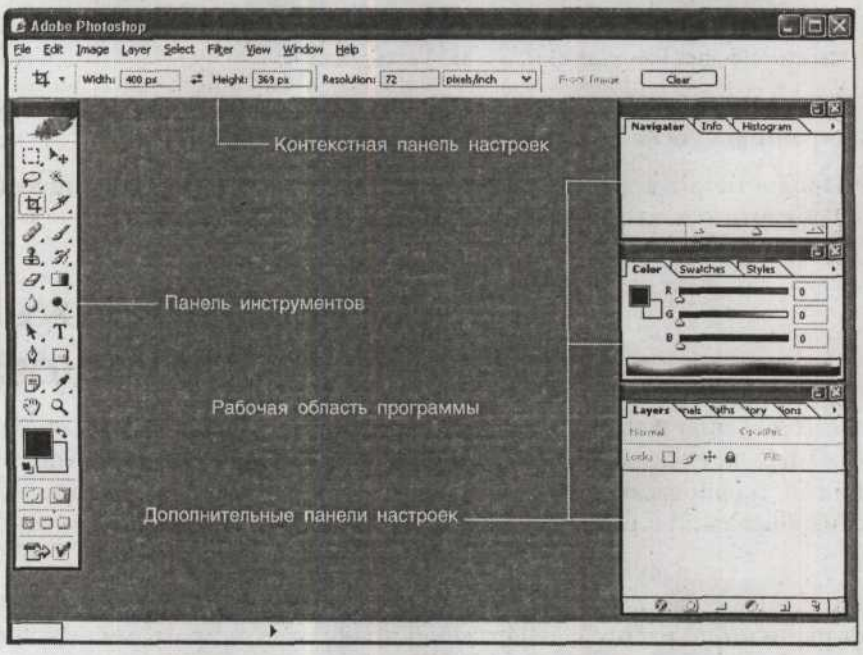


Рис. 4.16. Главное окно программы Adobe Photoshop

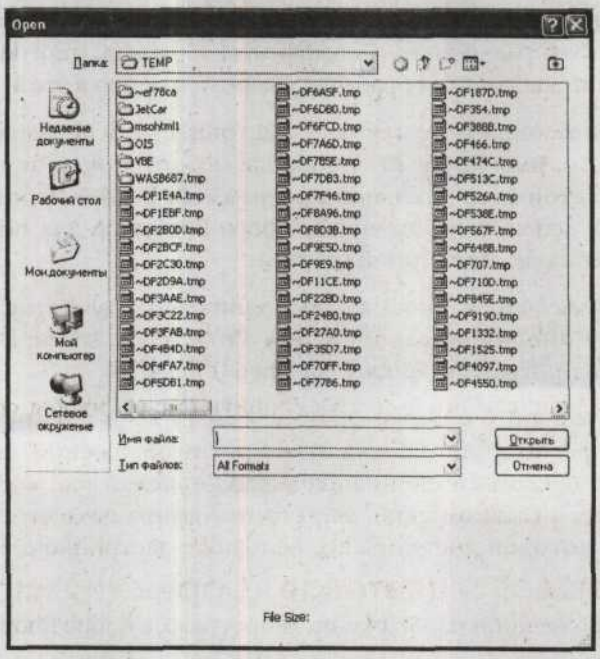


Рис. 4.17. Диалог ***Open*** *(Открыть)*

➢ Перейдите в этом диалоге к папке, в которой содержится нужный графический файл, и щелкните мышью на названии этого файла.

- Щелкните мышью на кнопке **Открыть** (Open), чтобы открыть файл и закрыть диалог.

После того как изображение изменено, его необходимо сохранить на жесткий диск. Если изображение раньше не сохранялось, то чтобы сохранить изображение в программе Adobe Photoshop:

- Выберите команду меню **File ◆ Save** (Файл ◆ Сохранить). Появится диалог **Save as** (Сохранить как), Рис. 4.18.

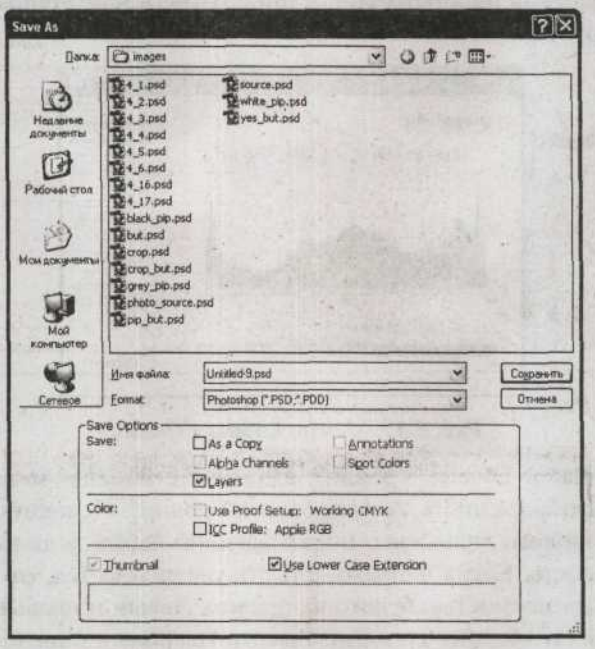


Рис. 4.18. Диалог ***Save as*** *(Сохранить как)*

- Перейдите к папке, в которой вы хотите сохранить файл.
- Щелкните мышью в поле ввода **Имя файла** (Filename) и введите название, под которым вы хотите сохранить файл изображения.
- Щелкните мышью на кнопке **Сохранить** (Save), чтобы сохранить файл и закрыть диалог.

Чтобы повторно сохранить изображение в файле, достаточно выбрать команду меню **File ◆ Save** (Файл ◆ Сохранить), изображение будет сохранено.

Тоновая и цветовая коррекция изображений

Как правило, прежде чем размещать изображения на Web-странице, необходимо провести их тоновую и цветовую коррекцию: осветлить слишком темные участки изображения, затемнить слишком светлые, немного изменить цвета, чтобы изображение было более естественным либо больше соответствовало художественному замыслу. Если на Web-странице планируется использовать несколько изо-

бражений, взятых из разных источников, то желательно «подогнать» их цвета друг к другу, чтобы между ними не было дисгармонии.

В программе Adobe Photoshop есть много инструментов для работы с тоновым и цветовым балансом изображений, но мы ограничимся рассмотрением основных из них.

Работа с диалогом Levels (Уровни)

Основным средством для тоновой коррекции, т. е. изменения соотношения светлых и темных участков изображения, в программе Adobe Photoshop является диалог **Levels** (Уровни), Рис. 4.19.

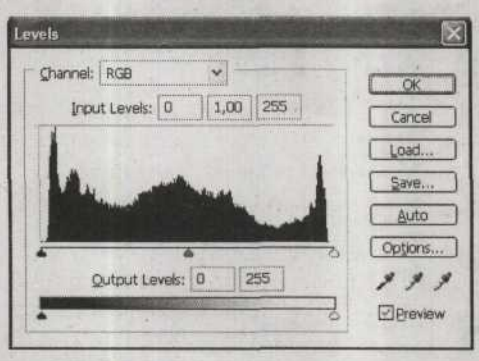


Рис. 4.19. Диалог ***Levels*** *(Уровни)*

Чтобы открыть диалог **Levels** (Уровни), выполните команду меню **Image ◆ Adjustments ◆ Levels** (Изображение ◆ Регулировка ◆ Уровни). Основную часть диалога занимает график тонового диапазона, показывающий, какая доля изображения имеет определенную яркость. Слева направо яркость увеличивается, от полностью черного на левом краю до полностью белого на правом. Левый и правый края графика называются «черной» и «белой» точками. Высота графика в определенной точков показывает, какая доля пикселов изображения имеет такую яркость.

Если на графике преобладает левая часть (Рис. 4.20), то в изображении, скорее всего, переизбыток темных тонов, если правая часть – то светлых (Рис. 4.21). В графике сбалансированного изображения будут и темные тона, и светлые (Рис. 4.19).

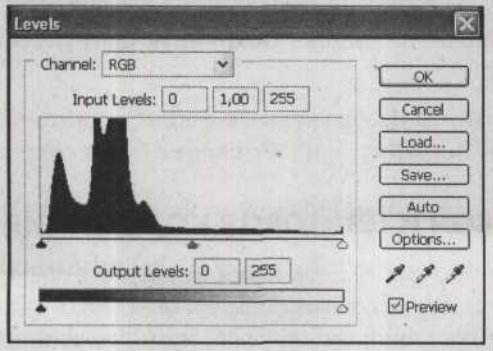


Рис. 4.20. График тонового диапазона для темного изображения

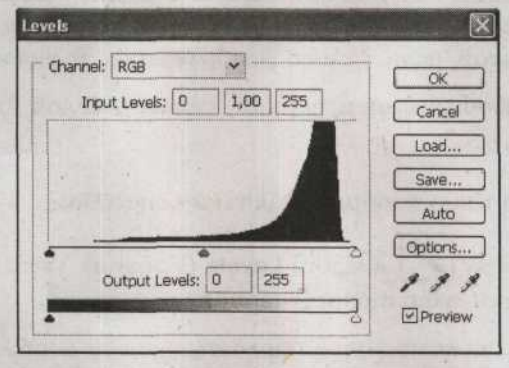


Рис. 4.21. График тонового диапазона для светлого изображения

Корректируя тоновый баланс, учитывайте характер самого изображения. Например, если на нем представлена ночная или вечерняя сцена, то, разумеется, должны преобладать темные тона. Сцена же снежного рассвета должна быть светлой. Но для большинства изображений оптимальной является сбалансированная яркость.

В диалоге **Levels** (Уровни) представлено несколько способов тоновой коррекции изображения. Если вы установите флажок **Preview** (Предварительный просмотр), то все ваши действия будет немедленно отображаться на самой обрабатываемой картинке.

Непосредственно под графиком тонового диапазона находятся три ползунковых регулятора – черный, серый и белый. Черный и белый регуляторы олицетворяют собой «черную» и «белую» точки изображения.

- Если вы сдвинете черный регулятор вправо по графику, то все точки изображения, находящиеся по яркости между регулятором и левым краем графика станут полностью черными. Яркость же остальных точек перераспределится в соответствии с новой черной точкой.
- Точно так же, перемещая влево белый регулятор, вы делаете все точки, находящиеся справа от него, белыми, а яркость остальных точек перераспределяется в соответствии с новой белой точкой.
- Серый ползунковый регулятор позволяет сдвигать баланс яркостей пикселов изображения. Место расположения этого регулятора на графике является точкой средней яркости. Сдвигая ее вправо, вы сместите баланс яркостей в темную сторону, а сдвигая влево – в светлую.

Над графиком тонового диапазона находятся три поля ввода, озаглавленные **Input Levels** (Входные уровни). В первое из них можно ввести положение черного ползункового регулятора, во второе – серого и в третье – белого. Положения черного и белого регулятора могут принимать значение яркости от «0» до «255», а положение серого регулятора задается отношением длины светлого участка на графике тонового диапазона к длине темного участка. Если значение в этом полем ввода больше единицы, то длина светлого участка больше, если же меньше, то наоборот.

Другим способом задания черной, белой и серой точек изображения, является использование для этой цели самого изображения. В диалоге **Levels** (Уровни) есть три кнопки с изображением черной, серой и белой пипеток:  ,  и  .

Чтобы задать черную точку изображения таким способом:

- Нажмите кнопку  диалога **Levels** (Уровни), указатель мыши приобретет вид пипетки, заполненной черной краской:  .
- Щелкните мышью на точке обрабатываемого изображения, уровень яркости которой хотите приравнять к черной точке. График в диалоге **Levels** (Уровни) изменится в соответствии с новой черной точкой.
- Если вас не устроил результат, повторите предыдущее действие.

Если вы хотите отменить выбор черной точки:

- Нажмите клавишу Alt. Кнопка **Cancel** (Отменить) диалога **Levels** (Уровни) сменится на кнопку **Reset** (Сбросить), как представлено на Рис. 4.22.

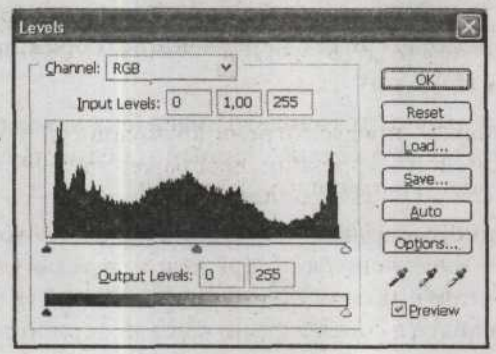


Рис. 4.22. Диалог ***Levels*** *(Уровни) при нажатой клавише* Alt

- Не отпуская клавишу Alt, щелкните мышью на кнопке **Reset** (Сбросить) диалога **Levels** (Уровни).

Чтобы задать белую точку изображения:

- Нажмите кнопку  диалога **Levels** (Уровни), указатель мыши приобретет вид пипетки, заполненной белой краской:  .
- Щелкните мышью на точке обрабатываемого изображения, уровень яркости которой хотите приравнять к белой точке. График в диалоге **Levels** (Уровни) изменится в соответствии с новой белой точкой.
- Если вас не устроил результат, повторите предыдущее действие.

Если вы хотите отменить выбор белой точки:

- Нажмите клавишу Alt. Кнопка **Cancel** (Отменить) диалога **Levels** (Уровни) сменится на кнопку **Reset** (Сбросить).
- Не отпуская клавишу Alt, щелкните мышью на кнопке **Reset** (Сбросить) диалога **Levels** (Уровни).

Чтобы задать серую точку изображения:

- Нажмите кнопку  диалога **Levels** (Уровни), указатель мыши приобретет вид пипетки, заполненной серой краской:  .
- Щелкните мышью на точке обрабатываемого изображения, уровень яркости которой хотите сделать средним уровнем яркости. График в диалоге **Levels** (Уровни) изменится в соответствии с новой серой точкой.
- Если вас не устроил результат, повторите предыдущее действие.

Если вы хотите отменить выбор серой точки:

- Нажмите клавишу Alt. Кнопка **Cancel** (Отменить) диалога **Levels** (Уровни) сменится на кнопку **Reset** (Сбросить).
- Не отпуская клавишу Alt, щелкните мышью на кнопке **Reset** (Сбросить) диалога **Levels** (Уровни).

В диалоге **Levels** (Уровни) предусмотрена возможность автоматически устанавливать параметры тоновой коррекции изображения. Все, что нужно для этого сделать – щелкнуть мышью на кнопке **Auto** (Автоматически) этого диалога. После этого автоматически определяются самая темная и самая светлая области изображения, они принимаются за черную и белую точку изображения, а яркости остальных точек изображения равномерно распределяются между ними.

Если вам не понравится полученный результат и вы захотите его отменить:

- Нажмите клавишу Alt. Кнопка **Cancel** (Отменить) диалога **Levels** (Уровни) сменится на кнопку **Reset** (Сбросить).
- Не отпуская клавишу Alt, щелкните мышью на кнопке **Reset** (Сбросить) диалога **Levels** (Уровни).

До сих пор мы узнавали, как с помощью диалога **Levels** (Уровни) расширить тоновый диапазон изображения – чтобы в нем присутствовали и светлые участки, вплоть до белого, и темные – до черного. Но встречается и обратная задача – «приглушить» изображение, убрать самые яркие участки, сделать самые светлые точки темнее, а темные – светлее. Для этого в диалоге **Levels** (Уровни) существует группа элементов управления **Output Levels** (Выходные уровни), состоящая из двух полей ввода и двух ползунковых регуляторов, черного и белого. Эти регуляторы располагаются под полосой уровней яркости: от минимальной – слева, до максимальной – справа.

Черный регулятор указывает на полосе уровней яркости минимальный уровень яркости для изображения. Передвигая этот регулятор вправо, вы делаете черную точку изображения уже не черной, а все более светлой. Точно так же, передвигая влево белый регулятор, вы делаете темнее белую точку изображения.

Минимальную и максимальную яркости изображения также можно задать, вводя непосредственные значения яркости в поля ввода **Output Levels** (Выходные уровни). В левое поле вводится минимальный уровень яркости изображения, а в правое – максимальный.

Настроив все параметры тоновой коррекции, примите изменения в изображении и закройте диалог **Levels** (Уровни), щелкнув мышью на кнопке **OK**.

Диалог Brightness/Contrast (Яркость/Контрастность)

Более простым средством тоновой коррекции в программе Adobe Photoshop служит диалог **Brightness/Contrast** (Яркость/Контрастность), Рис. 4.23. Чтобы открыть этот диалог выберите команду меню **Image** ♦ **Adjustments** ♦ **Brightness/Contrast** (Изображение ♦ Регулировка ♦ Яркость/Контрастность).

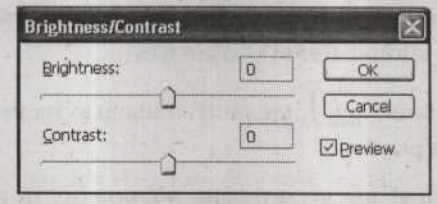


Рис. 4.23. Диалог ***Brightness/Contrast*** *(Яркость/Контрастность)*

Диалог содержит всего два ползунковых регулятора, продублированных полями ввода. Значения обоих полей могут изменяться в диапазоне от -100 до 100. Сдвигание ползункового регулятора вправо увеличивает значение в поле ввода, сдвигание регулятора влево – уменьшает.

- Если вы установите флажок **Preview** (Предварительный просмотр), то все изменения, произведенные в диалоге **Brightness/Contrast** (Яркость/Контрастность), будут немедленно отображаться на самом рисунке.
- С помощью первого регулятора **Brightness** (Яркость) можно установить общую яркость изображения. Сдвигая регулятор вправо, вы увеличиваете яркость всех точек изображения, сдвигая влево – уменьшаете.
- Ползунковый регулятор **Contrast** (Контрастность) устанавливает контрастность изображения. Контрастность – это различие в яркости между самой яркой и самой темной точками изображения. Сдвигая регулятор вправо, вы контрастность увеличиваете, сдвигая вправо – уменьшаете.

Выставив уровень яркости и контрастности изображения, щелкните мышью на кнопке **OK**, чтобы принять изменения и закрыть диалог.

Диалог Color Balance (Цветовой баланс)

Важную роль в подготовке изображений для использования на Web-страницах играет цветовая коррекция – изменение баланса цветов в изображении. Цветовая коррекция служит как для исправления дефектов изображения – например неправильных условий съемки фотографий, так и для реализации различных художественных замыслов – например, проведя небольшую цветовую коррекцию, можно превратить обычную земную пустыню в таинственный венерианский пейзаж.

Одним из средств цветовой коррекции программы Adobe Photoshop является диалог **Color Balance** (Цветовой баланс), Рис. 4.24. Чтобы открыть этот диалог, выполните команду меню **Image** ◆ **Adjustments** ◆ **Color Balance** (Изображение ◆ Регулировка ◆ Цветовой баланс).

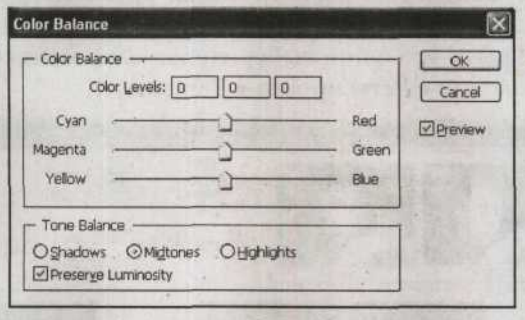


Рис. 4.24. Диалог ***Color Balance*** *(Цветовой баланс)*

Если вы установите флажок **Preview** (Предварительный просмотр), то все изменения, произведенные в диалоге **Color Balance** (Цветовой баланс), будут немедленно отображаться на самом рисунке.

Диалог **Color Balance** (Цветовой баланс) разделен на две группы элементов управления. В группе **Color Balance** (Цветовой баланс) собраны элементы, управляющие непосредственно балансом цветов, группа **Tone Balance** (Тоновый баланс) отвечает за то, к какому диапазону яркостей изображения применяется цветовая коррекция.

- В группе **Tone Balance** (Тоновый баланс) располагается переключатель с тремя положениями: **Shadows** (Тени), **Midtones** (Средние тона) и **Highlights** (Свет). Установив переключатель в положение **Shadows** (Тени), вы получаете возможность изменять баланс цветов теней изображения. Установив его в положение **Midtones** (Средние тона), можете регулировать баланс цветов, точек изображения средней яркости, а переключив в положение **Highlights** (Свет) – можете работать с цветами самых ярких точек изображения. Установка флажка **Preserve Luminosity** (Сохранить яркость тонов) воспрепятствует изменению тонового диапазона изображения при корректировании его цветового баланса.
- Группа элементов управления **Color Balance** (Цветовой баланс) состоит из трех ползунковых регуляторов, перемещая которые, вы изменяете

цветовой баланс изображения. Первый регулятор, **Cyan-Red** (Синий-Красный), отвечает за баланс между синим и красным цветами. Второй – **Magenta-Green** (Фиолетовый-Зеленый) – за баланс между фиолетовым и зеленым цветами. И третий – **Yellow-Blue** (Желтый-Голубой) – за баланс между желтым и голубым цветами. Перемещая регулятор в сторону одного цвета, вы увеличиваете долю этого цвета в изображении и, вместе с тем, уменьшаете долю цвета, находящегося с ним в паре.

Завершив настройку баланса цветов изображения, щелкните мышью на кнопке **OK**, чтобы принять изменения и закрыть диалог.

Диалог Variations (Варианты)

Удобным и простым средством регулирования яркости/контрастности изображения совместно с цветовой коррекцией является диалог **Variations** (Варианты), Рис. 4.25. Чтобы открыть это окно, выберите команду меню **Image** ♦ **Adjustments** ♦ **Variations** (Изображение ♦ Регулировка ♦ Варианты).

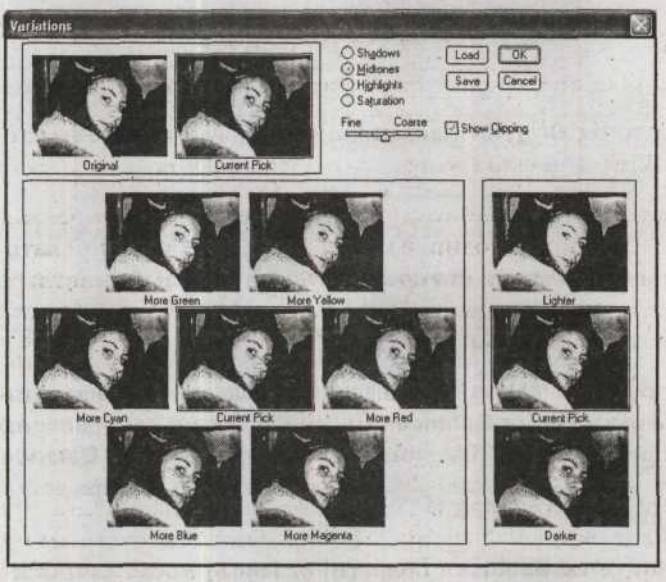


Рис. 4.25. Диалог ***Variations*** *(Варианты)*

В верхней части диалога **Variations** (Варианты) представлены две уменьшенные копии изображения: **Original** (Исходное изображение) и **Current Pick** (Текущий результат).

- Картинка **Original** (Исходное изображение) показывает, как выглядело изображение до обработки. Щелкнув мышью на этой картинке, вы сбросите все настройки, произведенные в диалоге **Variations** (Варианты).
- Картинка **Current Pick** (Текущий результат) показывает, как будет выглядеть изображение после принятия всех изменений, сделанных в диалоге **Variations** (Варианты).

В основной части диалога находится еще несколько уменьшенных версий изображения, с небольшими изменениями относительно его текущего состояния. Щелкнув мышью по одному из них, вы примете измененную версию изображения, сделав ее текущей. После этого все версии изображения опять обновятся.

Перечислим все картинки в основной части диалога по очереди.

- **More Green** (Больше Зеленого). Версия с усилением зеленого цвета.
- **More Yellow** (Больше Желтого). Версия с усилением желтого цвета.
- **More Cyan** (Больше Синего). Версия с усилением синего цвета.
- **More Red** (Больше Красного). Версия с усилением красного цвета.
- **More Blue** (Больше Голубого). Версия с усилением голубого цвета.
- **More Magenta** (Больше Фиолетового). Версия с усилением фиолетового цвета.
- **Lighter** (Светлее). Более светлая версия изображения.
- **Darker** (Темнее). Более темная версия изображения.

Справа от картинок **Original** (Исходное изображение) и **Current Pick** (Текущий результат) находится несколько элементов управления.

- Переключатели **Shadows** (Тени), **Midtones** (Средние тона) и **Highlights** (Свет). Установив один из них, вы будете регулировать баланс цветов теней изображения, его средних тонов или самых светлых участков.
- Ползунковый регулятор **Fine-Coarse** (Точно-Грубо). Передвигая его влево, вы уменьшаете «дозировку» изменений изображения, делая его настройку более точной, но и более медленной. Передвигая же регулятор вправо вы, напротив, увеличиваете «шаг», делая настройку более быстрой и грубой.
- Флажок **Show Clipping** (Показывать отсекаемые цвета). Если установить этот флажок, то на версиях изображения контрастными неоновыми цветами будут указываться области, которые после принятия версии уйдут из изображения, станут чистого белого или черного цвета.

Закончив настройку цветового баланса и яркости изображения, примите изменения и закройте диалог, щелкнув мышью на кнопке **OK**.

Редактирование изображений

Завершив тоновую и цветовую коррекцию изображения, можно приступать к редактированию его композиции. Обсудим по порядку важнейшие операции, наиболее часто выполняемые при работе над композицией.

Кадрирование

Важным этапом в обработке изображений, особенно фотографий, является кадрирование: определение новых границ изображения отсечением его краев.

Кадрирование компьютерных изображений проводится в двух разных случаях: когда необходимо обрезать картинку под определенный формат и в художественных целях – отрезание второстепенных деталей, выстраивание композиции кадра и т. д.

В программе Adobe Photoshop для проведения кадрирования предназначен инструмент **Crop** (Обрезка). Чтобы откадрировать с его помощью изображение:

- Щелкните мышью на кнопке **Crop** (Обрезка) на панели инструментов программы Adobe Photoshop. Указатель мыши примет следующий вид – .

На контекстной панели настроек можно установить некоторые параметры работы инструмента **Crop** (Обрезка), Рис. 4.26. В поля ввода **Width** (Ширина) и **Height** (Высота) можно ввести размеры кадрируемой области. Высота и ширина области кадрирования будут пропорциональны заданным размерам, а итоговое изображение будет иметь точно такие же размеры. Чтобы сбросить установки параметров **Width** (Ширина) и **Height** (Высота), щелкните мышью на кнопке **Clear** (Очистить).

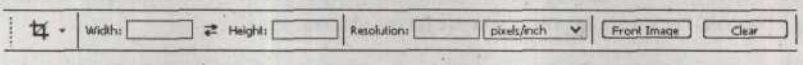


Рис. 4.26. Контекстная панель настроек инструмента ***Crop*** *(Обрезка)*

- Переместите указатель мыши к той точке изображения, где будет один из углов кадрируемой области. Нажмите левую кнопку мыши и, не отпуская ее, переместите указатель мыши к противоположному углу кадрируемой области.
- Отпустите левую кнопку мыши. Кадрируемая область будет окружена бегущей рамкой с маркерами, а оставшаяся часть изображения, которая будет обрезана, станет затененной (Рис. 4.27).

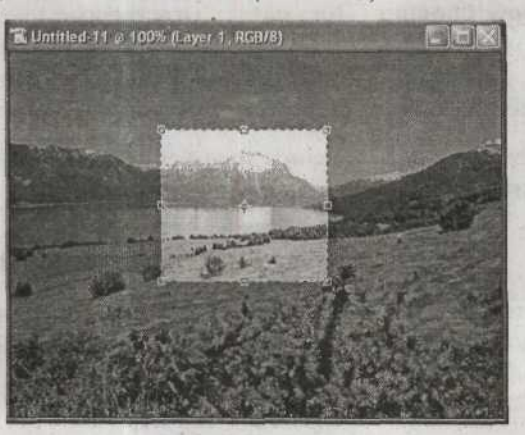


Рис. 4.27. Выделение кадрируемой области на изображении

Пока края окончательно не обрезаны, кадрируемую область можно подредактировать.

Чтобы передвинуть кадрируемую область:

- Переместите указатель мыши на область кадрирования, затем нажмите левую кнопку мыши и, не отпуская ее, перетащите область в другое место.
- Отпустите левую кнопку мыши. Область кадрирования будет передвинута.

Если вы хотите изменить размеры области кадрирования:

- Переместите указатель мыши на один из маркеров, находящихся по краю области кадрирования. Нажмите левую кнопку мыши и, не отпуская ее, переместите указатель мыши в другую позицию. Размеры области кадрирования будут изменяться.
- Отпустите левую кнопку мыши, размер области кадрирования будет изменен.
- Закончив настройку размеров и положения области кадрирования, нажмите клавишу Enter либо щелкните мышью на кнопке ✓. Изображение будет откадрировано.

Поворот изображений

Часто возникает необходимость повернуть изображение на определенный угол либо просто развернуть его. Для поворота изображений в программе Adobe Photoshop служит группа команд **Rotate Canvas** (Повернуть холст).

- Если вам необходимо повернуть изображение на 180 градусов, выполните команду меню **Image ◆ Rotate Canvas ◆ 180°** (Изображение ◆ Повернуть Холст ◆ 180°).
- Чтобы повернуть изображение на 90° по часовой стрелке, выполните команду меню **Image ◆ Rotate Canvas ◆ 90° CW** (Изображение ◆ Повернуть Холст ◆ 90° ЧС).
- Чтобы повернуть изображение на 90° против часовой стрелки, выполните команду меню **Image ◆ Rotate Canvas ◆ 90° CCW** (Изображение ◆ Повернуть Холст ◆ 90° ПЧС).
- Чтобы развернуть изображение по горизонтали, выполните команду меню **Image ◆ Rotate Canvas ◆ Flip Canvas Horizontal** (Изображение ◆ Повернуть Холст ◆ Развернуть Холст по Горизонтали).
- Чтобы развернуть изображение по горизонтали, выполните команду меню **Image ◆ Rotate Canvas ◆ Flip Canvas Vertical** (Изображение ◆ Повернуть Холст ◆ Развернуть Холст по Вертикали).

Если вам понадобилось повернуть изображение на произвольный угол, не кратный 90°, сделайте следующее:

- Выберите команду меню **Image ♦ Rotate Canvas ♦ Arbitrary** (Изображение ♦ Повернуть Холст ♦ Произвольно). Откроется диалог **Rotate Canvas** (Повернуть Холст), представленный на Рис. 4.28.

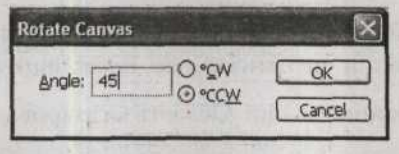


Рис. 4.28. Диалог ***Rotate Canvas*** *(Повернуть Холст)*

- Щелкните мышью в поле ввода **Angle** (Угол) и введите угол, на который хотите повернуть изображение.
- Установите переключатель в положение **CW** (ЧС), чтобы поворот шел в сторону по часовой стрелке или в положение **CCW** (ПЧС), чтобы поворачивать изображение против часовой стрелки.
- Щелкните мышью на кнопке **OK**, чтобы повернуть изображение и закрыть диалог.

Преобразование цветных изображений в оттенки серого цвета

Иногда требуется преобразовать цветное изображение в изображение, состоящее из оттенков серого цвета. Проведем такое преобразование:

- Выполните команду меню **Image ♦ Mode ♦ Grayscale** (Изображение ♦ Режим ♦ Оттенки серого). Откроется диалог запроса на подтверждение преобразования цветного изображения в изображение из оттенков серого цвета (Рис. 4.29).

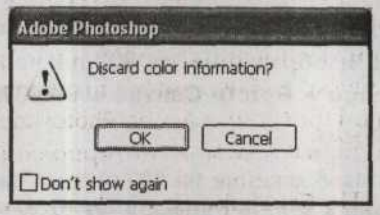


Рис. 4.29. Диалог с запросом на преобразование в полутоновое изображение

- Щелкните мышью на кнопке **OK**. Диалог закроется, а изображение будет преобразовано.

Изменение размеров изображений

Одной из последних стадий подготовки изображения к размещению его на Web-странице является установка подходящего размера изображения.

Размеры изображения в программе Adobe Photoshop изменяются с помощью диалога **Image Size** (Размеры Изображения), Рис. 4.30. Чтобы открыть этот диалог, выполните команду меню **Image ♦ Image Size** (Изображение ♦ Размеры Изображения).

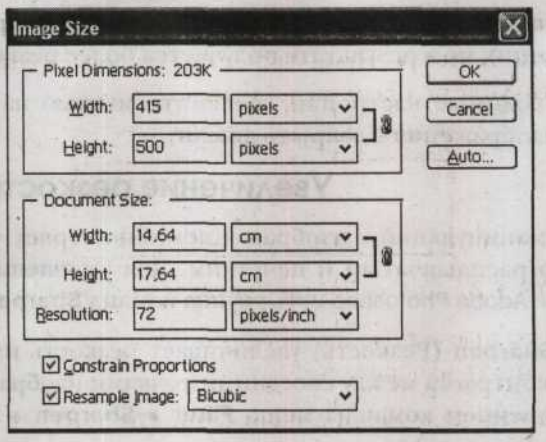


Рис. 4.30. Диалог ***Image Size*** *(Размеры Изображения)*

В группу элементов управления **Pixel Dimensions** (Пиксельные Размеры) сведены настройки, управляющие размером изображения в пикселах. В этой группе находятся два поля ввода, **Width** (Ширина) и **Height** (Высота), в которые вводятся желаемые высота и ширина изображения. Справа от этих полей находятся открывающиеся списки, состоящие из двух пунктов: **pixels** (пикселы) и **percent** (проценты). Если выбран пункт **pixels** (пикселы), то в поле ввода размеры указываются в точках, если же выбран пункт **percent** (проценты), – то в процентах от текущего размера изображения.

Если установлен флажок **Constrain Proportions** (Соблюдать Пропорции), то размеры изображения будут изменяться пропорционально: изменение ширины картинки будет вести к соответствующему изменению ее высоты, и наоборот.

В открывающемся списке **Resample Image** (Интерполяция изображения) можно выбрать способ, каким будут перерасчитываться пикселы изображения при изменении его размеров. При увеличении размеров изображения программа Adobe Photoshop рассчитывает цвета добавляемых пикселов изображения на основе цветов, уже существующих. Программа Adobe Photoshop поддерживает несколько способов такого перерасчета, называемого интерполяцией.

- **Nearest Neighbor** (По ближайшим соседям). Самый быстрый и наименее точный способ интерполяции.
- **Bilinear** (Билинейная). Немного более точный способ интерполяции, но работающий несколько медленнее.
- **Bicubic** (Бикубическая). Бикубическая интерполяция. Самый точный и медленный способ, дающий наилучший результат.
- **Bicubic Smoother** (Бикубическая сглаженная). Способ аналогичен бикубической интерполяции, но в результате получается более сглаженное изображение.

- **Bicubic Sharper** (Бикубическая резкая). Способ аналогичен бикубической интерполяции, но в результате получается более резкое изображение.

Проведя все необходимые настройки, щелкните мышью на кнопке **OK**, чтобы изменить размер изображения и закрыть диалог.

Увеличение резкости изображения

После различных манипуляций с изображением, оно теряет часть резкости, становится несколько расплывчатым и нечетким. Для улучшения резкости изображений в программе Adobe Photoshop есть группа команд **Sharpen** (Резкость).

- Команда **Sharpen** (Резкость) увеличивает резкость изображения за счет усиления контраста между соседними точками изображения. Запускается она выполнением команды меню **Filter ◆ Sharpen ◆ Sharpen** (Фильтр ◆ Резкость ◆ Резкость).

- Команда **Sharpen More** (Большая резкость) увеличивает резкость таким же способом, как и команда **Sharpen** (Резкость), но резкость увеличивается на большую величину. Запускается она выполнением команды меню **Filter ◆ Sharpen ◆ Sharpen More** (Фильтр ◆ Резкость ◆ Большая Резкость).

- Команда **Sharpen Edges** (Резкость границ) увеличивает резкость не всего изображения, а только границ с резкими переходами цветов. Запускается она выполнением команды меню **Filter ◆ Sharpen ◆ Sharpen Edges** (Фильтр ◆ Резкость ◆ Резкость границ).

В том случае, когда однократного применения одной из команд увеличения резкости недостаточно, ее можно применить повторно.

Оптимизация изображений для размещения их в Web

Когда изображение готово к размещению на Web-странице, необходимо подготовить его к размещенияю в Web. Для этого изображение необходимо сохранить в файле формата **GIF** или **JPEG**, чтобы при этом они занимали как можно меньше места, с минимальными потерями качества. Специально для такой оптимизации изображений, в программе Adobe Photoshop предназначен диалог **Save For Web** (Сохранить для Web), Рис. 4.31.

Открыть этот диалог можно, выбрав команду меню **File ◆ Save For Web** (Файл ◆ Сохранить Для Web).

Основную часть диалога занимает область предварительного просмотра изображения. Щелкая по ярлыкам над ним, вы можете выбрать вкладки с разными способами организации области предварительного просмотра.

- Вкладка **Original** (Исходное), представленная на Рис. 4.32. В диалоге будет представлено исходное изображение, до его оптимизации.

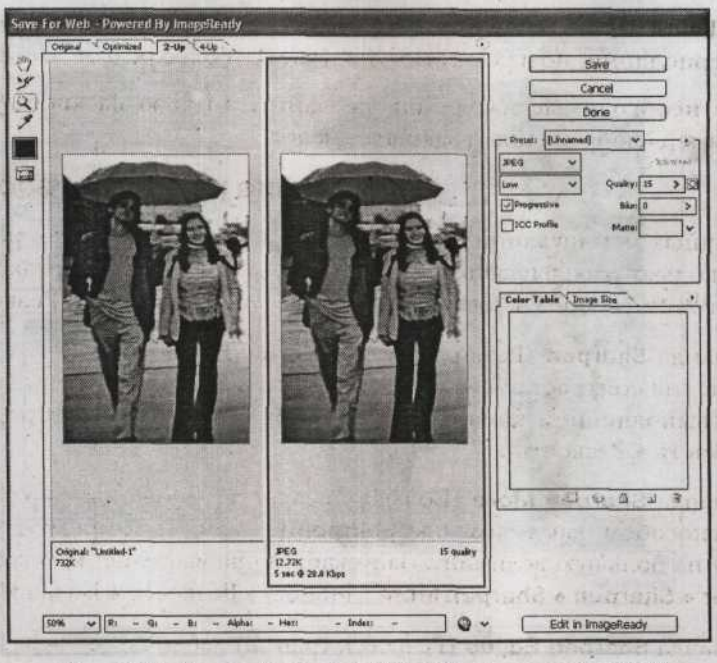


Рис. 4.31. Диалог ***Save For Web*** *(Сохранить для Web)*

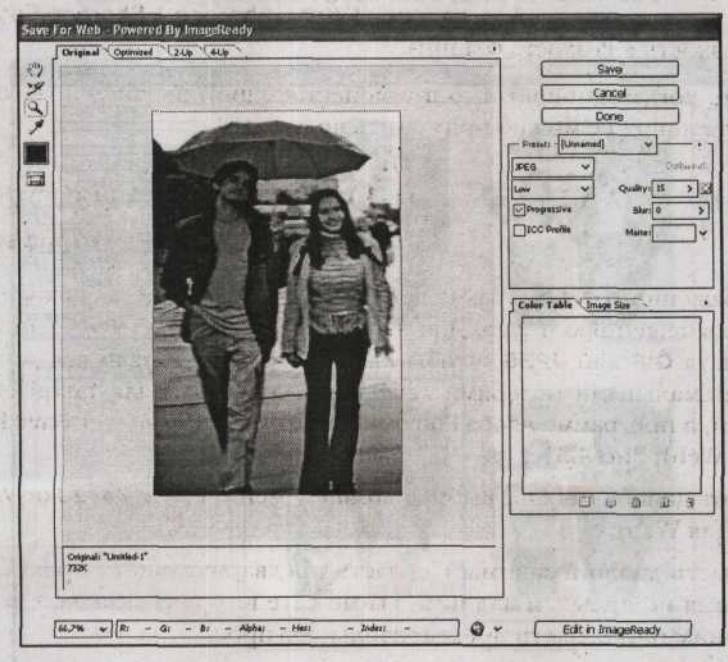


Рис. 4.32. Вкладка Original (Исходное) диалога ***Save For Web*** *(Сохранить для Web)*

- Вкладка **Optimized** (Оптимизированное), представленная на Рис. 4.33. Будет отображаться оптимизированное изображение.

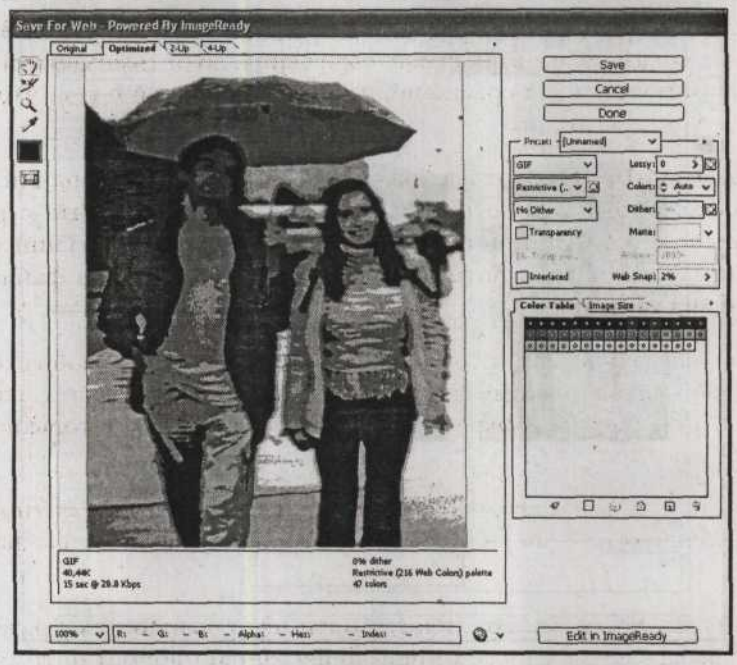


Рис. 4.33. Вкладка ***Optimized*** *(Оптимизированное) диалога* ***Save For Web*** *(Сохранить для Web)*

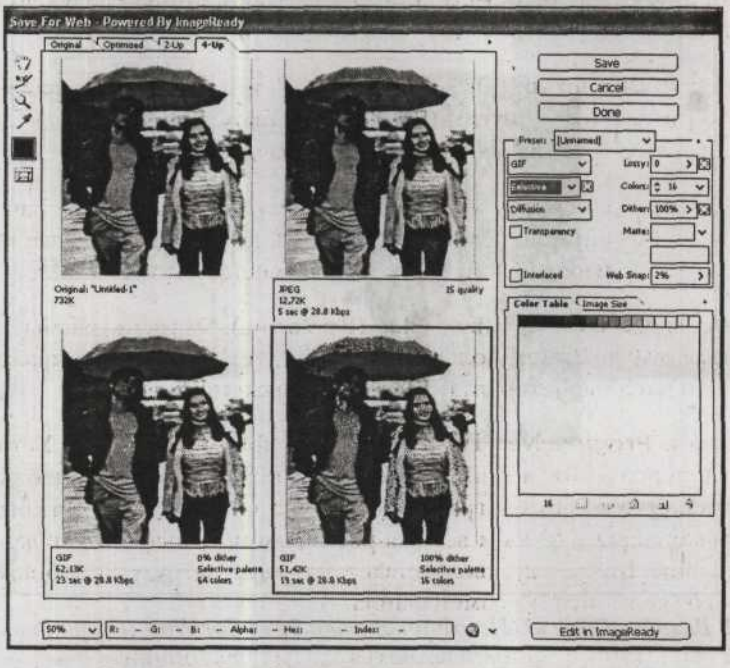


Рис. 4.34. Вкладка ***4-Up*** *(Четыре изображения) диалога* ***Save For Web*** *(Сохранить для Web)*

- Вкладка **2-Up** (Два изображения), представленная на Рис. 4.31. В диалоге будет показано два варианта изображения: до оптимизации и после. Этот режим полезен для сравнения исходной картинки с тем, что получается после оптимизации.

- Вкладка **4-Up** (Четыре изображения), представленная на Рис. 4.34. Будет отображено исходное изображение и три его варианта с разными настройками оптимизации. В этом режиме удобно сравнивать влияние разных параметров оптимизации и разных форматов файлов на внешний вид картинки.

В том случае, когда в диалоге отображаются два или четыре изображения, чтобы настроить параметры оптимизации того или иного варианта изображения, щелкните на нем мышью. Сохраняться, в итоге, тоже будет только выбранный вариант изображения.

В правой части диалога сосредоточены настройки параметров оптимизации изображения. Количество и вид элементов настройки зависят от выбранного формата файла.

Формат файла, в который сохраняется оптимизированное изображение, выбирается в открывающемся списке **Optimized file format** (Формат оптимизированного файла).

Если выбран формат файла **JPEG** (Рис. 4.35), то доступны следующие параметры оптимизации:

- Флажок **Optimized** (Оптимизация). Установка этого флажка включает дополнительную оптимизацию изображения, которая приводит к уменьшению файла без потери качества изображения.

- Поле ввода со счетчиком **Quality** (Качество). Отвечает за качество изображения. Значение может изменяться от 0 до 100. Чем больше число, тем выше качество изображения и тем больше места занимает файл.

- Поле ввода со счетчиком **Blur** (Размытие). Степень размытия может меняться от 0 до 2. Чем больше размытие, тем меньше места занимает файл, но тем расплывчатее становится изображение.

- Флажок **Progressive** (Прогрессивный формат файла). Установка этого флажка включает в файл несколько версий изображения с уменьшенным разрешением. При загрузке через Web сначала будет открыта самая маленькая версия, потом версия размером побольше, и так далее, до самой большой. Визуально увеличивает скорость загрузки изображения, но немного увеличивает размер файла.

Для формата **GIF** (Рис. 4.36) предлагаются другие настройки:

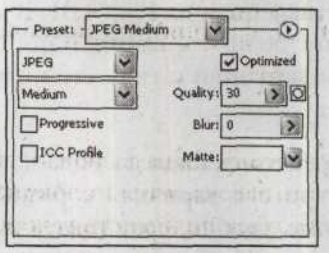


*Рис. 4.35 Параметры настройки файла формата **JPEG***

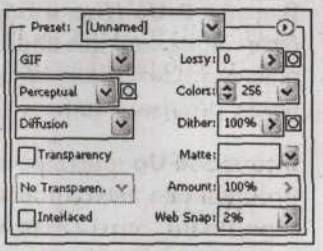


*Рис. 4.36. Параметры настройки файла формата **GIF***

Прямо под открывающимся списком выбора форматов находится открывающийся список **Color reduction algorithm** (Алгоритм сокращения количества цветов). В этом списке можно выбрать способ, каким количество цветов изображения уменьшается до палитры **GIF**. Возможны следующие варианты:

- **Perceptual** (Перцепционный). Цвета выбираются с учетом особенностей человеческого восприятия.
- **Selective** (Выборочный). Цвета выбираются также с учетом особенностей человеческого восприятия, но предпочтение отдается цветам из палитры безопасных Web-цветов. Речь о том, что это за безопасные цвета, пойдет чуть позже.
- **Adaptive** (Адаптивный). Выбираются цвета, наиболее часто встречающиеся в изображении.
- **Restrictive (Web)** (Ограниченные Web-палитрой). Используются только безопасные Web-цвета.
- **Black & White** (Черный и Белый). Палитра состоит из двух цветов – черного и белого.
- **Grayscale** (Оттенки серого). Цвета изображения преобразовываются в оттенки серого цвета.

Чуть ниже находится открывающийся список **Specify the dither algorithm** (Определите алгоритм сглаживания). В этом списке выбирается способ, каким будут имитироваться цвета, отсутствующие в палитре итогового изображения. Возможны следующие варианты:

- **No Dither** (Без сглаживания). Цвета не имитируются, а просто замещаются ближайшим похожим цветом из палитры.
- **Diffusion** (Диффузия). Цвета имитируются смешением цветов нескольких соседних точек, смешанных в случайном порядке.
- **Pattern** (Шаблон). Цвета имитируются смешением цветов соседних точек, выстроенных в определенном порядке.

- **Noise** (Шум). Похож на режим **Diffusion** (Диффузия), но несколько другой способ размещения точек.

Перечислим остальные параметры формата **GIF**:

- **Interlaced** (Чересстрочный). Установка этого флажка включает чересстрочный режим загрузки **GIF** файла. Сначала загружается половина строк, через одну, затем вторая половина. Это создает видимость ускорения загрузки.
- Поле ввода с счетчиком **Lossy** (С потерями). Позволяет несколько уменьшить размер файла за счет уменьшения качества изображения. Величина в этом поле может изменяться от 0 до 100.

0 – отсутствие ухудшения качества, 100 – максимальное ухудшение качества и минимальный размер файла.

- Поле ввода с счетчиком **Colors** (Цвета). В этом поле выбирается, какое количество цветов будет в палитре итогового изображения. Чем меньше цветов, тем меньше места занимает файл и тем хуже выглядит изображение. Количество цветов может быть от 2 до 256.
- Поле ввода с счетчиком **Dither** (Сглаживание). Определяет степень имитации цветов, не вошедших в палитру. Значение в поле может быть от 0 до 100%. 0% – отсутствие сглаживания, 100% – максимальное сглаживание.
- Поле ввода с счетчиком **Web-Snap** (Приближение к Web). Степень соответствия цветовой палитры изображения палитре безопасных Web-цветов. Может изменятся от от 0 до 100%. 0% – отсутствие соответствия, 100% – палитра полностью состоит из безопасных Web-цветов.

