МОУ «Лицей № 43»

(естественно-технический)

**СОЗДАНИЕ САЙТОВ**

Спирина Елизавета

10 класс

Саранск

2012

**Аннотация к реферату.**

Рассказывается о способах создания сайтов для новичков. Описываюся специально приспособленные для этого программы. В заключении предложен наипростейший способ создания сайта.

**Оглавление:**

**1.Введение.**

**2.Как создать сайт бесплатно.**

**3. Разработка шаблона сайта.**

**4.Разработка сайта без участия интернета.**

**5.** **Язык HTML, JAVASCRIPT и WWW.**

**6.** **Публикация сайта в Интернет и размещение сайта на сервере.**

**7. Простое решение по созданию сайта**

**1.Введение.**

Создание сайта – процесс довольно простой, однако таким он начинает казаться уже после того, как вы убьете значительную часть своего времени на изучение всего процесса, начиная с изучения HTML, и заканчивая поиском дешевого и качественного хостинга и публикации своих страниц. Попробуем вкратце описать процесс создания сайта, акцентировав внимание читателей на наиболее часто возникающих у начинающих веб-мастеров проблемах.

Итак, самое первое, с чем сталкивается решивший разместить в Интернет пару страничек человек, это HTML. HyperText Marcup Language – это язык разметки гипертекста, с помощью которого наши браузеры трансформируют документы в удобные для восприятия человеком странички. HTML не так сложен для изучения, как можно подумать, открыв впервые исходный код какой-нибудь веб-страницы. Смысл прост: HTML-документ содержит помимо текста служебные вставки – теги, в соответствии с которыми клиент (браузер) преобразует элементы страницы в должный вид. Благо, примеров предостаточно: страницу любого сайта можно открыть в браузере для просмотра исходного кода, и ознакомиться с HTML, а также CSS и JavaScript.

При создании сайта, веб-мастер, естественно, не набирает HTML-код вручную (хотя это довольно просто при наличии опыта), а использует программу-редактор, позволяющую быстро и без особого труда создавать HTML-страницы. Один из самых популярных редакторов HTML-кода – DreamWeaver, хотя, несомненно, есть и другие достойные альтернативы.

Научившись создавать сайты локально, можно приступать к публикации их в Интернет. Для этого нужен хостинг – сервер, размещенный в Сети, на котором будут храниться файлы сайтов. На пробу можно создать сайт на бесплатном хостинге. Лишь на пробу, т.к. такие ресурсы бесплатны лишь условно. Ведь хостинг-провайдер будет взамен размещать на ваших сайтах свою рекламу и зарабатывать деньги. Зачем это вам? Лучше, конечно, начинать создавать сайты сразу на платном хостинге. К счастью, цены на услуги хостинга сейчас уже не так высоки, как несколько лет назад, и за пару тысяч рэ в год можно купить неплохой профессиональный хостинг с поддержкой PHP, MySQL, FTP, почтой и прочими полезными сервисами.

Кроме хостинга, для создания сайта не хватает лишь одного – домена. Домен – это имя вашего сайта в сети. Всегда можно найти регистраторов доменов, проводящих различные акции, и покупать таким образом домены по смешным ценам. Например, webmoney сейчас регистрирует рушки по 100 р.

После того, как куплены хостинг и домен, можно приступать к публикации созданного сайта в Интернет. Для этой цели используются либо FTP-клиент, либо SSH (пользователи ОС Windows обычно предпочитают FTP). Посредством FTP (File Transfer Protocol), локальные файлы сайта копируются на сервер хостинг-провайдера, после чего созданный сайт становится доступен для просмотра посетителям.

**2.Как бесплатно создать сайт.**

Бесплатно создавать сайты сейчас становится все менее модно. Ведь стоимость услуг платного хостинга, как и стоимость регистрации доменов, сейчас стали гораздо доступнее, чем пару-тройку лет назад. Купить качественный хостинг можно за 1-2 тыс. руб. в год, а домены в зоне “.ru” продаются уже по 100 руб. – не не смешны ли такие цены. Даже самый маленький и невзрачный рекламный блок AdSense или Бегуна на сайте легко отобьет вложения. Конечно, не все зарабатывают (раз уж заговорили о заработке в Интернет) на одном-двух сайтах. Многие делают сайты тысячами, и, естественно, такие веб-мастера заинтересованы в том, чтобы каждый сайт обходился если не бесплатно, то, хотя бы, чтобы затраты на его создание были минимальными. Бесплатный хостинг является хорошей возможностью создавать такие сайты.

Наиболее популярными сервисами, предоставляющими бесплатно PHP, MySQL, Perl, .htaccess, неограниченное дисковое пространство, всегда были agava и fatal. Увы, в настоящее время регистрация новых сайтов временно прекращена на обоих.