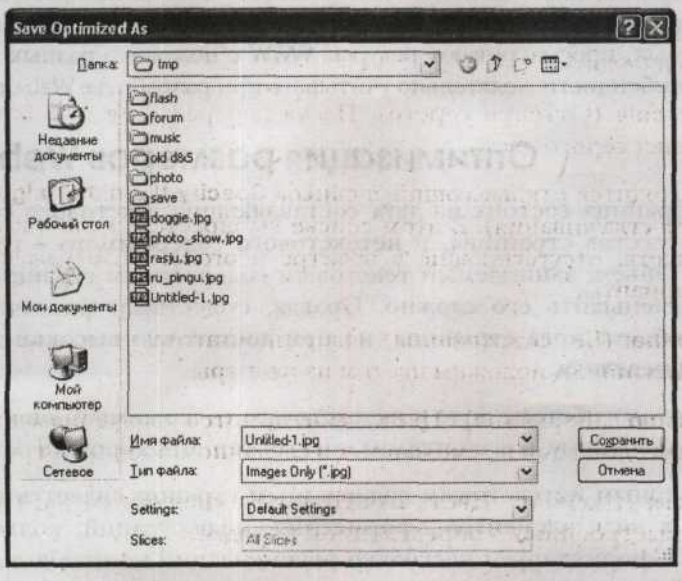


Рис. 4.37. Диалог сохранения оптимизированного файла

Закончив настройку параметров оптимизации изображения, сохраните его:

- Щелкните мышью на кнопке **Save** (Сохранить), откроется диалог **Save Optimized As** (Сохранить оптимизированный файл как), представленный на Рис. 4.37.
- Перейдите к папке, в которой вы хотите сохранить файл.
- Щелкните мышью в поле ввода **Имя файла** (Filename) и введите название, под которым вы хотите сохранить файл изображения.
- Щелкните мышью на кнопке **Сохранить** (Save), чтобы сохранить файл и закрыть диалоги **Save Optimized As** (Сохранить оптимизированный файл как) и **Save for Web** (Сохранить для Web).

Оптимизация кода HTML для Web

Размещение Web-страниц в сети Интернет предъявляет к страницам ряд требований, которые необходимо учитывать при их создании.

Существуют два основных момента, которые необходимо всегда иметь в виду.

- Первый – пропускная способность Интернет-соединений отнюдь не бесконечна и чем меньше места будет занимать ваша страница, тем быстрее она будет загружена пользователем и тем выше вероятность, что он дождется ее загрузки. Кроме того, за данные, прошедшие через Интернет-канал, часто приходится платить. Поэтому, важнейшей характеристикой Web-страницы является ее размер, желательно его минимизировать.
- И второй важный момент – компьютеры у всех пользователей разные. Люди пользуются разными мониторами, работают в разных операционных системах, просматривают ресурсы WWW с помощью разных браузеров. Все эти особенности желательно учитывать при разработке Web-страниц.

Оптимизация размеров Web-страниц

Любая Web-страница состоит из двух составляющих: текстового содержимого – HTML-кода и текстов страницы, и нетекстового содержимого – графики, звука, видео и т. д. Объем, занимаемый текстовым содержимым страницы обычно невелик, да и уменьшить его сложно. Правда, существуют различные методики уменьшения объема кода страницы, но при достаточно высокой трудоемкости, отдача от них невелика.

Совсем по другому обстоит дело с нетекстовым содержимым: оно обычно достаточно велико по объему, и его оптимизация может дать хорошие результаты.

Самым популярным нетекстовым содержимым страниц является графика – она используется в виде элементов оформления, иллюстраций, коллекций изображений и т. д. Сформулируем несколько рекомендаций по использованию графики на ваших Web-страницах.

- Не злоупотребляйте графикой в оформлении страниц. Все хорошо в меру. Основным содержанием вашей Web-страницы, если это не рекламный сайт и не галерея изображений, является текст. Оформление страницы служит для более удобного и красивого представления текстового содержимого и ни для чего иного.
- Оптимизируйте изображения. Подготавливая изображения для размещения в Web, пользуйтесь диалогом **Save For Web** (Сохранять для Web) программы Adobe Photoshop. Подбирайте параметры оптимизации так, чтобы изображение занимало минимально возможный объем и при этом сохраняло достойное качество. Не бойтесь экспериментировать, попробуйте несколько вариантов настроек оптимизации и выберите лучший.
- Не размещайте изображения слишком большого размера. Помните, что чем больше размер изображения, тем больше оно занимает места, поэтому старайтесь размещать изображения минимально необходимого размера.
- Используйте уменьшенные версии изображений для предварительного просмотра. Если вы хотите расположить на странице много изображений, например вашу фотогалерею, то лучше разместите на ней уменьшенные версии изображений, щелкая по которым посетитель сможет открыть изображения большого размера. Это ускорит загрузку страницы и позволит посетителю просмотреть только те изображения, которые его заинтересуют.

По поводу размещения видео и звука на ваших страницах можно привести практически те же рекомендации, что и для размещения графики: не злоупотребляйте такими объектами; старайтесь уменьшить их размер так, чтобы при минимальном размере обеспечивалось удовлетворительное качество.

Создание универсальных Web-страниц

Как уже было сказано выше, компьютеры у всех пользователей WWW разные. Разные у пользователей размеры мониторов, разрешение экранов, размеры окна браузера. При создании Web-страниц это важно учитывать.

- По статистике, разрешение экранов у большинства пользователей компьютеров 1024x768 точек и выше, но есть люди, на компьютерах которых выставлено разрешение не больше 800x600 точек. Количеством пользователей с меньшим разрешением экрана можно пренебречь. Желательно создавать Web-страницы так, чтобы они помещались в Web-браузерах, запущенных на компьютерах с разрешением экрана 800x600 точек. Но, вместе с тем, нельзя забывать и о пользователях с большими экранами. Разрешить это противоречие позволяет концепция резинового дизайна, рассмотренная в главе 3. Оформление страницы создается так, чтобы все элементы оформления занимали в ширину не более 750 точек, 50 точек отводится на границы окна браузера, полосы прокрутки и т.д. Если ширина окна браузера больше 750 точек, то содержимое страницы раздвигается, занимая весь свободный объем полезной информацией.

- Еще лет 10 назад большинство компьютеров не могли отображать на экране больше 256 цветов. В связи с этим была разработана специальная безопасная палитра Web-цветов. Она состояла из 216 цветов, которые могли быть отображены на любом компьютере с цветны экраном. Сейчас таких компьютеров осталось очень мало и безопасные Web-цвета давно потеряли свою актуальность. Но, тем не менее, если вы хотите, чтобы с вашими Web-страницами не испытывали проблем люди, пользующиеся такими компьютерами, используйте безопасную палитру, хотя это может вас сильно ограничить при создании графики.

- В настоящее время практически не осталось компьютеров, мониторы которых отображают только оттенки серого цвета. Но есть много людей с ослабленным зрением, плохо воспринимающие оттенки цвета, также множество людей страдает разными формами дальтонизма. Чтобы эти категории посетителей вашего сайта могли без труда знакомиться с содержимым ваших Web-страниц, необходимо чтобы страницы легко воспринимались, будучи преобразованными в режим оттенков серого цвета. Цвета, легко отличимые друг от друга в цветном режиме, при преобразовании в оттенки серого могут сливаться друг с другом.

- Как уже говорилось в главе 3, все изображения, размещаемые на ваших Web-страницах, необходимо снабжать альтернативными описаниями. Это важно для людей, отключающих загрузку графики с Web-страниц и для людей с ослабленным зрением, пользующихся речевыми браузерами, озвучивающими содержание Web-страниц.

Заключение

Этой главой завершается рассказ о создании статических Web-страниц. Вы знаете, как создавать такие страницы, знакомы с различными тонкостями их оформления, умеете делать страницы, насыщенные разнообразнейшим содержанием, так, чтобы они хорошо выглядели, быстро загружались и были доступны максимальному количеству людей.

Вы много достигли, но вскоре достигнете еще большего. Начинается новый этап освоения Web-технологий – создание динамических Web-сайтов. Этому посвящены следующие главы данной книги. По сути своей, динамические сайты являются расширением структуры и возможностей сайтов статических. Уже приобретенные вами знания, умения и опыт станут основой, которую вы дополните сведениями о технологиях создания динамических сайтов.

глава 5

Web-сайт с динамическими страницами

Пришла пора оживить ваши Web-страницы, добавить в них динамики и гибкости; этой главой начинается изучение технологий создания динамических Web-сайтов. В ней мы разберемся, какие бывают динамические сайты, как они выполняются на сервере и в браузере пользователя. Основное внимание в этой главе будет уделено написанию сценариев на языке JavaScript и внедрению их в HTML-страницы. Вы узнаете об основных командах и конструкциях этого языка, познакомитесь с тем, как сценарии JavaScript взаимодействуют с кодом HTML-страниц. Полученные знания мы применим на практике, создав действующую программу на языке JavaScript – инженерный калькулятор.

Как работают статические и динамические Web-сайты

В первой главе мы уже знакомились с особенностями выполнения статических и динамических Web-страниц на Web-сервере и в браузере пользователя. Рассмотрим этот процесс подробнее.

Статические Web-страницы лежат в каталогах сервера в виде уже готовых HTML-файлов. Работа сервера, в этом случае, заключается лишь в том, чтобы в ответ на запросы браузера передавать ему необходимые файлы. Браузер, в свою очередь,

отображает страницу, опираясь на правила языка HTML, и обеспечивает реакцию на действия пользователя, щелкающего по гиперссылкам. Все просто и надежно, но возможности такого сайта ограничены. Реакция браузера на действия пользователя ограничивается переходом по ссылкам, все страницы сайта должны быть созданы заранее и, чтобы изменить содержание сайта, необходимо переписать код страниц.

Ограничения статических Web-страниц преодолеваются в динамических Web-сайтах. Технологии создания таких сайтов делятся на две группы: серверные и клиентские технологии.

Серверные технологии создания динамических Web-сайтов

Серверные технологии работают следующим образом: браузер посылает Web-серверу запрос, в ответ на который сервер запускает программу, называемую сценарием или скриптом. Этот сценарий, в зависимости от параметров запроса, «на лету» формирует HTML-страницу, которая и передается браузеру. С помощью сценариев можно также генерировать изображения, анимации, создавать письма электронной почты и делать многое другое.

Серверные сценарии можно писать на множестве различных языков программирования, но чаще всего для этой цели используются языки, специально для этого предназначенные: PERL (Practical Extraction and Report Language – Практичный язык для создания выборок и отчетов) и PHP (расшифровывается как самоповторяющая аббревиатура PHP: Hypertext Preprocessor – PHP: препроцессор гипертекста).

К серверным сценариям прибегают в тех случаях, когда необходимо сохранять какую-либо информацию о посетителе на сервере, периодически обновлять содержимое сайта, не перелопачивая вручную код страниц, выдавать пользователю не всю информацию, а только соответствующую его запросу.

Серверные технологии необходимы при создании поисковых систем, форумов, Интернет-магазинов, новостных сайтов, почтовых служб типа **mail.ru** и **gmail.com** и во многих других случаях.

Одним из самых ярких применений серверных технологий является концепция «Web-приложений» – программ, аналогичных по своей функциональности приложениям, запускаемым на компьютере пользователя, но работающим на Web-сервере. Доступ к таким приложениям осуществляется с помощью обычного Web-браузера. Плюсом такой системы является то, что пользователь может работать с программой на любом компьютере, подключенном к сети Интернет. Причем, поскольку данные пользователя тоже хранятся на сервере, нет нужды переносить их с компьютера на компьютер. Главные минусы Web-приложений – необходимость быстрого Интернет-канала и меньшее удобство работы по сравнению с обычными программами. Наиболее популярным видом Web-приложений являются разнообразные Web-интерфейсы для работы с электронной почтой, которые есть на сайтах большинства почтовых служб.

При использовании серверных технологий никаких дополнительных требований к Web-браузеру пользователя не предъявляется, он может быть практически любым, вся нагрузка по созданию страниц и обработке данных ложится на сервер, который должен обладать достаточной мощностью, чтобы успевать обрабатывать запросы всех пользователей и иметь поддержку соответствующих языков сценариев.

Поведение Web-браузера при работе с сайтами, созданными при помощи серверных технологий, ничем не отличается от работы с обычными статическими сайтами. Браузер по-прежнему работает с уже готовыми HTML-страницами, разница лишь в том, что содержимое страниц формируется для каждого посетителя сайта отдельно.

Клиентские технологии создания динамических Web-страниц

В отличие от серверных технологий, клиентские технологии создания динамических Web-страниц позволяют влиять непосредственно на работу браузера. Основу клиентских технологий составляют сценарии на языке JavaScript, Jscript или VBScript, внедренные прямо в HTML-страницу либо подгружаемые дополнительно. С помощью таких сценариев можно создавать различные динамические эффекты, делать Web-страницы, изменяющиеся в ответ на действия пользователя, встраивать в них несложные игры и программы. Кроме того, современные Web-браузеры поддерживают механизм так называемых «плагинов» – небольших дополнений к браузеру, позволяющих включать в Web-страницу дополнительные динамические элементы – модули Flash, программы на языке JAVA, трехмерные объекты на языке VRML (Virtual Reality Modeling Language – Язык моделирования виртуальной реальности) и т. д.

С помощью клиентских сценариев, создаются различные «всплывающие» меню; кнопки, изменяющие внешний вид при наведении на них указателя мыши и т. д. Чтобы клиентские технологии правильно работали, необходимо, чтобы они поддерживались браузером посетителя Web-страницы, а вся вычислительная нагрузка по их обработке ложится на его компьютер.

Наилучших результатов часто можно достигнуть, комбинируя как серверные, так и клиентские технологии на одном сайте. На сервере может генерироваться содержимое страниц, а динамическое меню сайта может реализовываться с помощью клиентских сценариев. В этой главе мы рассмотрим клиентские технологии создания динамических Web-страниц с использованием языка JavaScript. Этот язык выбран нами по той причине, что он поддерживается максимальным количеством браузеров.

Создание клиентских сценариев на языке JavaScript

Технология совместного использования HTML вместе с клиентскими сценариями называется DHTML (Dynamic HTML – динамический HTML). Обе ее составляющие – код HTML и клиентские сценарии – тесно связаны друг с другом. Сценарии, внедренные в HTML-страницу, могут влиять практически на все ее элементы, создавать новые окна браузера с HTML-кодом и изображениями, открывать диалоги с предупреждениями и вопросами.

Объектная модель HTML-документа

Элементы HTML-документа называются объектами. Сценарии JavaScript манипулируют объектами HTML-страницы. Всю HTML-страницу можно представить как иерархию объектов. Можно представить следующий наглядный пример такой иерархии. Главный объект – квартира. Внутри квартиры три комнаты – это дочерние объекты квартиры, также их можно назвать ее потомками. А квартира для комнат является родительским объектом. Содержимое комнат (например, столы, стулья, шкафы) будет, в свою очередь, состоять из дочерних объектов для комнат. Дочерние объекты комнат являются дочерними объектами и для объекта квартиры, и наоборот, объект квартиры является родительским объектом и для объектов содержимого комнат.

Полное описание содержимого квартиры в терминах объектов будет называться объектной моделью, DOM (Document Object Model – объектная модель документа). Указание на определенный объект в комнате будет выглядеть следующим образом: **квартира.комната[0].стол**. Имена дочерних объектов указываются после родительских, через точку. Поскольку комнат в квартире несколько, в квадратных скобках указывается номер конкретной комнаты. Такие группы объектов, состоящие из нескольких однородных элементов, называются коллекциями. Элементы коллекции могут отличаться друг от друга именами либо нумерацией.

Документ HTML имеет достаточно большую иерархию объектов. Например, объект **window** отвечает за текущее окно браузера. Его дочерним объектом является объект HTML-документа в целом – **document**. У объекта **document** есть ряд дочерних коллекций объектов.

- **forms** – коллекция форм. Что такое формы, вы узнаете в следующей главе.
- **anchors** – коллекция якорей.
- **images** – коллекция изображений.
- **links** – коллекция ссылок.
- **plugins** – коллекция подключаемых модулей.

С помощью объектов можно узнать или изменить параметры большинства элементов HTML-страницы. В объектной модели JavaScript есть также несколько объектов, не касающихся напрямую документа HTML, например:

- **history** – объект, дающий доступ к истории посещенных ссылок
- **location** – объект, содержащий текущий URL.
- **screen** – объект, дающий доступ к характеристикам экрана.
- **Date** – объект, работающий с датой и временем.
- **Array** – объект для работы с массивами переменных.
- **Math** – объект математических функций.

Обратиться к одному из элементов коллекции объектов можно, зная номер этого элемента на странице. Номера присваиваются объектам автоматически в процессе открытия страницы браузером. Нумерация идет с «0» по возрастанию. Если рассмотреть страницу, код которой показан в Листинг 5.1, то в ней тегу, отображающему рисунок **photo1.jpg**, будет присвоен номер «0», тегу с рисунком **photo2.jpg** – номер «1», а тегу с рисунком **photo3.jpg** – номер «2». И места этих тегов в иерархии объектов страницы будут **document.images[0]**, **document.images[1]** и **document.images[2]** соответственно.

Листинг 5.1. Нумерация элементов ***Web****-страницы*

```
<html>
<head>
 <title>Нумерация элементов</title>
</head>
<body>
 <img src="photo1.jpg"><br>
 <img src="photo2.jpg"><br>
 <img src="photo3.jpg"><br>
</body>
</html>
```

Для удобства обращения к элементам Web-страницы, им можно присваивать имена. Имя тегу HTML задается при помощи атрибута **NAME**, например так: **<IMG NAME="rabbit" SRC="rabbit.gif">**. После этого, вместо безликого номер элемента в коллекции объектов, можно использовать осмысленное имя: **document.rabbit**;

События

Сценарии на Web-странице выполняются не постоянно, а только в случае наступления определенных событий, например нажатия кнопок, наведения указателя мыши на определенный участок страницы, прохождения определенного интервала времени или другого действия. При наступлении заданного события запус-

кается определенный сценарий JavaScript, называемый обработчиком события. После его завершения, система опять переходит в режим ожидания событий, практически не потребляя ресурсов компьютера.

Хотя чаще всего события назначают различным кнопкам, событие можно «повесить» практически на любой элемент HTML-страницы. Приведем полный список возможных событий, с которыми можно связать сценарии:

- **onLoad** – окончание открытия HTML документа.
- **onUnload** – выгрузка HTML документа.
- **OnClick** – щелчок мышью на поверхности элемента.
- **OnDblClick** – двойной щелчок мышью на поверхности элемента.
- **OnMouseDown** – нажатие кнопки мыши.
- **OnMouseOver** – перемещение мыши на элемент из-за его пределов.
- **OnMouseMove** – перемещение мыши над элементом.
- **OnMouseOut** – перемещение мыши с элемента за его пределы.
- **OnFocus** – элемент получает фокус ввода.
- **OnBlur** – элемент теряет фокус ввода.
- **OnKeyPress** – пользователь нажимает и отпускает клавишу.
- **OnKeyDown** – пользователь нажимает клавишу над элементом.
- **OnKeyUp** – пользователь отпускает клавишу над элементом.
- **OnSubmit** – данные из формы переданы Web-серверу.
- **OnReset** – форма очищена.
- **OnSelect** – пользователь выбирает текст в текстовом окне.
- **OnChange** – элемент теряет фокус ввода после изменения значения элемента.

Базовые понятия языка JavaScript

Чтобы вы могли самостоятельно создавать сценарии, вам необходимо разобраться с основами языка JavaScript. Базовые понятия этого языка, такие как переменные, операторы, функции, являются общими для большинства языков программирования и пригодятся вам при изучении серверных языков сценариев.

Сценарии

Любой сценарий представляет собой набор команд, последовательно исполняемых программой, выполняющей сценарий. В данном случае Web-браузером. Традиционно, изучение языков программирования начинается с написания программы, выводящей на экран монитора приветствие: «Hello, World». Не будем отступать от

этой традиции и мы. В Листинг 5.2 приведена HTML страница, содержащая сценарий, выводящий на экран диалог с надписью «Hello, world» (Рис. 5.1).

Листинг 5.2. Первый сценарий JavaScript

```
<html>
<body>
 <script language="JavaScript">
 window.alert("Hello, world!")
 </script>
</body>
</html>
```

Рис. 5.1. Диалог, вызванный сценарием JavaScript

Договоримся о терминологии. Особенности записи команды и ее параметров в сценарии, называются синтаксисом. В описании синтаксиса команды необязательные ее параметры заключаются в квадратные скобки – **[необязательный_параметр]**.

Переменные

Результаты промежуточных вычислений, обрабатываемые текстовые строки и другая информация, с которой работают программы, хранится в переменных. Переменные можно сравнить с грифельной доской, на которой вы можете записать некоторую информацию, затем стереть ее и записать что-нибудь другое. Чтобы прочитать содержимое переменной, или изменить его, нужно знать только название переменной.

Переменные могут содержать значения трех различных типов:

- Числовые. Числа могут целыми и дробными, например: 34 и 2.1234.
- Логические. Переменные могут принимать одно из двух логических значений: **True** (Истина) или **False** (Ложь).
- Строки. Переменные в JavaScript могут содержать строки из нулевого и большего количества знаков. При присвоении переменной строкового значения, строка заключается в двойные или одинарные кавычки, например так: **x="строка"**. Чтобы включить кавычку в строковую переменную, необходимо поставить перед ней знак «\». Выражение **x="\" Фраза в кавычках\""** запишет в переменную **x** значение "Фраза в кавычках".

В языке JavaScript названия переменных имеют следующие особенности:

- Название может состоять из символов латинского алфавита от «a» до «z» как строчных, так и прописных, цифр от «0» до «9» и символов подчеркивания «_».
- Название переменной не может начинаться с цифры. Название переменной **_3abc** – правильное, а **3_abc** – уже нет.
- JavaScript различает строчные и прописные буквы в названии переменной: **Abc** и **ABC** – две разные переменные.

В командах JavaScript переменные могут использоваться в качестве параметров наряду с числами и строками текста.

Операторы

Основные математические операции, манипуляции со строками и переменными производятся с помощью операторов. Основной оператор – оператор присваивания, =. Результат вычислений, проведенных справа от оператора =, записывается в переменную, имя которой стоит слева от него. Например, выражение **x=6** означает, что в переменную **x** записывается число **6**. Другой пример: **a=b**, – в переменную **a** записывается значение переменной **b**.

Математические вычисления проводятся при помощи стандартных математических операторов сложения, вычитания, деления, умножения и скобок: « + , - , / , * , ( ) », вычисления проводятся в согласии с правилами арифметики. Например так: **2*4+1**. Результатом выполнения этого выражения будет число 9. А в результате выполнения выражения **m=(3+1)*2**, переменная **m** примет значение 8.

Вычисления могут производиться и над значениями переменных. Например, если значение переменной **exp** равняется 25, то в результате выполнения выражения **res=exp/5** в переменную **res** будет записано число 5.

Оператор сложения используется также для объединения между собой текстовых строк. Результатом выражения **«амбар» + «цумян»** будет строка «амбарцумян».

Кроме стандартных математических действий, в языке JavaScript есть несколько дополнительных математических операций.

- **%** – модуль. Вычисляет остаток от целочисленного деления одного выражения на другое. Пример – **12%5**, результатом этого выражения будет 2.
- **+=** – добавление некоторого значения к переменной. Выражение **x+=y** добавляет к значению переменной **x** значение переменной **y** и записывает результат в переменную **x**. Это выражение является более короткой записью выражения **x=x+y**.
- **-=** – вычитание некоторого значения из переменной. Выражение **-=y** вычитает из значения переменной **x** значение переменной **y** и записывает результат в переменную **x**. Это выражение является более короткой записью выражения **x=x-y**.
- ***=** – умножение значения переменной на некоторую величину. Выражение **x*=y** умножает значение переменной **x** на значение переменной **y** и

записывает результат в переменную **x**. Это выражение является более короткой записью выражения **x=x*y**.

- **/=** – деление значения переменной на некоторую величину. Выражение **x/=y** делит значение переменной **x** на значение переменной **y** и записывает результат в переменную **x**. Это выражение является более короткой записью выражения **x=x/y**.
- **++** – инкремент. Увеличивает значение переменной на единицу. Записывается так: **x++**. В результате значение переменной **x** увеличится на 1.
- **--** – декремент. Уменьшает значение переменной на единицу. Записывается так: **x--**. В результате значение переменной **x** уменьшится на 1.

Логические операторы

Результатом логических, или Булевых, выражений, может являться одно из двух значений: **True** (истина) или **False** (Ложь). Простые логические выражения состоят из двух значений, или выражений, объединенных оператором сравнения: **1e_значение оператор_сравнения 2e_значение**. Существуют следующие операторы сравнения:

- **==** – равенство. Если оба значения равны, то результат выражения равен **True** (истина), иначе **False** (Ложь).
- **!=** – неравенство. Если значения не равны, то результатом выражения будет **True** (истина).
- **<** – меньше. Результат **True** (истина), если первое выражение меньше второго.
- **>** – больше. Результат **True** (истина), если 1-е выражение больше второго.
- **<=** – меньше либо равно. Результат **True** (истина), если 1-е выражение меньше либо равно второму.
- **>=** – больше либо равно. Результат **True** (истина), если 1-е выражение больше либо равно второму.

Приведем несколько примеров логических выражений. **2<3** – результат **True** (истина), поскольку 2 меньше 3; **2*2==4** – **True** (истина); **34>=45** – **False** (Ложь). Несколько простых логических выражений можно объединить в более сложное выражении при помощи логических операторов. Логическое выражение будет выглядеть так: **1e_логическое_значение логический_оператор 2e_логическое_значение**. Исключение составляет логический оператор **Not** (Не), который размещается перед единственным логическим значением.

Логические операторы бывают следующими:

- **!** – **Not** (Не). Меняет результат логического выражения на обратный, т. е. значение **True** (истина) меняет на **False** (Ложь), и наоборот.
- **&&** – **And** (И). Результатом выражения будет **True** (истина) в том случае, если значение **True** (истина) имеют оба логические значения.

- || – **OR** (Или). Результатом выражения будет **True** (истина), если значение **True** (истина) имеет хотя бы одно из логических значений.

Результатом выражения **2>5||5>2** будет **True** (истина), поскольку одно из выражений имеет результат **True** (истина). Результатом выражения **3>0 && 3>5** будет **False** (Ложь), поскольку одно из выражений имеет результат **False** (Ложь).

Условные переходы

При написании программ часто возникают ситуации, в которых необходимо, чтобы при одних значениях определенной переменной выполнялись одни команды, а при других значениях – другие. В таких случаях используются операторы условного перехода.

Можно привести следующий пример условного оператора из реальной жизни: «Если пойдет дождь, то я останусь дома, в противном случае я пойду гулять». В рассмотренном условном операторе условие выполнения команд определяет переменная «погода», если ее значение будет «дождь», то выполняются команды «остаться дома», при любых других значениях этой переменной, выполняется группа команд «пойти гулять».

В языке JavaScript, условные переходы создаются при помощи конструкции **IF ELSE**, см. Листинг 5.3. Если результат логического выражения **условие** равен **True** (истина), то выполняются команды **последовательность_команд_1**, в противном случае выполняются команды **последовательность_команд_2**. То есть, если условие выполняется, то исполняются команды, находящиеся в фигурных скобках непосредственно после оператора **IF**, если же условие не выполнено, – то команды, идущие за оператором **ELSE**. Если вместо последовательности команд нужно исполнить всего одну команду, то фигурные скобки использовать необязательно.

Листинг 5.3. Условный переход

```
if (условие)
  {последовательность_команд_1}
else
  {последовательность_команд_2}
```

Пример реального условного перехода в сценарии JavaScript приведен в Листинг 5.4. Если значение переменной **x** равняется 3, то на экран выводится диалоговое окно, сообщающее об этом. Если же значение этой переменной иное, то выводится другое диалоговое окно, сообщающее о том, что **x** не равен 3.

Листинг 5.4. Пример реального условного перехода

```
If (x==3)
      {window.alert("x=3")}
else
      {window.alert("x<>3"}
```

Циклы

При написании программ часто возникает необходимость повторять одно действие несколько раз подряд, возможно с небольшими изменениями. В таких случаях используются циклы. Цикл состоит из двух основных частей: тела и заголовка. В заголовке цикла задается условие, при выполнении которого исполнение цикла закончится. Тело цикла – команды, заключенные в фигурные скобки и выполняемые по кругу, пока условие выхода из цикла не будет выполнено. Если тело цикла состоит всего из одной команды, фигурные скобки использовать необязательно. Каждое выполнение цикла называется итерацией.

Циклы FOR

В языке JavaScript предусмотрено два вида циклов. Первый из них цикл **FOR**, его структура показана в Листинг 5.5. Цикл **FOR** является «циклом со счетчиком». Так называются циклы, в которых есть переменная-счетчик, значение которой изменяется при каждой итерации цикла. При достижении счетчиком определенного значения выполнение цикла заканчивается. Выражение **нач_знач** задает начальное значение переменной-счетчика. Выражение **условие** – логическое выражение, становящееся ложным при определенном значении переменной-счетчика. Когда условие перестает выполняться, цикл завершается. Выражение **приращение** изменяет переменную-счетчик при каждой итерации цикла.

Листинг 5.5. Структура цикла FOR

```
for (нач_знач; условие; приращение)
{
  тело_цикла
}
```

Код реального документа HTML с циклом **FOR** приведен в Листинг 5.6. На Рис. 5.2 показано, что отобразит Web-браузер при открытии данной страницы.

Листинг 5.6. Пример цикла FOR

```
<html>
<head>
  <title>Цикл FOR</title>
</head>
<body>
  <script language="JavaScript">
  for (i=1;i<=10;i++){
   document.write("Это строчка номер " + i + "<br>")
```

```
  }
  </script>
</body>
</html>
```

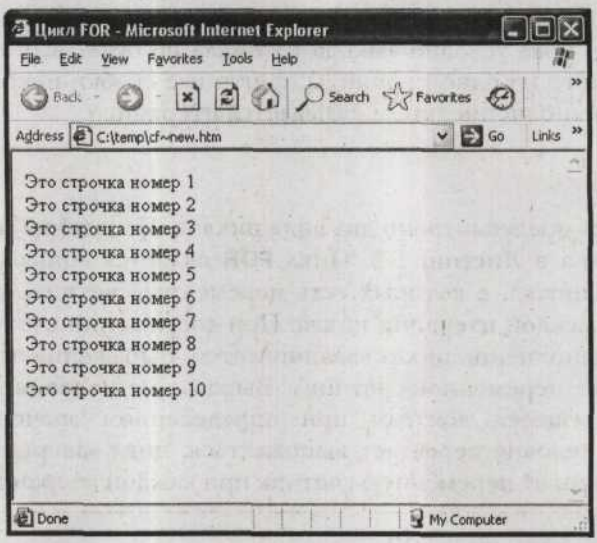


*Рис. 5.2. Отображение работы цикла **FOR** на странице Web-браузера*

Циклы WHILE

Другой вид циклов в языке JavaScript – циклы **WHILE**. В отличие от цикла **FOR**, в заголовке этого вида циклов осуществляется только проверка истинности условия. Структура цикла **WHILE** приведена в Листинг 5.7. Пока значение логического выражения **условие** равняется **True** (истина), выполнение цикла будет продолжаться. Если в качестве условия поставить выражение, истинное при любых обстоятельствах, например **2=2**, или просто **True**, то цикл будет выполняться бесконечно, или пока не будет принудительно прерван.

*Листинг 5.7. Структура цикла **FOR***

```
while(условие)
{
  тело_цикла
}
```

В Листинг 5.8 приведен пример цикла **WHILE**, выполняющего работу, аналогичную циклу **FOR** из Листинг 5.6. Обратите внимание, что задание начального значения переменной цикла и ее приращение осуществляется вне заголовка цикла.

Листинг 5.8. Пример цикла ***WHILE***

```
<html>
<head>
  <title>Цикл WHILE</title>
</head>
<body>
  <script language="JavaScript">
  i=1;
  while(i<=10){
   document.write("Это строчка номер " + i + "<br>")
   i++;
  }
  </script>
</body>
</html>
```

Команды, управляющие выполнением циклов

Для управления работой циклов используются команды **break** и **continue**. Команда **break** прерывает выполнение цикла и передает управление первой команде после цикла. Добавим к циклу **WHILE** из Листинг 5.8 условный переход, выполняющий команду **break**, при достижении переменной **i** значения 5. Результат приведен в Листинг 5.9. Как изменения отразились на получаемой **Web**-странице, видно на Рис. 5.3. При достижении переменной **i** значения 5, выполнение цикла принудительно прерывается.

Листинг 5.9. Пример использования команды ***break***

```
<html>
<head>
  <title>Цикл WHILE с командой break</title>
</head>
<body>
  <script language="JavaScript">
  i=1;
```

```
  while(i<=10){
   if (i==5)
     break
      document.write("Это строчка номер " + i + "<br>")
      i++;
  }
  </script>
</body>
</html>
```

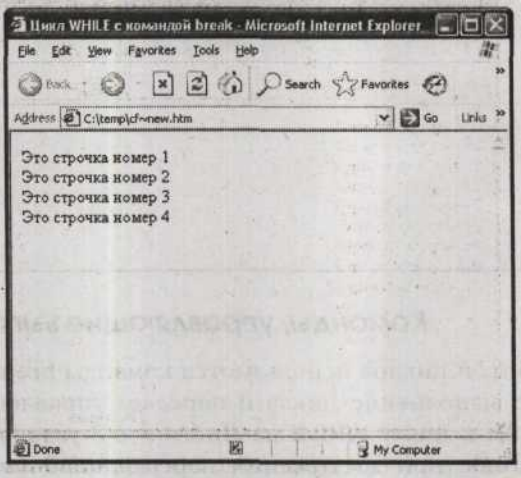


Рис. 5.3. Применение команды ***break***

Команда **continue** вызывает выполнение следующей итерации цикла, не заканчивая текущую. Добавим в код страницы в Листинг 5.6 условный переход, выполняющий команду **continue**, когда значение **i** достигнет значения 5. Полученный код показан в Листинг 5.10. В результате открытия такой страницы, браузер отобразит страницу, показанную на Рис. 5.4. На странице будет пропущена строка 5, поскольку команда **continue** запустила следующую итерацию цикла, не заканчивая текущую.

Листинг 5.10. Пример использования команды ***continue***

```
<html>
<head>
  <title>Цикл FOR</title>
</head>
<body>
```

```
  <script language="JavaScript">
  for (i=1;i<=10;i++){
   if (i == 5)
     continue
  document.write("Это строчка номер " + i + "<br>")
  }
  </script>
</body>
</html>
```

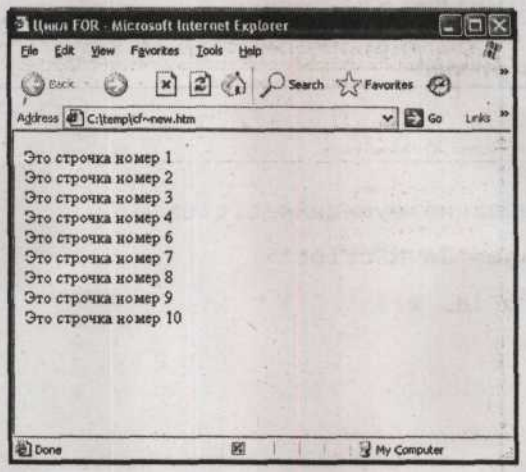


Рис. 5.4. Пример работы команды ***continue***

Функции

Последовательность из нескольких команд сценария можно объединить между собой, создав функцию. После этого команды функции можно выполнить, вызвав функцию по названию. В функции можно свести неоднократно используемые блоки кода чтобы вместо повторения кода, просто вызывать функцию. Кроме того, в функции можно вынести последовательности команд, дающие некий законченный результат, – это позволяет улучшить структуру программы.

Функции создаются по образцу, показанному в Листинг 5.11. **Имя_функции** – название, по которому функция будет вызываться в дальнейшем. **Параметры_функции** – переменные, значение которых передается в функцию. Внутри функции команды могут оперировать этими переменными.

Листинг 5.11. Структура функции

```
function имя_функции (параметры_функции)
{
```

```
  код_функции
}
```

В качестве результата своей работы функция может возвращать некоторое значение, которое можно присваивать переменным, использовать в вычислениях и т.д. Вызов функции происходит следующим образом: **имя_функции (параметры_функции)**. Можно присвоить значение функции переменной: **переменная = имя_функции (параметры_функции)**. Значение функции присваивается с помощью оператора **return**. Используется он так: **return значение_функции**.

В Листинг 5.12 показано практическое использование функции на примере функции, возводящей число **a** в целую положительную степень **b**. Код JavaScript страницы последовательно возводит двойку в степени от 0 до 10, а результаты работы можно увидеть на Рис. 5.5.

Листинг 5.12. Пример функции

```
<html>
<head>
  <title>Использование функций</title>
  <script language="JavaScript">
  function powerb (a, b){
   var i,atemp;
   atemp=a
   if (b>1){
     for (i=2;i<=b;i++)
      atemp*=a
   }
   if(b==0)
     atemp=1
   return atemp
  }
  </script>
</head>
<body>
  <script language="JavaScript">
  i=0;
  while(i<=10){
```

```
  document.write("2 в степени " + i + " равняется " + powerb(2,
i) + "<br>")
  i++;
  }
  </script>
</body>
</html>
```

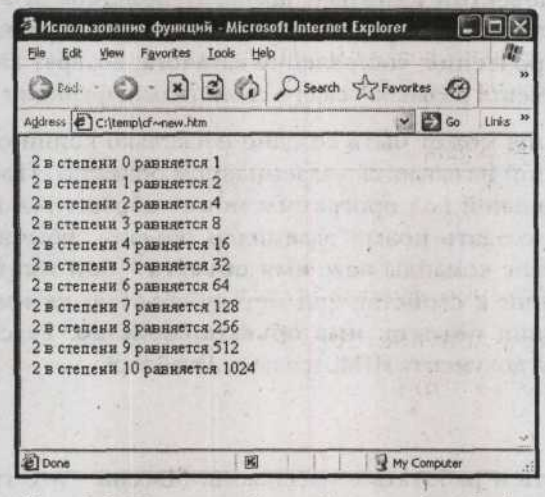


Рис. 5.5. Пример работы функции JavaScript

Обратите внимание на команду **var**. Эта команда создает переменные и задает их область действия. Если переменная создана командой **var** внутри функции, то и доступ к ней можно будет получить только внутри этой функции. Если ее создать в основной части сценария, то переменная будет доступна из любой части этого сценария, в том числе и из всех его функций. Явное задание переменных не является обязательным в языке JavaScript и используется в основном для того, чтобы разграничить зону действия переменных. Например, если в основной части сценария есть переменная **i**, а внутри функции используется переменная с таким же именем, то, если не задать в функции эту переменную явно, в функции будет использоваться переменная из основного сценария. В качестве эксперимента вы можете удалить команду **var** из документа в Листинг 5.12 и посмотреть, что получится в результате. При создании переменных командой **var**, можно задать их начальное значение, например: **var i=0, m="переменная"**.

Имена функций в JavaScript должны соответствовать тем же требованиям, что и имена переменных:

- Название может состоять из символов латинского алфавита от «a» до «z» как строчных, так и прописных, цифр от «0» до «9» и символов подчеркивания.

- Название функции не может начинаться с цифры. Название функции **_3abc** – правильное, а **3_abc** – уже нет.
- JavaScript различает строчные и прописные буквы в названии функции: **Abc** и **ABC** – две разные функции.

Объекты

Пришло время поближе познакомиться с объектами. Любой объект представляет собой контейнер, в который «сложено» некоторое количество переменных и функций, работающих с этими переменными. Проводя аналогию с обычным миром, представим объектом книжный шкаф. Переменными этого объекта будут сами книги, их количество и расположение на полке. Функциями – доставание книг с полки для прочтения, составление каталога, возврат книг обратно на полку. Переменные объекта называются его свойствами, функции – методами.

При работе сценария может быть создано несколько копий объектов одного типа, и каждая из копий называется «экземпляром объекта». После создания экземпляра объекта остальной код программы может обращаться к его методам и переменным. Чтобы создать новый экземпляр объекта, обычно используется команда **new**. Синтаксис команды **new: имя_объекта = new тип_объекта ([параметры])**. При обращении к свойству или методу объекта, их имена пишутся через точку после названия объекта: **имя_объекта.свойство**. Рассмотрим подробнее некоторые объекты документа HTML и языка JavaScript.

Объект Array

Позволяет создавать и работать с массивами. Массив – это группа переменных, объединенных под одним именем. Эти переменные называются элементами массива. Элементы массива пронумерованы от нуля и до некоего **N**, определяющего номер максимального элемента в массиве. Доступ к элементам массива осуществляется через указание названия массива и номера элемента в квадратных скобках: **название_массива[номер_элемента]**.

Новые массивы создаются при помощи команды new: **название_массива = new Array ([длина_массива])**. С помощью необязательного параметра **длина_массива** можно задать количество элементов в массиве. Нумероваться эти элементы будут от нуля до числа, на единицу меньшего значения выражения **длина_массива**. В Листинг 5.13 приведен пример использования массивов. В этом примере создается массив из 5 элементов и трем первым его элементам присваиваются значения.

Листинг 5.13. Пример использования массива

```
myArray = new Array (5)
myArray[0] = "мешок с цементом"
myArray[1] = "мешок с песком"
myArray[2] = "ведро с обойным клеем"
```

Если при создании массива не задавать количество его элементов, то будет создан массив нулевой длины. Но при задании значения некоторому элементу этого массива массив расширяется до этого номера. В программе из Листинг 5.14 создается массив нулевой длины, а затем, создается элемент этого массива под номером 99. Массив при этом автоматически расширяется до 100 элементов, от 0 до 99. С помощью свойства **length**, объекта **Array**, можно узнать количество элементов в массиве.

Листинг 5.14. Пример использования массива

```
myArray = new Array ()
myArray[99] = "сапог"
```

Объект Math

В этом объекте собраны различные математические функции и константы. Константы сохранены в свойствах объекта, доступ к математическим функциям осуществляется при помощи его методов.

Свойства объекта **Math**:

- **E**. Значение числа «е».
- **LN2**. Значение натурального логарифма числа 2.
- **LN10**. Значение натурального логарифма числа 10.
- **LOG2E**. Значение логарифма числа «е» по основанию 2.
- **LOG10E**. Значение десятичного логарифма числа «е».
- **PI**. Значение числа **пи**.
- **SQRT1_2**. Значение квадратного корня из 0.5
- **SQRT2**. Значение квадратного корня из 2.

Методы объекта **Math**:

- **abs**. Возвращает модуль аргумента. Синтаксис: **Math.abs(аргумент)**, **аргумент** – произвольное число.
- **acos**. Возвращает арккосинус аргумента. Синтаксис: **Math.acos(аргумент)**, **аргумент** – число от -1 до 1, если число выходит за эти границы, метод возвращает результат 0.
- **asin**. Возвращает арксинус аргумент. Синтаксис: **Math.asin(аргумент)**, **аргумент** – число от -1 до 1, если число выходит за эти границы, метод возвращает результат 0.
- **atan**. Возвращает арктангенс аргумент. Синтаксис: **Math.atan(аргумент)**, **аргумент** – число от -1 до 1, если число выходит за эти границы, метод возвращает результат 0.

- **ceil**. Возвращает значение аргумента, округленное в большую сторону. Синтаксис: **Math.ceil(аргумент)**, **аргумент** – произвольное число.
- **cos**. Возвращает косинус аргумента. Синтаксис: **Math.cos(аргумент)**, **аргумент** – произвольное число.
- **exp**. Возвращает экспоненту от аргумента. Синтаксис: **Math.exp(аргумент)**, **аргумент** – произвольное число.
- **floor**. Возвращает значение аргумента, округленное в меньшую сторону. Синтаксис **Math.floor(аргумент)**, **аргумент** – произвольное число.
- **log**. Возвращает натуральный логарифм аргумента. Синтаксис: **Math.log(аргумент)**, **аргумент** – любое положительное число.
- **max**. Возвращает значение: большего из двух аргументов. Синтаксис: **Math.max(аргумент1, аргумент2)**, **аргумент1, аргумент2** – произвольные числа.
- **min**. Возвращает значение: меньшего из двух аргументов. Синтаксис: **Math.min(аргумент1, аргумент2)**, **аргумент1, аргумент2** – произвольные числа.
- **pow**. Возводит число в произвольную степень. Синтаксис: **Math.pow(основание, степень)**. Число **основание** возводится в степень **степень**, **основание** и **степень** – произвольные числа.
- **random**. Возвращает случайное число в диапазоне от 0 до 1. Синтаксис: **Math.random()**.
- **round**. Возвращает значение аргумента, округленное по правилам округления. Синтаксис: **Math.round(аргумент)**, **аргумент** – произвольное число.
- **sin**. Возвращает синус аргумента. Синтаксис: **Math.sin(аргумент)**, **аргумент** – произвольное число.
- **sqrt**. Возвращает квадратный корень аргумента. Синтаксис: **Math.sqrt(аргумент)**, **аргумент** – неотрицательное число. В том случае, когда **аргумент** меньше нуля, метод возвращает значение 0.
- **tan**. Возвращает тангенс аргумента. Синтаксис: **Math.tan(аргумент)**, **аргумент** – произвольное число.

Объект window

Этот объект описывает открытое окно браузера и создается браузером при открытии нового документа HTML. Все объекты HTML-документа являются дочерними объектами объекта **window**. Для создания нового окна используется метод **open**, следующим образом: **название_окна=window.open ("URL", "имя_окна"[, "настройки_окна"])**.

- **название_окна** – название нового экземпляра окна.
- **URL** – адрес HTML-документа, открываемого в окне.

- **имя_окна** – название окна, используя которое можно обращаться к этому окну из тегов HTML.
- **настройки_окна** – дополнительные параметры, задаваемые для создаваемого окна. Параметры задаются в кавычках по парам: **название_параметра=значение_параметра**. Параметры перечисляются через запятую, например так: **"width=640,height=480,resize=no"**. Если при создании окна дополнительные параметры не задавать, то будут установлены параметры по умолчанию.

При создании нового окна ему можно задать такие параметры:

- **toolbar**. Панель инструментов. Может принимать значения **yes** и **no**. При значении **yes** у создаваемого окна будет панель инструментов.
- **location**. Адресная строка. Может принимать значения **yes** и **no**. При значении **yes** у создаваемого окна будет адресная строка.
- **status**. Строка статуса. Может принимать значения **yes** и **no**. При значении **yes** у создаваемого окна будет статусная строка.
- **menubars**. Строка меню. Может принимать значения **yes** и **no**. При значении **yes** у создаваемого окна будет строка меню.
- **scrollbars**. Полосы прокрутки. Может принимать значения **yes** и **no**. При значении **yes** у создаваемого окна будут полосы прокрутки в том случае, когда содержимое не будет помещаться в окне.
- **resizable**. Возможность изменения размеров окна. Может принимать значения **yes** и **no**. При значении **yes**, размеры созданного окна можно будет изменять.
- **width**. Ширина окна. Ширина задается в пикселах.
- **height**. Высота окна. Высота тоже задается в пикселах.

Команда **window.open("index.html", "win1", "width=640, height=480, resize=no, scrollbars=no")** создаст окно браузера размером 640x480 пикселов, без полос прокрутки и возможности изменения размера, содержащее HTML-документ **index.html**.

При обращении к методам, свойствам и дочерним объектам текущего окна, вместо названия экземпляра объекта **window** можно писать **window**, либо вообще ничего не писать. Записи **window.alert ()** и просто **alert ()**, вызывают один и тот же метод.

Методы объекта **window**:

- **alert**. Создает диалог с заданным текстом и кнопкой **OK** (Рис. 5.6). Синтаксис метода: **window.alert(сообщение)**, **сообщение** – текст, показываемый в диалоге.
- **close**. Закрывает окно. Синтаксис: **название_окна.close ()** название_окна – название экземпляра объекта **window**. Командой **window.close ()** или просто **close ()**, закрывается текущее окно.

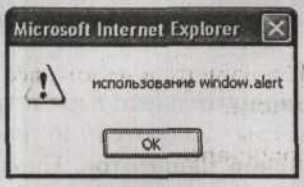


Рис. 5.6. Использование метода ***window.alert***

- **confirm**. Создает диалог с заданным текстом и кнопками **OK** и **Отмена** (Cancel), представленный на Рис. 5.7. Если пользователь щелкнет на кнопке **OK** этого диалога, то метод вернет значение **True** (Истина), если пользователь щелкнет на кнопке **Отмена** (Cancel), то метод вернет значение **False** (Ложь). Синтаксис метода: **window.confirm (сообщение)**, **сообщение** – текст, показываемый в диалоге.

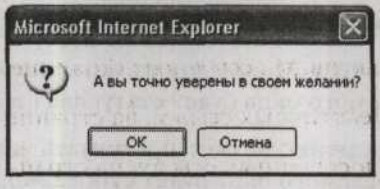


Рис. 5.7. Использование метода ***confirm***

- **open**. Открывает новое окно и создает новый экземпляр объекта **window**. Подробное описание этого метода уже давалось чуть выше в этой главе.
- **prompt**. Создает диалог с заданным текстом и полем ввода (Рис. 5.8). Синтаксис метода: **window.prompt (сообщение [,значение_по_умолчанию])**, **сообщение** – текст, показываемый в диалоге, **значение_по_умолчанию** – величина, являющаяся значением по умолчанию для поля ввода. Метод возвращает значение, введенное пользователем в поле ввода.

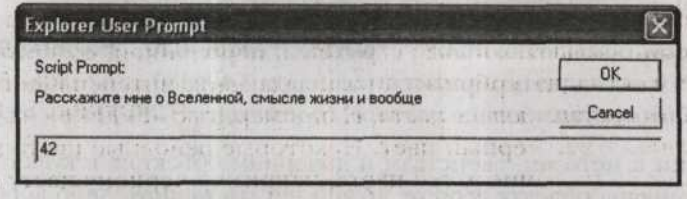


Рис. 5.8. Использование метода ***prompt***

Свойства объекта **window**:

- **defaultStatus**. Содержание строки статуса окна браузера по умолчанию, когда она не занята другими сообщениями, создаваемыми самим браузером или задаваемыми свойством **status**.
- **status**. Содержание статусной строки браузера. Является более приоритетным по сравнению со свойством **defaultStatus**.