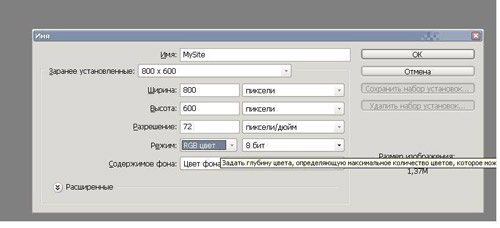
В качестве альтернативы, пожалуй, может подойти ucoz. Несмотря на отсутствие возможности использовать собственные PHP-скрипты, мерзкую службу поддержки, запрет на индексацию сайта поисковыми роботами в первый после публикации месяц, имеется способ продавать ссылки на страницах размещенных на этом бесплатном хостинге сайтов.

Также существуют бесплатные хостинги без поддержки PHP, но обычно они пользуются спросом только у совсем уж зеленых новичков: by.ru, nm.ru, narod.ru и многие другие. На таких сайтах тоже можно размещать рекламу, продавать ссылки, однако поисковые системы с пренебрежением относятся к бесплатно созданным сайтам, и привлечь посетителей на такой сайт будет очень не легко.

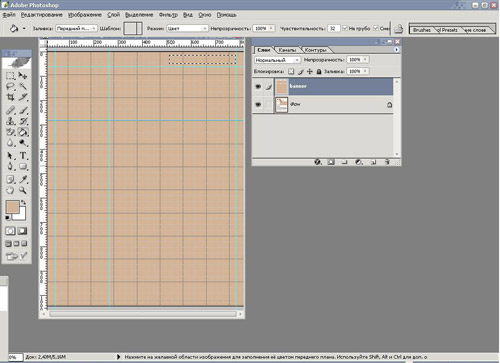
Что касается доменов, как правило, на бесплатном хостинге для сайтов выделяются субдомены, но позволяется размещать собственные домены второго или третьего уровня. Например, можно зарегистрировать бесплатно домен третьего уровня на co.cc, и припарковать его на свой блог на blogger.com.

**2.Разработка шаблона сайта.**

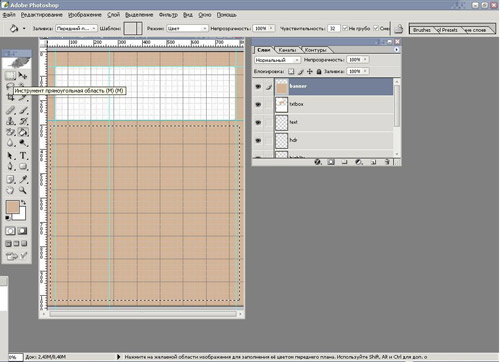
Как правило, этот этап работ выполняется в достаточно мощном графическом редакторе, например, Adobe Photoshop или CorelDraw. Фотошоп несколько проще в освоении и использовании, а возможности его очень велики. Именно поэтому среди веб-дизайнеров Photoshop пользуется вполне заслуженной популярностью. Воспользуемся им и мы.  
  
Создаем в Фотошопе новый документ. Задаем имя – MySite.   
Выбираем разрешение 800 х 600, это гарантирует корректное отображение у любого пользователя, размер по вертикали потом сможем увеличить. Выбираем разрешение 72 пикселя на дюйм и RGB-цвет. Эти установки делаем обязательно, именно они отвечают за корректное отображение веб-страницы.

[](http://www.internet-technologies.ru/uploads/pages/site_0.jpg)

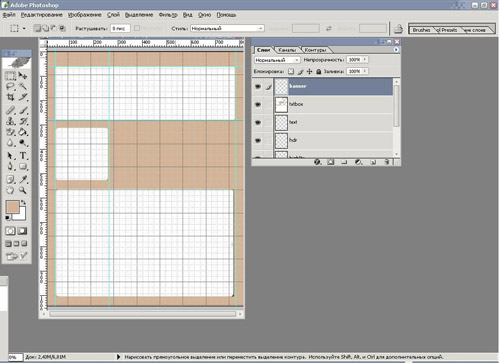
Устанавливаем цвет фона D3B497 в шестнадцатеричном виде или выбираем с помощью палитры цветов.   
Выбрав пункт меню «Вид» – «Направляющие» активируем отображение линеек и направляющих.  
В пункте меню «Вид» - «Показать» включаем сетку  
Выбираем инструмент «Перемещение» и, щелкнув на левой или верхней линейке, вытаскиваем вертикальные и горизонтальные направляющие  
Получаем размеченную область.  
В пункте меню «Вид» - «Привязка к» не забываем убедиться, что включена привязка к направляющим и границам документа.

[](http://www.internet-technologies.ru/uploads/pages/site_5.jpg)

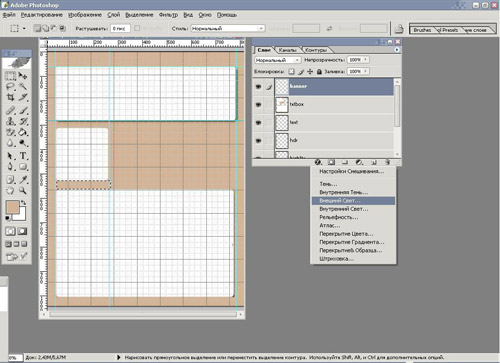
Командой «Слой» - «Новый» создаем слой с именем banner.   
Используя инструмент «Прямоугольная область» выделяем нужный нам фрагмент между направляющими.  
Рисуем в этой области прямоугольник со скругленными углами.  
Выбираем цвет заливки и заливаем его.

[](http://www.internet-technologies.ru/uploads/pages/site_6.jpg)

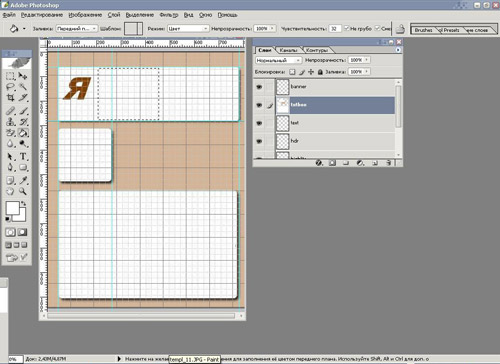
Таким же образом размечаем еще несколько областей – для логотипа, текста приветствия, и описательного текста. Слой для текста приветствия не заливаем.

[](http://www.internet-technologies.ru/uploads/pages/site_7.jpg)

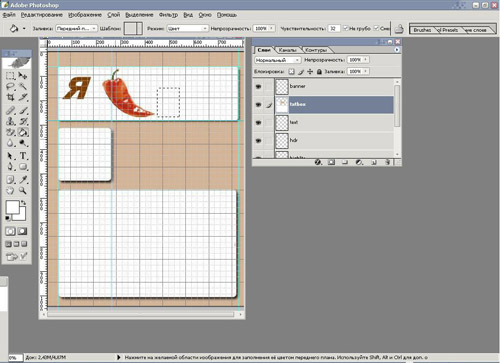
Для каждого из залитых слоев выбираем эффект «Внешний свет», задаем направление, получаем отбрасываемую тень

[](http://www.internet-technologies.ru/uploads/pages/site_8.jpg)

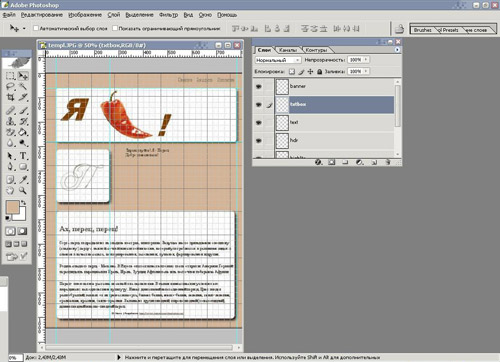
На области логотипа создаем слой с тем же цветом заливки.   
С помощью инструмента «Горизонтальный текст» вводим текст.

[](http://www.internet-technologies.ru/uploads/pages/site_10.jpg)

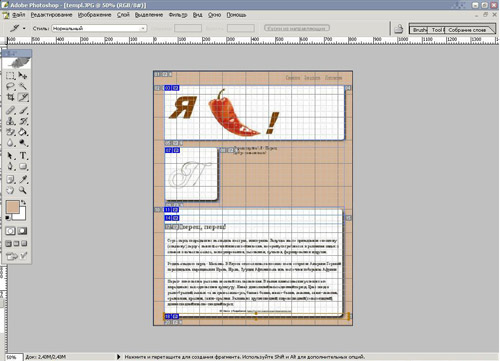
Создаем на области баннера новый слой. Открываем файл с изображением и просто перетаскиваем его на вновь созданный слой.

[](http://www.internet-technologies.ru/uploads/pages/site_11.jpg)

Аналогично заполняем изображениями и текстом остальные области

[](http://www.internet-technologies.ru/uploads/pages/site_f.jpg)

Наш макет готов!  
  
Теперь из него нужно вырезать фрагменты, необходимые для нашей страницы. Это – область баннера, область логотипа, верхняя и нижняя рамки для поясняющего текста и часть области, для фона поясняющего текста.  
С помощью инструмента «Фрагмент» выбираем необходимые части изображения.