Объект document

Объект, описывающий HTML-документ в целом, все элементы HTML-страницы являются его дочерними объектами.

Основные методы объекта **document**:

- **write**. Записывает произвольный текст в документ HTML. Этот метод позволяет генерировать с помощью JavaScript HTML-документы целиком, записывая в них, все теги и текст, включая сценарии JavaScript. Синтаксис: **document.write (текст)**, **текст** – текст, записываемый в документ HTML.
- **writeln**. Аналогичен методу **write**, с той лишь разницей, что после текста в документ HTML записывается символ перевода строки. Синтаксис: document.**writeln (текст)**, **текст** – текст, записываемый в документ HTML.

Основные свойства объекта **document**:

- **AlinkColor**. Цвет активных ссылок на странице.
- **LinkColor**. Цвет неактивных ссылок на странице.
- **VlinkColor**. Цвет посещенных ссылок на странице.
- **form**. Все формы документа.
- **BgColor**. Цвет фона страницы.
- **FgColor**. Цвет текста страницы.
- **LastModified**. Время последней модификации документа.
- **title**. Содержание заголовка страницы.
- **URL**. Полный URL документа.

Свойства **AlinkColor**, **LinkColor**, **VlinkColor**, **BgColor** и **FgColor** задают цвета документа. Цвета записываются в виде строковой переменной, содержащей в текстовом виде три шестнадцатеричных числа, задающих интенсивность красной, зеленой и голубой составляющих цвета. Например, так: «FFFFFF» – это белый цвет или так: «000000» – это черный цвет. Некоторые основные цвета можно задать, записав в свойство название этого цвета, например черному цвету соответствует значение «black». Приведем список названий некоторых основных цветов:

- **black**. Черный.
- **blue**. Голубой.
- **brown**. Коричневый.
- **cyan**. Синий.
- **gray**. Серый
- **green**. Зеленый.

- **magenta**. Малиновый.
- **orange**. Оранжевый.
- **pink**. Розовый.
- **red**. Красный.
- **turquoise**. Бирюзовый.
- **white**. Белый.
- **yellow**. Желтый.

Коллекция images

Коллекции представляют собой массивы, элементами которых являются объекты. Элементами коллекции **images** являются экземпляры объекта **image**. Объект **image** является объектом изображения в документе HTML. Коллекция **images** является дочерним объектом объекта **document** и содержит объекты всех изображений HTML-страницы. Свойство **length** коллекции **images**, позволяет узнать общее количество изображений на странице. Обращение к объектам конкретных изображений происходит как обращение к элементам массива **document.images[номер_изображения]**.

Объект **image** позволяет управлять практически всеми параметрами изображения на странице. У объекта **image** есть следующие свойства:

- **border**. Ширина границы изображения, в пикселов.
- **complete**. Логическая переменная, показывающая, загрузилось ли изображение. Когда изображение полностью загружено, принимает значение **True** (Истина).
- **height**. Высота изображения, в пикселов.
- **hspace**. Расстояние между краем изображения и текстом его окружающим, по горизонтали.
- **src**. Адрес, по которому располагается графический файл.
- **vspace**. Расстояние между краем изображения и текстом, его окружающим, по вертикали.
- **width**. Ширина изображения, в пикселов.

Изменяя параметры изображений на странице, можно создавать различные динамические эффекты. Следующий код, например, заменяет одну картинку на странице, на другую: **document.images[0].src="image.jpg"**.

Взаимодействие JavaScript и HTML

Код JavaScript сценариев, при использовании их на HTML-страницах, обычно делится на две части: определение функций выносится в заголовок страницы а вызываются они из тела страницы, при наступлении различных событий. При размещении в коде страницы, сценарии заключаются в контейнер **<SCRIPT></SCRIPT>**. С помощью атрибута **LANGUAGE** можно явно указать, на каком языке написан сценарий, заключенный в контейнер: **<SCRIPT LANGUAGE="JavaScript">**.

В целях дополнительной совместимости со старыми браузерами, которые не знают, что такое сценарии, перед кодом сценария часто вставляют: **<!--**, а после него: **-->**. Эта комбинация символов обозначает комментарий HTML – часть кода страницы, не предназначенная для рассмотрения браузером. Браузеры, не знающие о сценариях, благополучно пропускают этот участок, а браузеры, знакомые со сценариями, знают об этом трюке и обрабатывают содержимое таких комментариев. В Листинг 5.15 показано, как происходит внедрение кода JavaScript на HTML-страницу.

*Листинг 5.15. Внедрение **JavaScript** на **HTML** страницу*

```
<html>
<head>
  <title>В эту страницу внедрен код JavaScript </title>
  <script language="JavaScript">
  <!--
  function myFunc(parameters){
   ...
  }
  -->
  </script>
</head>
<body>
  <script language="JavaScript">
  <!--
  myFunc(parameters)
  -->
</body>
</html>
```

Вынесение функций в отдельный файл

Если одни и те же функции используются на нескольких страницах сайта, то целесообразно вынести определения функций в отдельный файл. Файл сценариев JavaScript обычно носит расширение **js**. В файл сценариев просто записываются определения функций, одной за другой. Чтобы использовать на странице HTML функции, определенные в файле сценариев, этот файл подключается к HTML-странице с помощью атрибута **SRC** тега **<SCRIPT>**: **<script language="JavaScript" src="main.js"></script>**. В приведенном теге к HTML-странице были подключены функции из файла **main.js**.

События и сценарии JavaScript

Как уже говорилось выше, сценарии JavaScript на HTML-страницах обычно связываются с наступлением различных событий. Реакцию на события можно добавлять практически ко всем тегам HTML-документа с помощью соответствующих атрибутов тега. В значении атрибута могут использоваться практически любые выражения JavaScript. Если необходимо на одно событие «повесить» несколько выражений, то они записываются через точку с запятой.

Событие onClick

Самым часто используемым событием, является событие **onClick** – щелчок мышью. В следующем примере мы добавим к тегу изображения **<IMG>** реакцию на щелчок мышью: **<IMG SRC="image.jpg" onClick= "openWin('балалайка')">**. После добавления атрибута **onClick** щелчок мышью на изображении будет приводить к выполнению функции **openWin** с аргументом «балалайка».

По правилам языка HTML значение атрибута **onClick** заключено в кавычки. Строковое значение, передаваемое функции, тоже должно быть заключено в кавычки. Поэтому, чтобы не возникло путаницы и программа работала правильно, аргумент функции заключен в одинарные кавычки. Этого правила следует придерживаться и в дальнейшем: если строка вызова функции заключена в кавычки, то с аргументами функции следует использовать кавычки другого типа.

Открытие нового окна браузера по щелчку мыши

Одним из самых частых случаев использования сценариев JavaScript является открытие новых окон браузера с заданными параметрами при щелчке на кнопке, или на ссылке. В Листинг 5.16 показано, как это можно делать. При щелчке мышью по ссылке **«щелкни меня»** будет выполняться функция **openwin**, открывающая новое окно браузера, лишенное практически всех элементов управления. Аргументами этой функции является название открываемого документа и размеры открываемого окна. Функция возвращает значение **False** (Ложь), которое передается тегу **<a>**. Получив от события **onClick** результат **False** (Ложь), тег **<a>** в ответ на щелчок мышью не открывает прописанную в нем ссылку.

Это сделано для совместимости с браузерами, не умеющими выполнять сценарии JavaScript и браузерами, в которых выполнение сценариев отключено. Работает

эта система таким образом: если браузер понимает сценарии, то срабатывает событие **onClick** и открывается окно заданного размера, с заданными свойствами. При этом работа тега **<A>** блокируется. Если же браузер не выполняет сценарии, то он игнорирует событие **onClick** и открывает окно с документом, прописанным в атрибуте **HREF** тега. Правда окно это будет произвольного размера и с настройками по умолчанию, но зато пользователь в любом случае получит доступ к нужной ему информации.

Листинг 5.16. Использование ***JavaScript*** *для открытия новых окон браузера*

```
<html>
<head>
  <script language="JavaScript" >
  <!--
  function openwin(w,par) {
   window.open (w,"dpw",par+"status=no, toolbar=no, menubar=no,
resizable = no, scrollbars=no, titlebar=no, location=no");
   return false;
  }
  -->
  </script>
  <title>Открытие окон с помощью JavaScript</title>
</head>
<body>
  <a href="newwin.html" target="_blank" onClick="return
openwin('newwin.html','width=670,height=360')">Щелкни меня</a>
</body>
</html>
```

Общий список возможных событий

Перечислим еще раз все возможные события, в ответ на которые можно запускать сценарии JavaScript:

- **onLoad** – окончание открытия HTML документа.
- **onUnload** – выгрузка HTML документа.
- **OnClick** – щелчок мышью по элементу.
- **OnDblClick** – двойной щелчок мышью по элементу.
- **OnMouseDown** – нажатие кнопки мыши.

- **OnMouseOver** – перемещение мыши на элемент из-за его пределов.
- **OnMouseMove** – перемещение мыши над элементом.
- **OnMouseOut** – перемещение мыши с элемента за его пределы.
- **OnFocus** – элемент получает фокус ввода.
- **OnBlur** – элемент теряет фокус ввода.
- **OnKeyPress** – пользователь нажимает и отпускает клавишу.
- **OnKeyDown** – пользователь нажимает клавишу над элементом.
- **OnKeyUp** – пользователь отпускает клавишу над элементом.
- **OnSubmit** – данные из формы переданы Web-серверу.
- **OnReset** – форма очищена.
- **OnSelect** – пользователь выбирает текст в текстовом окне.
- **OnChange** – элемент теряет фокус ввода, после изменения значения элемента.

Для удобства работы с объектом определенного тега HTML, можно присвоить ему имя. Это делается при помощи атрибута **NAME** тега, например так: **<IMG SRC="image.jpg" NAME="image1">**. После назначения имени тегу изображения к его объекту можно обращаться не только как к элементу коллекции images: **document.images[N]**, где **N** – номер изображения в документе, но и по имени: **document.image1**. Имя объекта можно подставить также вместо номера в коллекцию: **document.images["image1"]**.

Пример сценария JavaScript в HTML-документе

Для закрепления полученных знаний о языке JavaScript давайте сделаем небольшое приложение, выполняющее функцию калькулятора. Чтобы сделать удобный и красивый калькулятор, нам понадобится такой элемент HTML-страницы как форма.

Формы

Форма – это элемент страницы, позволяющий отправлять некие данные на сервер или передавать их сценариям JavaScript. Форма заключается в контейнер **<FORM></FORM>**. Составляющими частями формы являются поля ввода и кнопки, с помощью которых пользователь вводит данные. Подробнее о формах разговор пойдет в следующей главе, сейчас же рассмотрим ее элементы, необходимые нам для создания калькулятора.

Кнопки и поля ввода добавляются на HTML-страницу при помощи тега **<INPUT>**. Обязательным атрибутом тега **<INPUT>** является атрибут **NAME** – он задает название поля ввода. В зависимости от значения атрибута **TYPE** меняется тип поля ввода.

Стандарт HTML поддерживает достаточно много типов полей ввода, и более подробно мы рассмотрим их в следующей главе. Сейчас же нам понадобятся только два типа: **TEXT** и **BUTTON**.

Задав значение атрибута **TYPE** равным значению «TEXT», вы получите текстовое поле ввода (Рис. 5.9). С помощью этого поля вы будете вводить числа в программу калькулятора.

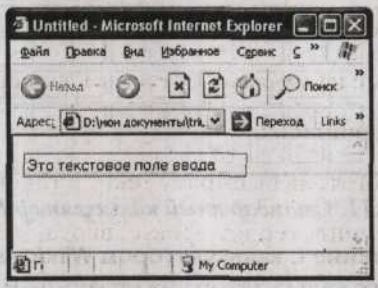


Рис. 5.9. Текстовое поле ввода в документе HTML

Для текстового поля ввода можно задать следующие атрибуты:

- **SIZE**. Этот атрибут устанавливает размер поля ввода текста.
- **MAXLENGTH**. Устанавливает максимальное количество символов, которые можно ввести в поле ввода текста или пароля.
- **VALUE**. Устанавливает значение поля по умолчанию.

Сделав значение атрибута **TYPE** равным значению «**BUTTON**», вы создадите кнопку (Рис. 5.10). Атрибут **VALUE** задает текст, отображаемый на кнопке.

Рис. 5.10. Кнопка в документе HTML

Принцип работы программы

Нажатиями на кнопки мы будем производить вычисления над числами. За основу калькулятора возьмем стандартный калькулятор Windows (Рис. 5.11). Чтобы было интереснее, используем инженерный его вариант с некоторыми упрощениями.

Чтобы получить представление о том, что в результате, запустите калькулятор и немного поработайте с ним. Чтобы запустить калькулятор, в главном меню Windows выберите команду **Все программы ♦ Стандартные ♦ Калькулятор**. Чтобы перевести калькулятор Windows в инженерный вид, выберите команду меню калькулятора **Вид ♦ Инженерный**.

Рис. 5.11. Стандартный калькулятор Windows

После того, как ознакомление с калькулятором Windows будет закончено, займемся созданием аналогичного калькулятора на странице HTML.

Основные части калькулятора

Наш калькулятор будет состоять из нескольких основных частей: области ввода чисел, области основных арифметических действий, области математических функций и области работы с памятью калькулятора. Для упрощения программы числа будут вводиться непосредственно в текстовом поле ввода, кнопок с цифрами делать не будем. Весь калькулятор будет оформлен в виде HTML-таблицы, с группами элементов в отдельных ячейках. Полученная, в итоге, страница будет выглядеть так, как показано на Рис. 5.12.

Рис. 5.12. Калькулятор, написанный на JavaScript

Костяк калькулятора составляет форма, в которую вставлена таблица с кнопками и полем ввода (Листинг 5.17). Необходимо связать действия с нажатием на кноп-

ки калькулятора. Для этого к кнопкам добавляется реакция на щелчок мышью событие **onClick**.

Листинг 5.17. Форма с элементами управления калькулятором

```
<form action="" name="calc" id="calc">
  <table cellspacing="2" cellpadding="2" border="0">
  <tr>
  <td colspan="3"><input type="text" name="numbers" value="0"
size="30" ><br></td>
  </tr><tr>
  <td valign="top">
  <input type="button" name="memosave" value="MS"<br>
  <input type="button" name="memorecall" value="MR"><br>
  <input type="button" name="memoclear" value="MC"></td>
  <td valign="top">
  <table cellspacing="3" cellpadding="0" border="0">
   <tr valign="top"><td>
   <input type="button" name="sin" value="sin"><br>
   <input type="button" name="cos" value="cos"><br>
   <input type="button" name="tg" value="tg"><br>
   <input type="button" name="Exp" value="Exp"><br>
   <input type="button" name="pow" value="x^y"><br></td>
   <td>
   <input type="button" name="sqr" value="x^2><br>
   <input type="button" name="ln" value="ln><br>
   <input type="button" name="fact" value="n"><br>
   <input type="button" name="inv" value="1/x><br>
   </td></tr>
  </table>
  </td>
  <td valign="top">
  <input type="button" name="divide" value><br>
  <input type="button" name="mult" value><br>
  <input type="button" name="minus" value><br>
  <input type="button" name="equals" value="=" ><br>
  <input type="button" name="clear" value="C" ><br>
  </td></tr>
  </table>
</form>
```

Форме присвоено имя **calc**, поле ввода носит название **numbers**. Поле является дочерним объектом объекта **form**, а тот, в свою очередь, является потомком объекта **document**. Значение поля, **value**, является свойством объекта поля. Поэтому обращение к значению поля записывается как **document.название_формы. название_поля.value**, а в данном случае, как **document.calc.numbers.value**.

Вычисление функций одного аргумента

Проще всего вычислять функции одного аргумента, например натуральный логарифм или возведение в квадрат. Достаточно в ответ на щелчок мышки присвоить полю значение, рассчитанное соответствующим методом объекта **Math**: **onClick = "document.calc.numbers.value = Math.log (document.calc.numbers.value)"**.

В объекте **Math** нет метода расчета факториалов, поэтому напишем свою функцию для их расчета, см. Листинг 5.18. Обратите внимание на начало функции. Здесь проверяется, является ли число, от которого берется факториал, целым. Если число равняется ему же, но округленному до целого, значит, оно и было целым, если же нет, то на экран выводится предупреждающий диалог. Если проверка пройдена, то производится вычисление факториала.

Листинг 5.18. Функция расчета факториалов

```
function factorial(n) {
if(Math.round(n)-n!=0){
  window.alert('факториалы можно брать только от целых чисел')
  return 0
}
else{
  if ((n == 0) || (n == 1))
   return 1
  else {
   result = (n * factorial(n-1) )
   return result
  }
}
}
```

Вычислений функций с двумя аргументами

Сложнее дело обстоит с операциями, в которых производятся действия с двумя аргументами, например умножение, или деление. Выполнение таких операций проводится в два этапа. Сначала пользователь вводит в поле ввода первое число, затем щелкает мышью на кнопке выбранной математической операции. В ответ на щелчок кнопка вызывает функцию, которая сохраняет значение поля ввода и выбранную операцию в специальные переменные. Затем пользователь вводит второе число и щелкает мышью на кнопке =. Кнопка вызывает функцию, которая берет из перемен-

ных первое число и название операции, из поля ввода берет второе число, проводит вычисления и возвращает результат. Как выглядят эти функции, показано в Листинг 5.19. Функция **storeop** сохраняет первый аргумент математической операции **variable** и тип операции **op** в переменных **x** и **oper**. Функция **total** производит вычисление результата математической операции с первым аргументом – **x** и вторым аргументом – **y**. Тип операции выбирается в зависимости от значения переменной **oper**.

Листинг 5.19. Вычисление функций с двумя аргументами

```
var x=0, oper=""
function storeop(variable, op){
x=variable
oper=op
return true
}
function total(y)
{
if (oper=='pow')
  return Math.pow(x,y)
if (oper=='div')
  return x/y
if (oper=='mult')
  return x*y
if (oper=='plus')
  return x+y
if (oper=='min')
  return x-y
if (oper=='')
  return 0
oper=''
}
```

Щелчок на одной из кнопок математических операций с двумя аргументами, приводит к выполнению функции **storeop** и очищению поля ввода: **<input type="button" name="mult" value="*" onClick="storeop(document.calc.numbers.value,'mult'); document.calc.numbers.value="">**. Щелчок на кнопке = вызывает функцию **total**, результаты ее выполнения записываются в поле ввода: **<input type="button" name="equals" value="=" onClick= "document.calc.numbers.value = total(document.calc.numbers.value)">**.

Работа с памятью

Последняя часть калькулятора – память. За эту часть отвечают две функции – **vsave** и **vrecall** и одна переменная – **tvalue**, см. Листинг 5.20. Функция **vsave** со-

храняет значение переданного ей аргумента в переменной **tvalue**. Функция **vrecall** возвращает значение этой переменной.

Листинг 5.20. Функции для работы с памятью калькулятора

```
var tvalue=0;
function vsave(v)
{
tvalue=v
return true
}
function vrecall()
{
return tvalue
}
```

При щелчке на кнопке записи в память **MS** вызывается функция **vsave**, ей передается текущее состояние поля ввода: **<input type="button" name="memosave" value="MS" onClick="vsave(document.calc.numbers.value)">**. Щелчок на кнопке восстановления из памяти, **MR**, записывает в поле ввода значение, возвращаемое функцией **vrecall**: **<input type="button" name="memorecall" value="MR" onClick = "document.calc.numbers.value = vrecall()">**. Щелчок на кнопке очищения памяти **MC** вызывает функцию **vsave** с нулем в качестве аргумента: **<input type="button" name="memoclear" value="MC" onClick="vsave(0)">**.

Общий код программы

Теперь, после того как рассмотрены все составляющие части калькулятора, можно записать его код целиком (см. Листинг 5.21).

*Листинг 5.21. Код калькулятора в документе **HTML***

```
<html>
<head>
<title>Калькулятор</title>
<script language="JavaScript">
<!--
var x=0, oper="", tvalue=0;
function factorial(n){
if(Math.round(n)-n!=0){
  window.alert('факториалы можно брать только от целых чисел')
  return 0
}
```

```
else{
 if ((n == 0) || (n == 1))
   return 1
 else {
   result = (n * factorial(n-1) )
   return result
 }
}
}
function storeop(variable, op){
x=variable
oper=op
return true
}
function total(y)
{
if (oper=='pow')
  return Math.pow(x,y)
if (oper=='div')
  return x/y
if (oper=='mult')
  return x*y
if (oper=='plus')
  return x+y
if (oper=='min')
  return x-y
if (oper=='')
  return 0
oper=''
}
function vsave(v){
tvalue=v
return true
}
function vrecall(){
return tvalue
}
-->
</script>
```

```
</head>
<body>
<h3>Калькулятор</h3>
<form action="" name="calc" id="calc">
  <table cellspacing="2" cellpadding="2" border="0">
  <tr>
  <td colspan="3"><input type="text" name="numbers" value="0"
size="30" ><br></td>
  </tr><tr>
  <td valign="top"><input type="button" name="memosave"
value="MS" onClick = "vsave (document.calc.numbers.value)"><br>
  <input type="button" name="memorecall" value="MR" onClick =
"document.calc.numbers.value=vrecall()"><br>
  <input type="button" name="memoclear" value="MC"
onClick="vsave(0)">
  </td><td valign="top">
  <table cellspacing="3" cellpadding="0" border="0">
   <tr valign="top"><td>
   <input type="button" name="sin" value="sin" onClick =
"document.calc.numbers.value =
Math.sin(document.calc.numbers.value)"><br>
   <input type="button" name="cos" value="cos" onClick =
"document.calc.numbers.value =
Math.cos(document.calc.numbers.value)"><br>
   <input type="button" name="tg" value="tg" onClick =
"document.calc.numbers.value =
Math.tan(document.calc.numbers.value)"><br>
   <input type="button" name="Exp" value="Exp" onClick =
"document.calc.numbers.value =
Math.exp(document.calc.numbers.value)"><br>
   <input type="button" name="pow" value="x^y" onClick =
"storeop(document.calc.numbers.value,'pow'); docu-
ment.calc.numbers.value=''"><br>
   </td><td>
   <input type="button" name="sqr" value="x^2" onClick =
"document.calc.numbers.value *= docu-
ment.calc.numbers.value"><br>
   <input type="button" name="ln" value="ln" onClick =
"document.calc.numbers.value =
Math.log(document.calc.numbers.value)"><br>
```

ременным можно присваивать сразу готовые значения, результаты выполнения каких-либо функций, результаты вычислений и значения других переменных.

При присвоении одной переменной значения другой переменной передается как значение, так и тип этой переменной. Например, если переменной массива вы присвоите числовое значение, то вместо массива вы получите число.

Проверка существования переменной

При передаче значений в сценарий PHP извне интерпретатор автоматически создает в сценарии переменные с именами передаваемых параметров. Часто возникает необходимость убедиться, был ли передан тот или иной параметр, т. е. была ли создана соответствующая переменная. Для этой цели используется функция **isset(имя_переменной)**. В том случае, если переменная задана, функция возвращает результат **True** (Истина).

Удаление переменных

После того как переменная исполнила свою роль, желательно ее удалить, чтобы освободить занимаемую память. Это делать необязательно, но если ваша программа оперирует большими объемами информации, то значительное количество переменных, хранящих массу неиспользуемых данных, может существенно замедлить работу сервера, особенно если учесть тот факт, что одновременно может выполняться несколько копий вашей программы, отдельно для каждого из посетителей сайта.

Если же ваш сценарий невелик, содержит всего несколько переменных и выполняется недолго, то уничтожением переменных можно пренебречь, поскольку после окончания работы сценария вся занимаемая память автоматические освобождается.

Переменные в языке PHP удаляются с помощью функции **unset(имя_переменной)**. После выполнения этой команды из памяти стираются все упоминания о переменной, и программа продолжает выполняться так, будто этой переменной и вовсе не существовало.

Мы рассмотрели все основные операции, которые можно проделать с любыми переменными, вне зависимости от их типов. Теперь же перейдем к операциям, касающимся только определенных типов переменных. Начнем с арифметических операций.

Математические операции

Математические вычисления проводятся при помощи стандартных математических операторов сложения, вычитания, деления, умножения и скобок : «+,-,/,*,()», вычисления проводятся в согласии с правилами арифметики. Например так: **2*4+1**. Результатом выполнения этого выражения будет число 9. А в результате вычисления выражения **$m=(3+1)*2;**, переменная **$m** примет значение 8.

Вычисления могут производиться и над значениями числовых переменных. Например, если значение переменной **$exp** равняется 25, то в результате выполнения выражения **$res=$exp/5;**, в переменную **$res** будет записано число 5.

ность_к_регистру – необязательный параметр логического типа, указывающий, нужно ли учитывать различия между прописными и строчными буквами при работе с именем константы. При значении параметра **чувствительность_к_регистру,** равном **True** (Истина), при обращении к константе необходимо учитывать регистр символов, то есть прописные и строчные символы считаются разными символами.

Пример задания константы показан в Листинг 7.11. Обратите внимание на следующие особенности этой программы: при задании константы ее имя указывается в кавычках, а при вызове константы ее имя указывается без кавычек и без знака «$», в отличие от вызова переменных.

Листинг 7.11. Пример задания константы

```
<?
define("pi",3.1415926,true);
echo pi;
?>
```

Проверка существования константы

Если возникает необходимость проверить, существует ли та или иная константа в программе, используется функция **defined(имя константы)**. Имя константы указывается в кавычках, например: **defined("pi")**. Если константа существует, функция возвращает значение **True** (Истина).

Стандартные константы PHP

Кроме констант, создаваемых в тексте программы, в PHP существует некоторый набор стандартных констант, создаваемых вне выполняемого сценария:

- **__FILE__**. Имя выполняемого в данный момент сценария. Обратите внимание, перед и после слова **FILE** находятся по два знака подчеркивания.
- **__LINE__**. Номер строки сценария, обрабатываемой интерпретатором PHP в данный момент.
- **PHP_OS**. Название и версия операционной системы, под которой выполняется PHP.
- **PHP_VERSION**. Версия языка PHP.

Операции с переменными

Оператор присваивания

Основным действием, совершаемым с переменными, является присвоение переменным некоторых значений. Для этого используется оператор присваивания, «=». В языке PHP он используется точно так же, как и в языке JavaScript. Имя переменной указывается слева от знака равно, а присваиваемое ей значение справа от него. Пе-

- **is_bool(переменная)** – если тип переменной **bool**, то функция возвращает значение **True** (Истина).
- **gettype (переменная)** – возвращает строку с названием типа переменной. Возможные варианты: «**integer**», «**double**», «**string**», «**array**», «**bool**» и «**unknown type**». Последний вариант возникает в том случае, если интерпретатор PHP не может определить тип данных.

В том случае, когда необходимо явно задать тип переменной, используется функция **settype(переменная, тип_переменной)**. **тип_переменной** – это название типа данных из тех, что возвращает функция **gettype (переменная)**. Например, команда **settype(a, "bool")**, установит тип переменной **$a** в значение **bool**. Если функция **settype(переменная, тип_переменной)** не может перевести переменную **переменная** в тип **тип_переменной**, то она возвращает значение **False** (Ложь).

Автоматическое преобразование типов

Одной из полезных особенностей языка PHP является возможность автоматического преобразования типов. Например, если есть две текстовые переменные, содержащие некоторые числовые значения, то при их перемножении результат примет числовой тип. Такая программа приведена в Листинг 7.10. Переменные **$a** и **$b** имеют текстовый тип данных, поскольку им присваиваются значения в кавычках. Но при попытке перемножить две текстовые строки в этих переменных, ошибки не возникает, поскольку интерпретатор PHP автоматически преобразует текстовые строки в числа и вычисляет результат.

Листинг 7.10. Пример автоматического преобразования типов

```
<?
$a="15";
$b="20";
echo $a*$b;
?>
```

Константы

Кроме переменных, в языке PHP используются константы. Константа – это некоторое постоянное значение, которому присвоено имя. В обычной жизни константой является, например, число p(Пи). Отличие констант от переменных в том, что, будучи однажды заданными, константы остаются неизменными на протяжении всего выполнения программы.

Задание констант

Задаются константы с помощью функции **define(имя_константы, значение_константы, чувствительность_к_регистру)**. **имя_константы** – это название, которое будет носить константа, **значение_константы** – ее значение, а **чувствитель-**

- Имя любой переменной в тексте программы начинается со знака «**$**», причем технически этот знак не является частью имени переменной, а просто помогает PHP отличить переменную от прочего текста программы. Если значение переменной передается программе PHP извне, например через CGI, то в вызове CGI, в именах переменных знак «**$**» указывать не надо.
- В принципе, имя переменной в PHP может состоять из любых символов, кроме пробелов, включая буквы русского языка. Но это может привести к ряду различных несовместимостей с другими программами, различными кодировками языков и т. д. Поэтому возьмите за правило ограничиваться в именах переменных только символами латинского алфавита, цифрами и знаком подчеркивания.
- Строчные и прописные буквы в названиях переменных PHP отличаются друг от друга, **$var** и **$Var** являются разными переменными.

Типы переменных

Переменные в PHP могут быть нескольких различных типов. При первом использовании переменной интерпретатор PHP самостоятельно решает, к какому типу отнести переменную, но, тем не менее, в некоторых случаях возникает необходимость задать переменной тип явно. Как это делается, мы рассмотрим чуть позже.

Сейчас же рассмотрим основные типы переменных в PHP:

- **integer**. Целые числа. Такие переменные могут содержать целые числа, в диапазоне значений от **-2147483648** до **2147483648.**
- **double** либо **float**. Вещественные числа.
- **string**. Текстовые строки. Для языка PHP этот тип данных является очень важным, поскольку большинство сценариев PHP занимаются именно обработкой строк данных.
- **array**. Массив. В отличие от языка JavaScript, массивы в PHP являются не объектами, а еще одним типом переменных.
- **bool**. Логические переменные. Могут принимать одно из двух значений: **True** (*Истина*) или **False** (*Ложь*).

Определение типа переменной и изменение его

Чтобы определить тип переменной, в языке PHP предусмотрено несколько функций:

- **is_int(переменная)** либо **is_integer(переменная)** – если тип переменной **integer**, то функция возвращает значение **True** (Истина).
- **is_double(переменная)** либо **is_float(переменная)** – если переменная вещественная, то функция возвращает значение **True** (Истина).
- **is_string(переменная)** – если тип переменной **string**, то функция возвращает значение **True** (Истина).
- **is_array(переменная)** – если тип переменной **array**, то функция возвращает значение **True** (Истина).

```
<body>
   <?echo "<h1>Hello, $any_name";?>
   </h1>
</body>
</html>
```

Результат выполнения этой программы будет таким же, как и результат неперепи-санной программы. Код, находящийся между угловыми скобками «<?» и «?>», обрабатывается PHP как код сценария, а остальной код страницы PHP передает браузеру в неизменном виде.

Комментарии

Важной частью сценариев PHP являются комментарии, которые позволяют оставлять в тексте программы пометки и замечания, дающие возможность позже вспомнить важные детали функционирования программ. Также комментарии позволяют временно исключить из программы часть команд, не удаляя их совсем.

Комментарии в языке PHP создаются тремя различными способами. Первые два аналогичны друг другу. Все символы строки, идущие за знаком «#» либо за знаком «//», PHP не обрабатывает и не передает браузеру. Третий способ создания комментариев окружение комментируемого текста знаками «/*» и «*/». Текст, находящийся между этими знаками, тоже считается комментарием. Преимущество третьего способа в том, что, во-первых, он позволяет создавать многострочные комментарии, а, во-вторых, может закомментировать не только конец какой-либо строки, но и середину или начало строки. Примеры комментариев всех трех типов приведены в Листинг 7.9.

Листинг 7.9. Примеры комментариев

```
# Комментарий
// Тоже комментарий
/* А этот комментарий
может быть многострочным
и длинным */
echo("Hello, world"); // Комментарий в конце строки
/* Комментарий в начале строки */ echo("Привет, мир");
```

Переменные

Поговорим несколько подробнее о переменных в языке PHP. Как и практически в любом языке программирования, в PHP существуют определенные правила задания переменных:

```
echo "<h1>Hello, $any_name";
echo "</h1></body></html>";
?>
```

Оформление программ PHP

Внимательнее рассмотрим созданную нами программу. Начинается она символами «<?», а заканчивается символами «?>», команды программы отделяются друг от друга точкой с запятой. Так оформляются все PHP-программы. Тело программы состоит из трех команд **echo**. Эта команда, как вы скорее всего уже догадались, служит для вывода HTML-кода, показываемого затем браузером. Выводимая строка заключается в кавычки: **echo "выводимая_строка"**. Команды программы отделяются друг от друга точкой с запятой. Первая команда программы **echo "<html><head><title>My first page</title></head><body>";**, выводит заголовок HTML-страницы.

Вторая команда программы, строка **echo "<h1>Hello, $any_name";**, выводит приветствие. Запись **$any_name** обозначает переменную, значение которой передается сценарию через адресную строку браузера с помощью механизма CGI. Переменные в языке PHP записываются со знаком «$» перед названием переменной. Завершается рассматриваемая нами программа третьей командой **echo**, выводящей последнюю часть HTML-кода, **echo "</h1></body></html>";** и знаком окончания PHP-сценария «**?>**».

Внедрение сценариев PHP в документы HTML

Только что нами был рассмотрен сценарий, формирующий HTML-страницу командами вывода **echo**, но существует и другой способ создания кода HTML с помощью PHP. Как мы уже отмечали раньше, отличительной чертой языка PHP является возможность встраивания PHP-сценариев прямо в код HTML-страниц. При этом отпадает необходимость выводить большую часть страницы с помощью команды **echo**, достаточно создавать с помощью сценария только те части кода HTML, которые могут изменяться. Перепишем программу из Листинг 7.7 так, чтобы сценарий PHP отвечал только за изменяющуюся часть страницы. В данном случае это будет строка с переменной, получаемой из адресной строки браузера. Переписанная программа показана в Листинг 7.8.

*Листинг 7.8. Программа **PHP**, внедренная в код **HTML***

```
<html>
<head>
   <title>
   My first  page
   </title>
</head>
```

Браузер отобразит HTML-страницу с приветствием (Рис. 7.25). Код получаемой HTML-страницы приведен в Листинг 7.6. Просмотреть его можно, выполнив команду меню браузера **Вид ♦ Просмотр HTML-кода** (View ♦ Source).

Рис. 7.25. Выполнение программы ***Hello, world***

Листинг 7.6. ***HTML****-код, генерируемый* ***PHP****-сценарием*

```
<html>
<head>
<title>My first  page</title>
</head>
<body>
<h1>Hello, World</h1>
</body>
</html>
```

Передача параметров из документа HTML в сценарий PHP

Немного усложним программу. Будем передавать ей значение переменной через адресную строку браузера, чтобы она передавала привет не всему миру, а одному конкретному человеку, имя которого мы передадим сценарию. Код измененной программы приведен в Листинг 7.7. Сохраните новый вариант кода вместо старого и наберите после этого в адресной строке браузера адрес **http://localhost/hello.php?any_name=Man**. В этой строке вы, через интерфейс CGI, передаете PHP-сценарию значение переменной **any_name=Man**. Напоминаем вам, что параметры, передаваемые сценариям через CGI методом GET, записываются в адресной строке браузера через знак «?» после названия сценария. Если параметров несколько, то они перечисляются через знак «&». Изменяя значение переменной в строке адреса, вы будете изменять и получаемую HTML-страницу.

Листинг 7.7. Программа, получающая значение переменной из адресной строки браузера

```
<?
echo "<html><head><title>My first  page</title></head><body>";
```

Основы языка PHP

Итак, все необходимые программы установлены и настроены. Можно приступать к изучению основ создания серверных сценариев на языке PHP и их практическому применению.

История PHP

Вначале, несколько слов о языке PHP. Язык программирования PHP специально разработан для создания серверных сценариев и сочетает достоинства таких языков программирования, как Perl и C. Одной из отличительных черт языка PHP является возможность встраивания программ PHP в код HTML-страниц наряду с обычными тегами.

История PHP началась в 1995 году, когда Расмус Лердорф (Rasmus Lerdorf) написал сценарий на языке Perl для подсчета посетителей его домашней страницы. Множество людей заинтересовались этим сценарием и Лердорф начал бесплатно раздавать его. Соответствующий инструментарий для работы со сценарием он назвал **Personal Home Page** (Личная домашняя страница), сокращенно PHP. Успех PHP побудил Лердорфа к развитию своего детища, и система начала быстро развиваться. В 1997 году было принято решение, что PHP отныне означает не **Personal Home Page**, а PHP Hypertext Processor (PHP – процессор гипертекста). На данный момент PHP является очень популярным средством написания серверных сценариев, позволяющим быстро создавать динамические Web-сайты.

При обращении браузера к сценарию PHP сервер запускает программу – интерпретатор языка PHP, которая преобразует текст сценария в специальный код (называемый byte-кодом), позволяющий выполнять сценарий быстрее, а затем приступает к его выполнению.

Первая программа на языке PHP

По традиции, начнем изучение языка PHP с создания программы, выводящей на экран сообщение «**Hello, world**». Код этой программы показан в Листинг 7.5. Сохраните эту программу под названием **hello.php** в каталоге **c:\www\html**, который мы определили как корневой каталог для HTML-документов на сервере. После этого, набрав в адресной строке браузера адрес **http://localhost/hello.php**, выполните сценарий.

*Листинг 7.5. Первая программа на **PHP***

```
<?
echo "<html><title>My first  page</title><body>";
echo "<h1>Hello, World";
echo "</h1></body></html>";
?>
```

Эти строчки подключают PHP к серверу Apache. Завершив настройку сервера, перезапустите его.

- Щелкните левой кнопкой мыши на иконке  в области уведомлений панели задач, появится меню.
- Выберите в появившемся меню пункт **Apache2 ◆ Restart** (Apache2 ◆ Перезапустить). Сервер будет перезапущен.

Проверка настроек PHP

Проверим правильность установки PHP. Создайте файл **test.php**, содержимое которого приведено в Листинг 7.4 и сохраните его в каталоге **c:\web\html**. После этого, введите в адресной строке браузера адрес **http://localhost/test.php**. В ответ вы должны увидеть в окне браузера страницу информации о PHP и сервере (Рис. 7.24). Это результат выполнения команды **phpinfo**();. Обратите внимание, что сценарий PHP вызывается не как CGI-программа, а как обычный документ HTML. Это одна из особенностей языка PHP, о которой мы поговорим чуть позже.

Листинг 7.4. Тестовый файл ***PHP***

```
<?
phpinfo();
?>
```

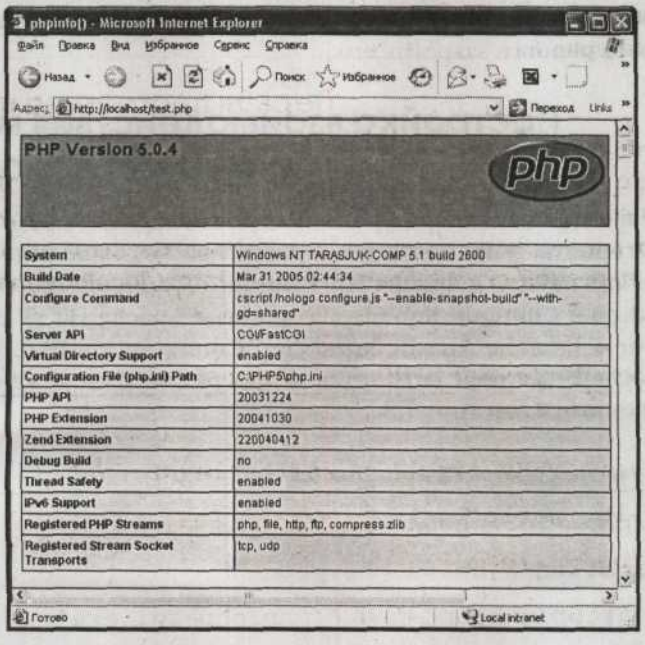


Рис. 7.24. Тестовая программа PHP

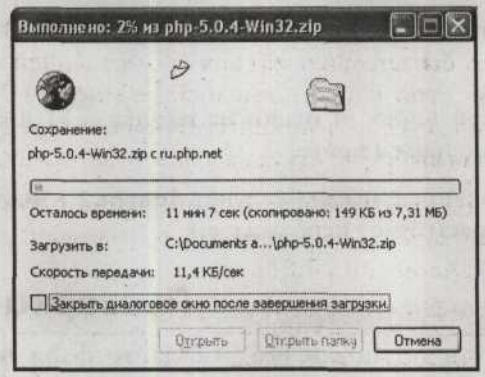


Рис. 7.23. Диалог, показывающий процесс загрузки дистрибутива

Настройка PHP

По окончании процесса загрузки дистрибутива PHP необходимо его установить на компьютер и настроить. Создайте каталог **C:\PHP5** и распакуйте в него загруженный **zip**-архив. В каталоге **C:\PHP5** будут созданы два файла настроек **php.ini-dist** и **php.ini-recommended**. Это два варианта настроек языка PHP. Выберем вариант **php.ini-recommended**. Переименуйте этот файл в **php.ini** и откройте для редактирования.

Файл настроек PHP начинается с команды **[PHP]**, строки комментариев начинаются со знака «;», параметры настроек приведены в виде **параметр=значение_параметра**. Найдите в файле настроек параметр **doc_root** и измените его значение на **doc_root="c:\web\html"**, чтобы указать средствам PHP каталог HTML-файлов. Сохраните файл настроек **php.ini** и закройте его.

Настройка взаимодействия между PHP и Web-сервером Apache

Программа PHP настроена, теперь нужно настроить сервер Apache на совместную работу с PHP. Откройте файл настроек сервера Apache, выполнив команду меню **Пуск** (Start) **Все Программы ♦ Apache HTTP Server ♦ Configure Apache Server ♦ Edit the Apache httpd.conf Configuration file** (Programs ♦ Apache HTTP Server ♦ Configure Apache Server ♦ Edit the Apache httpd.conf Configuration file). Файл настроек Web-сервера Apache **httpd.conf** откроется в программе Блокнот. Добавьте в него строчки, приведенные в Листинг 7.3.

*Листинг 7.3. Настройки сервера **Apache** для работы с **PHP***

```
ScriptAlias /php/ "c:/php5/"
AddType application/x-httpd-php.php
Action application/x-httpd-php "/php/php-cgi.exe"
```

- Щелкните мышью на ссылке **PHP X.X.X zip package**, где **X.X.X** – номер версии PHP, чтобы загрузить zip архив с последней версией PHP. На момент написания этой книги последней версией PHP была 5.0.4, именно она и рассматривается в этой книге. Откроется следующая страница сайта, предлагающая выбрать, откуда вы будете загружать архив (Рис. 7.20).
- Щелкните мышью на одной из ссылок, например на **ru.php.net**, чтобы приступить к загрузке. Откроется диалог, спрашивающий, что делать с загружаемым файлом (Рис. 7.21).

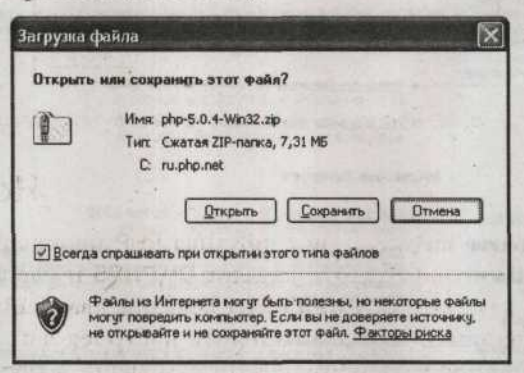


Рис. 7.21. *Диалог, спрашивающий, что делать с загружаемым файлом*

- Щелкните мышью на кнопке **Сохранить** (Save), откроется диалог **Сохранить как** (Save As), предлагающий выбрать каталог на вашем компьютере, в котором вы будете хранить получаемый файл (Рис. 7.22).

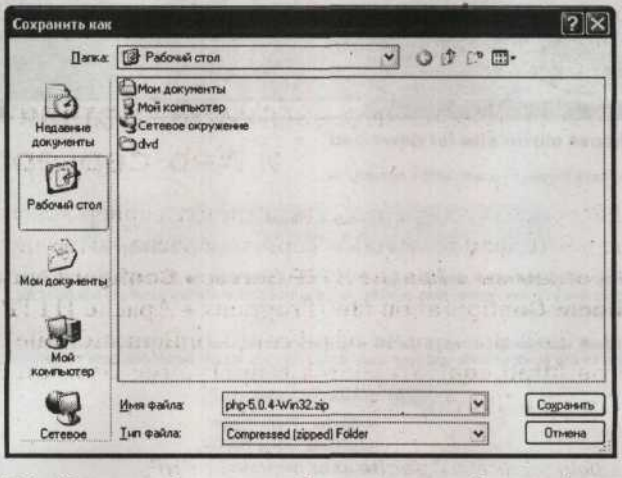


Рис. 7.22. *Диалог выбора папки для сохранения дистрибутива PHP*

- Выберите подходящий каталог и щелкните мышью на кнопке **Сохранить** (Save). Загрузка будет начата (Рис. 7.23).

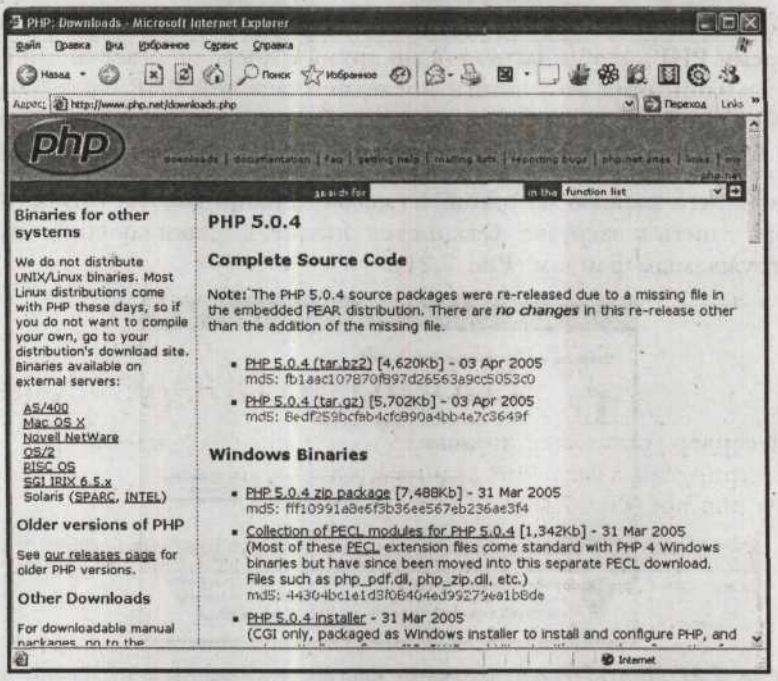


Рис. 7.19. Страница скачивания дистрибутивов PHP

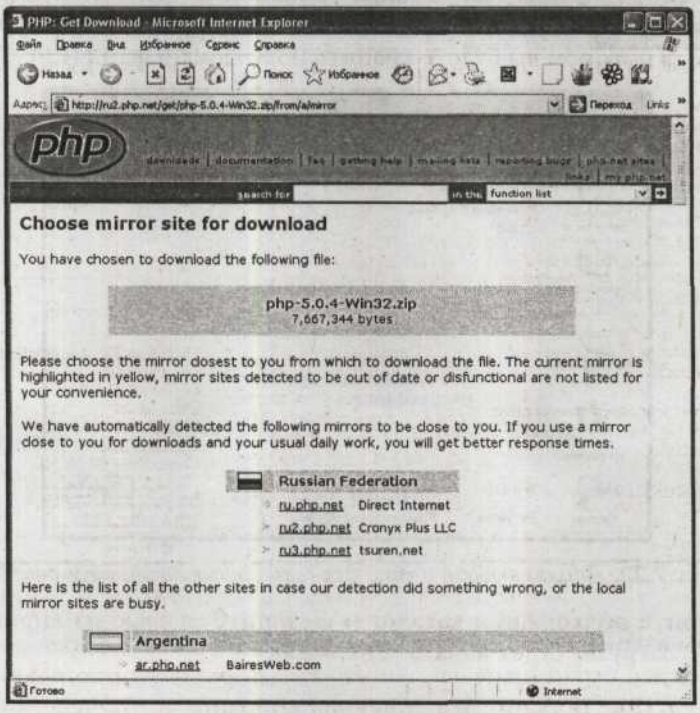


Рис. 7.20. Страница выбора зеркала с дистрибутивом PHP

*Листинг 7.2. Тестовый **CGI**-сценарий*

```
@echo off
echo Content-type: text/html
echo.
echo.
echo Hello!
```

Работа со средствами PHP

Итак, Web-сервер установлен, пришло время подключать к нему средства работы с PHP. Дистрибутив языка PHP скачивается с официального сайта программы, **http://www.php.net** (Рис. 7.18).

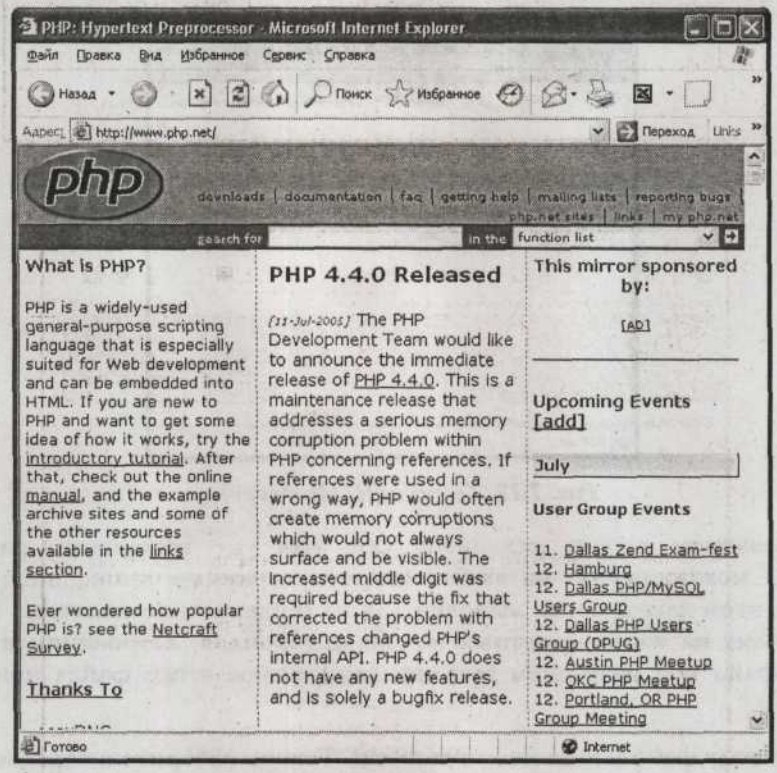


Рис. 7.18. Официальный сайт PHP

- Чтобы загрузить дистрибутив языка PHP, введите в адресную строку браузера адрес страницы загрузки файлов **http://www.php.net/downloads.php** (Рис. 7.19). Версии для Windows собраны в разделе, озаглавленном **Windows Binaries** (Бинарные Windows-файлы).