[](http://www.internet-technologies.ru/uploads/pages/site_12.jpg)

В меню «Файл» выбираем «Сохранить для Веб». Далее настраиваем качество изображений и сохраняем, в результате получаем множество графических фрагментов нашего шаблона. В папке, куда был сохранен сам шаблон, появилась папка images с изображениями. Отбираем нужные и переименовываем.  
  
Макет создан, необходимые фрагменты получены, можно переходить к верстке. Конечно, пока мы воспользовались лишь самыми примитивными возможностями Adobe Photoshop. Книги по веб-дизайну помогут более полно изучить его возможности и познакомиться с другими редакторами.  
  
Для быстрого старта зачастую выгоднее воспользоваться готовыми шаблонами, чем рисовать их самим. В разделе шаблоны сайтов можно найти много прекрасных шаблонов и создать на их основе свой сайт.

**4.Разработка сайта без участия интернета.**

Что нам понадобится, чтобы выполнить эту задачу?

   Первое что необходимо сделать, это обзавестись соответствущим программным обеспечением. Качаем из сети (версии для Windows):

   - Вебсервер Apache

   - Язык программирования Perl

   - Язык программирования PHP

   - Базу данных MySQL

   Первым желательно скачать сам Apache так как пока докачивается остальное, займемся его установкой и настройкой.

   Установка и настройка Apache

   Установка самого Apache не должна вызвать у вас каких-то вопросов и сложностей. Сначала вы выбираете директорию, куда устанавливать сам сервер, следом название группы (оставьте как есть), а потом конфигурацию установки - выбирайте Typical. Не спешите сразу запускать сервер - ничего хорошего из этого не выйдет, сначала надо его настроить. Переходим в директорию куда вы ставили Apache, затем в директорию conf внутри этой директории и любым текстовым редактором (лучше всего блокнотом) открываем файл httpd.conf на редактирование. Находим строчку '#Listen 12.34.56.78:80' (здесь и далее одинарные кавычки только для четкого определения строки, в файле конфигурации их быть не должно) и заменяем ее на 'Listen 127.0.0.1:80'. Чуть ниже видим строчку '#BindAddress \*', ее меняем на 'BindAddress 127.0.0.1'. Листаем файл еще ниже и обнаруживаем строку 'ServerAdmin you@your.address', здесь вместо you@your.address подставляем адрес своей электронной почты.

    Таким образом эта строка примет например такой вид "ServerAdmin oleg@digin.ru'. Листаем еще немного ниже и видим строку вида '#ServerName new.host.name'. Вместо нее, поскольку у нас локальная машина, вписываем 'ServerName 127.0.0.1'. А сейчас внимание. Создадим на вашем винчестере например на диске C директорию в которой будет находиться ваш сайт. Назовем ее server. Таким образом путь к вашему локальному серверу будет 'c:/server'. Запомним этот адрес, а лучше куда-нибудь запишем, он нам пригодится. После этого продолжаем конфигурировать Apache. Находим строку начинающуся со слов 'DocumentRoot', далее в ней идет некий адрес на вашем винчестере. Вместо этого адреса подставляем адрес только-что созданной серверной директории. После этого строка может выглядеть так 'DocumentRoot "c:/server"'. Обратите внимание, что в конце адреса мы не добавляем слэш. Далее находим такой небольшой абзац: '

   <Directory />

   Options FollowSymLinks

   AllowOverride None

   </Directory>

   ' и его полностью заменяем на приблизительно такой: '

   <Directory "c:/server">

   Options Indexes FollowSymLinks MultiViews +Includes

   AllowOverride None

   </Directory>

   ', где 'c:/server' адрес вашей серверной директории. Затем немного ниже видим строчку начинающуюся так '<Directory'. В этой строке заменяем адрес директории на адрес вашей серверной директории, например '<Directory "c:/server">'. Далее несколько ниже обнаруживаем строку 'Options Indexes FollowSymLinks MultiViews', ее заменяем на 'Options Indexes FollowSymLinks MultiViews +Includes'. Далее находим строку '#AddHandler server-parsed .shtml' и убираем перед ней значок хэша. Получаем 'AddHandler server-parsed .shtml'. На этом первоначальная конфигурация вебсервера Apache завершена, в том числе мы его настроили и для исполнения SSI директив. Сейчас, когда уже скачались Perl и PHP можно включить поддержку этих двух языков в ваш Apache, но до этого сначала поставим эти языки в систему.

   Ставим Perl

   Сначала добавим в систему Perl. Если Perl не нужен можно сразу перейти к следующему разделу. Установка самого Perl не вызовет у вас каких-либо затруднений. Только запомните путь к директории в которую вы его устанавливаете. Затем выключите Apache (если он у вас включен) и вновь откройте на редактирование конфигурационный файл Apache. Листаем его в самый верх и начинаем по порядку. Ищем строчку '#!c:/program files/perl/perl' и заменяем ее (в случае если ваш Perl вы ставили в 'c:/perl') на '#!c:/perl/bin/perl.exe'. Далее находим строку начинающуюся 'ScriptAlias /cgi-bin/' и меняем в ней путь к cgi-bin директории вашего сервера. Например эта строка будет выглядеть как 'ScriptAlias /cgi-bin/ "c:/server/cgi-bin/"'. Обратите внимание что вам надо создать директорию cgi-bin в вашей серверной директории и что в конце пути здесь добавляется слэш. Еще несколькими строчками ниже видим абзац начинающийся с '<Directory', и здесь тоже меняем адрес cgi-bin директории вашего сервера. Эта строка будет выглядеть например так '<Directory "c:/server/cgi-bin/">'. Еще листаем вниз и находим строчку '#AddHandler cgi-script .cgi'. Убираем перед ней хэш и она выглядит 'AddHandler cgi-script .cgi'. Все, теперь ваш Apache настроен и для исполнения Perl-скриптов.

   Установка и настройка PHP

   Те, кто не пользуется PHP могут этот раздел пропустить. Так-же как те, кто не пользуется Perl могли пропустить раздел предыдущий. Итак, устанавливаем PHP. Сама установка PHP происходит элементарно. После скачивания архива распакуйте его например в директорию 'c:/php'. При этом обратите внимание что в архиве может быть запакована корневая директория, в таком случае перепишите в 'c:/php' не ее саму а ее содержимое. После этого найдите в директории файл php4ts.dll и скопируйте его в директорию 'c:/windows/system' (или соответственно туда, где у вас находится подобная директория). После этого необходимо вновь поправить конфигурацию Apache. Листаем файл httpd.conf в самый верх и приступаем. Найдите строчку '# Dynamic Shared Object (DSO) Support' и после нее добавьте две строки 'LoadModule php4\_module c:/php/sapi/php4apache.dll' (если вы ставили PHP в 'c:/php') и 'AddModule mod\_php4.c'. После этого листаем вниз и находим строчку '#AddType application/x-httpd-php .php', убираем здесь хэш и получаем 'AddType application/x-httpd-php .php' а также строкой ниже добавляем 'AddType application/x-httpd-php .php3'. На этом установка PHP и настройка Apache для него закончены.

   Установка MySQL

   При установке базы данных MySQL серьезно отнеситесь к выбору логина и пароля для доступа к базе. Эти логин и пароль вы будете использовать для управления вашими данными. Адресом хоста вашей локальной базы будет 'localhost'. При переносе программного обеспечения, работающего с MySQL на вашей локальной машине, в интернет, не забудьте изменить данные доступа на соответствующие вашему хостингу. Apache настраивать для работы с MySQL не требуется.

   Когда все поставлено и настроено

   Перезагрузите машину. На всякий случай, хуже все равно уже не будет. После перезагрузки пробуем запустить все что мы тут наставили и нанастраивали. Поехали - из кнопки Start запустим Apache. Если все настроено верно то появится и не пропадет консольное окно с текстом вроде этого 'Apache/1.3.14 <Win32> PHP/4.1.3-dev running...' (в случае если мы ставили также и PHP). Если окошко выскочило и тут-же закрылось - имеет место ошибка конфигурации, но об этом позже. Итак, все запустилось, пишем какой-нибудь простенький html типа 'hello, world', сохраняем его в 'c:/server' как index.html и открываем в любом браузере адрес 'http://127.0.0.1'. Все:) Скажу еще что Perl-скрипты надо запускать из 'c:/server/cgi-bin' набирая в строке запроса что-то типа 'http://127.0.0.1/cgi-bin/test.cgi', а PHP скрипты и страницы можно запускать откуда угодно из вашей серверной директории. Да, SSI директивы также будут исполняться на вашем сервере, вот только такие файлы награждайте расширением shtml. Когда пишете скрипты, использующие в своей работе MySQL, не забывайте этот самый MySQL запускать. Выключайте Apache не закрывая консольное окно, а выбирая соответствующий пункт из программной группы Apache. И если у вас все с настройкой сервера в порядке, порекомендую вам пользоваться небольшой но потрясающе удобной утилиткой Apache Manager для запуска, перезапуска и выключения сервера. Скопировать ее нужно в директорию к Apache, а потом просто запустить.