*Листинг 7.1. Тестовая **HTML**-страница*

```
<html>
<head>
    <title>Просто тест</title>
</head>
<body>
Web-сервер работает!
</body>
</html>
```

Наберите в адресной строке браузера адрес **http://localhost/main.html**. Если вы правильно изменили настройки сервера, вы увидите страницу, отображенную на Рис. 7.17.

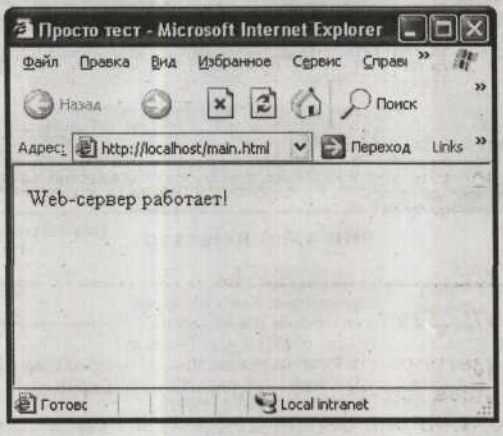


*Рис. 7.17. Тестовая **HTML**-страница*

Теперь проверим работу механизма CGI. Как уже говорилось ранее, CGI-программы можно писать на любом языке программирования, лишь бы программы на этом языке могли выполняться на серверном компьютере. Напишем CGI-программу на языке пакетных командных файлов, встроенном в Windows. Создайте файл под названием **test.bat**, содержимое этого файла приведено в Листинг 7.2.

Сохраните этот файл в каталоге **c:\web\cgi**. Теперь наберите в адресной строке браузера адрес **localhost/cgi/test.bat**. Результатом выполнения сценария должна стать строчка «Hello!» в окне браузера. Поздравляю вас, вы создали и выполнили свой первый CGI-сценарий.

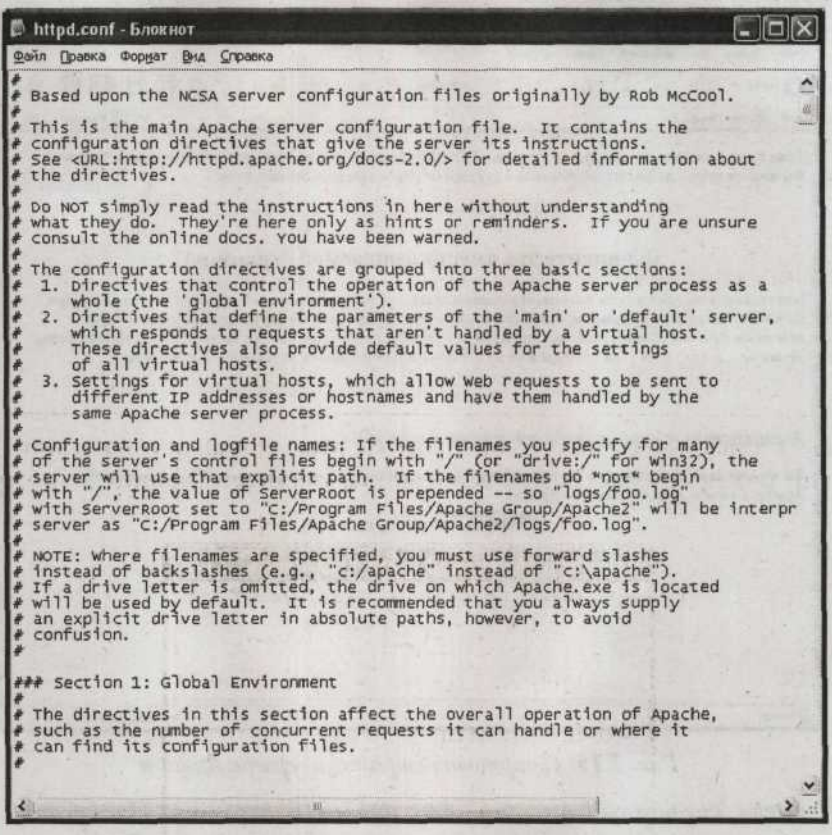


Рис. 7.16. Программа Блокнот с файлом настроек сервера Apache

Добавьте в файл настроек **httpd.conf** еще два параметра **ScriptAlias /cgi/ "c:/web/cgi"** и **ScriptAlias /cgi-bin/ "c:/web/cgi/"**. Этим вы создадите два «псевдонима» для каталога **c:\web\cgi**. Сервер будет автоматически подставлять в адресные запросы вместо строк **http://localhost/cgi-bin** и **http://localhost/cgi** строку **c:/web/cgi**. Чтобы Web-сервер воспринимал в качестве CGI-сценариев программы форматов **.exe .bat .cgi**, необходимо чтобы в файле настроек присутствовала строка **AddHandler cgi-script .bat .exe .cgi.**

Сохраните файл **httpd.conf** и перезапустите сервер. Чтобы перезапустить сервер:

- Щелкните левой кнопкой мыши на иконке  в области уведомлений панели задач, появится меню.
- Выберите в появившемся меню пункт **Apache2 ♦ Restart** (Apache2 ♦ Перезапустить). Сервер будет перезапущен.

Проверка настроек сервера

Проверим, правильно ли мы изменили настройки сервера. Создайте в каталоге **c:\web\html** файл **main.html**, содержание которого приведено в Листинг 7.1.

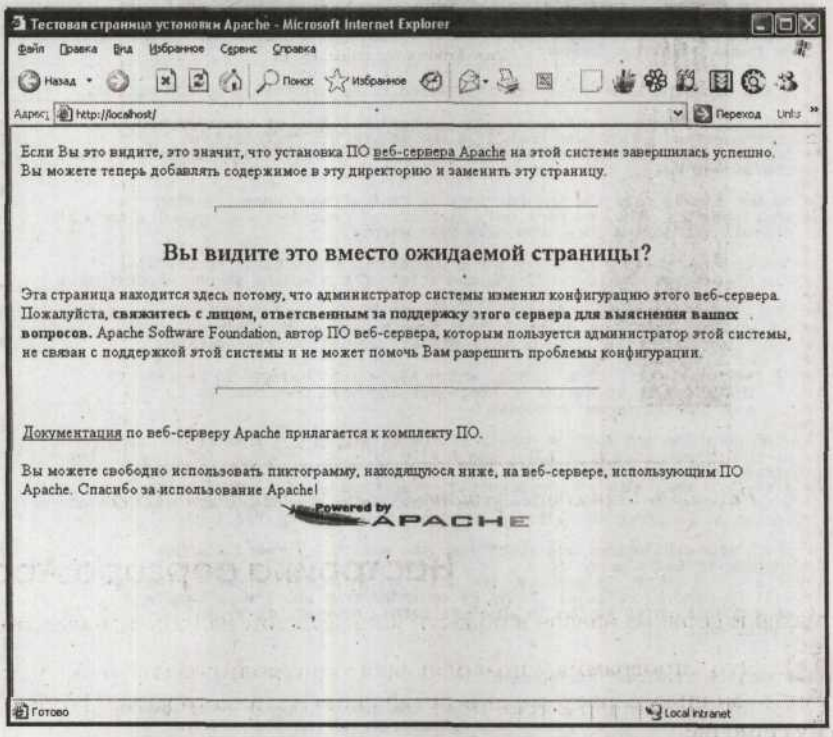


Рис. 7.15. Стартовая страница сервера Apache

Теперь, когда сервер установлен, его нужно настроить. Настройка сервера Apache осуществляется редактированием текстового файла **httpd.conf**, находящегося в каталоге **c:\Program Files\Apache Group\Apache2\conf**. Чтобы открыть его для редактирования, выполните пункт меню **Пуск** (Start) **Все Программы ♦ Apache HTTP Server ♦ Configure Apache Server ♦ Edit the Apache httpd.conf Configuration file** (Programs ♦ Apache HTTP Server ♦ Configure Apache Server ♦ Edit the Apache httpd.conf Configuration file). Файл **httpd.conf** откроется в программе Блокнот (Рис. 7.16). Все настройки сервера описываются строками этого файла в формате **параметр_настройки значение_параметра**. Строки файла, начинающиеся со знака «#», являются комментариями и настройками сервера не управляют.

По умолчанию корневым каталогом сайта установлен каталог **C:\Program Files\Apache Group\Apache2\htdocs**, что не очень удобно. Поэтому заменим его более удобным, например **C:\web**. Создайте этот каталог, а в нем два подкаталога **cgi** и **html**. В каталоге **cgi** будут находиться **CGI**-сценарии, а в каталоге **html – HTML**-файлы.

Найдите в файле **httpd.conf** параметр **DocumentRoot** и установите его новое значение: **DocumentRoot c:/web/html**. Этим вы укажете серверу расположение корневого каталога HTML-документов. Обратите внимание, что при указании пути к каталогу в файле настройки Apache используются знаки «/», а не «\», как обычно в Windows.

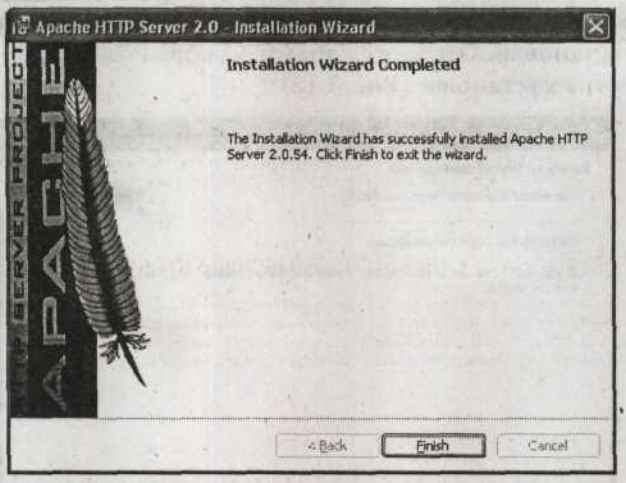


Рис. 7.14. Финальный установочный диалог сервера Apache

Настройка сервера Apache

После установки сервера Apache в области уведомлений на панели задач появится иконка  . Это программа, позволяющая контролировать работу сервера Apache. С ее помощью сервер можно останавливать и запускать. Чтобы остановить работу сервера:

- Щелкните левой кнопкой мыши на иконке  , появится меню.
- Выберите в появившемся меню пункт **Apache2 ◆ Stop** (Apache2 ◆ Остановить). Сервер будет остановлен, иконка сменится на другую –  .

Чтобы снова запустить сервер:

- Щелкните левой кнопкой мыши на иконке  , появится меню.
- Выберите в появившемся меню пункт **Apache2 ◆ Start** (Apache2 ◆ Запустить). Сервер будет снова запущен, иконка сменится на другую –  .

Проверим правильность установки сервера. Введите в адресной строке браузера адрес **http://localhost**. Если сервер правильно установился, браузер откроет стартовую страницу сервера Apache (Рис. 7.15).

- Щелкните мышью на кнопке **Next** (Далее), чтобы перейти к следующему диалогу установки. Откроется диалог, сообщающий о том, что Web-сервер Apache готов к установке (Рис. 7.12)

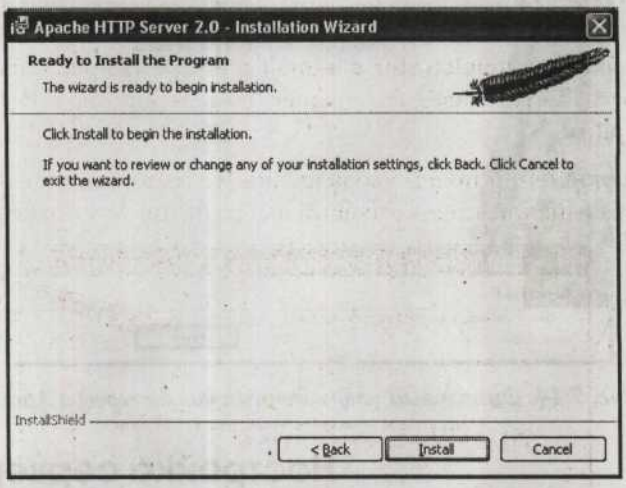


Рис. 7.12. *Диалог, предлагающий начать процесс установки сервера*

- Чтобы начать этот процесс, щелкните мышью на кнопке **Install** (Установить). Появится диалог, показывающий ход установки (Рис. 7.13).

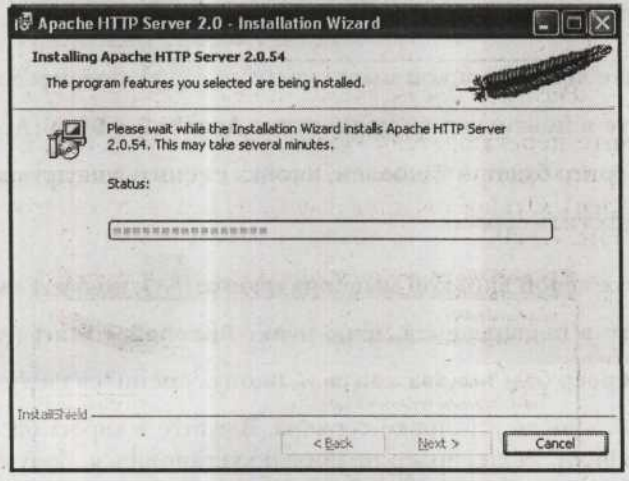


Рис. 7.13. *Диалог, показывающий процесс установки сервера*

- После окончания процесса установки появится последний установочный диалог (Рис. 7.14).
- Щелкните мышью на кнопке **Finish** (Завершить). Web-сервер Apache установлен.

торому смогут получать доступ другие пользователи сети Интернет, можете указать произвольное название, например **user.ru**.

- В поле ввода **Server Name** (Имя сервера) введите название сервера, например **www.user.ru**.
- В поле ввода **Administrator's e-mail address** (Адрес электронной почты администратора) введите адрес электронной почты, например **user@mail.ru**.
- После заполнения полей ввода щелкните мышью на кнопке **Next** (Далее). Откроется диалог, предлагающий выбрать тип установки (Рис. 7.10).

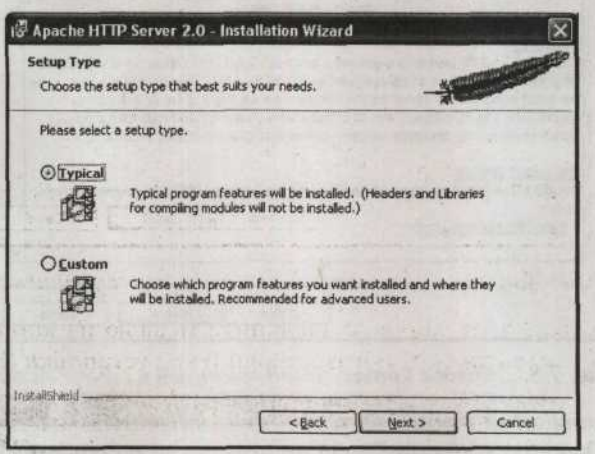


Рис. 7.10. Диалог выбора типа установки сервера

- Установите переключатель **Please select a setup type** (Укажите тип установки) в положение **Typical** (Обычный) и щелкните мышью на кнопке **Next** (Далее). Откроется диалог выбора папки, в которую устанавливается Apache (Рис. 7.11).

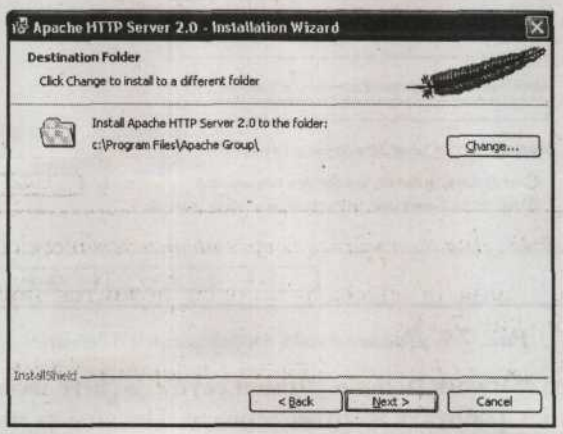


Рис. 7.11. Диалог выбора установочной папки для сервера Apache

- Чтобы подтвердить свое согласие с требованиями лицензионного соглашения, установите переключатель в положение **I accept the terms in the license agreement** (Я принимаю условия лицензионного соглашения) и щелкните мышью на кнопке **Next** (Далее). Откроется третий диалог установщика Apache. Этот диалог предоставляет краткую информацию о сервере Apache (Рис. 7.8).

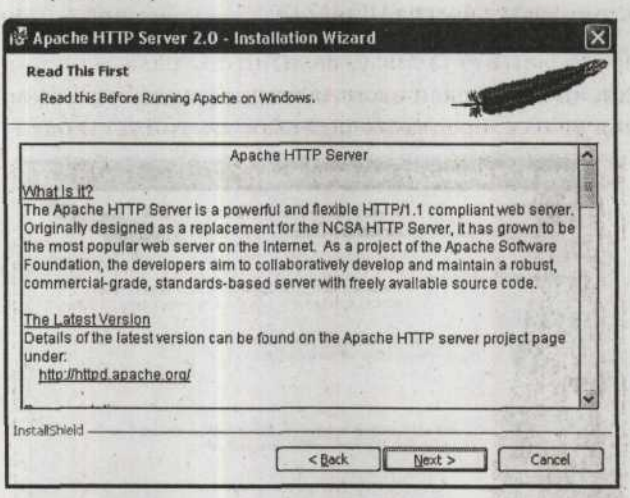


Рис. 7.8. Диалог с краткой информацией о сервере Apache

- Для продолжения процесса установки, щелкните мышью на кнопке **Next** (Далее). Появится диалог ввода информации о сервере (Рис. 7.9).

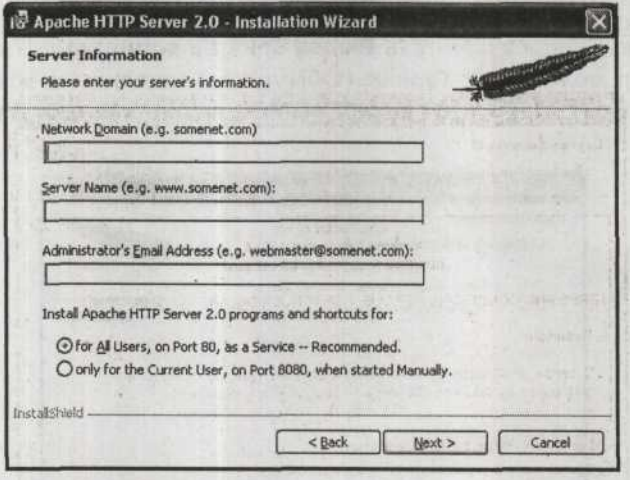


Рис. 7.9. Диалог ввода информации о сервере

- В поле ввода **Network Domain** (Домен сети) введите название домена сети, который будет работать на этом сервере. Поскольку мы не будем использовать ваш компьютер в качестве полноценного Интернет-сервера, к ко-

Установка сервера Apache

По окончании процесса загрузки Web-сервера необходимо его установить на компьютер.

- Запустите установочный файл **apache_2.0.54-win32-x86-no_ssl.msi**, чтобы приступить к процессу установки сервера Apache. Откроется первый диалог установщика Apache (Рис. 7.6).
- Чтобы продолжить установку, щелкните мышью на кнопке **Next** (Далее). Откроется второй диалог установщика Apache, показывающий лицензионное соглашение пользователя программы (Рис. 7.7).

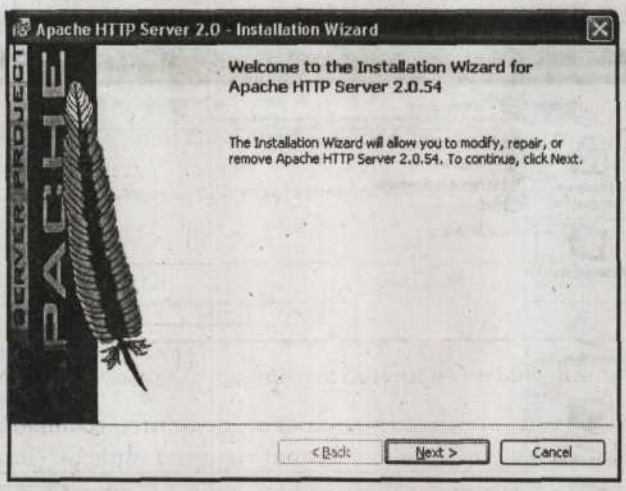


Рис. 7.6. Первый диалог установщика Web-сервера Apache

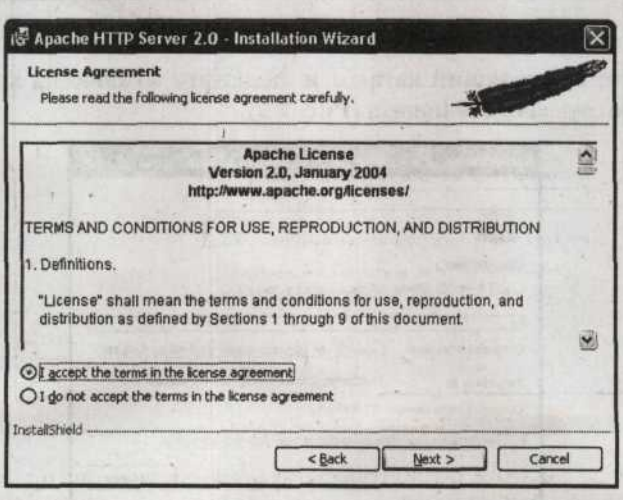


Рис. 7.7. Диалог с лицензионным соглашением Web-сервера Apache

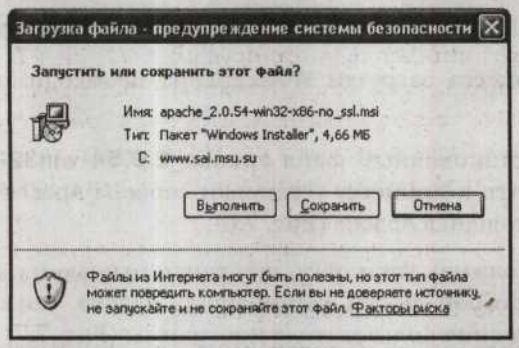


Рис. 7.3. Диалог, спрашивающий, что делать с загружаемым файлом

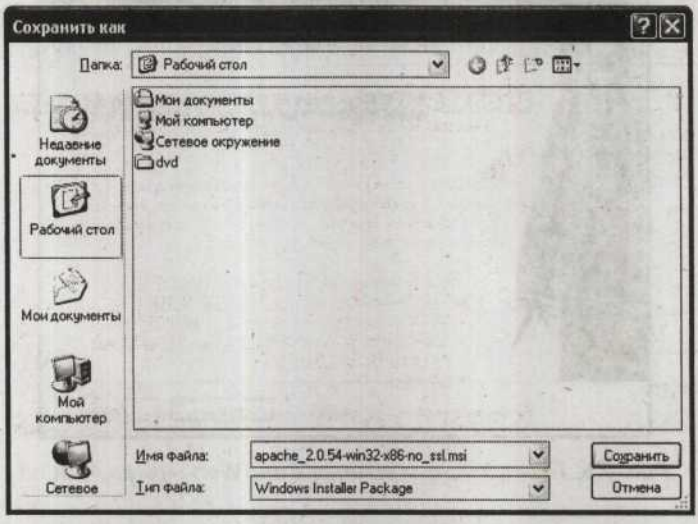


Рис. 7.4. Диалог ***Сохранить как*** *(Save as)*

- Выберите подходящий каталог и щелкните мышью на кнопке **Сохранить** (Save). Загрузка будет начата (Рис. 7.5).

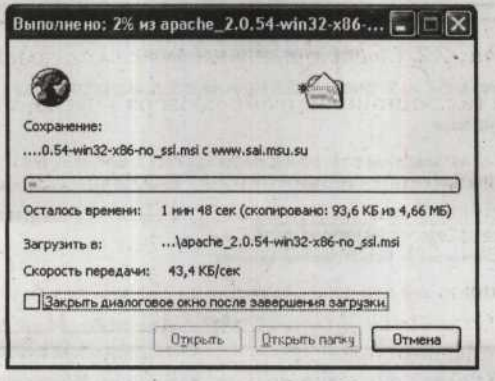


Рис. 7.5. Диалог скачивания файла

- Введите в адресной строке браузера адрес страницы загрузки сервера Apache – **http://httpd.apache.org/download.cgi** (Рис. 7.2). Ссылка на лучшую из существующих версий будет находиться под заголовком **Apache X.X.XX is the best available version** (Apache X.X.XX – лучшая из существующих версий). На месте букв **X** будет номер версии. На момент написания этой книги такой версией была 2.0.54, на нее мы и будем ориентироваться при рассмотрении сервера Apache. Под заголовком с номером версии сервера будут представлены несколько ссылок для загрузки различных вариантов Web-сервера Apache. Нас интересует версия для инсталляции под Windows: **Win32 Binary (MSI Installer): apache_2.0.54-win32-x86-no_ssl.msi**.

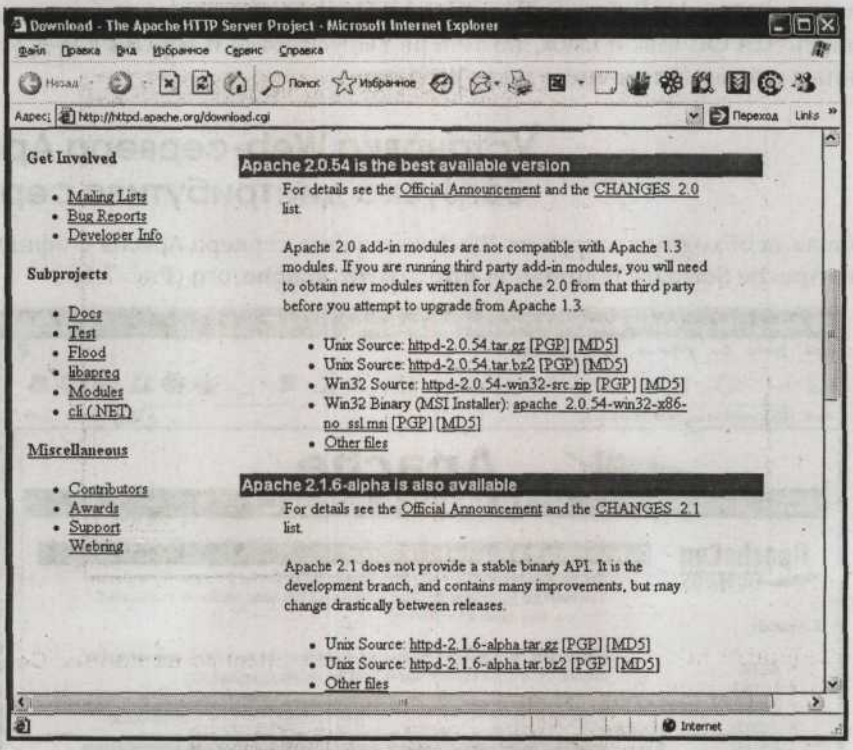


Рис. 7.2. Страница скачивания сервера Apache

Чтобы загрузить инсталляционный файл сервера Apache, выполните следующую пошаговую процедуру.

- Щелкните мышью на ссылке **apache_2.0.54-win32-x86-no_ssl.msi**, чтобы приступить к скачиванию программы. Появится диалог, спрашивающий, что делать с загружаемым файлом (Рис. 7.3).

- Щелкните мышью на кнопке **Сохранить** (Save), чтобы сохранить файл на жестком диске. Откроется диалог **Сохранить как** (Save as), предлагающий выбрать папку на вашем компьютере, в которую нужно сохранить получаемый файл (Рис. 7.4).

for Supercomputer Applications – Национальный центр по применению суперкомпьютеров). В 1994 группу, создававшую Web-сервер NCSA, покидает главный разработчик, и на этом история разработки этого Web-сервера заканчивается.

Но со временем начали появляться расширения и дополнения к этому серверу, называемые патчами (patch – заплатка). И вот, в апреле 1995 года выходит первая версия Web-сервера Apache, основанного на сервере NCSA, с добавлением всех патчей, существовавших на тот момент. Само название Apache является производной от слова patch – «A PatCHy».

Сейчас сервер Apache является самостоятельной разработкой и поддерживается организацией Apache Software Foundation. Первые версии этого сервера разрабатывались для ОС Unix и Linux, но теперь выпускаются и версии для других операционных систем, в том числе и для Windows.

Установка Web-сервера Apache

Загрузка дистрибутива сервера

Для начала необходимо загрузить Windows-версию сервера Apache с официального сайта Apache Software Foundation **http://www.apache.org** (Рис. 7.1).

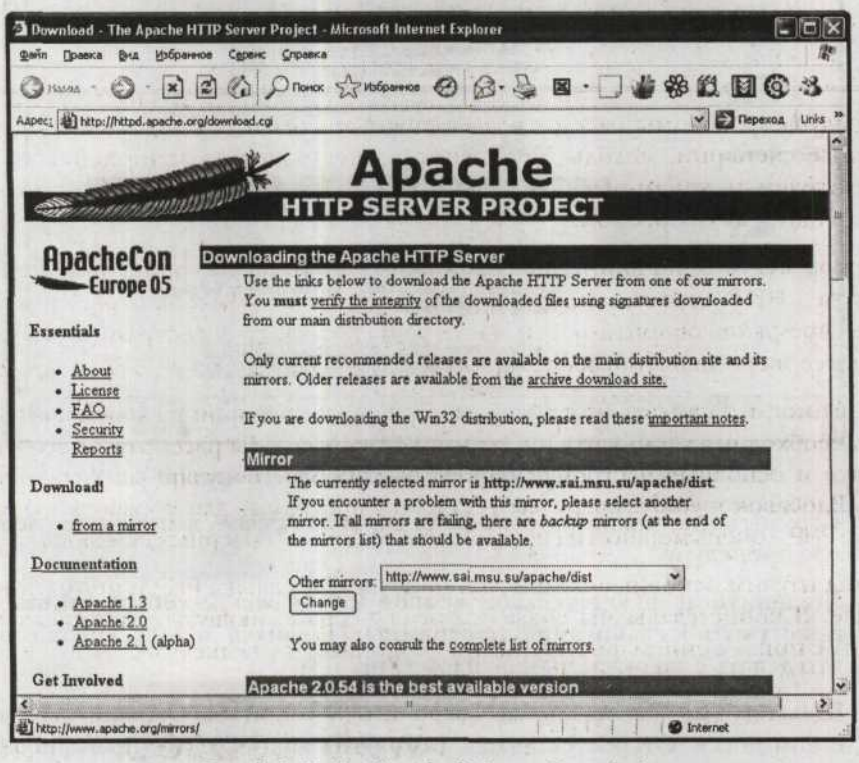
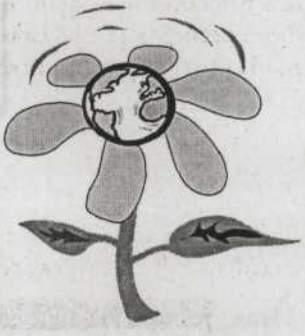


Рис. 7.1. Сайт Apache Software Foundation

глава 7

Настоящий профессиональный Web-сайт



В предыдущей главе мы рассмотрели основные случаи, в которых используются серверные сценарии, методы, которыми осуществляется взаимодействие между пользователем и динамическим Web-сайтом, узнали о том, как HTML-страницы могут «общаться» с сервером.

До сих пор мы уделяли внимание в основном клиентской стороне динамического Web-сайта – HTML-страницам, браузеру, действиям посетителя страницы. Теперь пришло время поговорить о той части динамических Web-страниц, за которую отвечает сервер – серверных сценариях.

Чтобы вы могли запускать и отлаживать серверные сценарии на своем компьютере, на него необходимо установить программу Web-сервер. Мы рассмотрим особенности установки и основные настройки одного из наиболее популярных Web-серверов – Apache. Вдобавок к Web-серверу мы установим программу для обработки сценариев на языке PHP – очень мощном и гибком языке создания серверных сценариев.

Затем мы изучим основные команды и конструкции языка PHP и применим их на практике. В конце главы мы создадим реально работающую динамическую Web-страницу с применением форм HTML и сценариями на языке PHP.

Web-сервер Apache

Первоначально, сервер Apache разрабатывался в Национальном центре разработок Суперкомпьютеров Иллинойского университета и назывался NCSA (National Center

Щелкнув мышью на кнопке **Войти**, посетитель передаст введенные логин и пароль сценарию **cgi-bin/reg_in**. Сценарий сверится с базой данных и, если найдет в ней подходящую пару из логина и пароля, то перезагрузит главную страницу сайта уже с учетом настроек данного конкретного пользователя. Если же пароль не подойдет или в базе данных не окажется нужного логина, сценарий сформирует HTML-страницу, предлагающую пройти процедуру регистрации.

Рис. 6.15. Форма для входа на сайт

Заключение

Итак, теперь вы знаете, каким именно образом происходит взаимодействие между браузером и серверными сценариями, как осуществляется передача данных от пользователя на сервер и создание **HTML**-страниц серверными сценариями. Вы познакомились с основными случаями, в которых применяются серверные технологии создания динамических Web-страниц, и рассмотрели практический пример применения этих технологий. Вы хорошо поработали и теперь обладаете достаточным багажом знаний и умений, чтобы приступить к созданию ваших собственных серверных сценариев.

В следующей главе мы познакомимся с самым популярным Web-сервером – Apache, установим его на компьютер и займемся созданием серверных сценариев на языке PHP. После этого освоение технологий создания Web-сайтов можно будет считать в общем и целом законченным.

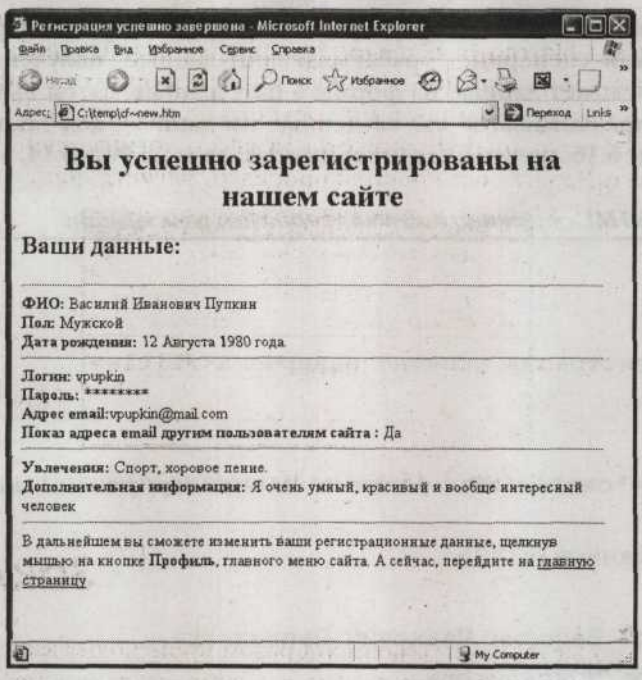


Рис. 6.14. Страница с итогами регистрации

На главной странице сайта размещается форма, введя в которую свои логин и пароль, пользователь может активизировать свои регистрационные данные. Код такой формы приведен в Листинг 6.17, ее внешний вид показан на Рис. 6.15.

Листинг 6.17. *Форма входа на сайт для зарегистрированного пользователя*

```
<html>
<head>
  <title>Вход на сайт</title>
</head>
<body>
  <form action="cgi-bin/reg_in" method="get">
   <b>Логин:</b> <input type="text" name="login" size="10"
max-length="10"> <b>
   Пароль:</b> <input type="password" name="pass" size="10"
maxlength="10">
   <input type="submit" value="Войти">
  </form>
</body>
</html>
```

человек. Браузер, на основе содержимого адресной строки генерирует запрос к серверу, вызывая сценарий, обрабатывающий данные, полученные из формы. Сценарий записывает данные из формы в базу данных, затем выводит посетителю страницу, показывающую все введенные им данные, код такой страницы показан в Листинг 6.16, полученная страница показана на Рис. 6.14.

*Листинг 6.16. **HTML**-страница, созданная сценарием регистрации*

```
<html>
<head>
  <title>Регистрация успешно завершена</title>
</head>
<body>
  <h1 align="center">Вы успешно зарегистрированы на нашем сай-
те</h1>
  <h2>Ваши данные:</h2>
  <hr>
  <b>ФИО:</b> Василий Иванович Пупкин<br>
  <b>Пол:</b> Мужской<br>
  <b>Дата рождения:</b> 12 Августа 1980 года.<br>
  <hr>
  <b>Логин:</b> vpupkin<br>
  <b>Пароль:</b> ********<br>
  <b>Адрес email:</b>vpupkin@mail.com<br>
  <b>Показ адреса email другим пользователям сайта :</b>  Да<br>
  <hr>
  <b>Увлечения:</b> Спорт, хоровое пение.<br>
  <b>Дополнительная информация:</b> Я очень умный, красивый и
вообще интересный человек<br>
  <hr>
  В дальнейшем вы сможете изменить ваши регистрационные данные,
щелкнув мышью на кнопке
  <b>Профиль</b>, главного меню сайта. А сейчас, перейдите на  <a
href="http://www.mysite.com/index.html">главную страницу</a>
</body>
</html>
```

```
    <option value="Настольные игры">Настольные игры</option>
    <option value="Азартные игры">Азартные игры</option>
    <option value="Рисование">Рисование</option>
    <option value="Хоровое пение">Хоровое пение</option>
  </select><br><br>
  <b>Дополнительная информация:</b>
<br>   <textarea cols="40" rows="8"
name="dop"></textarea><br>
  <hr>
  <input type="submit" value="Зарегистрировать"> <input
type="reset" name="Reset" value="Отменить ввод">
 </form>
</body>
</html>
```

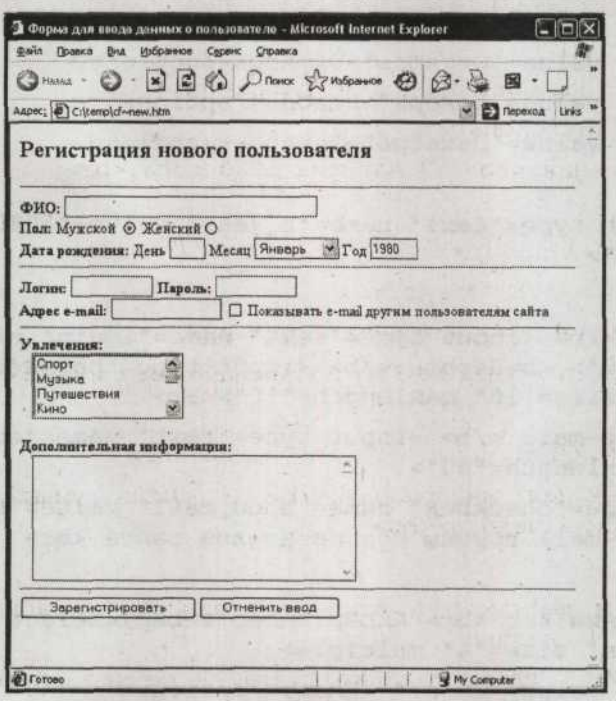


Рис. 6.13. Форма регистрации пользователя

Когда пользователь нажимает на кнопку ***Зарегистрировать***, параметры, введенные в форму, передаются адресной строке браузера. Содержимое адресной строки будет выглядеть примерно так: **http://www.mysite.com/cgi-bin/add_user?name = ВасилийИвановичПупкин&sex=male&b_day=12&b_month=Август&b_year=1980&login=vpupkin&pass=password&email=vpupkin@mail.com&show_mail=show&hobbies=Спорт&hobbies=Хоровое пение&dop=Я очень умный, красивый и вообще интересный**

```
   <b>Пол: </b>Мужской <input type="radio" name="sex"
value="male" checked> Женский<input type="radio" name="sex"
value="female"><br>
   <b>Дата рождения: </b> День <input type="text" name="b_day"
size="2" maxlength="2">
   Месяц <select name="b_month" size="1">
     <option value="Январь" SELECTED>Январь</option>
     <option value="Февраль">Февраль</option>
     <option value="Март">Март</option>
     <option value="Апрель">Апрель</option>
     <option value="Май">Май</option>
     <option value="Июнь">Июнь</option>
     <option value="Июль">Июль</option>
     <option value="Август">Август</option>
     <option value="Сентябрь">Сентябрь</option>
     <option value="Октябрь">Октябрь</option>
     <option value="Ноябрь">Ноябрь</option>
     <option value="Декабрь">Декабрь</option>
   </select>
   Год <input type="text" name="b_year" value="1980" size="4"
maxlength="4">
   <hr>
   <b>Логин:</b> <input type="text" name="login" size="10"
maxlength="10"> <b>Пароль:</b> <input type="password"
name="pass" size="10" maxlength="10"><br>
   <b>Адрес e-mail:</b> <input type="text" name="email"
size="15" maxlength="40">
   <input type="checkbox" name="show_mail" value="show">
Показывать e-mail другим пользователям сайта <br>
   <hr>
   <b>Увлечения:</b><br>    <select
name="hobbies" size="4" multiple>
     <option value="Спорт" >Спорт</option>
     <option value="Музыка">Музыка</option>
     <option value="Путешествия">Путешествия</option>
     <option value="Кино">Кино</option>
     <option value="Фотография">Фотография</option>
     <option value="Компьютерные игры">Компьютерные
игры</option>
```

ность даже создавать несколько вариантов оформления сайта, которые пользователь может выбирать по своему усмотрению. Авторы текстов получают возможность просто вводить тексты статей с иллюстрациями с помощью специального интерфейса либо просто как документ в формате **DOC** или **RTF**, а движок запишет статью в базу данных сервера. Менеджеры Интернет-магазинов получают возможность просто править параметры товаров и отмечать наличие/отсутствие товаров, с помощью специальной формы. Также движок может поддерживать форумы, чаты, ленты новостей, вести статистику посещений и т. д.

В ответ на запрос браузера пользователя к такому сайту движок достает из базы данных элементы оформления, заголовки статей и прочие необходимые элементы и показывает пользователю уже готовую HTML-страницу, полностью созданную персонально для него, основываясь на его предпочтениях. Хотя обычно движки делаются для больших сайтов, поддержкой которых занимается несколько человек, использование движка может оказаться полезным и на вашей домашней страничке, которую вы составляете и поддерживаете в одиночку.

Практический пример работы серверного сценария

Рассмотрим конкретный случай применения серверных сценариев на Web-странице. Допустим, у вас есть сайт, предусматривающий регистрацию пользователей. Чтобы посетитель был добавлен к списку зарегистрированных пользователей, он должен пройти процедуру регистрации. Для этого он должен заполнить форму, код которой представлен в Листинг 6.15. Получаемая в результате HTML-страница показана на Рис. 6.13.

Листинг 6.15. Форма регистрации пользователя

```
<html>
<head>
  <title>Форма для ввода данных о пользователе</title>
</head>
<body>
 <form action="cgi-bin/add_user" method="get">
   <h2>Регистрация нового пользователя</h2>
   <hr>
   <b>ФИО: </b> <input type="text" name="name" size="40" max-
length="70"><br>
```

необходимо предоставить ему возможность получать доступ только к той информации, которая ему действительно необходима и в том виде, в котором ему нужно.

Как это реализуется? На сервере устанавливается база данных, в которую записываются все статьи, новости и прочая информация сайта. Пользователь заполняет специальную форму, в которой указывает, документы с какими словами он ищет, по какой тематике и за какой период, а также, в каком порядке упорядочивать сведения. Эти данные передаются серверному сценарию, который проводит поиск по базе данных и формирует страницу HTML, на которой выводит ссылки на все документы сайта, подходящие пользователю.

Сходным образом работают поисковые машины, такие как **yandex.ru** и **google.com**, с той лишь разницей, что поиск они производят не по содержимому одного сайта, а по всему Интернету. Разумеется, если бы поисковая машина в ответ на запрос пользователя начинала обшаривать весь Интернет в поисках нужной страницы, это заняло бы огромное количество времени. Поэтому на службе у поисковых систем стоят специальные сценарии – «пауки» (Web-crowlers), которые «ползают» по сайтам сети Интернет и систематизируют и каталогизируют найденное. И когда пользователь вводит в поле ввода поисковой машины строку поиска, сценарии на сервере поисковика смотрят в своей базе данных список документов, в которых чаще всего встречаются искомые слова, и передают их адреса пользователю.

Другой стороной механизмов поиска и систематизации данных с помощью серверных сценариев является использование таких систем в Интернет-магазинах. Покупатель отмечает параметры товара, которые ему необходимы: тип товара, производителя, цвет, модельный ряд, диапазон цен. Затем заполненная форма передается на сервер, а серверный сценарий, на основе записей из базы данных, выводит список товаров, подходящих пользователю, и находящихся в наличии на складе.

Разделение сайта на дизайнерскую часть, данные и программную часть

В разработке и поддержке больших сайтов участвует масса людей: дизайнеры, программисты, авторы текстов, модераторы форумов и, наконец, администрация. Все эти люди участвуют в жизни и развитии сайта, но занимаются они различными частями сайта. Возникает необходимость сделать так, чтобы сферы их деятельности как можно меньше пересекались. Чтобы дизайнеры занимались только внешним видом сайта, не задаваясь проблемами кодирования страниц, чтобы авторы материалов сайта могли сосредоточиться на написании текстов, не задумываясь о том, как вставлять полученные статьи на сайт и т. д.

С помощью серверных сценариев эти проблемы легко решаются созданием так называемого «движка» сайта. Движок – это набор серверных сценариев, полностью отвечающих за функционирование сайта. Движок пишется программистами, остальные участники и посетители сайта им пользуются.

Дизайнеры получают возможность просто отрисовывать элементы дизайна, движок автоматически будет размещать их на нужных местах страниц. Появляется возмож-

лять комментарии к статьям. В общем, ситуация похожа на форум, в котором новые темы создают только модераторы. Кроме того, пользователи могут оценивать качество статей, выставляя им оценки, т. е. эта система полностью аналогична системе голосования.

Авторизация пользователя

Одним из важных применений серверных сценариев является авторизация пользователей – определение прав доступа пользователя к тому или иному разделу сайта, установка пользовательских настроек для сайта и т. д. В таких случаях на сервере устанавливается база данных, в которую записываются имена и пароли всех зарегистрированных пользователей сайта, а также их уровни доступа и личные настройки.

Зайдя на сайт, пользователь вводит в форму свое регистрационное имя и пароль, данные из формы отсылаются на сервер сценарию, который проверяет правильность введенных данных и, если все нормально, пускает пользователя дальше, попутно настроив сайт в соответствии с личными предпочтениями пользователя, зафиксированными в базе данных. Если же имя, либо пароль, оказываются неправильными, система предлагает пользователю либо ввести данные еще раз, либо зарегистрироваться, либо попытаться восстановить пароль, если пользователь его забыл.

Проходя процедуру регистрации, пользователь заполняет форму с различной информацией о себе, предпочтительными настройками сайта и т.д. После этого, данные отсылаются на сервер сценарию, который создает новую запись в базе данных.

Для восстановления пароля обычно используется адрес электронной почты, который пользователь указал при регистрации, пароль просто высылается на этот ящик электронным письмом.

Счетчик и анализ серверной статистики

С помощью сценариев можно узнать, сколько людей и в какое время посетили ваш сайт, на какие страницы они заходили, сколько времени в среднем провели на сайте. Вся эта информация позволяет определить посещаемость вашего ресурса, в какое время происходит наплыв пользователей, насколько интересны пользователям те или иные материалы вашего сайта. Особенно полезны такие системы сбора статистики для коммерческих сайтов; определение количества пользователей, посетивших сайт в то или иное время позволяет определить эффективность рекламных акций по «раскрутке» сайта, какой контингент пользователей больше интересуется сайтом и т. д.

Поиск и Сортировка записей базы данных

Если сайт содержит множество различной информации, посетитель может легко запутаться в ней и утонуть в информационных потоках. Чтобы этого не случилось,

Более сложной системой является форум. Форумы служат уже не столько для обратной связи с хозяином сайта, сколько для общения между посетителями. В отличие от гостевой книги, обсуждение в форуме делится по темам, причем новые темы для обсуждения могут создавать сами пользователи. Форумы обычно поддерживают режим регистрации пользователей, зарегистрировавшийся пользователь получает определенные преимущества – ведется статистика написанных им сообщений, своим сообщениям он может назначить некий графический образ «Я» – аватару, форум может отслеживать сообщения, появившиеся с последнего визита пользователя и т. д. «Продвинутый» форум отличается от гостевой книги примерно так же, как болид Формулы-1 от автомобиля «Запорожец».

Чтобы поддерживать форум, кроме написания сценариев, отвечающих за его работу, необходимы люди, которые будут следить за поддержанием порядка на форуме. Таких людей называют модераторами. Сценарии форума должны предусматривать, чтобы модераторы могли отключать определенных пользователей от форума, удалять сообщения, нарушающие правила, вести систему рейтингов пользователей и т. д.

Голосование

Частенько у владельцев сайтов возникает потребность узнать мнение посетителей об очередном новшестве на сайте, новом дизайне, узнать, желают ли посетители открытия нового раздела сайта, и т. д. Для проведения различного рода опросов на сайтах устраивают голосования.