   Подводим итоги

   У нас есть работающий домашний сервер, эдакий интернет в миниатюре. Запускаем Apache, MySQL и любимый текстовой редактор, открываем DOC'и, FAQ'и и прочую литературу и создаем шедевр. На этом прощаюсь, надеюсь был вам полезен;)

**5.** **Язык HTML, JAVASCRIPT и WWW.**

Волна бурного развития технологий World Wide Web не обошла стороной

и средства управления сценариями просмотра гипертекстовых баз данных. Сегодня

авторы домашних страниц Web получили в свое распоряжение новые языки JavaScript

и VBScript. Новые возможности по формированию баз данных Всемирной Паутины

приобрел также уже зарекомендовавший себя язык HTML, что особенно видно

в его версии 3.2 [1-8]. Темой данной статьи станет рассмотрение новых возможностей

HTML и JavaScript, используемых для программирования сценариев просмотра

страниц Web.

Как известно, язык гипертекстовой разметки HTML (HyperText Markup Language)

был предложен Тимом Бернерсом-Ли в 1989 году в качестве одного из компонентов

технологии разработки распределенной гипертекстовой системы World Wide

Web. В основу гипертекстовой разметки была положена таговая модель описания

документа, позволяющая представить документ в виде совокупности элементов,

каждый из которых окружен тагами. По своему значению таги близки к понятию

скобок "begin/end" в универсальных языках программирования и

задают области действия имен локальных переменных, определяют область действия

правил интерпретации текстовых элементов документа и т.п.

Общая схема построения элемента текста в формате HTML может быть записана

в следующем виде:

"элемент":= <"имя элемента" "список атрибутов">

содержание элемента </"имя элемента">

Конструкция перед содержанием элемента называется тагом начала элемента,

а конструкция, расположенная после содержания элемента, - тагом конца элемента.

Структура гипертекстовой сети задается гипертекстовыми ссылками. Гипертекстовая ссылка - это адрес другого HTML документа или информационного ресурса Internet, который тематически, логически или каким-либо другим способом связан с документом, в котором эта ссылка определена.

Для записи гипертекстовых ссылок в системе WWW была разработана специальная

форма - Universe Resource Locator. Типичным примером использования этой записи можно считать следующую запись:

Этот текст содержит <A HREF = "http://polyn.net.kiae.su/ altai/index.html">

гипертекстовую ссылку </A>.

Здесь элемент "A", который в HTML называют якорем (anchor),

использует атрибут "HREF", обозначающий гипертекстовую ссылку

для записи этой ссылки в форме URL. Данная ссылка указывает на документ

с именем "index.html" в каталоге "altai" на сервере

"polyn.net.kiae.su", доступ к которому осуществляется по протоколу

"http".

Структура HTML-документа позволяет использовать вложенные друг в друга

контейнеры. Собственно, сам документ - это один большой контейнер с именем

"HTML":

<HTML> Содержание документа </HTML>

Сам элемент HTML или гипертекстовый документ состоит из заголовка документа

(HEAD) и тела (BODY):

<HTML>

<HEAD>

Содержание заголовка

</HEAD>

<BODY>

Содержание тела документа

</BODY>

</HTML>

Приведенная форма записи определяет классический HTML-документ, например:

Пример 1.

<HTML>

<!-

Author: Irina Kuzina

Date: December 5, 1996

->

<HEAD>

<TITLE> This is a

Baner</TITLE>

</HEAD>

<BODY BACKGROUND=www\_wall.jpg

VLINK=0000FF LINK=FF0000>

<CENTER>

<TABLE>

<TR><TD><IMG

SRC="interne0.jpg"></TD>

<TD CENTER>

<H3> Администрирование

Internet</H3>

<I> Справочное руководство.

</I>

</TD></TR>

</TABLE>

</CENTER>

</BODY>

</HTML>

Каждый документ в WWW имеет свое имя, которое указывается в его заголовке.

Его можно видеть в первой строке программы-интерфейса. Контейнер BODY открывает

тело документа. В качестве фона в этом элементе определена картинка www\_wall.jpg,

которая в примере 1 задана частичной формой спецификации URL без указания

полного адреса ресурса в сети. Далее в документе определена таблица, состоящая

из двух ячеек: картинка и текстовый фрагмент. Текст определен как заголовок

третьего уровня, который должен отображаться стилем Italic.

Фреймы

Компания Netscape Communication расширила классическую форму документа

возможностью организации фреймов, а в Microsoft были разработаны фреймы

без визуальных границ, имеющие или нет механизм прокрутки - так называемые,

плавающие фреймы. Организация фрейма - это возможность разделить рабочее

окно программы просмотра на несколько независимых панелей (или фреймов).

В каждый фрейм может быть загружена своя страница HTML. Плавающие фреймы

позволяют вставить одну Web-страницу в другую. Тогда как обычные фреймы

позволяют лишь разделить область просмотра на несколько частей, плавающий

фрейм можно поместить где угодно на странице, указав его размер и отступ

справа или слева.