На сайт вывешивается форма, состоящая из нескольких переключателей с вариантами ответа. Например, на вопрос «Стоит ли заводить раздел сайта о желтых слониках?», варианты ответа могут быть следующими: «Конечно, стоит, желтые слоники – это здорово», «Никаких желтых слоников, и так от них тошно» и «А что за слоники такие, желтые?». Выбранный вариант ответа передается на сервер, где сценарий проверяет, не участвовал ли уже в этом опросе посетитель страницы. Если еще не участвовал, то его голос добавляется к результатам голосования, а информация о посетителе записывается, чтобы не посчитать его голос дважды. Затем подсчитываются средние баллы, и сценарий выдает HTML-страничку с посчитанными результатами голосования.

Лента Новостей

Обязательной частью сайтов с часто обновляемой информацией, являются ленты новостей. На главной странице сайта отображается краткое содержание новых материалов на сайте со ссылками, по которым можно перейти, чтобы ознакомиться с информацией подробнее.

По своей сути, ленты новостей являются чем-то средним между форумом и системой голосования. Функцию новых тем в форуме выполняют новые материалы сайта, но выкладывать новые материалы на сайт могут только люди, наделенные соответствующими полномочиями. Обычные пользователи могут только остав-

С помощью такой формы может быть устроена обратная связь с посетителями сайта. Вместо того чтобы посылать свои замечания и пожелания по адресу электронной почты, указанному на сайте, они могут просто ввести пожелания в соответствующую форму и отослать письмо буквально одним щелчком мыши.

На сервере могут выполняться сценарии периодически, через определенные промежутки времени, высылающие на определенный адрес электронной почты письма, содержащие информацию о статистике сайта: количестве посещений, объеме скачанной информации, новых пользователях сайта, наиболее часто посещаемых страницах.

Чтобы постоянные посетители вашего сайта были в курсе основных изменений на вашей странице, можно организовать почтовую рассылку. Специальный сценарий будет рассылать пользователям, подписавшимся на рассылку, информацию о новостях сайта: новых разделах, новых статьях, свежих файлах и т.д. Информация о новостях может составляться как вручную, так и автоматически, тем же сценарием.

Чат

Если сайт пользуется постоянной популярностью у некоторой группы людей и объединяет посетителей сети Интернет со схожими интересами, то логично будет поддержать постоянных пользователей, дав им возможность общаться друг с другом, не покидая пределы вашего сайта. Одним из способов наладить такое общение является создание чата.

Первая часть чата – модуль регистрации, он проверяет имя и пароль пользователя и регистрирует новых пользователей. Имена и пароли всех пользователей хранятся на сервере в базе данных. В том случае, если пользователь успешно зарегистрировался, он допускается до второй части чата – комнаты общения.

Попав в комнату, пользователь может вводить сообщения, которые будут передаваться на сервер и записываться в файл. Затем, последние несколько строк файла выводятся всем пользователям, находящимся в комнате общения, чтобы все видели последние реплики. Чтобы отличать сообщения одного пользователя от сообщений другого, все сообщения помечаются именем пользователя, примерно так: **<имя_пользователя>: сообщение**.

Гостевая книга и форум

Наладить контакт между посетителями сайта и его владельцем позволяет гостевая книга. Любой посетитель сайта может внести свои пожелания, замечания и заметки в форму, затем эти данные передаются сценарию, который записывает их в файл вместе с именем посетителя, временем и датой заметки и адресом электронной почты посетителя. Содержимое файла с помощью сценария отображается на странице гостевой книги так, чтобы все посетители могли ознакомиться с ним и участвовать в обсуждении.

- **Server**. Передает название и версию программного обеспечения Web-сервера. Пример заголовка: **Server Apache/1.3.23 (Unix) (Mandrake/Linux) mod_ssl/2.8.7 OpenSSL/0.9.6b DAV/1.0.3 PHP/4.3.4 mod_perl/1.26 configured**.
- **User-Agent**. Передает серверу версию браузера. Пример заголовка: **User-Agent: Mozilla/5.0**.

Типы MIME

Данные, передаваемые от браузера к серверу и наоборот, могут быть различными – это может быть и простой текст, и графика, и исполняемый код программы. Чтобы различать виды данных, можно задать тип данных, называемый MIME-типом. Обработка полученных данных ведется в зависимости от их MIME-типа. Тип MIME состоит из двух частей: **тип_данных/подтип**, где **тип_данных** – это общая категория данных, например звуковая информация или текст, а **подтип** – конкретный формат. Например, формат **mp3** или **exe**.

При передаче данных между браузером и Web-сервером тип данных может задаваться при помощи HTTP-заголовка **Content-type**. Перечислим основные MIME-типы:

- **application**. Приложение, программа или документ, связанный с определенной программой. Пример: **application/pdf**.
- **audio**. Звуковой файл. Пример: **audio/mp3**.
- **image**. Графический файл, например: **image/gif**.
- **text**. Текстовая информация. **text/html** – текст в формате HTML, **text/plain** – обычный текст.

Для чего нужны серверные сценарии

В каких же целях используют серверные сценарии? Рассмотрим основные способы их применения и случаи, в которых к ним прибегают.

Создание и прием писем

Очень часто серверные сценарии используются для передачи данных, введенных в форму посетителем страницы, на некоторый адрес электронной почты. Например, такие формы используются в Интернет-магазинах для заказа товаров. Покупатель вносит в поля формы название товара, нужное его количество, свои реквизиты и щелчком на кнопке, отправляет заполненную форму на сервер. На сервере сценарий получает данные из формы, формирует на их основе письмо электронной почты и отправляет его по определенному адресу. Это письмо получает сотрудник Интернет-магазина, связывается с покупателем и оформляет доставку товара.

В этом случае данные сценарию передаются уже после запроса, не подвергаясь никаким изменениям со стороны сервера, длина передаваемых данных, в символах, задается заголовком **Content-length**. Как и в случае с методом **GET**, языки серверных сценариев, вроде PHP, переводят полученные данные в удобный для дальнейшей обработки сценарием вид.

Обычно метод **POST** используется для передачи на сервер больших объемов данных, например, файлов. Еще этот метод может быть полезен в том случае, когда нужно скрыть передаваемые параметры от пользователя.

Заголовки HTTP

В запросе, который браузер посылает на Web-сервер, кроме **GET** или **POST** заголовка могут быть и другие строки, содержащие дополнительную информацию, передаваемую серверу. Сервер в свою очередь тоже пересылает браузеру строки с различной служебной информацией. Эти строки называются HTTP-заголовками. Рассмотрим основные заголовки HTTP:

- **Accept**. Этот заголовок сообщает серверу о типах данных, поддерживаемых браузером. Пример использования заголовка **Accept**: **Accept: text/html, text/plain, image/jpeg**. В этом примере браузер сообщает серверу, что он поддерживает формат HTML, обычный текст и графику формата JPEG. О возможных типах данных мы поговорим чуть позже. Если браузер поддерживает все возможные типы данных, то он может просто передать серверу заголовок **Accept: *.***.
- **Content-type**. Обозначает тип передаваемых данных. По умолчанию этот заголовок имеет значение **Content-type: application/x-www-form-urlencoded**. Так указывается тип данных, в котором все управляющие символы, то есть не являющиеся алфавитно-цифровыми, особым образом кодируются.
- **Content-length**. С помощью этого заголовка передается длина передаваемых данных, при использовании метода передачи **POST**.
- **GET**. Этот заголовок используется при методе передачи данных **GET**, его мы уже рассматривали ранее. Заголовок имеет следующий формат: **GET название_сценария?параметры HTTP/1.0**.
- **POST**. Используется при передаче данных методом **POST**. Уже рассматривался при изучении соответствующего метода. Заголовок имеет такой формат: **POST название_сценария HTTP/1.0**.
- **Location**. Указывает серверу **URL**, по которому тот немедленно должен перейти. Пример использования заголовка: **Location: http://www.mysite.com**.
- **Pragma**. Используется для различных целей, например для запрета сохранения временной копии документа. В этом случае заголовок будет выглядеть так: **Pragma: no-cache**.

Данные передаются на сервер в виде пар: ***имя_поля_ввода=значение_поля_ввода***. Эти пары значений записываются в адресной строке браузера после адреса серверного сценария, отделяясь от него знаком вопроса – «?». Если на сервер передается несколько параметров, то они отделяются друг от друга знаком «&» – амперсанд: **http://localhost/cgi-bin/add_name?user_name=Anthony&age=12**.

Браузер выделяет из полученной строки имя сервера, затем соединяется с ним и посылает на сервер запрос, вид которого показан в Листинг 6.13. В этом запросе **\n** означает перевод строки, а **\n\n** – обозначает окончание запроса. Строки запроса, включая первую, начинающуюся с **GET**, называют заголовками. Заголовки формирует браузер, для передачи параметров в сценарий достаточно первой строки запроса.

Листинг 6.13. Запрос ***GET***

```
GET cgi-bin/add_name?user_name=Anthony&age=12 HTTP /1.0 \n
...еще несколько строк информации...
\n\n
```

Сценарий, запускаемый при выполнении запроса, получает доступ к строке с переданными параметрами, например **user_name=Anthony&age=12**; ему достаточно лишь разобраться, где имена параметров, а где их значения. Но языки программирования, разработанные для написания серверных сценариев, такие как PHP, берут эту грязную работу на себя, предоставляя сценарию информацию уже в удобном виде.

Преимуществом метода **GET** является то, что все данные, передаваемые формой на сервер, отображаются в адресной строке браузера, и пользователь, при желании, может сохранить запрос в закладках браузера и повторить его позже. Это может быть полезным, например, при работе с поисковыми машинами. Сохраняя содержимое адресной строки в закладках браузера, пользователь может сохранять интересующие его поисковые запросы.

Метод POST

В том случае, когда данные передаются на сервер с использованием метода **POST**, адресная строка браузера в передаче данных не участвует, и передаваемые параметры в ней отображаться не будут. Браузер просто формирует запрос к серверу, в виде, показанном в Листинг 6.14.

Листинг 6.14. Запрос ***POST***

```
POST имя_сценария HTTP/1.0 \n
Content-length: 5\n\n
передаваемые_данные.
```

Методы передачи данных на сервер

Взаимодействие между браузером пользователя и сервером может осуществляться двумя различными способами: **GET** и **POST**.

Метод GET

При использовании метода передачи **GET** форма передает данные из формы через адресную строку браузера. Рассмотрим работу этого механизма на примере формы, код которой приведен в

Листинг 6.12, а получаемая Web-страница показана на Рис. 6.12. Допустим, пользователь ввел в текстовое поле **user_name** имя «Anthony», Когда пользователь щелкает мышью на кнопке **Добавить**, в адресной строке браузера формируется примерно следующая текстовая строка: **http://localhost/cgi-bin/add_name?user_name=Anthony**.

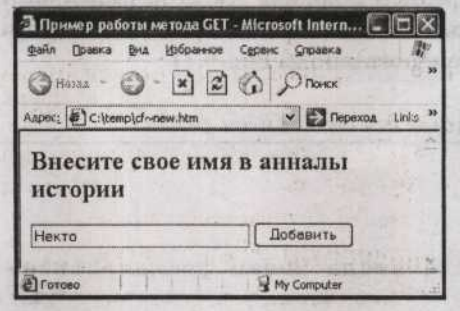


Рис. 6.12. Форма, передающая данные методом ***GET***

Листинг 6.12. Пример работы метода ***GET***

```
<html>
<head>
  <title>Пример работы метода GET </title>
</head>
<body>
  <form action="cgi-bin/add_name" method="get">
  <h2>Внесите свое имя в анналы истории</h2>
   <input type="text" name="user_name" value="Некто" size="30"
maxlength="30">
   <input type="submit" value="Добавить">
  </form>
</body>
</html>
```

Рис. 6.10. Форма с кнопками ***RESET***

Тип FILE

При значении атрибута **TYPE="FILE"** создается поле ввода названия файла. С помощью этого поля можно передавать на сервер различные файлы. С помощью атрибута **SIZE** можно задать размер поля для ввода названия файла в символах. Пример формы с полем для ввода названия файла приведен в Листинг 6.11, как отображает эту страницу Web-браузер, показано на Рис. 6.11

Листинг 6.11. Форма с полем для ввода имени файла

```
<html>
<head>
    <title>Поле для ввода имени файла</title>
</head>
<body>
<form action="">
<input type="file" name="picture_file" size="40"><br>
</form>
</body>
</html>
```

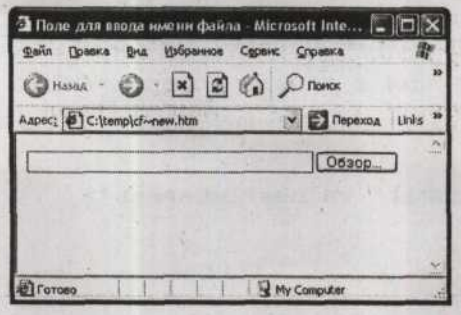


Рис. 6.11. Страница с полем ввода для имени файла

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

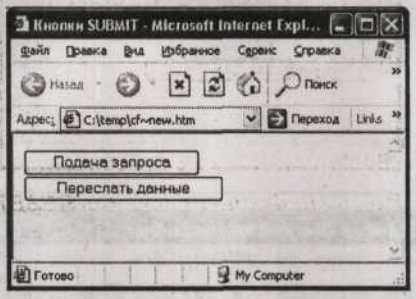


Рис. 6.9. Форма с кнопками ***SUBMIT***

Тип RESET

При значении атрибута **TYPE="RESET"**, создается кнопка сброса введенных данных. Нажатие на такую кнопку приводит к отмене ввода информации в форму и установке всех полей в значение по умолчанию. С помощью атрибута **VALUE** можно задать текст, отображающийся на кнопке. Если атрибут **VALUE** не задан, текст на кнопке будет определять сам браузер. Пример формы с кнопкой **RESET** приведен в Листинг 6.10, получившаяся страница показана на Рис. 6.10. В этом примере, в форме размещены две кнопки **RESET**, текст на одной из них назначается атрибутом **VALUE**, а на другой задается браузером.

Листинг 6.10. Форма с кнопками RESET

```
<html>
<head>
  <title>Кнопки RESET</title>
</head>
<body>
  <form action="">
   <input type="reset"><br>
   <input type="reset" value="Отменить ввод данных">
  </form>
</body>
</html>
```

```
    <input type="button" name="but1" value="Слоник">
    <input type="button" name="but2" value="Бегемотик">
    <input    type="button"    name="but3"    value="Сиреневенький
ушастик" >
  </form>
</body>
</html>
```

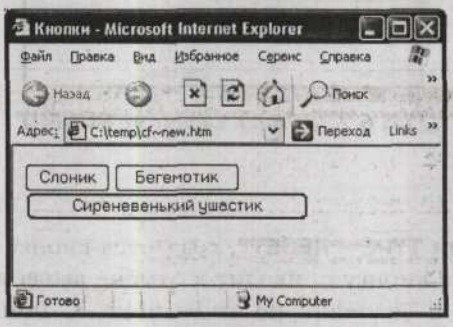


Рис. 6.8. Кнопки на странице HTML

Тип SUBMIT

При значении атрибута **TYPE="SUBMIT"** создается кнопка передачи данных на сервер. Нажатие на эту кнопку приводит к тому, что все данные, введенные в форму, пересылаются по адресу, указанному в атрибуте **ACTION** тега **<FORM>**. С помощью атрибута **VALUE** можно задать текст, отображающийся на кнопке. Если атрибут **VALUE** не задан, текст на кнопке будет определять сам браузер. Код формы с кнопкой **SUBMIT** приведен в Листинг 6.9, полученная страница отображена на Рис. 6.9. В этом примере в форме размещены две кнопки **SUBMIT**, текст на одной из них назначается атрибутом **VALUE**, а на другой задается браузером.

Листинг 6.9. Форма с кнопками SUBMIT

```
<html>
<head>
  <title>Кнопки SUBMIT</title>
</head>
<body>
  <form action="">
   <input type="submit"><br>
   <input type="submit" value="Переслать данные">
  </form>
```

```
  <input type="radio" name="Field_of_glory" value="лось"
>Купить лося<br>
  <input type="radio" name="Field_of_glory" value="порося"
checked>Или порося <br>
  <input type="radio" name="Field_of_glory" value="гусь" >А
может и гуся
  </form>
</body>
</html>
```

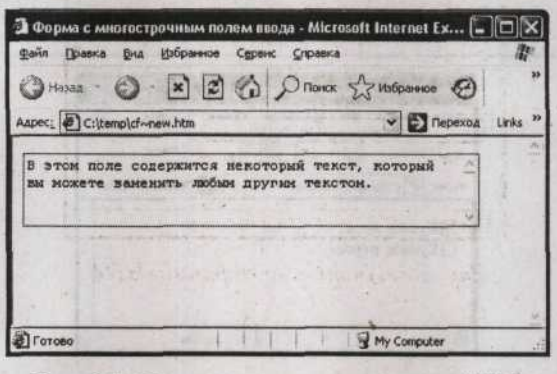


Рис. 6.7. Переключатели на странице HTML

Тип BUTTON

При значении атрибута **TYPE="BUTTON"** создается кнопка. Непосредственно кнопки не могут влиять на данные, отправляемые из формы на сервер, но на нажатие кнопки можно назначить событие **onClick**, к которому можно привязать произвольный сценарий JavaScript. Этот сценарий может, например, определенным образом заполнять часть полей формы. Атрибут **VALUE** задает надпись, которая будет отображена на кнопке. Пример формы с кнопками приведен в Листинг 6.8, внешний вид страницы показан на Рис. 6.8.

Листинг 6.8. Форма с кнопками

```
<html>
<head>
  <title>Кнопки</title>
</head>
<body>
  <form action="">
```

```
</head>
<body>
  <form action="">
   <input type="checkbox" name="Field_of_glory" value="лось"
checked>Купить лося<br>
  <input type="checkbox" name="Field_of_glory"
value="порося">Купить порося
  </form>
</body>
</html>
```

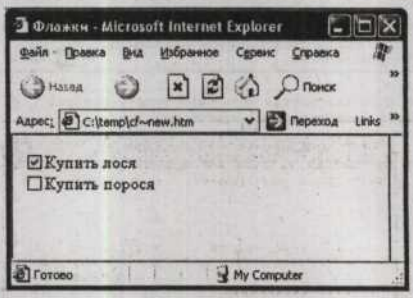


Рис. 6.6. Флажки на странице HTML

Тип RADIO

При значении атрибута **TYPE="RADIO"** создается переключатель. Несколько переключателей с одинаковым значением атрибута **NAME** группируются вместе так, что выбран может быть только один из переключателей группы. При передаче данных формы на сервер передается значение **VALUE** выбранного переключателя группы. Атрибут **CHECKED** позволяет указать на переключатель, выбранный по умолчанию. Пример использования переключателей приведен в Листинг 6.7, внешний вид получившейся страницы показан на Рис. 6.7.

Листинг 6.7. Форма с переключателями

```
<html>
<head>
  <title>Переключатели</title>
</head>
<body>
  <form action="">
```

Листинг 6.5. Поле для ввода паролей

```
<html>
<head>
  <title>Поле для ввода паролей</title>
</head>
<body>
  <form action="">
   <input          type="password"          name="Field_of_glory"
value="оксилентрибутадиен" size="50" maxlength="100">
  </form>
</body>
</html>
```

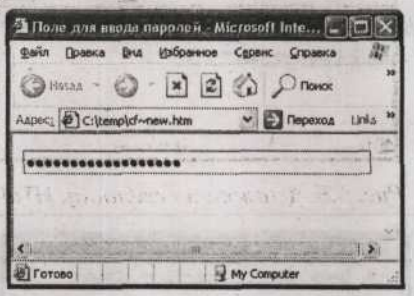


Рис. 6.5. Поле для ввода паролей

Тип CHECKBOX

При значении атрибута **TYPE="CHECKBOX"** отображается флажок, который можно установить либо сбросить. Если задать для флажка атрибут **CHECKED**, то он будет по умолчанию установлен. Несколько флажков могут иметь одинаковые имена **NAME**. При передаче заполненной формы на сервер будут передаваться значения атрибута **VALUE** установленных флажков. Пример использования флажков приведен в Листинг 6.6, как будет выглядеть эта страница в браузере показано на Рис. 6.6.

Листинг 6.6. Использование флажков

```
<html>
<head>
  <title>Флажки</title>
```

Тип TEXT

При значении атрибута **TYPE="TEXT"** будет создано однострочное поле ввода. Максимальное количество символов в поле определяется атрибутом **MAXLENGTH**, размер поля – атрибутом **SIZE**, а значение по умолчанию – атрибутом **VALUE**. Пример однострочного поля ввода приведен в Листинг 6.4, как выглядит однострочное поле ввода в браузере показано на Рис. 6.4.

Листинг 6.4. Однострочное поле ввода

```
<html>
<head>
  <title>Однострочное поле ввода</title>
</head>
<body>
  <form action="">
   <input type="text" name="Field_of_glory" value="Ондатровый
воротник" size="50" maxlength="100">
  </form>
</body>
</html>
```

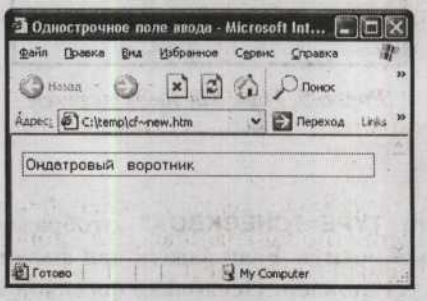


Рис. 6.4. Однострочное поле ввода

Тип PASSWORD

При значении атрибута **TYPE="PASSWORD"** создается текстовое поле для ввода паролей. При вводе текста вместо символов отображаются звездочки. Максимальное количество символов в поле определяется атрибутом **MAXLENGTH**, размер поля – атрибутом **SIZE**, а значение по умолчанию – атрибутом **VALUE**. Пример поля для ввода паролей приведен в Листинг 6.5, его отображение в браузере показано на Рис. 6.5.

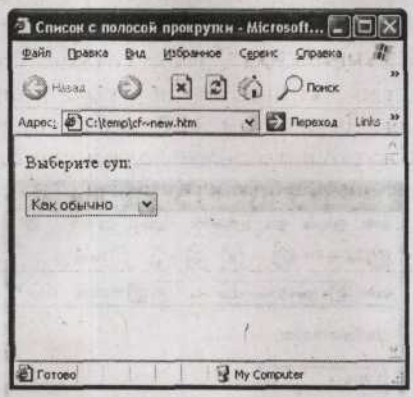


Рис. 6.3. Выпадающий список

Тег <INPUT>

В отличие от остальных элементов форм, тег **<INPUT>** является одиночным и не требует себе в пару закрывающего тега. С помощью тега **<INPUT>** реализуется целая группа элементов управления, размещаемых в формах **HTML**: однострочные текстовые поля, поля для ввода паролей, переключатели, флажки и различные кнопки. Параметры тега **<INPUT>** настраиваются с помощью следующих атрибутов:

- **NAME**. Имя поля. Является обязательным для всех типов поля ввода **<INPUT>**, кроме вариантов **SUBMIT** и **RESET**.
- **SIZE**. Если тег **<INPUT>** используется в виде текстового поля ввода или поля ввода пароля, этот атрибут позволяет задать размер поля ввода. Размер задается в символах.
- **MAXLENGTH**. С помощью этого атрибута задается максимальное количество символов, которое можно ввести в текстовое поле ввода или в поле ввода пароля.
- **VALUE**. Для текстовых полей ввода и полей ввода пароля этот атрибут устанавливает значение по умолчанию. Для кнопок этот атрибут определяет текст, размещаемый на кнопке.
- **READONLY**. Задание этого атрибута для текстовых полей ввода и полей ввода пароля делает поле доступным только для чтения и запрещает любые его изменение.
- **CHECKED**. Определяет выбранные по умолчанию переключатели и установленные по умолчанию флажки.
- **TYPE**. Этот атрибут определяет, какой элемент управления описывает тег **<INPUT>**.

Атрибут **TYPE** тега **<INPUT>** может принимать множество различных значений. От этого атрибута зависит, какого типа будет поле ввода.

```
  </form>
</body>
</html>
```

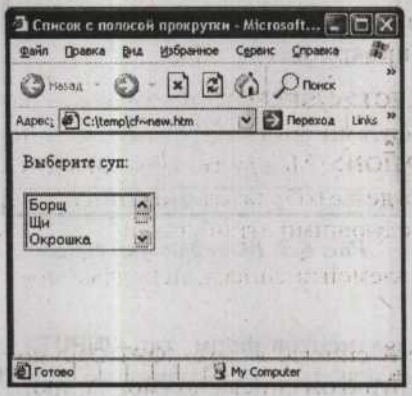


Рис. 6.2. Список с полосой прокрутки

***Листинг 6.3.** Форма с выпадающим списком*

```
<html>
<head>
  <title>Выпадающий список</title>
</head>
<body>
  Выберите суп:
  <form action="">
  <select name="soup" size="1">
   <option value="1">Борщ</option>
   <option value="2">Щи</option>
   <option value="3">Окрошка</option>
   <option value="4">Молочный суп</option>
   <option value="default" selected>Как обычно</option>
  </select>
  </form>
</body>
</html>
```

открывающийся список значений. Если значение атрибута SIZE больше единицы, то значения будут отображаться в виде списка с полосой прокрутки.

- **MULTIPLE**. Если в теге **<SELECT>** указать этот атрибут, не присваивая ему никаких значений, то в создаваемом списке можно будет одновременно выбрать несколько его элементов. При задании атрибута **MULTIPLE** вне зависимости от значения атрибута **SIZE** значения отображаются в виде списка с полосой прокрутки.

Внутри контейнера **<SELECT></SELECT>** последовательно перечисляются значения, которые будут элементами списка. Каждый элемент списка образуется контейнером **<OPTION></OPTION>**. Между тегами **<OPTION>** и **</OPTION>** записывается название, которое будет отображаться в списке, остальные параметры элемента списка задаются следующими атрибутами:

- **VALUE**. Значение элемента списка, передаваемое серверу при выборе этого пункта.
- **SELECTED**. Элемент списка, в теге **<OPTION>** которого указан этот атрибут, будет выбран пунктом списка по умолчанию.

В Листинг 6.2 приведен код HTML-страницы со списком с полосой прокрутки. Как созданный элемент будет выглядеть в браузере, показано на Рис. 6.2. Значением по умолчанию в этом списке назначен пункт «Как обычно». Поскольку в теге **<SELECT>** указан атрибут **MULTIPLE**, то пользователь может выбрать сразу несколько супов, если необходимо. Код страницы с теми же элементами, но оформленными в виде выпадающего списка, приведен в Листинг 6.3, ее отображение в браузере показано на Рис. 6.3.

Листинг 6.2. Список с полосой прокрутки

```
<html>
<head>
  <title>Список с полосой прокрутки</title>
</head>
<body>
  Выберите суп:
  <form action="">
  <select name="soup" size="3" multiple>
   <option value="1">Борщ</option>
   <option value="2">Щи</option>
   <option value="3">Окрошка</option>
   <option value="4">Молочный суп</option>
   <option value="default" selected>Как обычно</option>
  </select>
```

Текст, помещенный между тегами **<TEXTAREA>** и **</TEXTAREA>**, будет значением поля ввода по умолчанию. Этот текст будет отображен в поле ввода. Код HTML-документа с многострочным полем ввода показан в Листинг 6.1. Как такое поле ввода отображается браузером, показано на Рис. 6.1.

Листинг 6.1. Форма с полем ввода ***<TEXTAREA>***

```
<html>
<head>
 <title>Форма с многострочным полем ввода</title>
</head>
<body>
 <form action="">
 <textarea cols="50" rows="4" name="sometext">
  В этом поле содержится некоторый текст, который вы можете заменить любым другим текстом.
  </textarea>
  </form>
</body>
</html>
```

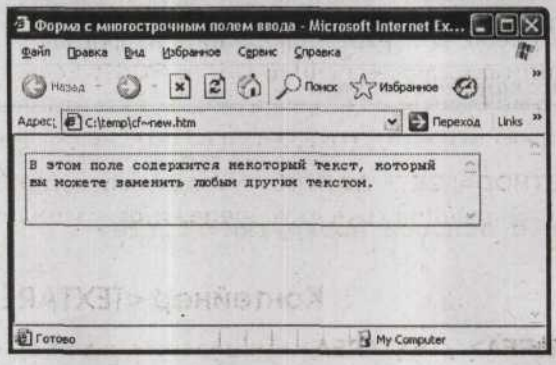


Рис. 6.1. Форма с многострочным полем ввода

Контейнер <SELECT></SELECT>

С помощью этого контейнера в формах **HTML** создаются списки с полосой прокрутки и выпадающие списки значений. Для контейнера **<SELECT></SELECT>** можно задать следующие атрибуты:

- **NAME**. Имя поля.
- **SIZE**. Число строк списка, одновременно отображаемых на экране. Если не задавать атрибут SIZE или присвоить ему значение 1, то будет создан

- **<TEXTAREA></TEXTAREA>**. С помощью этого контейнера в форму добавляются поля, в которые можно вводить многострочный текст.
- **<SELECT></SELECT>**. Этот контейнер создает списки и открывающиеся списки с вариантами выбора.
- **<INPUT>**. Этот тег позволяет использовать в форме следующие элементы управления: переключатели, флажки, кнопки и однострочные поля ввода.

Рассмотрим теги форм по очереди.

Контейнер <FORM></FORM>

В контейнере **<FORM></FORM>** могут содержаться как теги формы: **<TEXTAREA>**, **<SELECT>**, **<INPUT>**, так и другие элементы HTML-страницы: ссылки, таблицы, изображения. Ограничений на количество элементов в форме и количество самих форм нет, но формы нельзя вкладывать друг в друга. Если внутри контейнера **<FORM></FORM>** поместить еще один тег **<FORM>**, то он будет проигнорирован. У тега **<FORM>** есть три основных атрибута:

- **NAME.** Имя формы можно задать для удобства обращения к форме посредством сценариев JavaScript.
- **ACTION**. Адрес, по которому пересылается заполненная форма. Обычно указывается адрес серверного сценария, который обрабатывает данные, полученные из формы. Если форма используется исключительно для взаимодействия со сценариями JavaScript, то значение атрибута **ACTION** можно не задавать.
- **METHOD**. Способ, которым информация из формы передается на сервер. Может принимать два значения: **GET** и **POST**. Если атрибут **METHOD** не задавать, то он будет иметь значение GET. Чем отличаются эти способы и в каких случаях используется каждый из них, мы рассмотрим чуть позже.

Форма **<FORM METHOD="GET" ACTION="cgi-bin/megascript">** будет пересылать результаты своего заполнения сценарию **megascript**, расположенному в каталоге **cgi-bin**, сервера.

Контейнер <TEXTAREA></TEXTAREA>

Контейнер **<TEXTAREA></TEXTAREA>** применяется в тех случаях, когда необходимо ввести в форму более одной строки текста. Контейнер **<TEXTAREA></TEXTAREA>** хорошо подходит для ввода различных комментариев, сообщений в форумы и гостевые книги и других случаев, в которых от пользователя ожидается значительный объем текстовой информации. У контейнера **<TEXTAREA></TEXTAREA>** есть следующие атрибуты:

- **NAME**. Имя поля.
- **ROWS**. Число строк в поле ввода.
- **COLS**. Количество символов в строке поля.

Знакомство с интерфейсом CGI

Запуск серверных сценариев в ответ на запросы Web-браузера производится с помощью CGI-интерфейса. Аббревиатура CGI (Common Gateway Interface – Общий шлюзовой интерфейс) обозначает набор правил и соглашений, соблюдаемых Web-сервером при выполнении сценариев. Интерфейс CGI является промежуточным звеном между программами, установленными на Web-сервере и браузером. Серверные сценарии часто называют также CGI-программами, и размещаются они как правило в каталоге сервера **cgi-bin**.

CGI-программой, в принципе, может быть любая программа, выводящая какой-либо результат. С помощью механизмов CGI вывод программы перенаправляется на Web-сервер. А сервер в свою очередь передает полученный результат в браузер посетителя страницы. Программе совершенно не нужно «знать», что результаты ее работы передаются в Интернет, она может «полагать», что просто выводит данные на экран монитора.

В действительности далеко не всякая программа может быть серверным сценарием. Для правильной работы системы браузер–Web-сервер–сценарий необходимо, чтобы программа, кроме самого документа, выводила еще определенную дополнительную информацию, например, указывала тип выводимых данных. Такая информация называется заголовками сервера. Например, чтобы указать серверу, что выводимые данные являются документом HTML, необходимо отправить ему заголовок **Content-type: text/html**.

Чтобы серверный сценарий мог генерировать различные документы в зависимости от действий пользователя, необходимо передавать сценарию определенные параметры. К примеру, чтобы сообщить программе **script.pl** имя пользователя и адрес его электронной почты, необходимо вызвать эту программу с нужными параметрами: **script.pl? name=Вася&mail=vpupkin@mail.com**. Этим запросом в программу передается параметр **name**, значение которого «Вася», и параметр **mail** со значением «vpupkin@mail.com».

Хотя чаще всего для написания серверных сценариев используются специально предназначенные для этого языки программирования, сценарии могут быть написаны на любом языке, лишь бы соответствующие программы могли выполняться на компьютере, на котором установлен Web-сервер.

Формы

Чтобы пользователь мог передавать информацию серверным сценариям, на страницах **HTML** используются формы. Некоторые элементы форм мы уже использовали в предыдущей главе, теперь же пришло время познакомиться с формами поближе.

Как уже говорилось ранее, основное предназначение форм – передача информации, введенной пользователем, на сервер. Вся форма заключается в контейнер **<FORM></FORM>**. Составляющими частями формы являются поля ввода, кнопки и переключатели, с помощью которых пользователь вводит данные. Элементы ввода данных в форму создаются при помощи трех разных тегов:

глава 6

CGI-сценарии: новые возможности Ваших страниц

В предыдущей главе мы начали рассмотрение динамических Web-страниц и познакомились с технологиями создания клиентских сценариев. Клиентские сценарии являются неотъемлемой частью любого современного сайта и их изучение очень важно, но не менее важным является знание серверных технологий создания динамических Web-сайтов.

Эта глава посвящена изучению основ взаимодействия между браузером и серверными сценариями, здесь рассмотрены основные технологии передачи данных из браузера на сервер и обратно. Также, мы рассмотрим основные случаи, в которых применяются серверные сценарии, и увидим каким именно образом.

Формы и сценарии CGI

В предыдущей главе уже обсуждалось, как Web-сервер обрабатывает запросы браузера. В том случае, сайт статический, сервер передает браузеру уже готовую HTML-страницу или изображение. Если же сайт построен на основе серверных технологий создания динамических сайтов, то в ответ на запрос браузера, будет выполнен определенный серверный сценарий. Этот сценарий может создать HTML-страницу, изображение или другой документ, который и будет передан браузеру. Причем для браузера оба варианта действий сервера выглядят совершенно одинаково: он подал запрос и получил некий документ в ответ.

```
    <input type="button" name="fact" value="n!" onClick =
"document.calc.numbers.value = factrial(document.calc.
numbers.value)"><br>
    <input type="button" name="inv" value="1/x" onClick =
"document.calc.numbers.value =
1/document.calc.numbers.value"><br>
    </td></tr>
  </table>
  </td><td valign="top">
  <input type="button" name="divide" value="/" onClick =
"storeop(document.calc.numbers.value,'div'); docu-
ment.calc.numbers.value = ''"><br>
  <input type="button" name="mult" value="*" onClick =
"storeop(document.calc.numbers.value,'mult'); docu-
ment.calc.numbers.value=''"><br>
  <input type="button" name="minus" value="-" onClick =
"storeop(document.calc.numbers.value,'min'); docu-
ment.calc.numbers.value=''"><br>
  <input type="button" name="equals" value="=" onClick =
"document.calc.numbers.value = to-
tal(document.calc.numbers.value)"><br>
<input type="button" name="clear" value="C" onClick =
"document.calc.numbers.value = ''"><br>
  </td></tr>
</table>
</form>
</body>
</html>
```

Заключение

Итак, вы создали свою первую динамическую Web-страницу, которая в ответ на действия посетителя выполняет сценарии JavaScript, проводит расчеты и выводит полученный результат на экран. Поздравляю вас, вы сделали большой шаг от статических технологий к динамическим. Теперь перед вами открыты замечательные возможности по созданию Web-страниц, дружественных посетителю, реагирующих на каждый его запрос, подстраивающихся под его нужды.

Следующим шагом, после освоения клиентских сценариев, будет погружение в мир серверных динамических страниц. Этим мы и займемся в следующих главах книги.

Кроме стандартных математических действий, в языке PHP, также как и в языке JavaScript, есть несколько дополнительных математических операций.

- **%** – модуль. Вычисляет остаток от целочисленного деления одного выражения на другое. Пример – **12%5**, результатом этого выражения будет 2.
- **+=** – добавление некоторого значения к переменной. Выражение **$x+=$y;** добавляет к значению переменной **$x** значение переменной **$y** и записывает результат в переменную **$x**. Это выражение является более короткой записью выражения **$x=$x+$y;**.
- **-=** – вычитание некоторого значения из переменной. Выражение **$x-=$y;** вычитает из значения переменной **$x** значение переменной **$y** и записывает результат в переменную **$x**. Это выражение является более короткой записью выражения **$x=$x-$y;**.
- ***=** – умножение значения переменной на некоторую величину. Выражение **$x*=$y;** умножает значение переменной **$x** на значение переменной **$y** и записывает результат в переменную **$x**. Это выражение является более короткой записью выражения **$x=$x*$y**.
- **/=** – деление значения переменной на некоторую величину. Выражение **$x/=$y;** делит значение переменной **$x** на значение переменной **$y** и записывает результат в переменную **$x**. Это выражение является более короткой записью выражения **$x=$x/$y;**.
- **++** – инкремент. Увеличивает значение переменной на единицу. Записывается так: **$x++**. В результате, значение переменной **$x** увеличится на 1.
- **--** – декремент. Уменьшает значение переменной на единицу. Записывается так: **$x--**. В результате значение переменной **$x** уменьшится на 1.

Логические операции

Результатом логических, или Булевых, выражений, может являться одно из двух значений: **True** (истина) или **False** (Ложь). Простые логические выражения состоят из двух значений, или выражений, объединенных оператором сравнения: **1е_значение оператор_сравнения 2е_значение**. Существуют следующие операторы сравнения:

- **==** – равенство. Если оба значения равны, то результат выражения **True** (истина), иначе **False** (Ложь).
- **!=** – неравенство. Если значения не равны, то результатом выражения будет **True** (истина).
- **<** – меньше. Результат **True** (истина), если первое выражение меньше второго.
- **>** – больше. Результат **True** (истина), если выражение больше второго.
- **<=** – меньше либо равно. Результат **True** (истина), если выражение меньше либо равно второму.

- >= – больше либо равно. Результат **True** (истина), если 1-е выражение больше либо равно второму.

Приведем несколько примеров логических выражений. **2<3** – результат **True** (истина), поскольку 2 меньше 3; **2*2==4** – **True** (истина); **34>=45** – **False** (Ложь). Несколько простых логических выражений можно объединить в более сложное выражении при помощи логических операторов. Логическое выражение будет выглядеть так: **1е_логическое_значение логический_оператор 2е_логическое_значение**. Исключение составляет логический оператор **Not** (Не), который размещается перед единственным логическим значением.

Логические операторы бывают следующими:

- **!** – **Not** (Не). Меняет результат логического выражения на обратный. **True** (истина) на **False** (Ложь) и наоборот.
- **&&** – **And** (И). Результатом выражения будет **True** (истина) в том случае, если значение **True** (истина) имеют оба логические значения.
- || – **OR** (Или). Результатом выражения будет **True** (истина), если значение **True** (истина) имеет хотя бы одно из логических значений.

Результатом выражения **2>5||5>2** будет **True** (истина), поскольку одно из выражений имеет результат **True** (истина). Результатом выражения **3>0 && 3>5** будет **False** (Ложь), поскольку одно из выражений имеет результат **False** (Ложь).

Основные конструкции языка PHP

Основные конструкции языка PHP очень похожи на конструкции языка JavaScript, который мы рассматривали ранее, поэтому они врядли вызовут у вас затруднения.

Оператор условного перехода

Одной из важнейших конструкций, безусловно, является оператор условного перехода **IF-ELSE**. Синтаксис его показан в Листинг 7.12.

Листинг 7.12. Оператор ***IF-ELSE***

```
if (условие)
{
последовательность_команд_1
}
else
{
последовательность_команд_2
}
```

Если результат логического выражения **условие** равен **True** (истина), то выполняются команды **последовательность_команд_1**, в противном случае выполняются команды **последовательность_команд_2**. То есть, если условие выполняется, то исполняются команды, находящиеся в фигурных скобках непосредственно после оператора **IF**, если же условие не выполнено, – то команды, идущие за оператором **ELSE**. Если вместо последовательности команд нужно исполнить всего одну команду, то фигурные скобки использовать необязательно. Синтаксис оператора **IF-ELSE** для такого случая приведен в Листинг 7.13.

*Листинг 7.13. Оператор **IF-ELSE**, сокращенный вариант*

```
if (условие)
   команда_1;
else
   команда_2;
```

Если нет необходимости выполнять какие-либо команды в случае невыполнения условия, оператор **IF-ELSE** можно еще более сократить, см. Листинг 7.14.

*Листинг 7.14. Оператор **IF-ELSE** без команды **ELSE***

```
if (условие)
   команда_1;
```

Пример программы, использующей оператор условного перехода, приведен в Листинг 7.15. Если передать этому сценарию переменную **name**, значение которой будет «**Vasja**», **http://localhost/if.php?name=Vasja**, то в ответ вы получите текстовую строку **"Здравствуй, Вася, я давно тебя ждал"** (**Рис. 7.26**). Если же переменная **name** будет носить другое значение, вы получите в ответ строку **"Ты не Вася, я тебя не знаю, но все равно Здравствуй!"**.

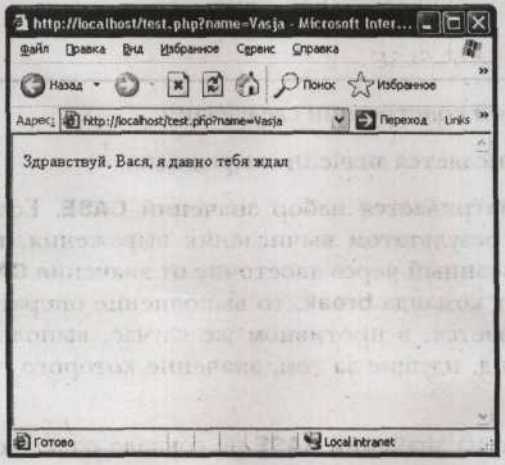


*Рис. 7.26. Вывод программы, использующей оператор **IF-ELSE***

Листинг 7.15. Пример использования оператора ***IF-ELSE***

```
<?
if($name==Vasja)
   echo "Здравствуй, Вася, я давно тебя ждал";
else
   echo "Ты не Вася, я тебя не знаю, но все-равно Здравствуй!";
?>
```

Оператор множественного выбора

Разновидностью оператора условного перехода является оператор множественного выбора – **SWITCH-CASE**. Этот оператор позволяет выполнить одно из множества действий, в зависимости от результата определенного выражения. Этот оператор удобно использовать в тех случаях, когда необходимо реализовать условный переход для пяти и более вариантов. Синтаксис оператора **CASE-SWITCH** приведен в Листинг 7.16.

Листинг 7.16. Синтаксис оператора CASE-SWITCH

```
switch (выражение)
{case значение_1 : команды_1; [break;]
case значение_2 : команды_2; [break;]
....
case значение_N : команды_N; [break;]
[default: команды_по_умолчанию;]
}
```

Алгоритм работы этой конструкции следующий:

- Сначала вычисляется значение выражения.
- Затем просматривается набор значений **CASE**. Если одно из значений совпадает с результатом вычисления выражения, то выполняется блок команд, записанный через двоеточие от значения **CASE**. Если после блока команд стоит команда **break**, то выполнение оператора **SWITCH-CASE** на этом прерывается, в противном же случае, выполняются команды всех блоков команд, идущие за тем, значение которого совпало со значением выражения.
- Если ни одно из значений **CASE** не совпало со значением выражения, то выполняется блок команд **default**, если он есть.

Пример использования оператора **CASE-SWITCH** приведен в Листинг 7.17. Что будет выведено на экран при значении переменной **$animal**, равном «кошка», показано на Рис. 7.27. Чтобы на экран выводилась только одна строка, относящаяся к конкретному животному, нужно использовать команду **break**. Измененный код программы приведен в Листинг 7.18. Полученный результат проиллюстрирован на Рис. 7.28.

Рис. 7.27. Программа с оператором ***CASE-SWITCH***

Листинг 7.17. Пример использования оператора CASE-SWITCH

```
<html>
<head>
   <title>Конструкция SWITCH-CASE</title>
</head>
<body>
   <?
$animal="кошка";
switch ($animal)
{
   case "собака": echo "Гав <br>";
   case "кошка": echo "Мяу <br>";
   case "петух": echo "Ку-ка-ре-ку <br>";
default: echo "Мертвая тишина <br>";
}
   ?>
</body>
</html>
```

Листинг 7.18. Использование оператора CASE-SWITCH с командой break

```
<html>
<head>
   <title>Конструкция SWITCH-CASE</title>
</head>
<body>
   <?
$animal="кошка";
switch ($animal)
{
   case "собака": echo "Гав <br>"; break;
   case "кошка": echo "Мяу <br>"; break;
   case "петух": echo "Ку-ка-ре-ку <br>"; break;
default: echo "Мертвая тишина <br>";
}
   ?>
</body>
</html>
```

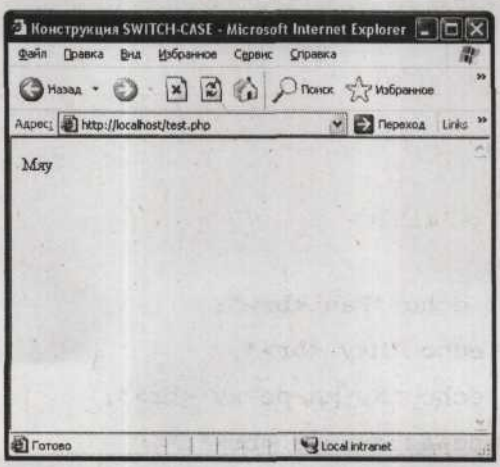


Рис. 7.28. Использование команды ***break***

Циклы

При написании программ часто возникает необходимость повторять одно действие несколько раз подряд, возможно с небольшими изменениями. В таких случаях ис-

пользуются циклы. Цикл состоит из двух основных частей: тела и заголовка. В заголовке цикла задается условие, при выполнении которого исполнение цикла закончится. Тело цикла – команды, выполняемые по кругу, пока условие выхода из цикла не будет выполнено. Каждое выполнение цикла называется итерацией.

Циклы FOR

В языке PHP предусмотрено несколько основных видов циклов. Первый из них – цикл **FOR**, его структура показана в Листинг 7.19. Цикл **FOR** является «циклом со счетчиком». Так называются циклы, в которых есть переменная-счетчик, значение которой изменяется при каждой итерации цикла. При достижении счетчиком определенного значения выполнение цикла заканчивается. Выражение **нач_знач** задает начальное значение переменной-счетчика. Выражение **условие** – логическое выражение, становящееся ложным при определенном значении переменной-счетчика. Когда условие перестает выполняться, цикл завершается. Выражение **приращение** изменяет переменную-счетчик при каждой итерации цикла. Если в теле цикла всего одна команда, то фигурные скобки использовать необязательно. Код реального документа PHP с циклом **FOR** приведен в Листинг 7.20. Получаемый результат представлен на Рис. 7.29.

Листинг 7.19. Синтаксис цикла FOR

```
for (нач_знач; условие; приращение)
{
   тело_цикла
}
```

Листинг 7.20. Пример цикла FOR

```
<html>
<head>
   <title>Цикл FOR</title>
</head>
<body>
   <?
      for ($i=1; $i<=10; $i++)
         echo "Это строчка номер $i <br>";
   ?>
</body>
</html>
```

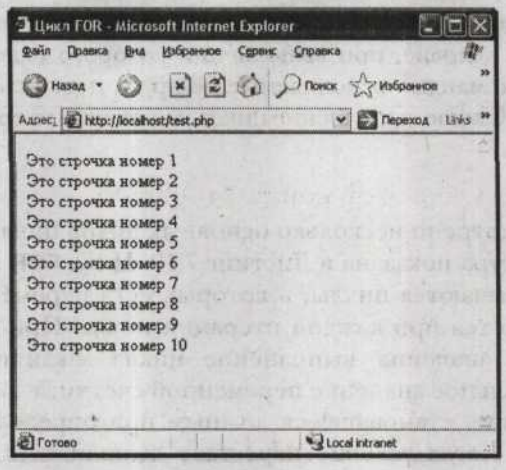


Рис. 7.29. Использование цикла ***FOR***

Циклы WHILE

Другой вид циклов в языке PHP – циклы **WHILE**. В отличие от цикла **FOR**, в заголовке этого вида циклов осуществляется только проверка истинности условия. Структура цикла **WHILE** приведена в Листинг 7.21. Пока значение логического выражения **условие** равняется **True** (истина), выполнение цикла будет продолжаться. Если в качестве условия поставить выражение, истинное при любых обстоятельствах, например **2=2** или просто **True** (истина), то цикл будет выполняться либо бесконечно, либо до тех пор, пока не будет принудительно прерван.

Листинг 7.21. Синтаксис цикла WHILE

```
while(условие)
{
   тело_цикла
}
```

В Листинг 7.22 приведен пример цикла **WHILE**, выполняющего работу, аналогичную циклу **FOR** из Листинг 7.20. Обратите внимание, что задание начального значения переменной цикла и ее приращение осуществляется вне заголовка цикла.

Листинг 7.22. Пример цикла WHILE

```
<html>
<head>
   <title>Цикл WHILE</title>
</head>
<body>
```

```
    <?
        $i=0;
        while ($i<=10)
       {
           echo "Это строчка номер $i <br>";
        $i++;
        }
    ?>
</body>
</html>
```

Циклы DO WHILE

Третьей разновидностью циклов в PHP являются циклы **DO WHILE**. Их отличие от циклов **WHILE** заключается в том, что выполнение условия проверяется уже после выполнения команд цикла. Синтаксис цикла **DO WHILE** показан в Листинг 7.23.

Листинг 7.23. Структура цикла DO WHILE

```
do
{
    тело_цикла
}
while(условие);
```

Изменим тип цикла в программе из Листинг 7.22 с **WHILE** на **DO WHILE**. Полученная программа приведена в Листинг 7.24.

Листинг 7.24. Пример цикла DO WHILE

```
<html>
<head>
    <title>Цикл DO WHILE</title>
</head>
<body>
    <?
        $i=0;
        do
       {
```

```
        echo "Это строчка номер $i <br>";
        $i++;
    }
    while ($i<=10);
  ?>
</body>
</html>
```

Команды, управляющие выполнением циклов

Для управления работой циклов используются команды **break** и **continue**. Команда **break** прерывает выполнение цикла и передает управление первой команде после цикла. Добавим к циклу **WHILE** из Листинг 7.22 условный переход, выполняющий команду **break**, при достижении переменной **$i** значения 5. Полученный код приведен в Листинг 7.25. При достижении переменной **$i**, значения 5, выполнение цикла принудительно прерывается, результат показан на Рис. 7.30.

Листинг 7.25. Пример использования команды break

```
<html>
<head>
   <title>Цикл DO WHILE</title>
</head>
<body>
   <?
      $i=0;
      while ($i<=10)
      {
      if ($i==5)
         break;
      echo "Это строчка номер $i <br>";
      $i++;
      }
   ?>
</body>
</html>
```

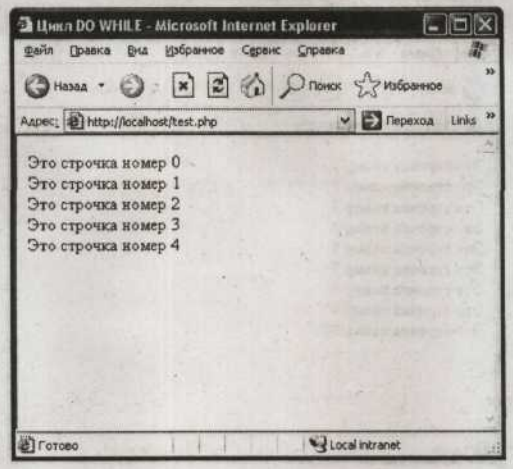


Рис. 7.30. Использование команды ***break*** *в циклах*

Команда **continue** вызывает выполнение следующей итерации цикла, не заканчивая текущую. Добавим в код страницы в Листинг 7.20 условный переход, выполняющий команду **continue**, когда значение **$i** достигнет 5. Полученный код показан в Листинг 7.26. На странице будет пропущена строчка 5, поскольку команда **continue** запустила следующую итерацию цикла, не заканчивая текущую. Получаемый результат показан на Рис. 7.31.

Листинг 7.26. Пример использования команды continue

```
<html>
<head>
   <title>Цикл FOR</title>
</head>
<body>
   <?
      for ($i=1; $i<=10; $i++)
      {
        if ($i == 5)
            continue;
         echo "Это строчка номер $i <br>";

      }
   ?>
</body>
</html>
```

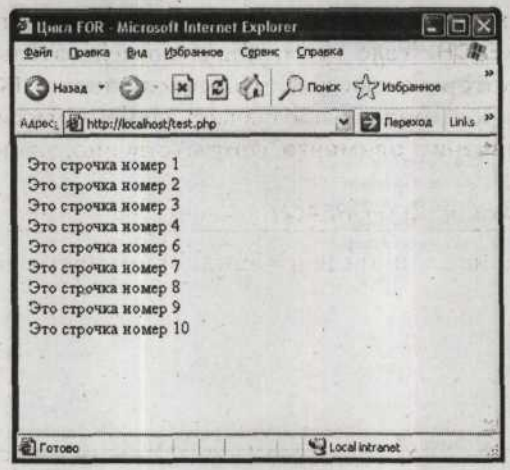


Рис. 7.31. Пример использования команды ***continue***

Цикл FOREACH

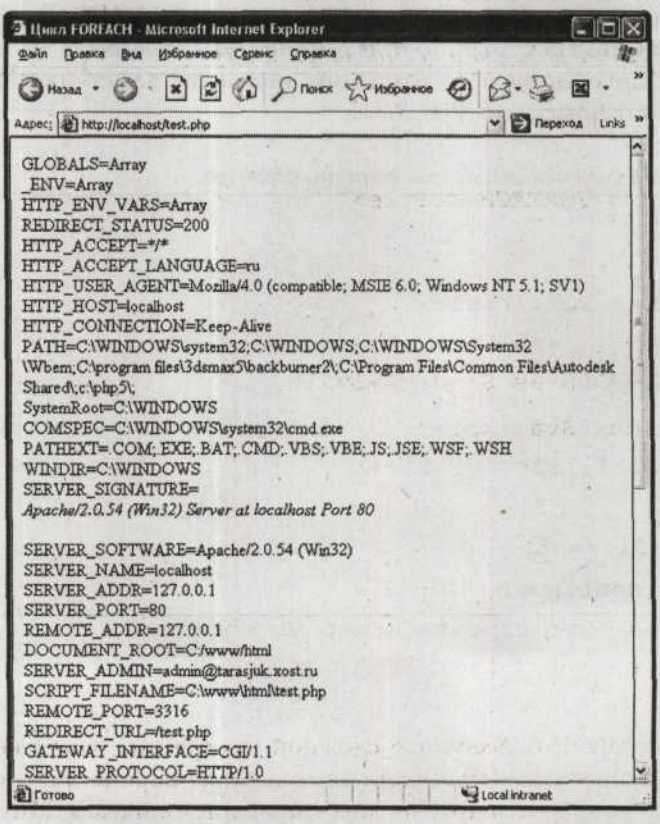


Рис. 7.32. Использование цикла ***FOREACH***

В последних версиях PHP появилась еще одна разновидность циклов: цикл перебора массивов, **FOREACH**. Тело этого цикла последовательно выполняется для всех элементов некоторого массива. Синтаксис цикла **FOREACH** приведен в Листинг 7.27, где строка **индекс** обозначает элемент массива, по которому выполняется цикл, а **значение_элемента**, соответственно, значение элемента.

Листинг 7.27. Синтаксис цикла FOREACH

```
foreach (имя_массива as индекс=>значение_элемента)
{
тело цикла
}
```

В Листинг 7.28 приведен пример использования цикла **FOREACH**. Цикл перебирает значения стандартного массива PHP, **$GLOBALS**, содержащего все глобальные переменные PHP, и выводит названия элементов массива и их значения. Полученный результат отображен на Рис. 7.32.

Листинг 7.28. Пример использования цикла FOREACH

```
<html>
<head>
   <title>Цикл FOREACH</title>
</head>
<body>
   <?
foreach ($GLOBALS as $index=>$val)
   echo "$index=$val <br>"
   ?>
</body>
</html>
```

Массивы

При создании любой, более-менее сложной программы, совершенно необходимыми являются массивы – упорядоченные наборы переменных. Отдельные элементы массива отличаются друг от друга значением индекса, который однозначно определяет место элемента в массиве.

Работа с массивами

В языке PHP массивы являются просто одним из типов переменных. Чтобы создать массив, достаточно присвоить значение одному из его членов, например **$seasons[0]="зима"**. В квадратных скобках указывается значение индекса. Чтобы добавить к массиву новый элемент, нужно точно так же присвоить ему некоторое значение, например, **$seasons[1]="весна"**. Обратиться к элементу массива можно по имени массива и значению индекса – **echo $seasons[0]**.

Если количество элементов в массиве точно известно, то все элементы массива можно последовательно вывести в цикле, как показано в Листинг 7.29.

Листинг 7.29. Вывод элементов массива

```
<html>
<head>
   <title>Конструкция Вывод элементов массива</title>
</head>
<body>
   <?
$seasons[0]="зима";
$seasons[1]="весна";
$seasons[2]="лето";
$seasons[3]="осень";
for ($i=0; $i<4;$i++) echo "$seasons[$i] <br>";
   ?>
</body>
</html>
```

Функция count()

Если точное количество элементов неизвестно, то можно использовать функцию **count(название_массива)**, возвращающую точное количество элементов массива. Перепишем программу, выводящую элементы списка, используя функцию **count()**. Полученная программа приведена в Листинг 7.30.

Листинг 7.30. Вывод элементов массива с использованием функции count()

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

```
   <title>Вывод элементов массива с использованием функции
count()</title>
</head>
<body>
   <?
$seasons[0]="зима";
$seasons[1]="весна";
$seasons[2]="лето";
$seasons[3]="осень";
for ($i=0; $i<count($seasons);$i++) echo "$seasons[$i] <br>";
   ?>
</body>
</html>
```

Автоматическая нумерация элементов массива

Язык PHP поддерживает автоматическую нумерацию элементов массива. Если при создании элемента массива не указывать индекс, например так: **$seasons[]="зима";**, то элементу будет присвоен наименьший незанятый индекс.

Отдельной разновидностью массивов являются ассоциативные массивы. Ассоциативными называются массивы, индексами которых являются строки, например, такие: **$seasons["winter"]="Зима";**. Индекс ассоциативного массива называется ключом. Обращаться к элементам ассоциативного массива можно точно так же, как и к элементам обычного массива, по ключу: **echo $seasons["winter"];**.

Строки

Большинство сценариев на языке PHP работают со строками текста, поэтому строковый тип данных очень важен. Строковые значения можно записывать двумя способами: 1) присваивая им текстовые строки в двойных кавычках; 2) присваивая им текстовые строки в одинарных кавычках. Эти способы значительно отличаются друг от друга.

Дело в том, что строка, заключенная в одинарные кавычки, считается просто текстовой строкой. Все символы, входящие в нее, сохраняются в переменной или передаются функции как есть, без изменений. Дополнительно преобразовываются только сочетания символов «\'» и «\\». Первое сочетание преобразуется в одинарную кавычку, а второе – в обратную наклонную черту, «\».

Строка, заключенная в двойные кавычки, может обрабатывать некоторые специальные сочетания символов, а кроме того, включать значения переменных. Пример программы, показывающей различие между строками с одинарными и двойными кавычками, показан в Листинг 7.31.

Листинг 7.31. Использование двойных и одинарных кавычек

```
<?
$str="Строка";
echo "$str "; //строка в двойных кавычках
echo '$str'; //строка в одинарных кавычках
?>
```

В строках, заключенных в двойные кавычки, могут использоваться следующие специальные сочетания символов:

- **\n** – символ новой строки
- **\r** – символ возврата каретки
- **\t** – символ табуляции
- **\$** – знак доллара
- **\"** – двойная кавычка
- **** – обратная косая черта
- **\xNN** – символ, код которого равняется **NN** в шестнадцатеричной системе.

Строковые операции

В языке PHP предусмотрены две операции, которые можно производить над строками, все остальные действия над строками производятся с помощью различных функций. Первая операция – конкатенация, объединяющая между собой две текстовые строки, **первая_строка.вторая_строка**. И вторая операция – **строка[N]**, выводящая N'ный символ строки. Нумерация символов в строке начинается с нуля. В Листинг 7.32 показан пример использования строковых операций.

Листинг 7.32. Операции над строками

```
<?
$str1="Объединенная ";
$str2="строка";
$res=$str1.$str2;
echo "$res"; //Вывод объединенной строки
echo $res[10]; //Вывод десятого символа строки
?>
```

Вызов внешних программ

Иногда возникает необходимость вызвать из сценария некоторую внешнюю программу и вывести результаты ее работы либо сохранить их в переменной. Для этого используются строки в обратных кавычках «`», на клавиатуре этот знак вводится той же клавишей, что и знак тильда «~». В качестве примера вызовем команду **dir**, которая выводит содержимое текущего каталога. Код программы, выводящей HTML-страницу со списком файлов в директории, приведен в Листинг 7.33. Результирующая страница показана на Рис. 7.33.

Листинг 7.33. Вызов внешней программы

```
<html>
<head>
    <title>Вызов программы</title>
</head>
<body><pre>
<? echo `dir`;?>
</pre>
</body>
</html>
```

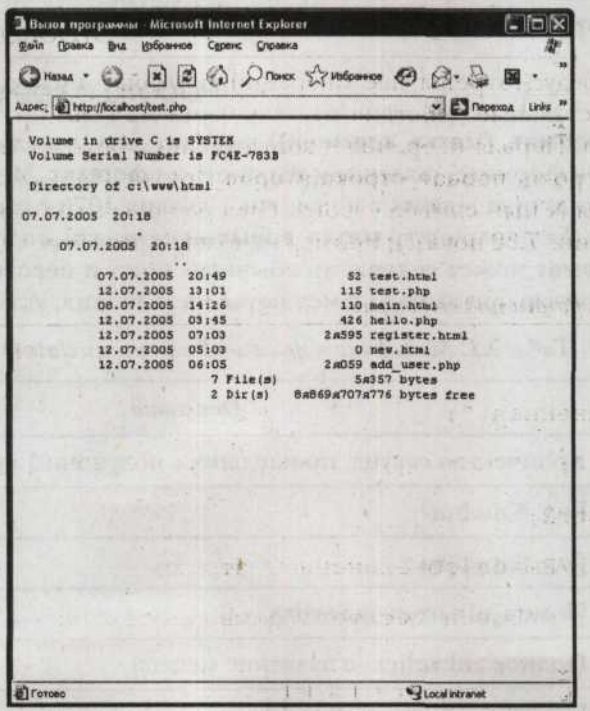


Рис. 7.33. Результаты вызова внешней программы

Основные функции работы со строками

Функция **strlen(строка);** возвращает количество символов, содержащееся в строке. В Листинг 7.34. приведен пример использования этой функции.

Листинг 7.34. Определение длины строки

```
$str="Некоторая строка";
echo strlen($str); //Эта команда выведет число 16
```

Функция **strpos(строка, подстрока, [ст_поз])** или осуществляет поиск подстроки **подстрока** в строке, начиная с позиции **ст_поз**. Если значение **ст_поз** не указывать, то поиск будет осуществляться с нулевой позиции. Функция возвращает номер позиции в строке, с которой начинается вхождение подстроки, а в том случае если подстрока не найдена, функция вернет значение **False** (Ложь). Результатом выполнения команды **strpos("This is a string","str");** будет число 10.

Функция **substr(строка, ст_поз, длина)**, возвращает подстроку строки **строка**, начиная с символа под номером **ст_поз** и длиной в **длина** символов. Результатом выполнения функции **substr("Это строка", 1,2);**, будет строка «то».

Стандартные функции PHP

Далее рассмотрены некоторые стандартные функции языка PHP, полезные при написании сценариев.

Работа с временем и датой

Функция **date (формат, [метка_времени])** выводит дату, закодированную числом **метка_времени**, в формате, указанном параметром **формат**. Метка времени является числом секунд, прошедших с полуночи 1 января 1970 года, времени основания системы Unix. Если параметр **метка_времени** не указан, то выводится текущая дата. Строка **формат** может содержать обычный текст и некоторые символы модификаторы, которые при выводе заменяются на значения, указанные в Табл.7.1

Табл. 7.1. Модификаторы вывода функции ***date()***

Модификатор	*Описание*
U	Количество секунд, прошедших с полуночи 1 января 1970 года
Y	Год, 4 цифры
y	Год, 2 цифры
z	Номер дня от начала года
F	Полное английское название месяца
m	Номер месяца, от 01 до 12

Модификатор	*Описание*
n	Номер месяца без предваряющего нуля, от 1 до 12
M	Трехбуквенное английское сокращение названия месяца, например «Jan»
d	Номер дня в месяце, от 01 до 31
j	Номер дня в месяце, без предваряющего нуля, от 1 до 31
l	Английское название дня недели, например «Sunday»
w	Номер дня недели, 0 – Воскресенье и т. д.
D	Трехбуквенное английское сокращение дня недели, например «Sun»
A	Отметка «До» или «После» полудня, «AM» либо «PM»
a	Отметка «До» или «После» полудня, «am» либо «pm»
H	Часы в 24-часовом формате, от 0 до 23
h	Часы в 12-часовом формате, от 0 до 11
i	Минуты, от 00 до 59
s	Секунды, от 00 до 59

Пример использования функции **date**: **echo date ("Сегодняшняя дата: d.m.Y");**. В результате будет выведена строка, содержащая текущие день, месяц и год.

Функция **time()** возвращает «метку времени» текущего момента. Эта функция позволяет «законсервировать» текущие дату и время, чтобы затем можно было, например, вывести их с помощью функции **date()**.

Функция **mktime([часы [, минуты [, секунды [, месяц [, день [, год [, летнее_время]]]]]]])** возвращает метку времени для указанной даты. Все параметры функции являются необязательными. Параметр **летнее_время** устанавливается равным 1, если дата относится к летнему времени, 0 – в противном случае и -1, если эта информация неизвестна. Пример: команда echo **date("M-d-Y", mktime(0, 0, 0, 1, 1, 1998))** вернет значение «Jan-01-1998»;

Генератор случайных чисел

Для генерирования случайных чисел в языке PHP используется функция **mt_rand(минимальное_значение, [максимальное_значение])**. Функция **mt_rand()** возвращает некоторое случайное значение, в диапазоне от целого числа **минимальное_значение** до целого числа **максимальное_значение**. Если **максимальное_значение** не задано, то вместо него используется значение по умолчанию – константа **RAND_MAX**. Ее значение можно узнать с помощью функции **mt_getrandmax()**.

Программа, представленная в Листинг 7.35, выводит последовательность из пяти чисел, в диапазоне от 20 до 50.

Листинг 7.35. Использование генератора случайных чисел

```
<html>
<head>
   <title>Использование генератора случайный чисел</title>
</head>
<body>
   <?
      for($i=0;$i<=5;$i++)
      {
         echo mt_rand(20,50);
         echo "<br>";
      }
   ?>
</body>
</html>
```

Математические функции

Математические функции PHP достаточно просты в использовании, поэтому сведем их описания в Табл. 7.2.

Табл. 7.2. Математические функции

Функция	*Описание*
Min (числовые_аргументы)	Возвращает значение наименьшего из переданных функции аргументов, в функцию передается несколько числовых аргументов. Также в качестве аргумента может быть числовой массив, в этом случае функция возвращает значение наименьшего из элементов массива
Max (числовые_аргументы)	Возвращает значение наибольшего из переданных функции аргументов, в функцию передается несколько числовых аргументов. Также в качестве аргумента может быть числовой массив, в этом случае функция возвращает значение наибольшего из элементов массива

Функция	*Описание*
acos(аргумент)	Возвращает арккосинус аргумента функции
asin(аргумент)	Возвращает арксинус аргумента функции
atan(аргумент1, аргумент2)	Возвращает арктангенс числа **аргумент1/ аргумент2**. Результат функции возвращается в радианах
sin(аргумент)	Возвращает синус аргумента функции. Аргумент задается в радианах
cos(аргумент)	Возвращает косинус аргумента функции. Аргумент задается в радианах
tan(аргумент)	Возвращает тангенс аргумента функции. Аргумент задается в радианах
pi()	Возвращает число Пи
sqrt(аргумент)	Возвращает квадратный корень аргумента функции Аргумент должен быть неотрицательным
log(аргумент)	Возвращает натуральный логарифм аргумента функции
exp(аргумент)	Возвращает экспоненту аргумента функции
pow(число,степень)	Возвращает результат возведения числа **число** в степень степень

Пользовательские Функции

Последовательность из нескольких команд сценария можно объединить между собой, создав функцию. После этого команды функции можно выполнить, вызвав созданную функцию по названию. В функции можно свести неоднократно используемые блоки кода, чтобы затем вместо повторения в теле программы этого кода просто вызывать нужные функции. Кроме того, в функции можно вынести последовательности команд, дающие некий законченный результат, – это позволяет улучшить структуру программы.

Создание новых функций

Функции создаются по образцу, показанному в Листинг 7.36. **Имя_функции** – название, по которому функция будет вызываться в дальнейшем. **Параметры_функции** – переменные, значение которых передается в функцию. Внутри функции команды могут оперировать этими переменными.

Листинг 7.36. Структура функции

```
function имя_функции (параметры_функции)
{
   код_функции
}
```

Вызов функций

В качестве результата своей работы функция может возвращать некоторое значение, которое можно присваивать переменным, использовать в вычислениях и т. д. Вызов функции происходит следующим образом: **имя_функции (параметры_функции);**. Можно присвоить значение функции переменной: **переменная = имя_функции (параметры_функции);**. Значение функции присваивается с помощью оператора **return**. Используется этот оператор так: **return значение_функции;**.

Пример использования пользовательской функции

В Листинг 7.37 показано практическое использование функции, на примере функции, возводящей число **$a** в целую положительную степень **$b**. Код PHP страницы последовательно возводит двойку в степени от 0 до 10. Результаты работы этой программы приведены на Рис. 7.34.

Листинг 7.37. Пример использования функции

```
<html>
<head>
   <title>Использование функций</title>
</head>
<body>
   <?
function powerb ($a, $b){
      $atemp=$a;
      if ($b>1){
         for ($i=2;$i<=$b;$i++)
            $atemp*=$a;
      }
      if($b==0)
         $atemp=1;
      return $atemp;
   }
   $i=0;
```

```
    while($i<=10){
    echo "2 в степени $i равняется ";
    echo powerb(2, $i);
    echo "<br>";
    $i++;
    }
    ?>
</body>
</html>
```

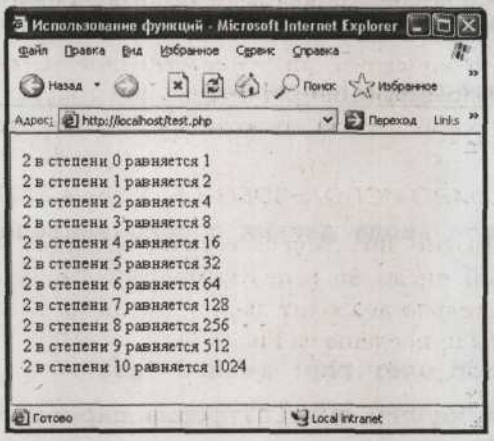


Рис. 7.34. Использование пользовательской функции

Имена функций

Имена функций в PHP должны соответствовать следующим требованиям:

- В принципе, имя функции в PHP может состоять из любых символов, кроме пробелов, включая буквы русского алфавита. Но это может привести к ряду различных несовместимостей с другими программами, различными кодировками языков и т. д. Поэтому возьмите за правило ограничиваться в именах переменных только символами латинского алфавита, цифрами и знаком подчеркивания.
- PHP не различает строчные и прописные буквы в названии функции: **Abc()** и **ABC()** – одна и та же функция.

Практическое использование HTML-форм и серверных сценариев

В завершение этой главы, мы рассмотрим практический пример применения серверных технологий. Мы создадим сценарий, который будет обрабатывать па-

раметры, полученные от HTML-страницы, и создавать на их основе другие HTML-страницы. Таким образом, наша задача разбивается на две части. Первая часть – создание HTML-страницы с формой, которую будет заполнять пользователь. И вторая – создание сценария PHP, обрабатывающего полученные данные.

Форма HTML передачи информации на сервер

В качестве документа HTML, передающего данные сценарию, используем немного измененную HTML-страницу регистрации пользователя, заполненную нами в прошлой главе. Исходный код ее приведен в Листинг 7.38. Важные детали ее устройства приведены в комментариях к коду. Сохраните этот HTML-документ названием **register.html** в папке HTML-документов сервера Apache, **c:/web/html**. Полученный HTML-документ показан на Рис. 7.35.

Листинг 7.38. Форма регистрации пользователя

```
<html>
<head>
   <title>Форма для ввода данных о пользователе</title>
</head>
<body>
  <form action="add_user.php" method="get">
  <!--При нажатии кнопки submit, форма передает все данные сценарию add_user.php-->
      <h2>Регистрация нового пользователя</h2>
      <hr>
      <b>ФИО: </b> <input type="text" name="name" size="40" maxlength="70"><br>
<!-- ФИО сохраняется в текстовой переменной name, максимальная ее длина 70 знаков-->
      <b>Пол: </b>Мужской <input type="radio" name="sex" value="male" checked>Женский<input type="radio" name="sex" value="female"><br>
<!-- В зависимости от положения переключателя, переменная sex принимает значение "male" либо "female"-->
      <b>Дата рождения: </b> День <input type="text" name="b_day" size="2" maxlength="2">
      Месяц <select name="b_month" size="1">
         <option value="Январь" SELECTED>Январь</option>
         <option value="Февраль">Февраль</option>
```

```
        <option value="Март">Март</option>
        <option value="Апрель">Апрель</option>
        <option value="Май">Май</option>
        <option value="Июнь">Июнь</option>
        <option value="Июль">Июль</option>
        <option value="Август">Август</option>
        <option value="Сентябрь">Сентябрь</option>
        <option value="Октябрь">Октябрь</option>
        <option value="Ноябрь">Ноябрь</option>
        <option value="Декабрь">Декабрь</option>
      </select>
      Год <input type="text" name="b_year" value="1980"
size="4" maxlength="4">
      <hr>
      <b>Логин:</b>  <input type="text" name="login" size="10"
maxlength="10"> <b>Пароль:</b>  <input type="password"
name="pass" size="10" maxlength="10"><br>
      <b>Адрес e-mail:</b> <input type="text" name="email"
size="15" maxlength="40">
      <input type="checkbox" name="show_mail" value="show"> По-
казывать e-mail другим пользователям сайта <br>
      <hr>
      <b>Увлечения:</b><br>    <select
name="hobbies[]" size="4" multiple>
<!-- После переменной hobbies добавлены квадратные скобки, это
сигнал интерпретатору PHP, что сценарию передается переменная-
массив с несколькими значениями-->
        <option value="Спорт" >Спорт</option>
        <option value="Музыка">Музыка</option>
        <option value="Путешествия">Путешествия</option>
        <option value="Кино">Кино</option>
        <option value="Фотография">Фотография</option>
        <option value="Компьютерные игры">Компьютерные иг-
ры</option>
        <option value="Настольные игры">Настольные иг-
ры</option>
        <option value="Азартные игры">Азартные игры</option>
```

```
        <option value="Рисование">Рисование</option>
        <option value="Хоровое пение">Хоровое пение</option>
      </select><br><br>
      <b>Дополнительная информация:</b>
<br>   <textarea cols="40" rows="8"
name="dop"></textarea><br>
      <hr>
      <input type="submit" value="Зарегистрировать"> <input
type="reset" name="Reset" value="Отменить ввод">
   </form>
</body>
</html>
```

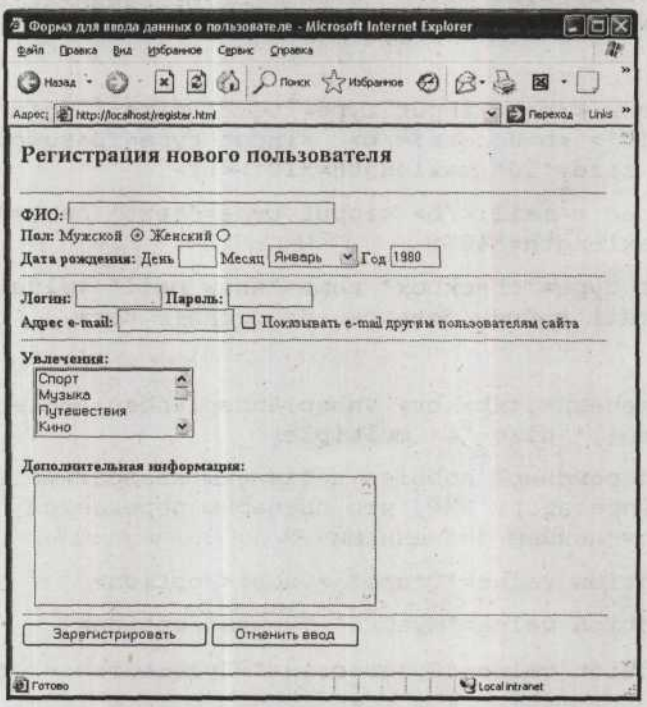


Рис. 7.35. Форма для ввода информации о пользователе

Сценарий PHP, обрабатывающий запросы

После нажатия кнопки «Зарегистрировать» данные из формы передаются сценарию **add_user.php**, также находящемуся в каталоге **c:/web/html**. Код этого сценария приведен в Листинг 7.39. Все важные участки сценария снабжены комментариями, поэтому разобраться в нем вам будет несложно.

Листинг 7.39. Сценарий, обрабатывающий данные из формы

```
<?
if ($name!=""&&$login!=""&&pass!="")
//Проверка, заполнены ли поля с ФИО, логином и паролем
{
//Если заполнены, то начинается формирование страницы HTML,
//сообщающей об успешной регистрации.
   echo "<html> <head>  <title>Регистрация успешно
завершена</title> </head> <body> <h1 align=\"center\">";
//Обратите внимание, кавычки в строке вывода предваряются
//знаком \.
   echo "Вы успешно зарегистрированы на нашем сайте</h1>
<h2>Ваши данные:</h2>  <hr> <b>Время регистрации:</b> ";
   echo date ("D.m.Y H:i:s");
// Вывод времени и даты регистрации.
   echo "<br> <b>ФИО:</b> $name";
// Вывод ФИО
   echo "<br> <b>Пол: </b>";
   if ($sex="male") echo "Мужской<br>";
   else echo "Женский <br>";
/* Если в HTML-форме переключатель «Пол» был установлен в поло-
жение Мужской, то переменная $sex, имеет значение "male". Если
эта переменная имеет другое значение, то пол, соответственно,
женский.*/
   echo "<b>Дата рождения:</b>";
   $birthday=$b_day." ".$b_month." ".$b_year;
//Объединяем три переменные, содержащие дату рождения, в одну
//строку.
   echo "$birthday <br> <hr> <b>Логин:</b>";
//И выводим ее.
   echo "$login <br> <b>Пароль:</b>";
   $starpass="";
   for ($i=0;$i<strlen($pass);$i++) $starpass=$starpass."*";
/*Поскольку пароль необходимо выводить на страницу в виде звез-
дочек, создаем строковую переменную $starpass и добавляем в эту
строку столько звездочек, сколько символов в пароле*/
   echo "$starpass <br> <b>Адрес email:</b>";
   echo "$email <br> <b>Показ адреса e-mail другим пользовате-
лям сайта :</b>";
```

```
   if (isset($show_mail)) echo "Да";
   else echo "Нет";
/*Если флажок "Показ адреса e-mail другим пользователям сайта",
в HTML-форме был сброшен, то переменная, передающая значение
флажка в сценарий, не создавалась. Если же эта переменная суще-
ствует, значит флажок был установлен. */
   echo "<br> <hr> <b>Увлечения:</b>";
   for ($i=0;$i<count($hobbies);$i++)
   {
     if ($i>0) echo", ";
     echo "$hobbies[$i]";
   }
/* Последовательно выводим элементы массива $hobbies, содержа-
щего информацию об увлечениях. Перед всеми, кроме первого, эле-
ментами массива вставляем запятые.*/
   echo ".<br> <b>Дополнительная информация:</b>";
//Вывод дополнительной информации о пользователе.
   echo "$dop <br><hr> В дальнейшем вы сможете изменить ваши
регистрационные данные, щелкнув мышью на кнопке ";
   echo  "<b>Профиль</b>, главного меню сайта. А сейчас, перей-
дите на   <a href=\"index.html\">главную страницу</a>";
   echo "</body></html>";
}
else
{
 if ($name=="")
//В том же случае, если не было введено ФИО пользователя...
{
 echo "<html> <head>  <title>Регистрация не выполнена</title>
</head> <body>";
// Выводим HTML-страницу, сообщающую об ошибке регистрации
 echo "<div align=\"center\"> Не введено ФИО. Щелкните на ссыл-
ке \"Ввести данные\", чтобы заполнить форму снова. </div><hr>";
  echo "<div align=\"center\"><a href=\"register.html\">Ввести
данные</a></div></body></html>";
//И предлагаем пройти процедуру регистрации заново.
 }
 else
 {
```

```
//Если ФИО введено, но не введены логин, либо пароль
  echo "<html> <head>  <title>Регистрация не выполнена</title>
</head> <body>";
// Выводим HTML-страницу, сообщающую об этой ошибке регистрации
  echo "<div align=\"center\"> Не введены логин или пароль.
Щелкните на ссылке \"Ввести данные\", чтобы заполнить форму
снова. </div><hr>";
  echo "<div align=\"center\"><a href=\"register.html\">Ввести
данные</a></div></body></html>";
//И, опять же, предлагаем пройти процедуру регистрации заново
 }
}
?>
```

Рассмотрение полученных результатов

Возможны три варианта выполнения сценария **add_user.php**. Далее все они описаны по порядку.

Успешное завершение регистрации

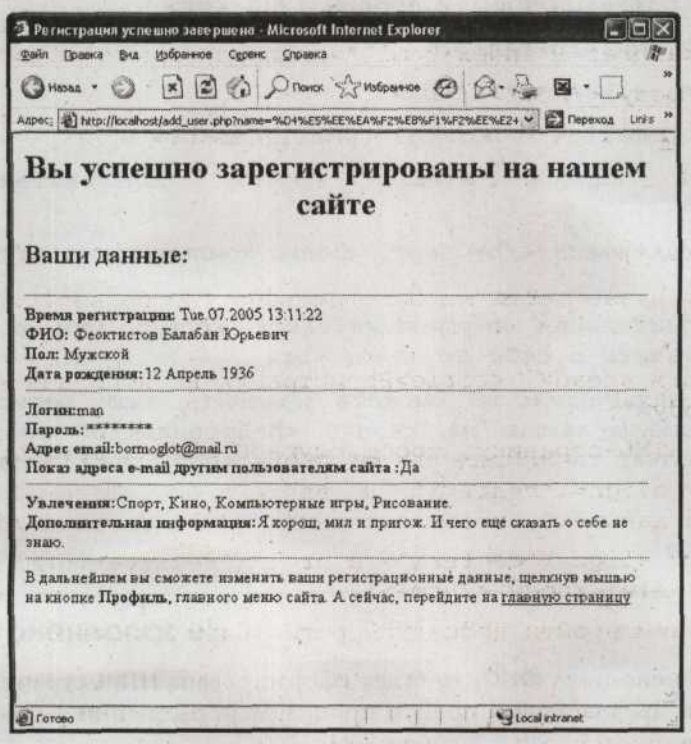


Рис. 7.36. HTML-страница, информирующая об успешном завершении регистрации

Если все необходимые поля формы заполнены, сценарий формирует HTML-страницу, информирующую об этом пользователя (Рис. 7.36). Код этой страницы приведен в Листинг 7.40.

*Листинг 7.40. **HTML**-страница, информирующая об успешном завершении регистрации*

```
<html>
<head>
   <title>Регистрация успешно завершена</title>
</head>
<body>
   <h1 align="center">Вы успешно зарегистрированы на нашем сайте</h1>
   <h2>Ваши данные:</h2>
   <hr>
   <b>Время регистрации:</b> Tue.07.2005 06:51:28<br>
   <b>ФИО:</b> Феоктистов Балабан Юрьевич<br>
   <b>Пол: </b>Мужской<br>
   <b>Дата рождения:</b>12 Апрель 1936 <br>
   <hr><b>Логин:</b>man<br>
   <b>Пароль:</b>********<br>
   <b>Адрес email:</b>bormoglot@mail.com<br>
   <b>Показ адреса e-mail другим пользователям сайта :</b>Да<br>
   <hr><b>Увлечения:</b>Спорт, Кино, Компьютерные игры, Рисование.<br>
   <b>Дополнительная информация:</b>Я хорош, мил и пригож. И чего еще сказать о себе не знаю. <br>
   <hr> В дальнейшем вы сможете изменить ваши регистрационные данные, щелкнув мышью на кнопке <b>Профиль</b> главного меню сайта. А сейчас перейдите на <a href="index.html">главную страницу</a>
</body>
</html>
```

Не заполнено поле ФИО

Если не заполнено поле **ФИО**, то будет сформирована HTML-страница, сообщающая об этом и предлагающая пройти процедуру регистрации заново (Рис. 7.37). Код этой страницы показан в Листинг 7.41

Листинг 7.41. ***HTML****-страница, информирующая о незаполненном поле* ***ФИО***

```
<html>
<head>
   <title>Регистрация не выполнена</title>
</head>
<body>
   <div align="center">
      Не введено ФИО. Щелкните на ссылке "Ввести данные", чтобы
заполнить форму снова.
   </div>
   <hr>
   <div align="center">
      <a href="register.html">Ввести данные</a>
   </div>
</body>
</html>
```

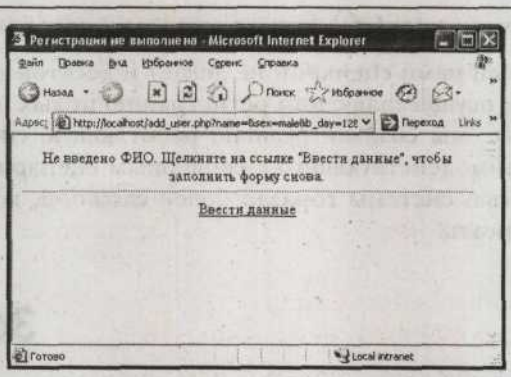


Рис. 7.37. HTML-страница, информирующая о не заполненном поле ***ФИО***

Не введен логин либо пароль

Если же не введен логин либо пароль, будет сформирована HTML-страница, сообщающая об этом (Рис. 7.38). Код этой страницы приведен в Листинге 7.42

Листинг 7.42. ***HTML****-страница, информирующая о незаполненных полях логина либо пароля*

```
<html>
<head>
   <title>Регистрация не выполнена</title>
```

```
</head>
<body>
   <div align="center"> Не введены логин или пароль. Щелкните
на ссылке "Ввести данные", чтобы заполнить форму снова.
   </div><hr><div align="center">
      <a href="register.html">Ввести данные</a>
   </div>
</body>
</html>
```

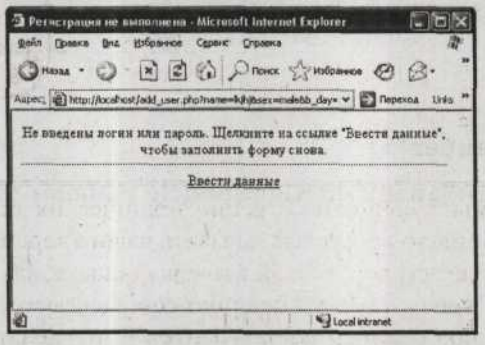


Рис. 7.38. HTML-страница, информирующая о незаполненных полях логина или пароля

Разумеется, созданный нами сценарий не делает и десятой доли того, что делает реальный сценарий, осуществляющий регистрацию новых пользователей на сайте. Но, тем не менее, мы создали реально работающую систему, состоящую из HTML-документа, взаимодействующего с серверным сценарием PHP, которая легко может стать частью системы гораздо более сложной, которую, я уверен, вы вскоре сможете написать.

Заключение

Этой главой завершилось изучение таинств создания динамических Web-страниц. Несколько глав назад мы начали с изучения клиентских сценариев JavaScript, затем рассмотрели способы взаимодействия между сервером и браузером пользователя и, наконец, объединили воедино HTML-страницы и серверные сценарии. Теперь вам известны практически все основные приемы создания сайтов любой степени сложности. Конечно, вы еще не знаете некоторых тонкостей, которые, возможно понадобятся вам в будущем, но после пройденного вами пути вы их без проблем осилите.

В следующей главе вы узнаете о том, что же делать с Web-сайтом после того, как он создан, как разместить ваш Web-проект в сети, познакомить с ним максимальное количество пользователей, а также о многом другом.

глава 8

Выгрузка сайта в Web и его продвижение

В предыдущей главе мы закончили изучение технических таинств создания сайтов для сети Интернет. Пришло время выкладывать ваши творения в естественную среду обитания, на Интернет-сервер. В этой главе вы узнаете, как можно выложить Web-сайт в Интернет, как заявить всему Интернет-сообществу о его существовании, как отвоевать свое место под солнцем у конкурентов и что делать с завоеванными территориями, чтобы, не потерять их. Этой главой заканчивается изучение премудростей Web-мастеринга и дальше начинается реальная жизнь в сети Интернет.

Варианты размещения Web-сайта в сети Интернет

Итак, ваш сайт готов, настала пора познакомить с ним обитателей сети Интернет. Как это сделать? Существует множество способов выложить ваш сайт в сети Интернет.

Виртуальный хостинг

Вашему сайту выделяется каталог на Web-сервере, из которого он и запускается. Название хостинг происходит от слова хост (от англ. Host – главный, хозяин). Хостом называется сервер, на котором размещается каталог с сайтом. Таких каталогов на сервере может быть несколько сотен и даже тысяч, но работать эта система будет так, будто для сайта выделен отдельный сервер, поэтому к слову хостинг часто добавляется слово «виртуальный».

Преимуществом такой системы является доступность. В зависимости от условий, предоставляемого дискового объема, ежемесячного Интернет-трафика и наличия различных дополнительных услуг, виртуальный хостинг стоит от пяти долларов в месяц. Некоторые хостинг-провайдеры, т. е. фирмы, занимающиеся предоставлением хостинга, предоставляют виртуальный хостинг и вовсе бесплатно.

Недостатки виртуального хостинга являются продолжением его достоинств. Вычислительную мощность сервера и пропускную способность канала, по которому он подключен к Интернету, вашему сайту придется делить еще с несколькими сотнями конкурентов. Поэтому скорость выполнения сценариев на вашем сайте и быстрота доступа к его страницам могут желать лучшего. В таких случаях лучше воспользоваться одним из других способов размещения сайта в Интернете.

Другим недостатком виртуального хостинга является зависимость от программного обеспечения, установленного на сервере. Вы не сможете поменять Web-сервер, установленный в системе, подключить поддержку дополнительных языков серверных сценариев и т. д.

Выделенный сервер

Этот вариант позволяет избавиться от большинства недостатков виртуального хостинга. Для сайта выделяется отдельный сервер, на который устанавливается программное обеспечение, по выбору владельца сайта.

Основными недостатками этого способа размещения сайтов являются достаточно высокая цена такого решения и необходимость использовать только то оборудование, которое предоставляет хостинг-провайдер. Кроме того, свобода в выборе программного обеспечения сервера приносит с собой необходимость это программное обеспечение устанавливать и настраивать, так как хостинг-провайдер несет в данном случае ответственность только, аппаратную часть сервера.

Выделенные серверы используются, как правило, для размещения достаточно крупных коммерческих проектов, окупающих затраты на поддержание сервера и его администрирование. Еще одним возможным случаем использования выделенного сервера является размещение на нем нескольких Web-проектов, работающих одновременно. Использование отдельного сервера позволяет более гибко контролировать работу каждого из проектов по сравнению с использованием виртуального хостинга.

Совместное размещение

Совместное размещение (на сленге – «co-location» – совместное размещение) – это подключение сервера, принадлежащего заказчику, к Интернет-каналу хостинг- провайдера. Содержимое сервера, как программное, так и аппаратное полностью определяется потребностями владельца сайта или сайтов, размещенных на сервере. Хостинг-провайдер предоставляет только выход в Интернет, электропитание сервера и физическое пространство для размещения сервера.

Совместное размещение позволяет удовлетворить потребности любого, самого взыскательного сайтовладельца. Но этот вариант приносит с собой и некоторое количество проблем, связанных с тем, что обслуживание сервера полностью ложится на плечи владельца сайта, за работоспособность оборудования и программ хостинг-провайдер ответственности не несет.

Используется совместное размещение практически в тех же случаях, что и выделенный сервер: для крупных проектов с высокой посещаемостью либо для размещения нескольких Web-сайтов одного владельца.

Размещение сайта на своем компьютере

Одним из вариантов обеспечения доступа к своему сайту из Интернета, является размещение его на своем компьютере, подключенном к Сети.

Преимущества такого решения ясны. Не нужно платить за оборудование хостинг-провайдера и за размещение сервера, только за Интернет трафик. Можно устанавливать любое программное и аппаратное обеспечение без каких-либо ограничений. Сервер находится всегда под рукой, а не у хостинг-провайдера. И т. д.

Минусов у такого варианта тоже достаточно много. Необходимо быстрое и постоянное подключение к Интернету с выделенным IP-адресом. Нужно постоянно держать компьютер включенным. Нужно выделять часть ресурсов компьютера на обслуживание сервера и т. д.

На своем личном компьютере имеет смысл размещать только Web-сайты, для которых не будет критичным периодическое его пропадание из сети, при отключении электричества и перезагрузках. Также не стоит размещать таким образом сайты с большой посещаемостью и сложными серверными сценариями, иначе нагрузка на компьютер будет слишком велика.

Особенности бесплатного хостинга

Самым простым и популярным способом размещения Web-сайтов в сети был и остается виртуальный хостинг. Это решение подходит для большинства случаев, именно этот вариант мы и рассмотрим подробнее.

Как уже отмечалось ранее, виртуальный хостинг бывает платным и бесплатным. Разумеется, ничего полностью бесплатного не существует и бесплатный хостинг на самом деле таким не является. Услуги бесплатных хостинг-провайдеров приходится оплачивать не деньгами, а чем-то другим.

Чаще всего за бесплатный хостинг приходится платить размещением рекламы на своем сайте. К тому же, хостинг-провайдер не отвечает ни за что, качество услуг может быть соответствующим.

Существует и другой вариант, когда бесплатный хостинг идет в дополнение к какой-нибудь другой услуге или товару. Например, большинство Интернет-провайдеров в числе прочих услуг предлагают своим клиентам бесплатно разместить Web-страницу. Этого, кстати, делать не стоит, поскольку в том случае, если вы поменяете Интернет-провайдера, вы потеряете и свою Web-страницу.

Кроме того, большинство хостинг-провайдеров предлагает на своих бесплатных тарифных планах очень маленький набор дополнительных услуг, маленький размер дискового пространства, ограничение на размер файлов и их типы. Эти ограничения вводятся, в том числе и для того, чтобы пользователь перешел, со временем, на платный тарифный план, не меняя хостинг-провайдера.

Бесплатный хостинг идеально подходит для размещения ваших первых Web-сайтов, поскольку многочисленные эксперименты с сайтами и первые «блины комом», которых вы вряд ли избежите, не будут вам стоит ничего. В дальнейшем,

когда эксперименты закончатся, вы сможете перейти на платный хостинг либо вообще поставить выделенный сервер, но для некоторых проектов можно так и ограничиться бесплатным хостингом. Если вы планируете впоследствии поменять хостинг-провайдера, то выбирайте хостинг с поддержкой сторонних доменов второго уровня, чтобы поменять сервер с минимальными проблемами.

Рекомендации по использованию бесплатного хостинга

Чтобы в дальнейшем у вас не возникало затруднений и сложностей, четко оговорим область применения бесплатного хостинга. Бесплатный хостинг можно применять для:

- Экспериментов. Это как раз то, чем мы с вами будем заниматься в этой главе.
- Домашних страниц. Если вы создаете страничку для неформальных целей, чтобы познакомить Интернет-сообщество с собой, своими увлечениями и пристрастиями и т. д., нет необходимости за это платить деньги. Если человек захотел узнать о вас побольше, он, скорее всего, не будет очень расстраиваться и не уйдет рассержено, если ваш сайт будет грузиться не слишком быстро, а в углу примостится рекламный баннер.
- Некоммерческих проектов. Если проект не приносит денег, то желательно, чтобы он и тратил их как можно меньше, поэтому бесплатный хостинг в таком случае является хорошим решением. Хотя, если посещаемость вашего ресурса будет достаточно велика, можно вместо рекламы, вставляемой бесплатным хостингом, повесить на сайте рекламный баннер и оплачивать из вырученных денег услуги платного хостинга.
- Фан-ресурсов на начальной стадии работы. Допустим, вы создали сайт, посвященный вашему увлечению, вы знаете, что ваше увлечение разделяют и другие люди, и им будет интересно посещать ваш сайт. Но пока о нем еще никто не знает и на уровень окупаемости, например за счет показа рекламы, сайт выйдет нескоро. В этом случае, можно сначала разместить его на сервере бесплатного хостинг-провайдера, а затем, после повышения популярности ресурса, перейти на платный хостинг. Хотя лучшим вариантом будет все-таки и начинать с платного хостинга.

Бесплатный хостинг не следует использовать для:

- Солидных коммерческих и официальных сайтов – представительств фирм, общественных организаций, государственных учреждений. Доменный адрес, принадлежащий бесплатному хостинг-провайдеру, наверняка подорвет доверие к солидной фирме, которая экономит несколько долларов на своем сайте. Кроме того, хозяева бесплатных хостингов, как правило, не очень хорошо относятся к размещению коммерческих сайтов на их территории.
- Крупных проектов. Бесплатный сервер просто захлебнется от наплыва посетителей и одновременного запуска кучи серверных сценариев.

- Файловых архивов. Хостинг-провайдеры не жалуют такой способ использования бесплатного Web-пространства и всячески противодействуют ему. Ограничивают максимальный размер файлов, запрещают закачивать файлы с определенными разрешениями и т. д.

Требования, предъявляемые к серверу бесплатного хостинга

Итак, для начала нам нужен бесплатный хостинг. Давайте разберемся, какие основные требования нужно предъявлять к бесплатным хостингам, на что следует обратить внимание.

Скорость работы сервера

Это требование понятно, чем быстрее сервер будет реагировать на запросы пользователя, тем ниже вероятность, что пользователь закроет Web-сайт, не дождавшись загрузки. Скорость работы определяется экспериментальным путем. Либо создайте тестовую страницу на сервере и проверьте ее работу, либо походите по страницам, уже размещенным на этом бесплатном хостинге.

Ограничения на типы и размеры файлов

Многие бесплатные хостинг-провайдеры устанавливают ограничение на размер размещаемых файлов и их тип. Чаще всего, Web-страницы содержат достаточно небольшие HTML-файлы и картинки, поэтому в таких ограничениях нет ничего страшного.

Но не исключен вариант, что вам понадобится разместить на своем сайте несколько музыкальных композиций в формате **mp3**, видеоклипы или что-нибудь столь же объемистое. В таком случае при выборе бесплатного хостинга обратите внимание на условия размещения файлов.