Для описания плавающих фреймов используется таг <IFRAME>...</IFRAME>.

В отличие от спецификации Netscape, не разрешающей пересечение фреймов,

нотации Microsoft позволяет накладывать одни фреймы на другие с помощью

отрицательного отступа.

В каждый фрейм может быть загружена своя HTML-страница, представляющая

собой классический HTML-документ.

В примере 2 представлен документ, состоящий из четырех окон внутри рабочего

окна программы просмотра, в каждое из которых загружается обычный документ.

Верхний и нижний фреймы имеют размер 100 пикселей, а оставшаяся область

рабочего окна разбивается на две части: на левую отводится 30%, а на правую

- 70% пространства. Если программа просмотра не поддерживает организацию

фреймов: то отображается информация, заключенная между тагами <NOFRAMES>

... </NOFRAMES>, если же эта возможность реализована, то такая область

на экране не отображается.

Пример 2.

<HTML>

<!-

Author: Irina Kuzina

Date: December 5, 1996

->

<HEAD>

</HEAD>

<FRAMESET ROWS = "100, \*,

100">

<NOFRAMES>

<BODY>

Sorry, there is not a frame

support in your browser.

</BODY>

</NOFRAMES>

<FRAME SRC = booter.html

NAME = boot>

<FRAMESET COLS = "%30, %70">

<FRAME SRC = menli.html NAME

= left>

<FRAME SRC = t2.html NAME =

right>

</FRAMESET>

<FRAME SRC = footer.html

NAME = foot>

</FRAMESET>

</HTML>

Создание многооконного интерфейса

С фреймами тесно связана возможность именования окон - любой фрейм может

иметь свое собственное имя. Для того, чтобы документ загружался именно

в то окно, которое определил автор страницы, в контейнер гипертекстовой

ссылки вводится атрибут TARGET. При организации фреймов (пример 2) каждому

из них было присвоено имя (атрибут NAME контейнера FRAME). Для загрузки

документа в конкретный фрейм (left), автор должен использовать гипертекстовую

ссылку вида:

<A HREF = frame.html TARGET = left>

В данном случае документ, указанный в атрибуте HREF, будет загружен

во фрейм left.

Если в программе просмотра нет окна с именем left, то будет открыто

новое окно с таким именем и в него будет загружен требуемый документ. Приведем

пример открытия дополнительного окна (пример 3).

Пример 3.

<HTML>

<HEAD>

</HEAD>

<BODY BGCOLOR=#FFbcbc >

<H1> <A HREF = frame.html target = left> How to create

a multi-window interface

</A> </H1>

<h3>

In the document "frame.html"

there is no frame named

"left".<br>

That"s why the program will

open a new window </h3>

</BODY>

</HTML>

Открывшееся окно содержит разбиение рабочей области на поименованные

фреймы.

Если теперь пользователь выберет гипертекстовую ссылку из окна notleft,

то файл new.html будет загружен в окно фрейма right (примеры 4, 5).

Пример 4.

<HTML>

<HEAD>

</HEAD>

<FRAMESET ROWS = "100,\*,100">

<FRAME SRC = t2.html NAME =

top>

<FRAMESET COLS = "%40, %60">

<FRAME SRC = t2.html NAME =

notleft>

<FRAME SRC = t2.html NAME =

right>

</FRAMESET>

<FRAME SRC = t2.html NAME =

bottom>

</FRAMESET>

</HTML>

Пример 5.

<HTML>

<HEAD>

</HEAD>

<BODY bgcolor = #AF0000>

<A HREF = new.html TARGET =

right> <b> This is an

example of loading into

the right window </b> </A>

<center>

You can do it using the

TARGET element

</BODY>

</HTML>

В качестве реального примера можно привести открытие окна подсказки

в базе данных проекта Radleg. http://www.polegn.kial.su/redleeg.

Окно подсказки появляется после выбора гипертекстовой ссылки "Help".

При этом в данное окно загружается страница с фреймами, где среднее окно

- это контекстный Help, верхнее - стандартная заставка базы данных, а нижнее

окно - переход к "домашней" странице Radleg и закрытие окна подсказки.