Место, выделяемое под сайт

Среднестатистический Web-сайт, особенно недавно начавший работу, занимает довольно мало места, и двадцати мегабайт минимального объема, предлагаемого бесплатными хостинг-провайдерами, вам скорее всего хватит с хорошим запасом. К тому же, на многих бесплатных хостингах имеется возможность впоследствии увеличить предоставляемый объем дискового пространства. Поэтому этот параметр не слишком важен, смотрите только, чтобы вам не предоставили объем порядка одного-двух мегабайт, этого места вам скорее всего не хватит.

Возможность работы с CGI и PHP

Если вы хотите разместить на бесплатном хостинге динамический Web-сайт, использующий серверные сценарии, то от хостинга вам потребуется поддержка технологий CGI и языка PHP. К сожалению, далеко не все бесплатные хостинг-провайдеры поддерживают такую возможность.

Чаще всего хостингом предоставляется некий предустановленный набор сценариев, создающих гостевую книгу, чат, набор счетчиков и т.д., без возможности добавить свои сценарии. Если вы размещаете на сервере домашнюю страницу, то

предустановленные сценарии вам еще могут быть полезны, если же вы делаете какой-то более-менее навороченный динамический сайт, то вам, безусловно, понадобится полноценная поддержка серверных сценариев.

Реклама на странице

Бесплатность размещения Web-сайтов хостинг-провайдер окупает, как правило, размещением своей рекламы на их страницах, хотя некоторые хостеры получают прибыль с бесплатных страниц другими способами, например привлекая новых клиентов на свой платный хостинг или повышая с помощью бесплатных страниц популярность своего доменного имени.

Реклама, размещаемая хостинг-провайдерами, делится на два вида: рекламные баннеры на страницах сайтов и динамические рекламные вставки, pop-up'ы (от англ. pop-up – выскочить), закрывающие часть страницы, а затем автоматически исчезающие с экрана. Выбирайте из двух зол ту, что кажется вам меньшей. Баннер можно органично вписать в дизайн вашей страницы, зато динамические вставки, исчезнув, не отвлекают посетителя и не уменьшают рабочий объем страницы.

Адрес сайта

Большинство бесплатных хостингов предлагают своим пользователям доменные имена третьего уровня **http://www.название_страницы.имя_хостинг_провайдера.ru**, например: **http://www.vasya.narod.ru** или **http://www.rot.front.ru**. Другим вариантом является представление сайта пользователя, как папки на сервере хостинг-провайдера, например **http://www.chat.ru/~username**. Такое имя выглядит хуже, создается впечатление, будто ваш сайт не самостоятельный проект, а просто один из разделов сервера, да и запомнить такое название сложнее.

Следует отметить, что адрес сайта, указывающий на размещение его на одном из популярных серверов бесплатных страниц, негативно сказывается на рейтинге страницы. Дело в том, что большинство страниц, размещаемых на серверах бесплатных страниц, например, **narod.ru** или **by.ru**, – это слепленные по одному шаблону, скучные и некрасивые страницы, все содержание которых сводится к «здесь был Вася». Чтобы доменное имя сервера не работало против вас, выберите хостинг-провайдера с не таким раскрученным именем. Но здесь вас поджидают другие подводные камни. Если доменное имя не популярно то, возможно, для этого есть объективные причины. Например, хостинг не очень хорош сам по себе или недавно открыт и еще не пережил «болезни роста», негативно сказывающиеся на стабильности и удобстве работы.

Некоторые бесплатные хостинг-провайдеры предлагают услугу по размещению сайта пользователя под произвольным доменом второго уровня. Но для этого этот домен нужно приобрести, заплатив некоторую сумму денег. Преимущество такого решения заключается в том, что даже перейдя в последствии на платный хостинг, вы сохраните свое доменное имя, а недостаток – за него нужно платить деньги. А, учитывая тот факт, что заказывая платный хостинг у некоторых компаний, домен второго уровня вы получаете бесплатно, в таком случае имеет смысл подумать о платном хостинге. Если вы уже готовы платить за свой сайт, то почему бы не заплатить и за хостинг, избавившись от всех недостатков бесплатного сервера.

Способ обновления сайта

Некоторые бесплатные хостинг-провайдеры предлагают единственный способ обновления файлов на сайте – через специальную Web-форму, с помощью браузера. Безусловно, таким способом можно добавить 1-2 файла на сайт, но если вам придется таким образом обновлять несколько сотен файлов, вы недобрым словом помянете разработчиков этого инструмента. Единственно правильный способ работы с файлами на сервере – по протоколу FTP (File Transfer Protocol – Протокол передачи файлов), когда вы можете обращаться с файлами сайта, как с файлами в папке на вашем жестком диске, т.е. просто и функционально. Обязательно обращайте внимание на этот параметр при выборе хостинга, благо возможность FTP доступа предоставляет большинство хостинг-провайдеров.

Сравнение различных серверов бесплатного хостинга

Найти сервера бесплатного хостинга можно просто введя в поисковую систему Яndex запрос «бесплатный хостинг». Сейчас же, мы перечислим несколько популярных бесплатных хостинг-провайдеров и отметим их основные особенности, см Табл. 8.1.

Табл. 8.1. *Сравнительная характеристика бесплатных хостинг- провайдеров*

Хостер	*Дисковое пространство*	*Адрес сайта*	*Серверные сценарии*	*Способы обновления*	*Тип рекламы*
www.narod.ru	Неограничен.	**www.назв._сайта.narod.ru**	Только набор предустановленных сценариев	Web-форма, FTP	Всплывающее окно
by.ru	Неогр.	**www.назв._сайта.by.ru**, поддержка произвольных доменов второго и третьего уровней	Поддержка CGI и PHP	Web-форма, FTP	Баннер
www.webservis.ru	Неогр.	**Назв._сайта.al.ru** **Назв._сайта.bip.ru** **Назв._сайта.dem.ru** **Назв._сайта.fud.ru** **Назв._сайта.hobi.ru**	Поддержка CGI и PHP, набор предустановленных сценариев	Web-форма, FTP	Баннер

Хостер	*Дисковое пространство*	*Адрес сайта*	*Серверные сценарии*	*Способы обновления*	*Тип рекламы*
www.pochta.ru	20 Мб	**www.назв._сайта.pochta.ru**, возможны варианты: **fromru.com, front.ru, hotbox.ru, krovatka.net, land.ru, mail15.com, mail333.com, pisem.net, pochtamt.ru, pop3.ru, rbcmail.ru, smtp.ru**	Нет	Web-форма, FTP	Без рекламы
www.holm.ru	Неогр.	**www.назв._сайта.h15.ru**	Поддержка CGI, PHP	FTP	Баннер
www.hut.ru	100 Мб, с возможностью дальнейшего увеличения	**www.назв._сайта.hut.ru**, возможна поддержка произвольных доменов второго уровня	Поддержка CGI-сценариев на языке PERL, PHP.	FTP	Баннер вверху страницы
www.mail.ru	50 Мб, возможно увеличение	**www.название_сайта.mail.ru**	Сценарии не поддерживаются	Web-форма, FTP	Баннер вверху всех страниц, размером более 2 Кбайт
www.chat.ru	10 Мб.	**www.назв._сайта.chat.ru, либо www.chat.ru/~название_сайта**	Только предустановленные сценарии	Web-форма, FTP	Баннер

Регистрация на сервере бесплатного хостинга

Допустим, вы подобрали себе бесплатный хостинг по вкусу, теперь необходимо на нем зарегистрироваться. Процедура регистрации зависит от сервера хостинга, но в общем и целом они похожи. Рассмотрим эту процедуру на примере сервера **http://www.webservis.ru**.

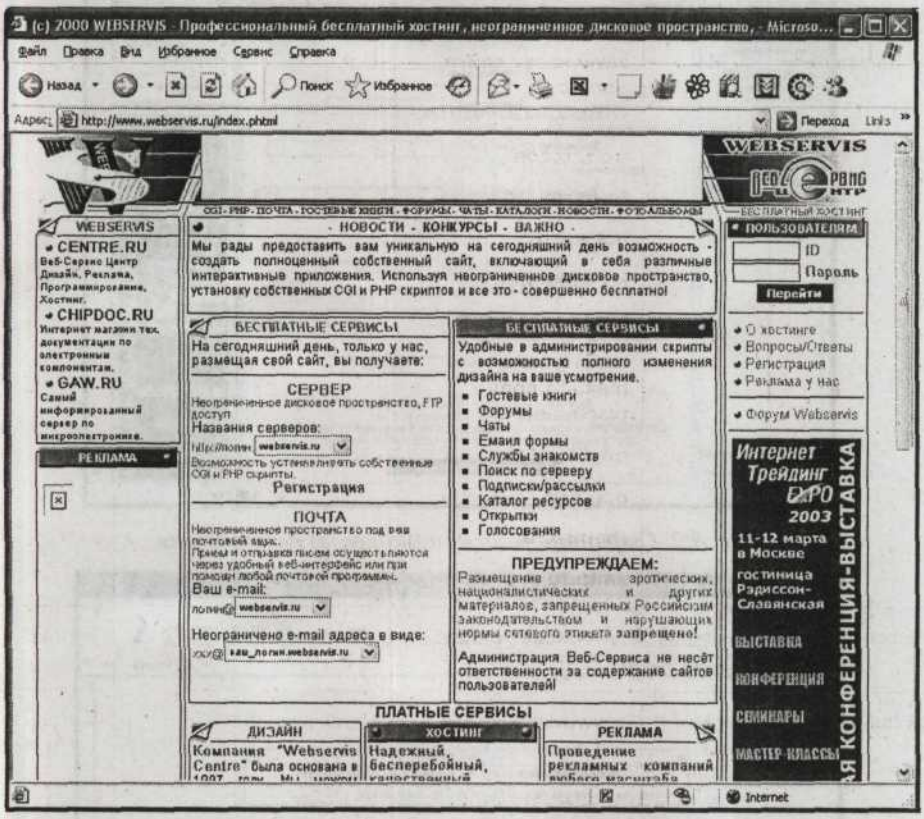


Рис. 8.1. Главная страница сайта хостинг-провайдера Webservis.ru

- В адресной строке браузера введите адрес сайта хостинг-провайдера, **http://www.webservis.ru**, после чего откроется его главная страница (Рис. 8.1). На ней вы найдете упоминания о последних новостях, предоставляемых услугах и ссылку на страницу регистрации новых пользователей хостинга.

- Щелкните мышью на ссылке **Регистрация** в правой части страницы. Откроется страница с регистрационной формой (Рис. 8.2).

- Щелкните мышью на ссылке **СОГЛАШЕНИЕ о предоставлении услуг по бесплатному хостингу**. Откроется страница, на которой будут приведены основные условия предоставления хостинга, ограничения на размещение информации и файлов и правила поведения, которым вы должны следовать, пользуясь услугами этого хостинг-провайдера (Рис. 8.3).

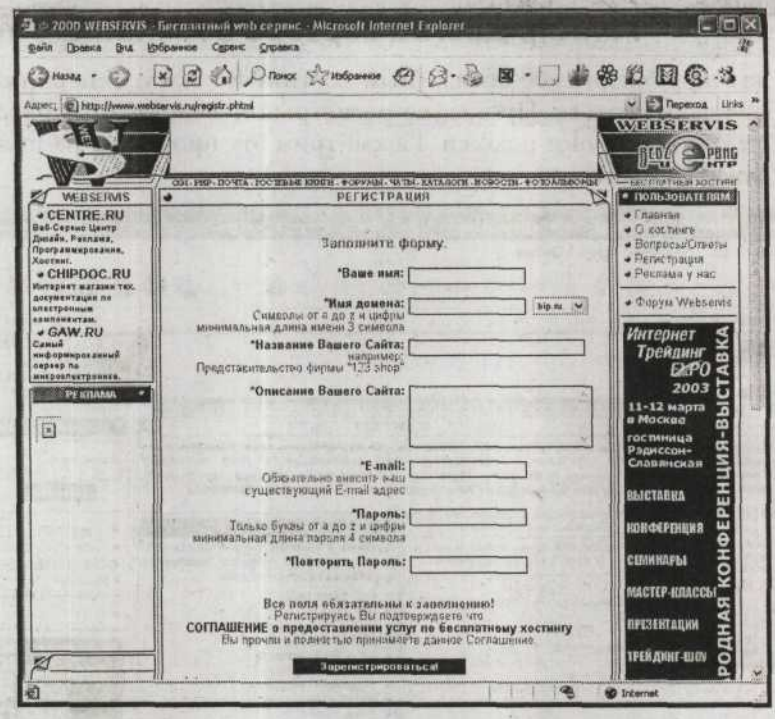


Рис. 8.2. Страница регистрации нового пользователя

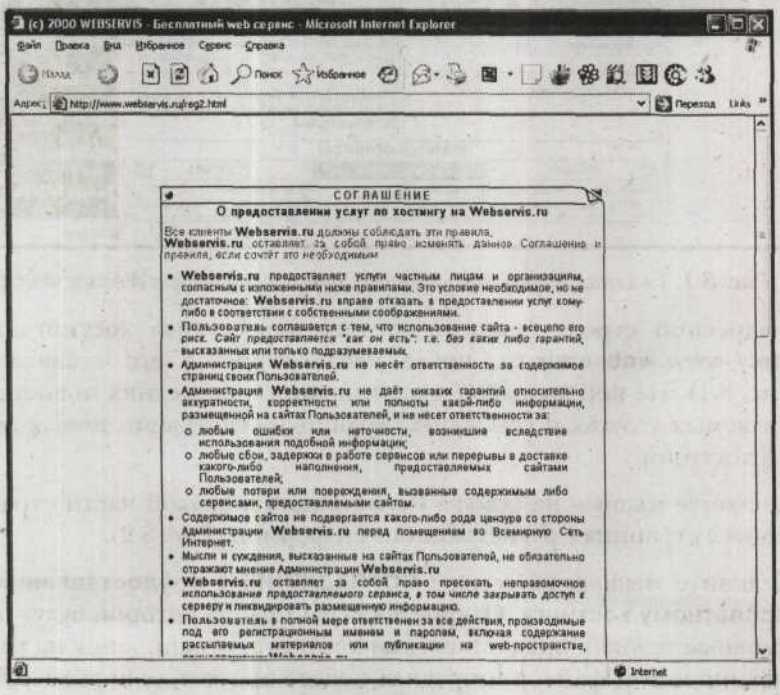


Рис. 8.3. Соглашение о предоставлении услуг хостинга

- Если вам подходят условия соглашения, закрывайте эту страницу и приступайте к заполнению регистрационной формы, если же какие-то пункты вам категорически не подходят, значит, этот хостинг не для вас, выберите другого хостинг-провайдера.
- В регистрационной форме звездочками помечаются поля, обязательные к заполнению. Введите в соответствующие поля ввода свое имя, желаемое название домена третьего уровня. Выберите из выпадающего списка желаемый домен второго уровня. Напишите название сайта и краткое его описание. Эти данные будут использованы при размещении сайта в каталоге хостинг-провайдера. Наконец, укажите свой адрес электронной почты и пароль доступа.
- Щелкните мышью на кнопке **Зарегистрировать**. Если все данные введены правильно и выбранное вами доменное имя сайта еще не занято, вы будете зарегистрированы (Рис. 8.4). Если же в форме были ошибки либо выбранное доменное имя уже кем-то занято, вас попросят изменить часть введенных сведений.

Через некоторое время после выполнения процедуры регистрации, на ящик электронной почты, указанный в регистрационной форме, придет письмо, информирующее вас об окончании процесса создания вашей учетной записи и содержащее дальнейшие указания по работе с вашей учетной записью виртуального хостинга.

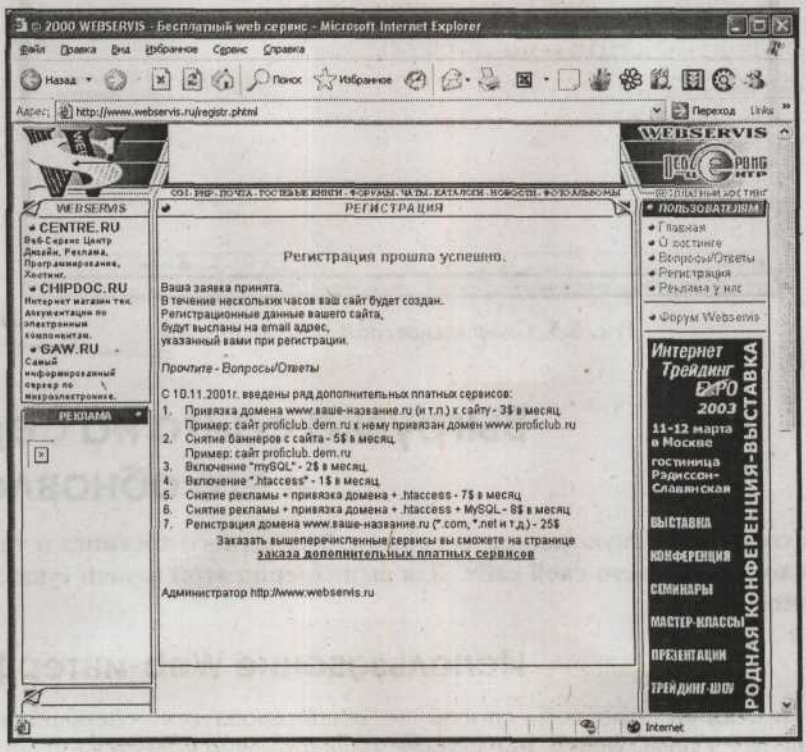


Рис. 8.4. Страница, сообщающая об успешной регистрации

Введите в адресной строке браузера адрес вашего сайта, приведенный в регистрационном письме. Вы увидите страницу, сообщающую о том, что сайт находится в разработке (Рис. 8.5). Эту страницу хостинг автоматически разместил на вашем виртуальном сервере, как главную страницу сайта, и она будет видна всем посетителям вашего сайта, пока вы не замените ее какой-либо другой страницей. О том, каким образом закачать свои собственные файлы на сервер, мы поговорим в следующем разделе.

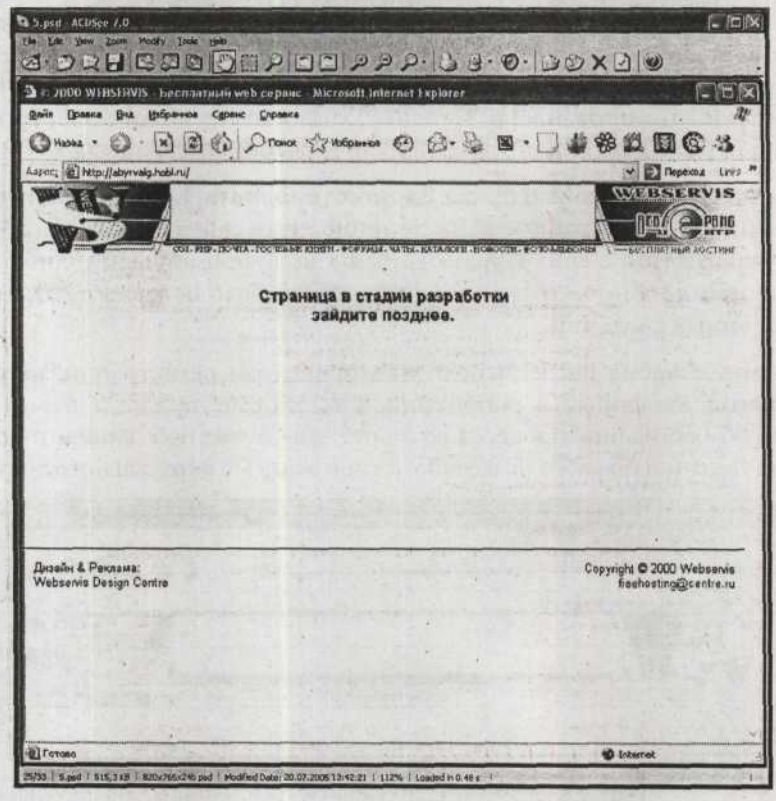


Рис. 8.5. Содержимое сайта по умолчанию

Выгрузка сайта на сервер и его обновление

Итак, вы создали учетную запись на сервере бесплатного хостинга и теперь вам нужно выложить на него свой сайт. Для выполнения этой задачи существует несколько способов.

Использование Web-интерфейса

Первый и самый простой способ – воспользоваться специальным Web-интерфейсом. Такую услугу предлагает большинство хостинг-провайдеров. У разных хостингов Web-интерфейсы тоже разные, но в общих чертах они доста-

точно похожи. Рассмотрим Web-интерфейс управления файлами, предлагаемый хостингом **http://www.webservis.ru**. Получить доступ к Web-интерфейсу можно через страничку управления сайтом.

Введите в адресной строке браузера адрес главной страницы сайта хостинга, **http://www.webservis.ru**, в правой части открывшейся страницы будут два поля ввода для ID и пароля зарегистрировавшихся пользователей.

Введите в эти поля ID и пароль, полученные вами в регистрационном письме, и щелкните мышью на кнопке **Перейти**. Откроется страница с информацией для зарегистрированных пользователей (Рис. 8.6).

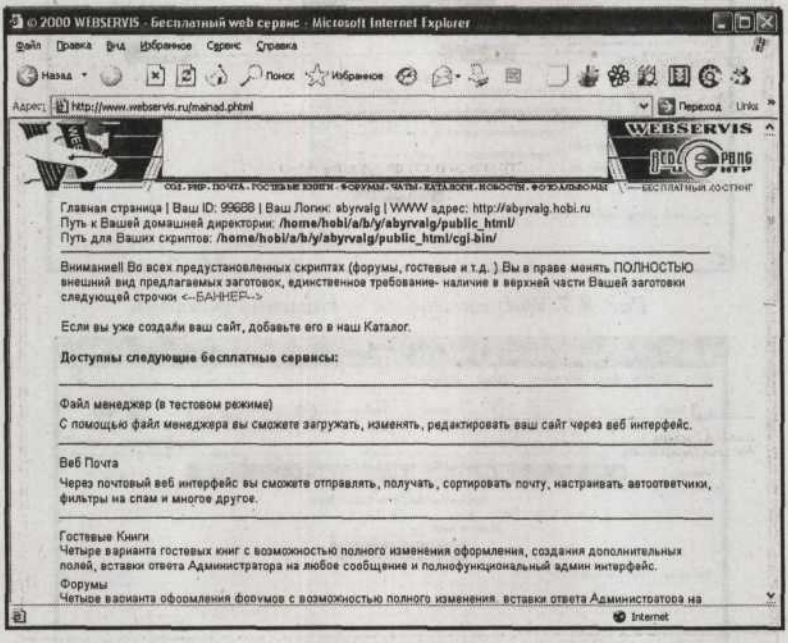


Рис. 8.6. Страница с информацией для зарегистрированных пользователей

В списке бесплатных услуг найдите ссылку **Файл менеджер** и щелкните на ней мышью, откроется страница Web-интерфейса управления файлами (Рис. 8.7). Большую часть страницы занимает список папок и каталогов.

- Если вы щелкнете мышью на названии папки, в окне откроется ее содержание. Чтобы вернуться к содержимому вышестоящей папки, щелкните мышью на двоеточии (..) в верхней части списка.

Чтобы переименовать файл либо папку, щелкните мышью на ссылке **rename**, справа от имени файла или папки. Появится окно переименования (Рис. 8.8). Введите новое название и щелкните мышью на кнопке **переименовать**.

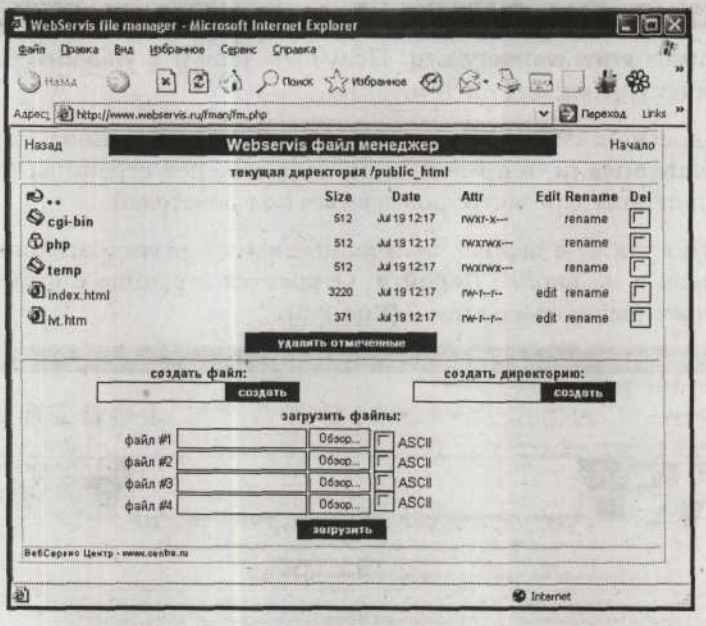


Рис. 8.7. Web-интерфейс управления файлами

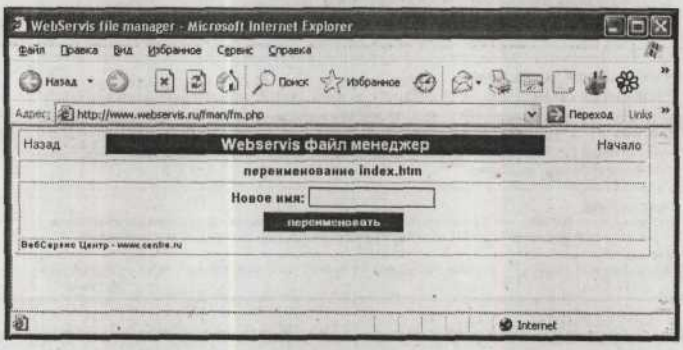


Рис. 8.8. Форма переименования файлов

- Если вы щелкнете на ссылке **edit** справа от названия файла, откроется страница редактирования файла, позволяющая немного подправить код той или иной страницы либо сценария (Рис. 8.9). Закончив редактирование, щелкните мышью на ссылке **Сохранить**.
- Чтобы удалить файлы или папки с сервера, установите флажки в столбце **Del** (Удалить) напротив имен удаляемых файлов и папок, а затем щелкните мышью на кнопке **удалить отмеченные**.
- Чтобы создать новую папку, введите название папки в поле ввода **создать директорию**, а затем щелкните мышью на кнопке **создать** справа от этого поля.
- Чтобы создать новый файл, введите его название в поле ввода **создать файл**, а затем щелкните мышью на кнопке **создать**, справа от этого поля

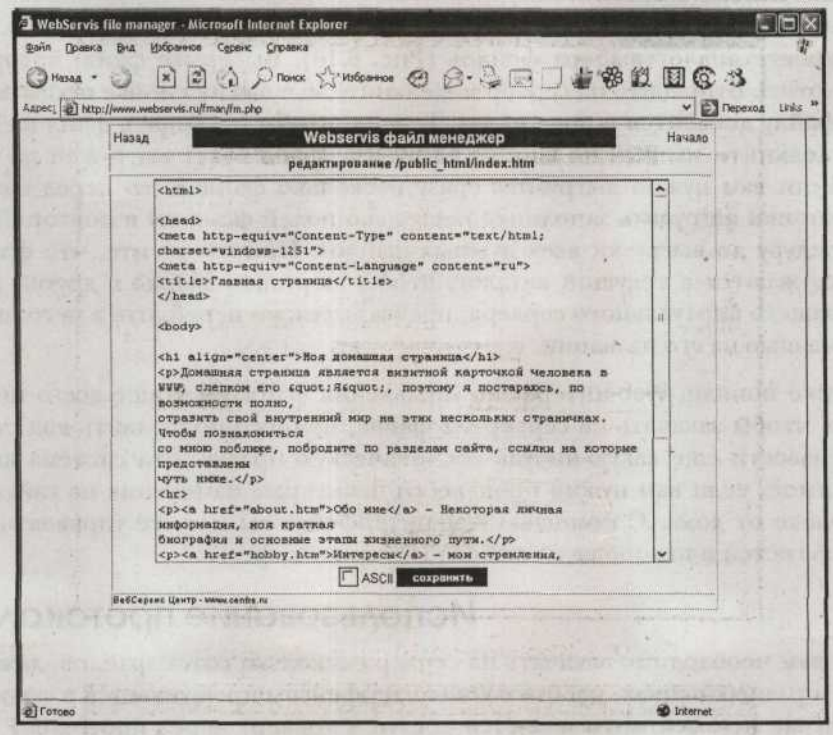


Рис. 8.9. Страница редактирования файлов

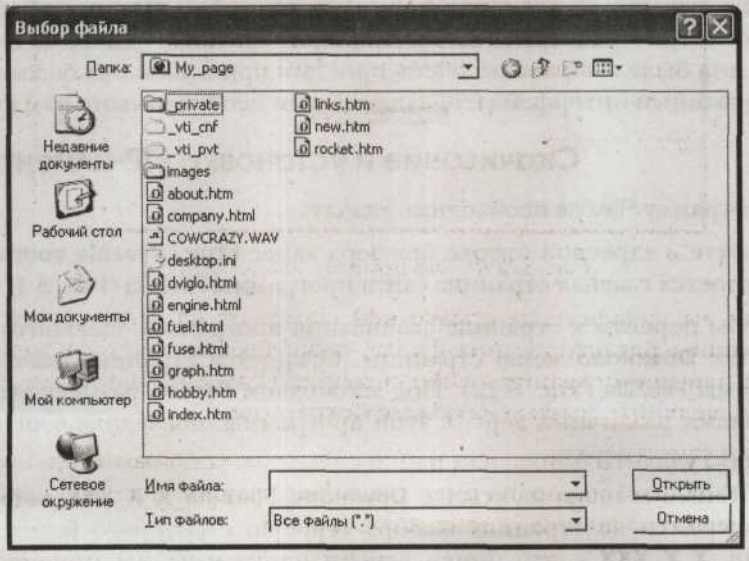


Рис. 8.10. Диалог выбора файлов

- Чтобы выгрузить на Web-сервер файл из папки на вашем компьютере, в Web-форме предназначен раздел под названием **загрузить файлы**. Этот раздел состоит из четырех полей ввода **файл #N** с кнопками **обзор** справа

от них и кнопкой **загрузить**. Щелкнув мышью на кнопке **обзор**, вы откроете диалог выбора файлов (Рис. 8.10). Выберите файл, который вы хотите отправить на сервер и щелкните мышью на кнопке **открыть**, путь к файлу добавится в поле ввода. Теперь, чтобы отправить файл на сервер, щелкните мышью на кнопке **загрузить**, файл будет выгружен на сервер. Если вам нужно выгрузить сразу несколько файлов, то перед нажатием кнопки **загрузить** заполните несколько полей **файл #N** и повторяйте процедуру до выгрузки всех нужных файлов. Причем учтите, что файлы загружаются в текущий каталог, чтобы загрузить файлы в другой каталог вашего виртуального сервера, предварительно перейдите в него, щелкнув мышью на его названии.

Как вы уже поняли, Web-интерфейс управления файлами лучше всего подходит для того, чтобы закачать на сервер 1-2 файла, немного подправить код страниц, или произвести еще какую-нибудь косметическую правку. Эта система является незаменимой, если вам нужно произвести некоторые изменения на сайте, находясь вдалеке от дома. С помощью Web-интерфейса вы можете управлять вашим сайтом из гостей или вообще из любого Интернет-кафе.

Использование протокола FTP

Если же вам необходимо закачать на сервер несколько сотен файлов, лежащих к тому же в разных папках, работа с Web-интерфейсом превращается в сущий ад. В таком случае незаменимым является доступ к серверу через протокол FTP. Для работы с сервером по протоколу FTP нужна специальная программа FTP-клиент. FTP-клиентов существует великое множество, на любой вкус и любые потребности. Мы рассмотрим работу FTP-клиента на примере лишь одного из них – FileZilla. Эта программа была выбрана по очень простым причинам: она бесплатна, имеет русифицированный интерфейс и обладает всеми необходимыми нам функциями.

Скачивание и установка FTP-клиента FileZilla

Сначала программу FileZilla необходимо скачать.

- Введите в адресной строке браузера адрес **http://filezilla.sourceforge.net**. Откроется главная страница сайта программы FileZilla (Рис. 8.11).
- Чтобы перейти к странице скачивания программы, щелкните мышью на пункте **Download** меню страницы. Откроется страница скачивания программы FileZilla (Рис. 8.12). Под заголовком FileZilla размещается список с файлами различных версий этой программы, последняя версия размещается в самом верху.
- Щелкните мышью на ссылке **Download FileZilla_X_X_XXX_setup.exe**, чтобы перейти на страницу выбора сервера, с которого будете скачивать файл. **X_X_XXX** – это номер версии программы (на момент написания этой книги, номер последней версии был **2_2_14b**). На странице выбора сервера будет представлен список серверов, с которых можно скачать программу, с указанием их месторасположения (Рис. 8.13).

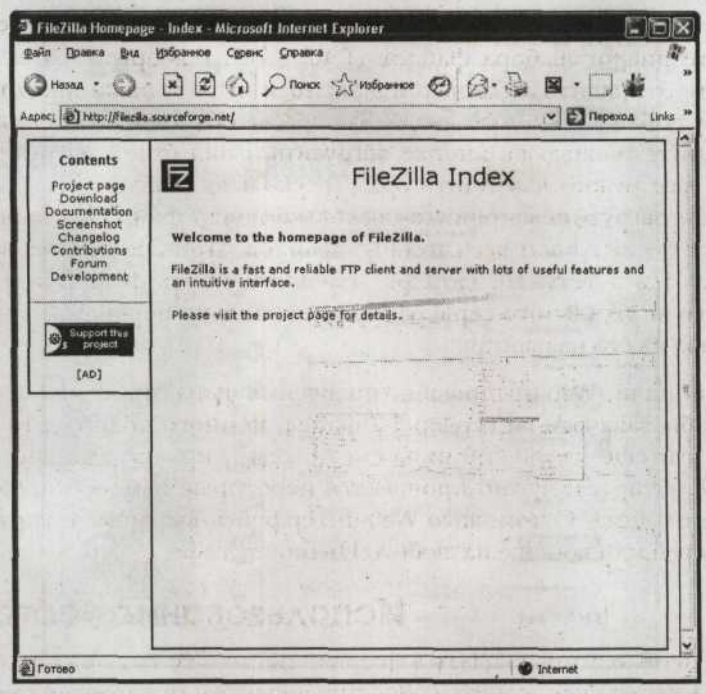


Рис. 8.11. Главная страница сайта программы FileZilla

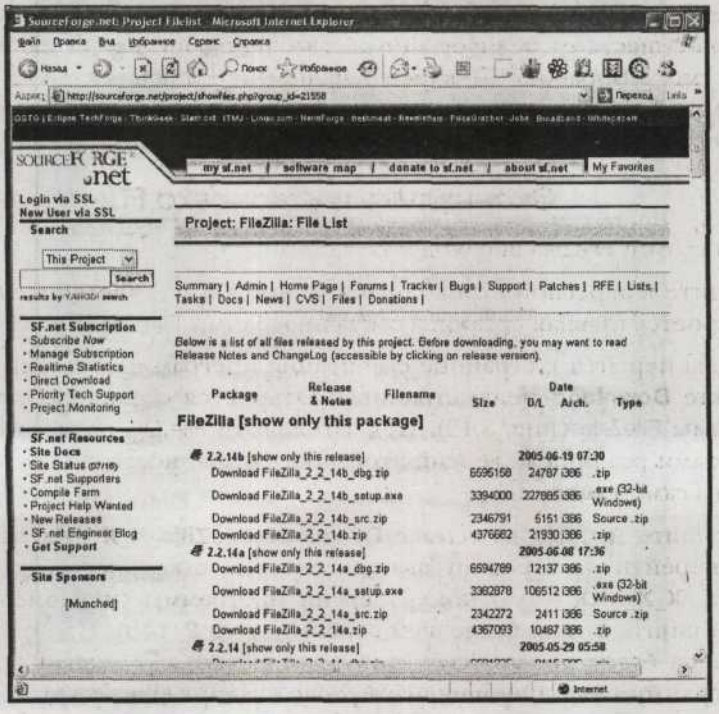


Рис. 8.12. Страница скачивания программы FileZilla

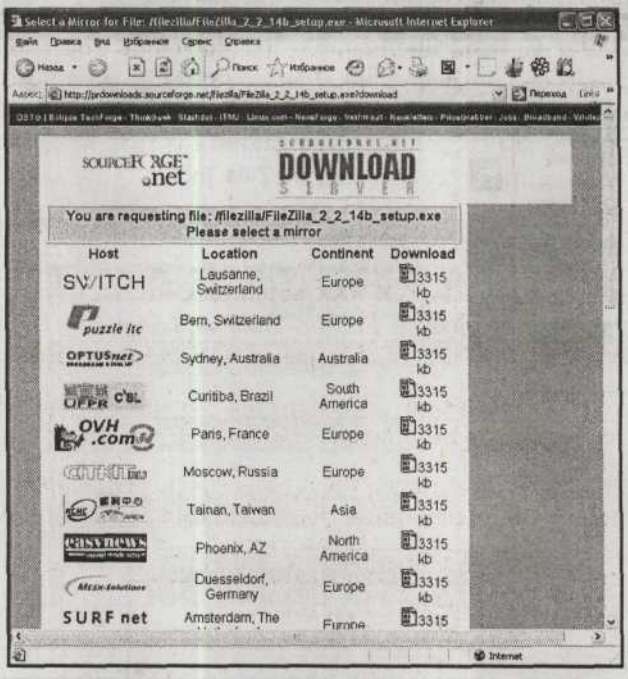


Рис. 8.13. Страница выбора сервера, для скачивания программы FileZilla

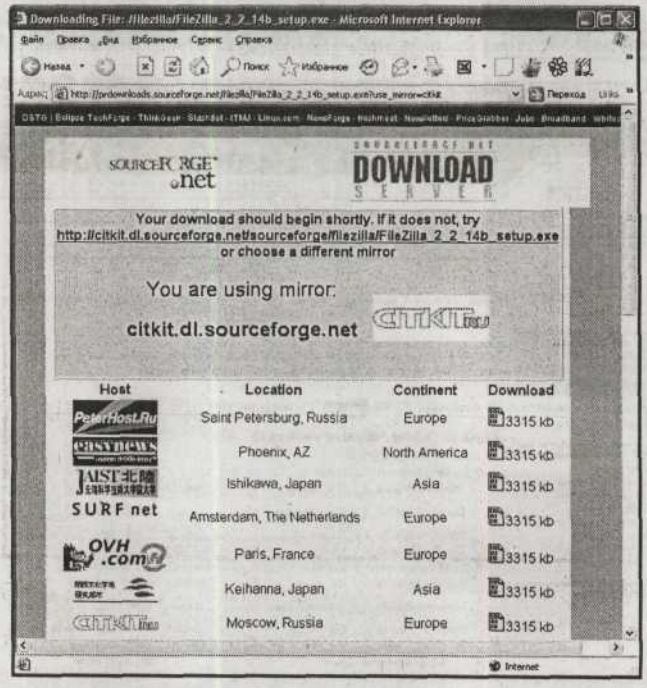


Рис. 8.14. Страница, сообщающая о выбранном сервере, для скачивания программы FileZilla

- Щелкните мышью на иконке столбца **Download** той строки списка, в которой указан сервер, наиболее близко к вам расположенный. Например, можно выбрать сервер Citkit.ru, находящийся в Москве. После щелчка на иконке откроется страница, сообщающая о том, какой сервер вы выбрали, и начнется процесс скачивания.

После того как программа полностью загружена, необходимо ее установить на компьютер.

- Запустите файл **FileZilla_X_X_XXX_setup.exe**, чтобы начать процесс установки программы. Откроется диалог **Installer Language** (Язык инсталляции), предлагающий выбрать язык программы и системы установки (Рис. 8.15).
- Выберите пункт **Russian** (Русский) из раскрывающегося списка, чтобы программа общалась по-русски.
- Щелкните мышью на кнопке **OK**, чтобы продолжить процесс установки. Откроется диалог с лицензионным соглашением программы FileZilla (Рис. 8.16).

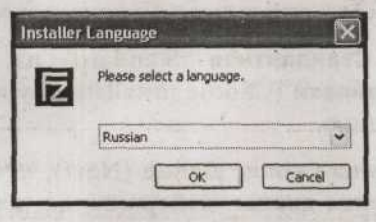


Рис. 8.15. Диалог ***Installer Language*** *(Язык инсталляции)*

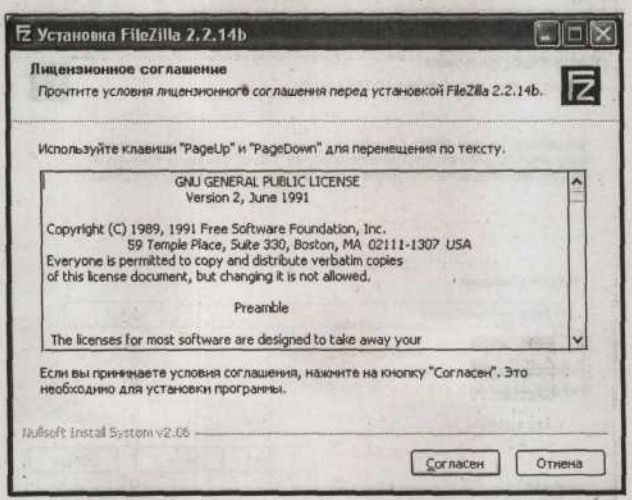


Рис. 8.16. Диалог с лицензионным соглашением программы FileZilla

- Щелкните мышью на кнопке **Согласен** (Accept), чтобы принять условия лицензионного соглашения. Появится диалог выбора компонентов, которые необходимо устанавливать (Рис. 8.17).

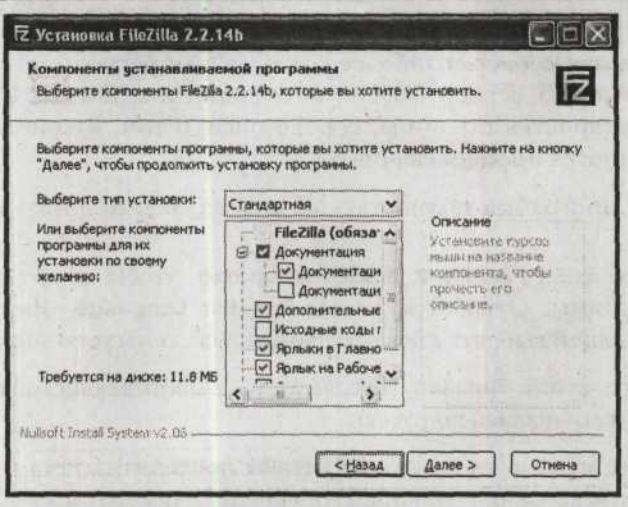


Рис. 8.17. Диалог выбора устанавливаемых компонентов программы FileZilla

- Выберите пункт **Стандартная** (Standard) из открывающегося списка **Выберите тип установки** (Choose installation type), чтобы выбрать тип установки по умолчанию.
- Щелкните мышью на кнопке **Далее** (Next), чтобы продолжить процесс установки. Откроется диалог выбора папки, в которую будет устанавлена программа FileZilla (Рис. 8.18).

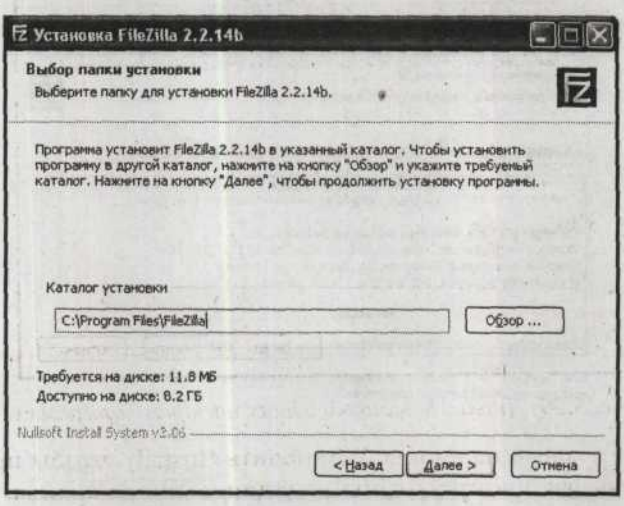


Рис. 8.18. Диалог выбора установочной папки

- Щелкните мышью на кнопке **Далее** (Next), чтобы подтвердить установку программы в папку по умолчанию. Откроется диалог выбора папки в меню **Пуск** (Start), куда будут установлены иконки программы (Рис. 8.19).

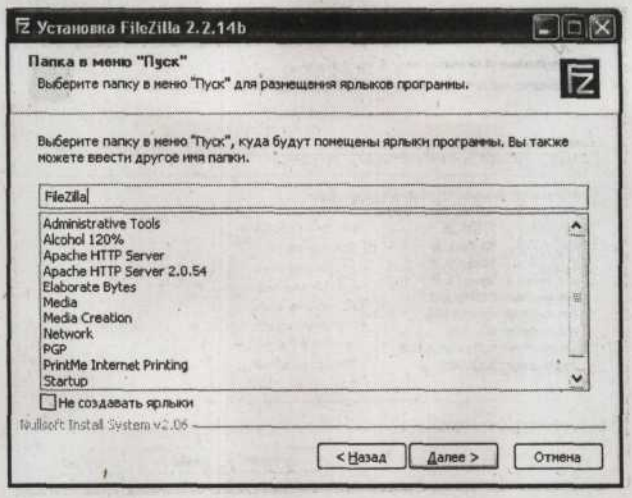


Рис. 8.19. *Диалог выбора установочной папки в меню* ***Пуск*** *(Start)*

- Щелкните мышью на кнопке **Далее** (Next), чтобы подтвердить установку иконок в папку меню **Пуск** (Start) по умолчанию. Откроется диалог с общими настройками программы (Рис. 8.20).

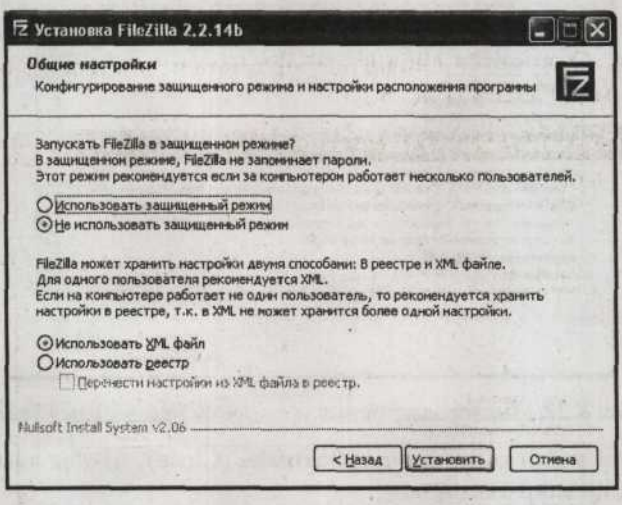


Рис. 8.20. *Диалог установки общих настроек программы*

- Щелкните мышью на кнопке **Установить** (Install), чтобы приступить к копированию файлов программы из установочного архива. Появится диалог копирования файлов, показывающий процесс установки файлов программы на компьютер (Рис. 8.21). Когда копирование файлов будет завершено, отобразится диалог завершения установки (Рис. 8.22).

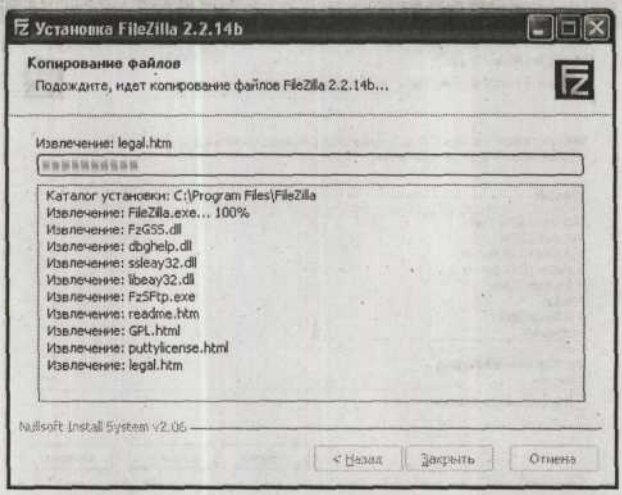


Рис. 8.21. Диалог копирования установочных файлов

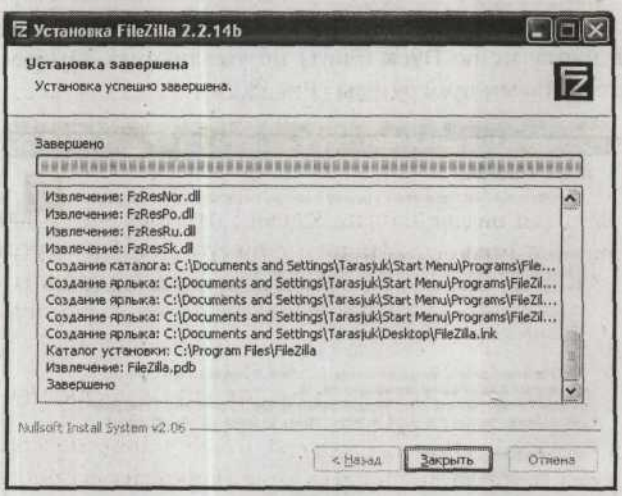


Рис. 8.22. Диалог завершения установки программы FileZilla

➢ Щелкните мышью на кнопке **Закрыть** (Close), чтобы завершить процесс установки и закрыть диалог..

Работа с FTP-клиентом FileZilla

Чтобы запустить программу FileZilla, выполните команду меню **Пуск** (Start) **Все программы ◆ FileZilla ◆ FileZilla** (Programs ◆ FileZilla ◆ FileZilla)

Главное окно программы FileZilla разделено на четыре части (Рис. 8.23). Внизу – область обработки заданий, в ней отображается процесс передачи файлов с компьютера на удаленный сервер и наоборот. В верхней части окна, находится область сообщений, в ней отображаются команды, посылаемые серверу, и отклик на них.

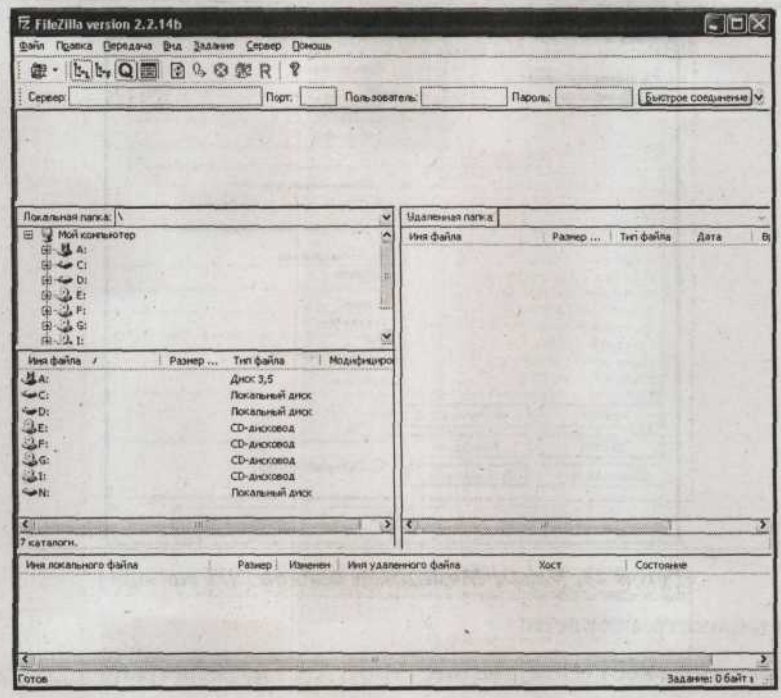


Рис. 8.23. *Главное окно программы FileZilla*

Середина окна делится на две части. В левой отображаются каталоги дисков вашего компьютера, а в правой, каталоги сервера. Чтобы передать файл, группу файлов – или каталог на сервер, достаточно просто перетащить их из левой части окна в правую. Точно так же, совершается и перенос файлов с сервера на компьютер, только переносить файлы нужно уже справа налево.

Настройка параметров подключения к серверу

Но чтобы получить доступ к файлам сервера, нужно к нему подсоединиться, для этого необходимо задать настройки сервера в менеджере сайтов программы FileZilla.

- Выполните команду меню **Файл ♦ Менеджер сайтов** (File ♦ Site manager). Откроется диалог **Менеджер сайтов** (Site manager), Рис. 8.24. В этом диалоге можно выбрать сервер, к которому вы будете подключаться, и добавить новый сервер к списку уже настроенных.
- Щелкните мышью на кнопке **Новый** (New). К списку **Дерево FTP сайтов** (My FTP Sites), добавится новый пункт. Задайте какое-нибудь название серверу, например **webservis**.
- В правой части диалога **Менеджер сайтов** (Site manager) задаются настройки сервера. Основные настройки сервера, которые вам нужно ввести, чтобы получить доступ к каталогу вашей страницы, приведены в регистрационном письме, в разделе **Для входа на FTP**.

Рис. 8.24. Диалог ***Менеджер сайтов*** *(Site manager)*

Настроим параметры сервера:

- В поле ввода **Хост** (Host) введите строку Хост из регистрационного письма.
- В поле ввода **Имя пользователя** (Username) введите строку Логин из регистрационного письма.
- В поле ввода **Пароль** (Password) введите строку Пароль из регистрационного письма.
- Установите переключатель **Тип идентификации** (Logontype) в положение **Обычная** (Normal).
- Щелкните мышью на кнопке **Дополнительно** (Advanced), чтобы указать дополнительные настройки сервера. Откроется диалог дополнительных настроек (Рис. 8.25).
- В поле ввода **Удаленный каталог по умолчанию** (Default remote) введите «**/public_html/**», чтобы на сервере автоматически открывалась папка HTML-файлов.
- Щелкните мышью на кнопке с многоточием справа от поля ввода **Локальный каталог по умолчанию** (Default local), чтобы выбрать каталог вашего компьютера, с которого вы будете обычно копировать файлы на этот сервер. Откроется диалог **Обзор папок** (Browse folders), Рис. 8.26.
- Выберите нужную папку и щелкните мышью на кнопке **OK**, чтобы принять выбор и закрыть диалог **Обзор папок** (Browse folders).
- Установите переключатель **Настройки пассивного режима** в положение **Активный** (Use active mode).

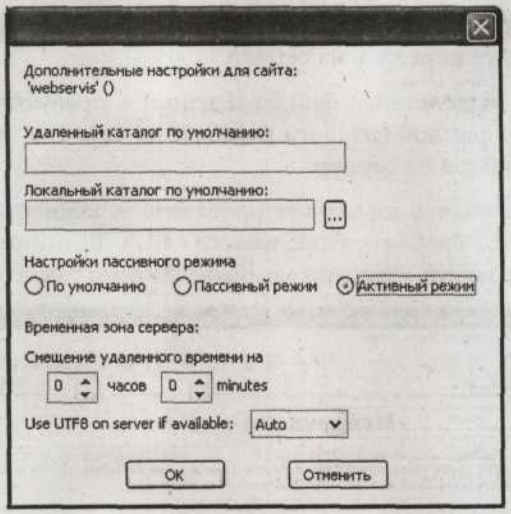


Рис. 8.25. Диалог дополнительных настроек сервера

Рис. 8.26. Диалог ***Обзор папок***

- Щелкните мышью на кнопке **OK**, чтобы принять установки и закрыть диалог дополнительных настроек.
- Щелкните мышью на кнопке **Соединить** (Connect), чтобы запомнить настройки и подключиться к серверу. Не забудьте только предварительно подключиться к сети Интернет.

Закачка файлов на сервер

После того, как FTP-клиент FileZilla подключится к серверу, можете закачивать на него файлы.

- В левой части окна отображено дерево каталогов вашего компьютера. Выберите каталог, содержимое которого вы хотите отправить на сервер.

- В появившемся списке файлов этого каталога выделите, файлы и папки, которые хотите передать на сервер.
- Перетащите выделенные файлы и папки в правую часть окна, где находится список файлов каталога **public_html** сервера. Начнется процесс закачивания файлов на сервер.

Когда файлы будут закачаны, вы можете проверить успешность этого процесса. Введите в адресной строке браузера адрес вашего сайта. Если процесс закачки прошел успешно, вы увидите главную страницу закачанного вами сайта (Рис. 8.27).

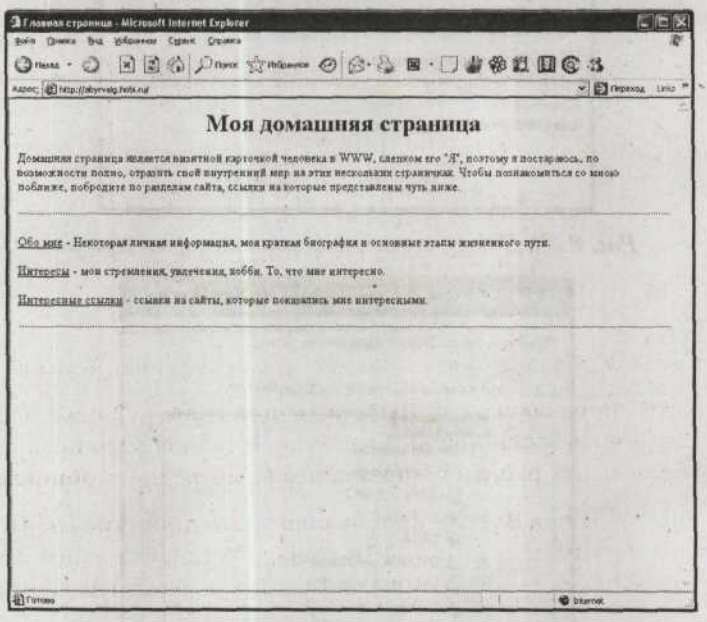


Рис. 8.27. Главная страница сайта

Обновление файлов на сервере

Чтобы обновить сайт на сервере после каких-либо изменений, которые вы внесли в его локальную копию, сделайте следующее.

- В программе FileZilla выполните команду меню **Файл ♦ Менеджер сайтов** (File ♦ Site manager). Откроется диалог **Менеджер сайтов** (Site manager).
- В списке **Дерево FTP сайтов** (My FTP sites) щелкните мышью на названии сервера, на котором вы хотите обновить сайт.
- Щелкните мышью на кнопке **Соединить** (Connect), чтобы подключиться к серверу.
- В дереве каталогов вашего компьютере в левой части главного окна программы FileZilla выберите каталог, содержимое которого вы хотите обновить на сервере.

- В появившемся списке файлов этого каталога, выделите обновляемые файлы и папки. Необязательно выделять только файлы, которые были изменены, проще выделить все файлы, относящиеся к сайту.
- Перетащите выделенные файлы и папки в правую часть окна, где находится список файлов каталога **public_html** сервера. Начнется процесс выгрузки файлов на сервер.
- Если файл с таким же именем, как один из закачиваемых файлов, уже содержится на сервере, появится диалог, спрашивающий, что делать в такой ситуации (Рис. 8.28).

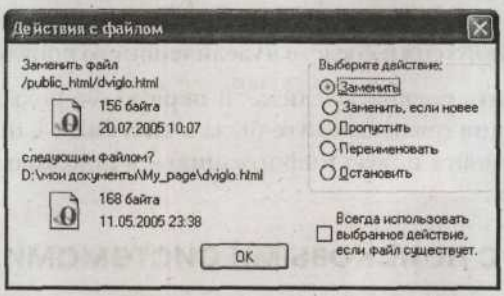


Рис. 8.28. Диалог, спрашивающий, как поступить с файлом

- Установите переключатель **Выберите действие** (Choose action) в положение **Заменить, если новее** (Overwrite, if newer). Это позволит заменять на сервере только файлы, изменившиеся с последнего обновления сайта.
- Установите флажок **Всегда использовать это действие, если файл существует** (Always use this action, if file exist). Установка этого флажка позволит вам не лицезреть этот диалог на каждом перезаписываемом файле.
- Щелкните мышью на кнопке **OK**, чтобы закрыть диалог и продолжить процесс обновления сайта.

а) Раскрутка сайта

б) Завоевание популярности

Вот наконец ваш сайт создан, размещен на Интернет-сервере и успешно выполняет свои функции. Вы полагаете, что работа закончилась и можно почивать на лаврах? Вы ошибаетесь, все только начинается!

Задайте себе вопрос, какую цель вы ставили, создавая сайт. Поведать миру о себе? Познакомить людей со своими творениями? Заняться торговлей, или создать фан-клуб любимой группы? Для достижения любой из этих целей, да и для большинства других, важнейшим фактором является аудитория сайта.

Эти люди должны найти ваш сайт среди миллионов других, зайти на него один раз и продолжать посещать в дальнейшем. Как этого добиться? Успех практиче-

ски любого Web-проекта состоит из двух компонентов: качественного содержания и грамотной рекламной политики.

С помощью рекламы на сайт привлекаются новые посетители, а вот зайдут ли они к вам еще хоть раз, определяется исключительно содержанием сайта. Поэтому нельзя впадать в крайности. Излишне увлекаясь рекламой или, как еще говорят, раскруткой сайта, в ущерб его наполнению, вы потратите силы впустую, поскольку люди будут уходить с вашего ресурса, не задерживаясь на нем. Если же, напротив, вы все силы будете тратить исключительно на содержательную сторону вашего сайта, не тратя сил на рекламу, популярность ваша, скорее всего, расти не будет. К сожалению, в современном мире практически не работает правило «хороший товар в рекламе не нуждается», чтобы ваш сайт был «на слуху» без усилий с вашей стороны, нужно сначала вложить очень много сил и средств в увеличение его популярности.

С чего же стоит начать раскрутку сайта? В первую очередь, необходимо сделать так, чтобы информация о вашем сайте была в системах, с помощью которых Интернет-пользователи ищут новую информацию – в поисковых системах и информационных каталогах.

Работа с поисковыми системами и каталогами

Как работают поисковые системы, уже кратко обсуждалось в данной книге. Существуют специальные программы, называемые пауками, или поисковыми роботами, которые посещают сайты Интернета и записывают в специальную базу данных выжимку информации, хранящейся на сайте, – индекс. Сам процесс составления такой выжимки, называется индексированием. Процесс индексирования является периодическим и пауки время от времени заходят на один и тот же сайт, чтобы проверить, не появилось ли на нем новой информации. В процессе индексирования, паук обращает внимание на ссылки, ведущие с данного сайта на другие и, если ссылка ведет на сайт, которого еще нету в базе поисковой системы, то этот сайт будет автоматически туда добавлен, а в впоследствии проиндексирован.

Совершенно необязательно ждать, пока поисковик вас автоматически проиндексирует, это может занять не один месяц. К тому же, чтобы на других сайтах размещали ссылки на ваш ресурс, нужно в определенной степени раскрутиться. Существует другой способ добавления сайта в поисковую базу ручная регистрация. На сайтах всех поисковых систем есть специальная форма, заполнив которую, вы добавите свой сайт в базу данных поисковика. Рассмотрим процедуру ручного добавления сайта в базу поисковика на примере двух популярных российских поисковых систем Яndex и Rambler.

Регистрация в поисковой системе Яndex

Чтобы указать на свой сайт системе Яndex:

- Введите в адресной строке браузера адрес поисковой системы, **http://www.yandex.ru**. Откроется главная страница поисковика (Рис. 8.29).
- Щелкните мышью на ссылке **Все службы** под полем ввода поисковых запросов. Откроется страница со списком всех услуг системы Яndex, (Рис. 8.30).

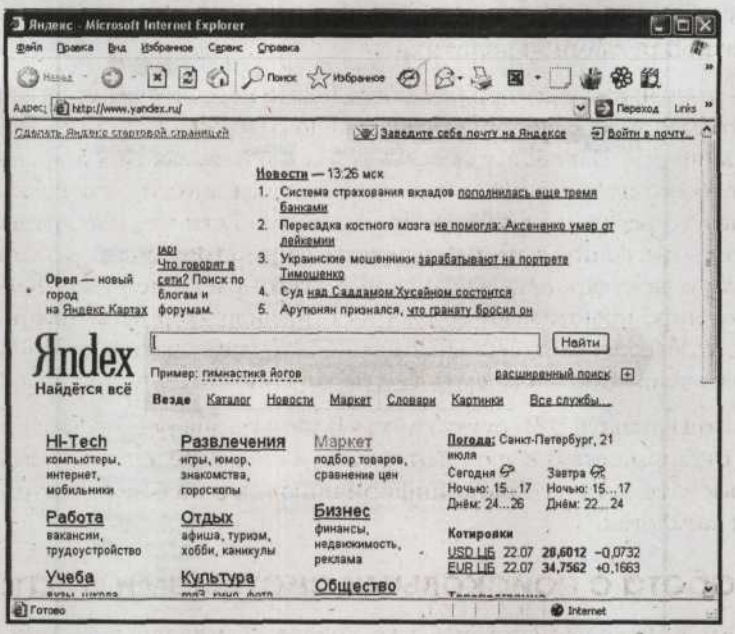


Рис. 8.29. Главная страница поисковой системы Яndex

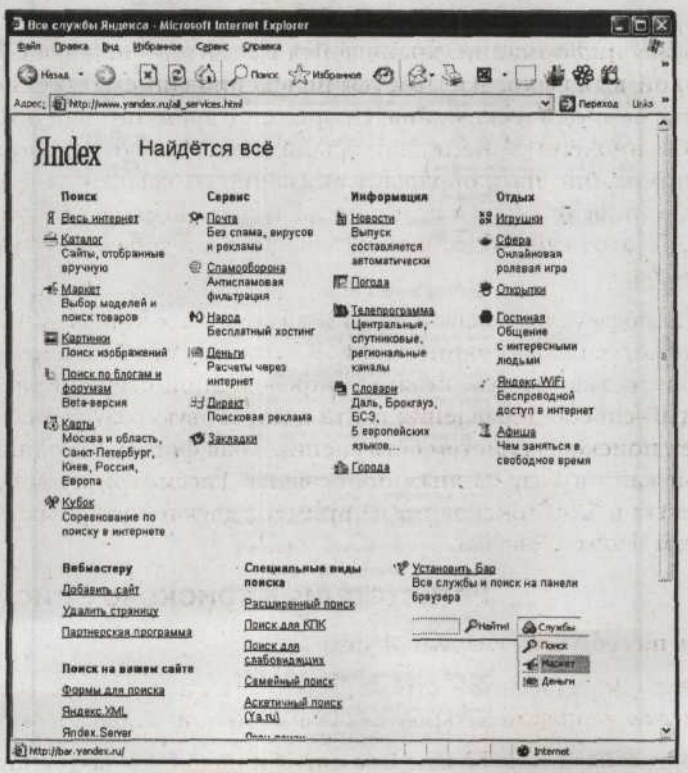


Рис. 8.30. Страница услуг, предоставляемых системой ***Яndex***

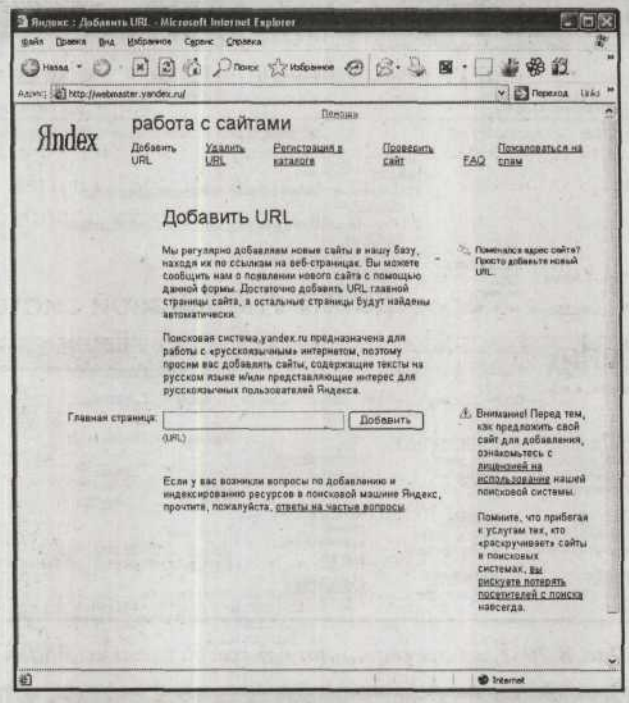


Рис. 8.31. Страница добавления новых адресов к поисковой базе Яndex

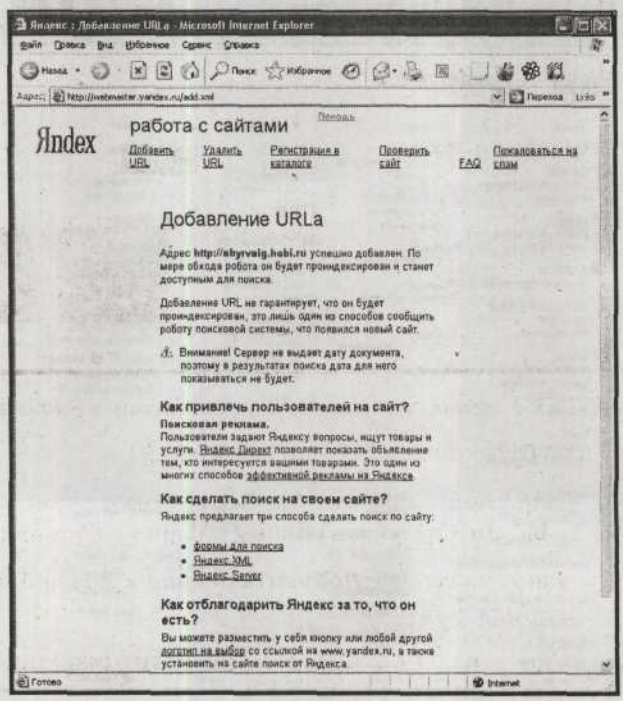


Рис. 8.32. Страница, сообщающая об успешной регистрации сайта в базе Яndex

- Щелкните мышью на ссылке **Добавить сайт** в разделе страницы **Вебмастеру**. Откроется страница добавления нового сайта к базе поисковой системы (Рис. 8.31).

В поле ввода **Главная страница** введите полный адрес вашего сайта и щелкните мышью на кнопке **добавить**. Откроется страница, сообщающая о том, что ваш сайт был добавлен к поисковой базе и со временем будет проиндексирован поисковым роботом (Рис. 8.32).

Регистрация в поисковой системе Rambler

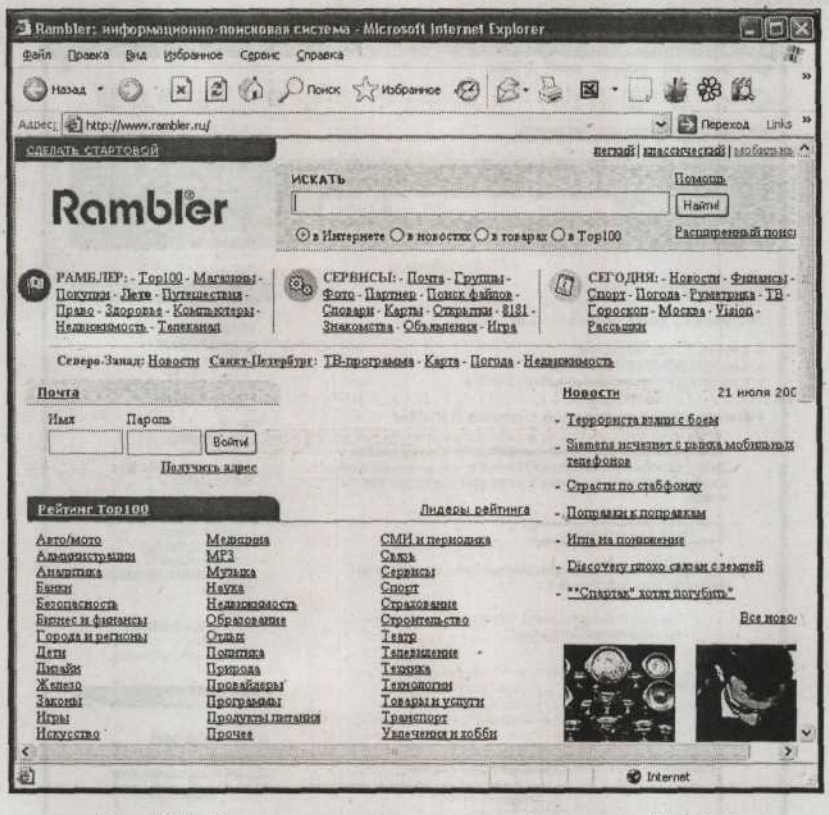


Рис. 8.33. Главная страница поисковой системы Rambler

Чтобы добавить сайт в базу поисковой системы Rambler:

- Введите в адресной строке браузера адрес поисковой системы, **http://www.rambler.ru**. Откроется главная страница поисковика (Рис. 8.33).
- Щелкните мышью на ссылке **Добавить ресурс** в нижней части страницы. Откроется страница добавления сайтов (Рис. 8.34).
- Щелкните мышью на кнопке **Начать регистрацию**, чтобы перейти на страницу регистрации сайта (Рис. 8.35).

- В поле ввода **Название сайта** введите название, под которым сайт будет представляться в поисковой системе.
- В поле ввода **URL головной страницы** введите адрес главной страницы сайта.
- В многострочном поле ввода **Короткое описание регистрируемого сайта** введите краткое описание содержимого вашего сайта.
- В поле ввода **Контактное лицо** введите ваши имя и фамилию.
- В поле ввода **E-mail контактного лица** введите свой адрес электронной почты.

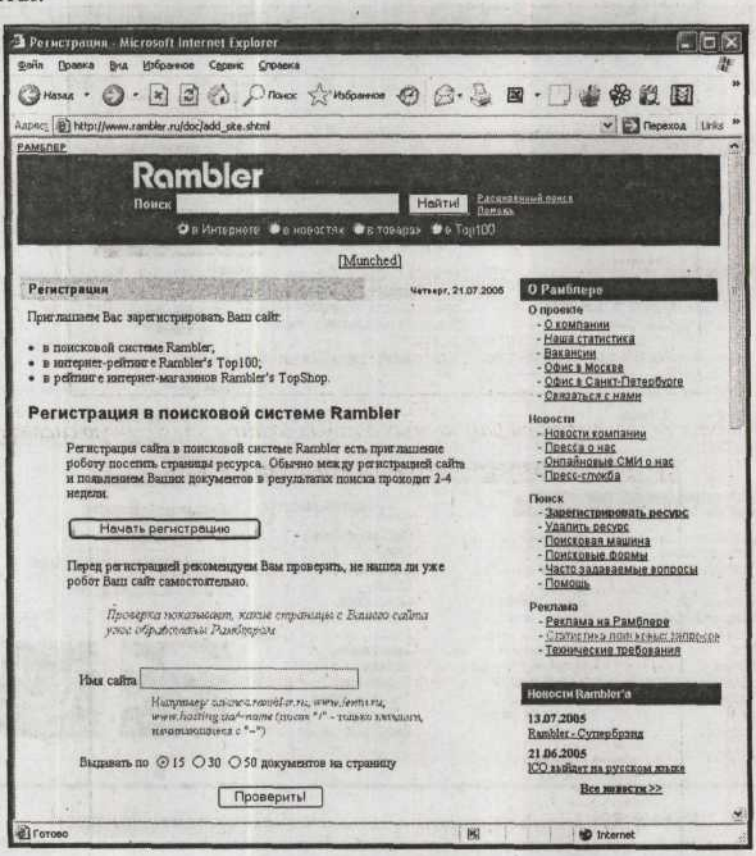


Рис. 8.34. Страница начала регистрации новых сайтов в системе Rambler

- Заполнив все поля формы регистрации, щелкните мышью на кнопке **Зарегистрировать**. Откроется страница, сообщающая об успешном принятии сайта к регистрации (Рис. 8.36).

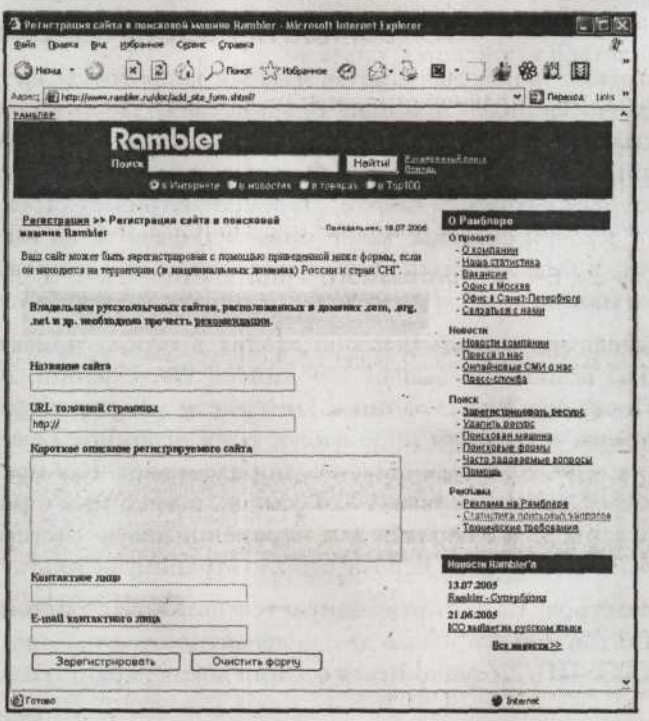


Рис. 8.35. Страница ввода регистрационных данных о сайте в поисковую систему Rambler

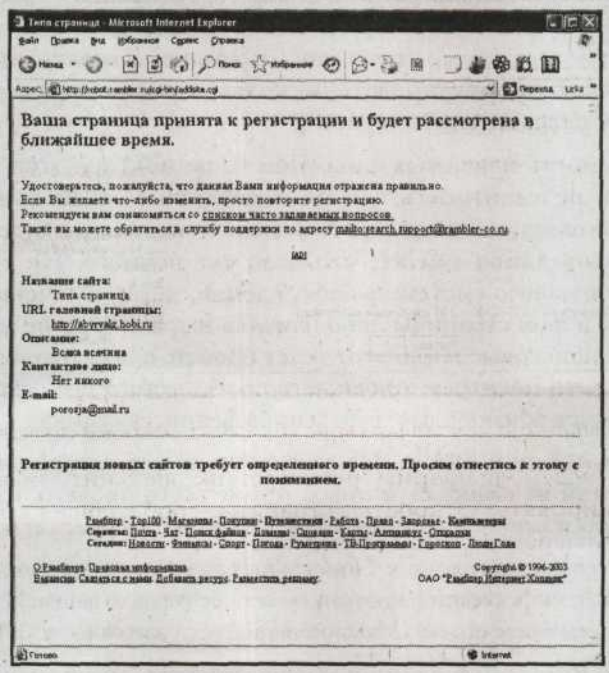


Рис. 8.36. Страница, сообщающая об успешной регистрации в поисковой системе Rambler

Особенности работы поисковых систем

В свете рассмотрения поисковых систем, важным вопросом является принцип, по которому производится индексирование страниц сайта. Достоверно алгоритмы индексирования страниц неизвестны, они являются «know-how» поисковых систем, но основные их принципы не скрываются. Основное внимание поисковые пауки уделяют так называемым ключевым словам. Паук просматривает страницы сайта и отмечает в базе поисковой системы, какие слова встречаются на вашей странице. А скорее, наоборот, в базе хранится список слов, а при индексировании страниц, в базу добавляются упоминания, на каких Web-страницах содержатся эти слова.

Когда пользователь поисковой машины вводит в строку поиска определенное слово или фразу, поисковик выдает ему список Web-страниц, на которых она упоминается. Поскольку Web-страниц в Интернете многие миллионы, страниц, на которых встречается то или иное слово, тоже огромное количество, и поисковая система их определенным образом упорядочивает. Как правило, пользователь просматривает только первые 10-20 ссылок, выводимых в результатах поиска и, чтобы посетители заходили к вам через поисковую систему, необходимо, чтобы ваша страница попала на первую-вторую страницу результатов поиска.

Основных параметров, на которые опирается поисковая система при упорядочивании результатов поиска, ровно два: релевантность страницы и величина индекса цитирования, ИЦ. Познакомимся с этими двумя параметрами подробнее.

Релевантность

Релевантность, – это степень соответствия содержимого страницы ключевому слову или фразе. Релевантность определяется по многим параметрам с помощью сложных алгоритмов, которые тщательно скрываются хозяевами поисковых систем. В целом, можно сформулировать несколько простых правил, которые помогут увеличить вам релевантность страниц.

- Повторяемость ключевых слов. Чем чаще повторяются ключевые слова, тем выше релевантность. Но не переборщите если ключевое слово повторяется слишком часто, то поисковая система не будет учитывать его вовсе, справедливо считая, что дело тут нечисто. Не старайтесь также ввести поисковую систему в заблуждение, делая ключевые слова того же цвета, что и фон страницы либо изменяя их размер до не читаемо маленького, чтобы поисковые пауки этот текст прочли, а посетители страницы его не заметили. Эти нехитрые трюки легко раскусывают все основные поисковые системы и исключают такие сайты из базы поиска вообще.
- Параметр **ALT** тега **<IMG>**. Не забывайте делать альтернативные описания изображений на ваших страницах. Кроме всего прочего, изображения поисковыми роботами не индексируются, а альтернативные описания вполне.
- Ключевые слова в заголовках. Поисковики считают, что в заголовках сообщается самая важная информация, поэтому и «вес» ее гораздо выше. И чем выше уровень заголовка, тем выше его вес. Максимальный вес у заголовков **<H1></H1>**.
- Мета-теги. В код HTML-страниц можно добавить специальный тег **<META>**, с помощью которого можно прямо сообщить поисковой системе о ключевых

словах на странице. Такие теги называются мета-тегами, поскольку непосредственно на формирование страницы они никакого влияния не оказывают, а создаются для других программ, таких как поисковый паук. С помощью тега **<META>** с параметром **name="Description"** можно задать краткое описание страницы сайта, например: **<meta name="Description" Content="краткое описание">**. Длина описания не должна превышать 200 символов. Тег **<META>** с параметром **name="Keywords"** позволяет явно задать ключевые слова страницы, например: **<meta name="Keywords" Content="ключевые слова">**. Ключевые слова, или фразы отделяются друг от друга запятыми. Максимальная длина строки ключевых слов 800 символов.

- Тег **<TITLE></TITLE>**. Ключевые слова, содержащиеся в заголовке сайта, заключенном в контейнер **<TITLE></TITLE>**, пользуются наибольшим приоритетом.

Хочется сделать несколько замечаний по поводу оптимизации информации сайта под поисковые запросы.

- Первое. Позаботьтесь о ключевых словах до регистрации сайта в поисковых системах, поскольку до повторной индексации сайта может пройти очень продолжительный промежуток времени.
- Второе. Не увлекайтесь поисковой оптимизацией. С одной стороны, оптимизация текста под поисковые запросы не должна ухудшать воспринимаемость информации на сайте, а с другой стороны, за излишне усердную поисковую оптимизацию вас могут и вовсе исключить из поискового индекса.
- Третье. Не указывайте в мета-тегах информацию, не соответствующую действительности, с целью повысить посещаемость страницы. Конечно, велик соблазн указать в ключевых словах самые популярные слова, такие как «секс», «халява», «голые знаменитости» и т. д., но результат не оправдает ваших ожиданий. Поисковые машины отнюдь не приветствуют такой обман, посетители, завлеченные обманными посулами, уйдут, чтобы больше не вернуться, а имидж вашего сайта ощутимо пострадает.
- И четвертое. Старайтесь указывать ключевые слова, которые встречаются не слишком часто. Хотя поисковые запросы по ним бывают реже, но и конкуренция за верхние строчки в результатах поиска будет гораздо меньше.

Индекс цитирования

И второй важнейший параметр индекс цитирования. Он напрямую зависит от того, как часто на ваш ресурс ссылаются другие сайты. Причем, чем выше ИЦ у сайта, ссылающегося на вас, тем больше вырастет и ваш ИЦ. Кроме увеличения ИЦ, ссылки на ваш сайт приведут и к прямому росту посещаемости сайта. Существует несколько способов повысить ИЦ.

- Ссылки на статьи. Напишите, если умеете, несколько статей, расскажите о них на тематических форумах, предложите нескольким сайтам подходящей тематики разместить эти статьи у них, с обязательной ссылкой на первоисточник.

Если вам повезет, то таким образом вы достаточно быстро сможете значительно повысить свой ИЦ.

- Прямой обмен ссылками. Договоритесь с сайтами с близкой к вашей тематикой о размещении на их страницах ссылок на ваш сайт в обмен на то, что вы тоже разместите у себя ссылку на их ресурсы. Сложности с обменом ссылками, скорее всего, будут возникать на начальных стадиях раскрутки вашего Web-проекта, пока у вашего сайта низкий рейтинг и не слишком высокая посещаемость.
- Полезность сайта. Если на вашем сайте есть незаменимая и всегда актуальная информация либо вы предлагаете востребованные и полезные услуги, то рано или поздно, при условии определенной активности с вашей стороны, на ваш ресурс начнут ссылаться как на полезный сайт.

Каталоги и рейтинги

Вариацией на тему поисковых систем являются различные каталоги и рейтинги. Рейтинги представляют собой тематический набор ссылок на сайты, отсортированный по популярности ресурсов. Популярность чаще всего определяется по посещаемости сайта. Чтобы участвовать в рейтинге, нужно в нем зарегистрироваться и повесить на страницах своего сайта специальный ярлычок, кнопку, по числу загрузок которой будет рассчитываться посещаемость страницы и, соответственно, ее рейтинг. Самыми известными российскими рейтингами являются Top 100 Рамблера, **http://top100.rambler.ru/**, Top-List, **http://top.list.ru/** и Rax.ru, **http://www.rax.ru/**.

Каталог, как и рейтинг, тоже является сборником ссылок на различные ресурсы определенной тематики либо ряда тематик. Новые сайты могут добавляться в каталог самими владельцами сайтов либо сотрудниками службы каталога. Пример каталога: list.ru, **http://list.mail.ru**.

Хотя каталоги и рейтинги не так часто посещаются, как поисковые системы, но зато и количество сайтов, внесенных в них, гораздо меньше, а сами сайты гораздо четче структурированы. Благодаря этому через каталоги и рейтинги приходит гораздо меньше случайных посетителей и больше людей из целевой аудитории сайта.

Баннеры

Одним из самых популярных способов раскрутки сайтов является использование баннерной рекламы. Баннер – это прямоугольное изображение с рекламным содержанием, размещаемое на Web-странице. Щелчок мышью на баннере приводит к переходу на страницу, им рекламируемую. Баннеры бывают различных размеров, но самым популярным является размер 468x60 пикселов. Прежде чем размещать у себя баннер, учтите, что баннер увеличивает объем страницы в среднем на 10-20 кб, баннеры раздражают многих посетителей страницы, а также баннеры бывает достаточно сложно вписать в дизайн сайта.

Баннерные сети

Основной объем баннерной рекламы осуществляется при помощи так называемых баннерообменных сетей. Участвовать в такой сети можно несколькими способами.

Первый способ – бартер. Вы некоторое количество раз показываете на своем сайте баннеры других участников сети, а они в обмен примерно такое же количество раз показывают ваш баннер. Недостаток такого способа заключается в том, что если у вашей страницы мало посетителей, то вы покажете баннер малое число раз и так же мало раз покажут ваш баннер. А если учесть тот факт, что щелкают по баннеру в лучшем случае 0,5% процента от увидевших его людей, то вы практически не получите отдачи от такой схемы, если у вас не менее нескольких тысяч посетителей в день.

Другой вариант участия в баннерообменной сети – покупка показов баннера. Вы платите некоторую сумму, а ваш баннер показывают определенное число раз на других сайтах. Такая схема всем хороша, за исключением того, что вам приходится платить деньги за рекламу вашего сайта. Прежде чем заказывать баннерную рекламу, посчитайте, окупится ли вложение средств в баннерные показы, учитывая количество людей, которые посетят ваш сайт, щелкнув по баннеру и то, что далеко не все из них захотят посетить его вновь либо совершить у вас покупку, если вы чем-нибудь торгуете.

Третий вариант участия в баннерообменных сетях прямо противоположен второму. Вы показываете на своем сайте баннеры и получаете за это деньги. Чтобы получать значительные суммы за показ баннеров нужно иметь очень хорошо раскрученный сайт но, по крайней мере, так вы сможете окупить платный хостинг.

Существуют уменьшенные версии баннерообменных сетей, называемые Web-rings (Web-кольца). Они объединяют в баннерную сеть сайты сходной друг с другом тематики, что значительно повышает вероятность того, что по вашему баннеру щелкнут.

Другие способы раскрутки сайта

Не забывайте везде, где можно указывать адрес своего сайта: в подписи электронной почты, на ваших визитных карточках, если они у вас есть, в строке подписи на форумах, на которых вы общаетесь. В Интернет-общении ссылайтесь на материалы своего сайта, если такая ссылка уместна и т. д. Хотя каждый из этих способов принесет вам считанные единицы посетителей, примененные вместе, они могут дать вам весомую прибавку посещаемости. Кроме того, если вы общаетесь с людьми, которым может быть интересна тематика вашего сайта, то велика вероятность, что они станут вашими постоянными посетителями.

Запрещенные приемы раскрутки сайта

Есть ряд приемов раскрутки сайта, которыми не стоит пользоваться ни в коем случае, чтобы не потерять в одночасье уважение Интернет-сообщества, остаться без привлеченных с большим трудом постоянных посетителей и получить отказ в обслуживании от своего хостинг-провайдера.

- Спам. Одним из самых страшных грехов в Интернете считается рассылка спама. Спам – это непрошенная почтовая корреспонденция рекламного характера, заполоняющая электронные почтовые ящики. В последнее время спам настолько осточертел даже самым неискушенным пользова-

телям Интернета, что использование его в рекламных целях приводит скорее к обратному эффекту, получив спам-сообщение с рекламой определенного сайта, интернетчик не посетит его, даже если в последствии наткнется на его адрес в других источниках информации. Кроме того, в современном Интернете спам безоговорочно признается преступлением против всего Интернет-мира и тот, кого уличили в рассылке спама, становится изгоем в Сети. Хостинг-провайдер на законных основаниях откажет ему в поддержке сайта, Интернет-провайдер может отключить от сети Интернет, а если станут известны физические координаты спамера, то ему грозит и физическая расправа.

- Поисковый спам. Вторым смертным грехом, правда только для поисковых систем, является поисковый спам, или чрезмерная оптимизация Web-страниц для повышения рейтингов в поисковиках. В случае уличения владельца сайта в поисковом спаме, его сайт просто навсегда исключают из индексирования в поисковике. Поэтому будьте очень осторожны, применяя на практике советы по оптимизации сайта для поисковых машин, чаще всего игра не стоит свеч. Поисковые системы являются одним из основных источников новых посетителей, и их потеря будет достаточно серьезным препятствием на пути развития вашего сайта.
- Назойливая самореклама. Хотя мы рекомендовали ссылаться как можно чаще на свой сайт, но необходимо делать это ненавязчиво, чтобы не набить оскомину потенциальной аудитории. Каждый раз подумайте, уместно ли размещать ссылку, прежде чем сделать это.
- Накрутка счетчиков. Существует масса способов обмануть счетчики посещения ваших страниц, создав видимость повышенной их посещаемости. Делается это для улучшения позиции сайта в рейтингах, а также для повышения привлекательности в глазах рекламодателей, размещающих на сайте баннеры. Почему не стоит пользоваться такими методами? Потому же, почему не стоит заниматься поисковым спамом. В конечном итоге обман раскроется и тогда вас «попросят» изо всех рейтингов.
- Использование чужих информационных материалов без указания источника. Делать это не стоит в первую очередь по морально-этическим соображениям, а во вторую, чтобы не потерять уважение ваших постоянных посетителей, когда факт заимствования всплывет, а это произойдет, и скорее рано, чем поздно. Если вам уж очень хочется разместить у себя материалы с другого сайта, обязательно укажите авторство и дайте ссылку на первоисточник, а еще лучше спросите разрешения автора на публикацию его работы.

В общем и целом, можно сформулировать основное правило работы в сети Интернет: «честным быть выгодно и удобно». И еще одно замечание, в завершение раздела о методах раскрутки сайта. Главным ресурсом сети Интернет являются люди. И главная задача любого Web-мастера – привлечь на сайт людей и удержать их. Если первая задача решается грамотной раскруткой Web-ресурса, то вторая – исключительно дело качественного и хорошо оформленного содержания сайта. Не раскру-

ченный сайт с добротным и интересным содержанием худо-бедно, но будет набирать популярность, а раскрученная пустышка непременно вскоре канет в небытие.

Проблемы роста Web-сайта

Вот вы создали сайт, разместили его на своем виртуальном хостинге, зарегистрировали в основных поисковых системах и начали постепенно раскручивать его, увеличивая популярность. И вот, с ростом популярности сайта, возникают и новые проблемы. Проблемы можно разделить на две группы: технические проблемы и проблемы содержания.

Технические проблемы

Технические проблемы скорее всего возникнут в том случае, если при создании сайта вы разместили его на бесплатном хостинге либо на коммерческом, но по самому дешевому тарифному плану, решив не рисковать своими деньгами. Либо поначалу задумывался м-а-а-а-ленький сайтик, с посещаемостью человек 20 в сутки, а он разросся до совершенно неожиданных размеров.

В любом случае, сервер перестал справляться с наплывом посетителей: сайт иногда открывается несколько минут, места категорически не хватает и т. д. Если вы пользуетесь платным хостингом, то все просто: либо переходите на другой тарифный план, либо меняете хостинг-провайдера, сохраняя свой прежний доменный адрес. Если же вы пользовались услугами бесплатного хостинг-провайдера, то тут вариантов гораздо больше.

- Вариант первый – хостинг-провайдер поддерживает сайты с произвольными доменами второго уровня и вы, воспользовавшись этой возможностью, зарегистрировали себе такой адрес. В таком случае вы ничем не связаны с бесплатным хостингом кроме воспоминаний. Подключайтесь к любому сервису платного хостинга, по вашему вкусу.
- Вариант второй, похуже. Вы пользуетесь доменным адресом, принадлежащим бесплатному хостингу, но имеете возможность перейти на платный хостинг у того же провайдера. У вас два варианта. Либо пойти по пути наименьшего сопротивления и остаться у провайдера, сменив тарифный план, либо поступить как в варианте третьем, сменив хостинг-провайдера и поменяв адрес сайта. Этот вариант сложнее, затратнее и чреват потерей части постоянных посетителей, но зато развязывает вам руки, вы больше не будете привязаны к одному хостингу.
- Вариант третий – вы пользуетесь доменным адресом, принадлежащим бесплатному хостинг провайдеру и не имеете возможности перейти на платный хостинг у него же. В этом случае у вас только один выход: заказать хостинг у другого провайдера и поменять адрес. Разумеется, если вы это проделаете, не предприняв дополнительных мер, вы потеряете большую часть своих постоянных посетителей и процесс раскрутки страницы вам придется начать заново. Чтобы этого не случилось, разместите

на месте прежнего сайта одну лишь страницу, примерное содержание которой указано в Листинг 8.1. Эта страница сообщит пользователям о переезде вашего сайта и предложит на него перейти (Рис. 8.37). Другой вариант – автоматическая переадресация, через некоторое время, если посетитель не нажал на ссылку, браузер сам автоматически перейдет на новый сайт. Код такой страницы приведен в Листинг 8.2. Полученная страница показана на Рис. 8.38.

Листинг 8.1. Код страницы ручной переадресации

```
<html>
<head>
    <title>мы переехали</title>
</head>
<body>
   Наш сайт переехал <br>
   Наш новый адрес: <a
href="http://new_adress.ru">http://new_adress.ru</a><br>
</body>
</html>
```

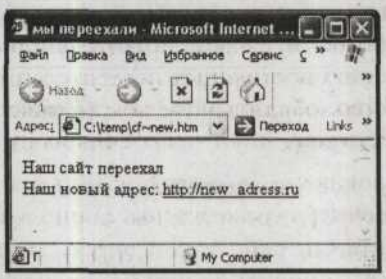


Рис. 8.37. Ручная переадресация страницы

Листинг 8.2. Код страницы автоматической переадресации

```
<html>
<head>
<META HTTP-EQUIV="refresh" content="3;
URL=http://new_adress.ru">
    <title>мы переехали</title>
</head>
<body>
   Наш сайт переехал <br>
   Наш новый адрес: <a href="http://new_adress.ru">
http://new_adress.ru </a><br>
   Если в течение трех секунд браузер не перейдет на новый сайт
автоматически, щелкните по ссылке.
</body>
</html>
```

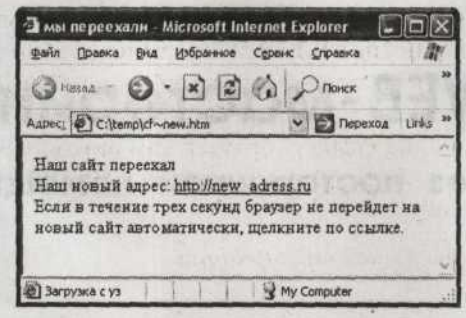


Рис. 8.38. Страница автоматической переадресации

Проблема удержания посетителей

Проблемы содержания обычно сводятся к тому, что приходят все новые и новые посетители, чтобы они периодически заходили на сайт, необходимо создавать все новые и новые материалы, что занимает все больше и больше вашего времени, пока не пожирает его целиком.

Частичным решением такой проблемы является реализация идеи, что пользователи сами должны создавать содержание сайта. Добавьте на сайт форум. Наделите нескольких пользователей правами модераторов, чтобы они могли направлять течение жизни в форуме. Предложите наиболее активным посетителям писать статьи для вашего сайта и так далее. Если количество постоянных посетителей превышает некоторую критическую массу, то сообщество, образованное вокруг вашего сайта, будет самоподдерживающимся. Такое сообщество называют также **community** (общность).

Развитие сайта до такого уровня уже позволит окупать его содержание размещением одного-двух баннеров. Эти баннеры даже будут вам приносить некоторую прибыль.

Образование **community** вовсе не означает, что вы достигли максимума возможностей, все еще только начинается, впереди огромные резервы для развития, но это уже тема для другой книги.

Заключение

Вот и закончилась наша книга. Начинали мы ее со знакомства с всемирной компьютерной сетью Интернет, а закончили созданием и размещением в Сети и раскруткой полноценных Web-проектов, созданных по всем правилам современных компьютерных технологий. Вы проделали большой путь, но еще больший вам предстоит, поскольку в одной книге невозможно описать все многообразие Интернет-технологий и все возможности, которые предоставляет это величайшее изобретение человечества. Да это невозможно сделать и в сотне томов, ведь Интернет – это живой организм, постоянно изменяющийся и развивающийся. Постоянно появляются новые технологии, отмирают старые, жизнь идет. Но мы постарались дать вам основу, зная которую, вы всегда сможете легко ориентироваться в море информационных технологий, которое представляет собой Интернет. Удачного вам плаванья!

Б. Артанов

WEB-мастеринг

без посторонней помощи

Отдел распространения издательской группы «ТРИУМФ»:

- ✔ «Издательство Триумф»
- ✔ «Лучшие книги»
- ✔ «Технический бестселлер»
- ✔ «Только для взрослых»
- ✔ «Технолоджи – 3000»
- ✔ «25 КАДР»
- ✔ «100 КНИГ»

Телефон: (095) 720-07-65, (095) 772-19-56. E-mail: opt@triumph.ru

Интернет-магазин: www.3st.ru

КНИГА-ПОЧТОЙ: 125438, г.Москва, а/я 18 «Триумф».
E-mail: post@triumph.ru

Главный редактор издательства «100 книг» А.Г. Жадаев
Над текстом книги работал Т.В.Татуревич
Выпускающий редактор И.Г. Колмыкова
Дизайн обложки А.В. Чубарь
Литературный редактор С.Л. Крючкова
Компьютерная верстка А.В. Чубарь .

Подписано в печать с оригинал-макета 03.11.2005 г.
Формат 70×100$^{1}/_{16}$. Печать офсетная. Печ. л. 21. Заказ № 1558.

Издательская группа «ТРИУМФ»

ООО «ТЕХНОЛОДЖИ – 3000»
125438, г. Москва, а/я 18.

Отпечатано в полном соответствии с качеством предоставленных диапозитивов
в ОАО «Можайский полиграфический комбинат»
143200, г. Можайск, ул. Мира, 93