Использование HTML для определения полей заголовка

сообщения протокола HTTP

Важным свойством современных версий HTML является возможность записи

полей заголовка сообщения протокола HTTP. Реализуется эта возможность через

контейнер заголовка HTML - документа <META>. Используя атрибут HTTP-EQUIV,

в заголовок HTTP - сообщения можно включить любое поле, которое определено

стандартом протокола HTTP (текущая версия 1.0). Рассмотрим в качестве примера

организацию рекламных роликов. Идея состоит в том, чтобы использовать время

обновления HTML страниц для их автоматической подмены при просмотре. Время

жизни определяется полем Refresh заголовка сообщения HTTP:

Refresh = время; URL = документ <LF>

Используя контейнер META с атрибутом HTTP-EQUIV, автор HTML - документа

может определить следующий документ или кадр рекламного ролика:

<META HTTP-EQUIV = "Refresh", CONTENT = "0, URL=ref2.http">

В общем случае можно построить замкнутую или незамкнутую цепочку кадров,

которая и будет из себя представлять рекламный ролик.

В примере 6 описан документ, который после своей загрузки будет немедленно

заменен документом ref2.html. В реальной практике такая немедленная замена

происходит не мгновенно и зависит от скорости коммуникаций и особенностей

ПО и ОС.

Пример 6.

<HTML>

<HEAD>

<TITLE> type\_Document\_ Title\_

here </TITLE>

<META HTTP-EQUIV = "Refresh"

CONTENT = "0; URL =

ref2.htm">

</HEAD>

<BODY>

<H1>Document 1</H1>

</BODY>

</HTML>

Программирование сценариев просмотра. JavaScript

С задачей управления отображением во фреймы и окна тесно связано написание

сценариев просмотра домашних страниц Web. Такие сценарии чаще всего пишутся

на языке Java Script.

Java Script - это язык управления сценарием отображения документа. Все

операции, которые можно исполнять в программе на JavaScript, описывают

действия над хорошо известными и понятными объектам - элементами рабочей

области программы Netscape Navigator и контейнерами языка HTML. В Java

Script не реализованы классы объектов, наследование, инкапсуляция и полиморфизм.

Имеются объекты с набором свойств и набор функций над объектами, которые

называются методами. Функция пользователя выполняется по наступлении некоторого

события: onChange, onClick, onLoad и др., а внутренние функции языка используются

непосредственно.

Не вдаваясь в подробности описания Java Script, сконцентрируем внимание

на контейнерах Java Script и примерах использования Java Script кодов.

Контейнер <SCRIPT LANGUAGE= "Java Script"> ... </SCRIPT>

может использоваться как в заголовке, так и в теле документа. Не все программы

просмотра способны распознавать и исполнять скрипты, поэтому само тело

скрипта помещается в контейнер комментария. Лучше всего описать применение

Java Script на примерах.

Идущие часы и бегущая строка

Данный тип объектов наиболее часто используется на страницах WWW. В

примере 7 приведен фрагмент программы, реализующей эти объекты:

Пример 7.

<HTML>

<HEAD>

<TITLE>JavaScript</TITLE>

<SCRIPT LANGUAGE =

"JavaScript">

<!- Hide script from user

adv\_string = "Internet\"

status\_string = adv\_string +

adv\_string + adv\_string +

adv\_string + adv\_string +

adv\_string

i=0

function background()

{

window.status = status\_

string.substring (i,i+180)

...

current\_date = new Date()

window.document.form1.

clock.value = current\_

date.getHours() + ":" +

current\_date.getMinutes

() + ":" + current\_

date.getSeconds()

id = setTimeout("back

ground()", 500)

window.document.form1.

kuku.value = "number"+i

}

//This is the end of code

definition ->

</SCRIPT>

</HEAD>

<BODY onLoad = "background()"

BACKGROUND=www\_wal0.jpg>

<H1>JavaScript</H1>

<FORM NAME=form1 ACTION =

"new\_window()">

<INPUT NAME = clock TYPE =

text SIZE = 8MAXLENGTH=8>

<HR>

<INPUT TYPE=button NAME =

help Value = "HELP"

onClick = "window.open

("clock.htm",

"Clock\_Window",

"scrollbars = yes,width =

450, height=350") ">.

<HR>

<INPUT NAME = kuku type =

text>

<HR>

</FORM

<P>

</BODY>

</HTML>

Для создания скрипта в заголовок документа (контейнер HEAD) включен

контейнер SCRIPT. До недавнего времени атрибут LANGUAGE в этом контейнере

не являлся обязательным. Но с момента появления VBSCRIPT смысл в указании

типа языка появился - Navigator не понимает скриптов на других языках.

Отсутствие атрибута может привести к ошибкам при отображении гипертекстовых

страниц. Далее в тексте страницы определен комментарий, в который включен

текст скрипта. Начинается комментарий строкой:

<!- Hide script from user

И кончается строкой:

// This is the end of code definition->

Функция, реализующая бегущую строку, использует метод open - встроенную

в JavaScript функцию, которая определена над объектом window. В результате

открывается новое окно Netscape Navigator и в него загружается документ.

Вызов функции осуществляется после нажатия на соответствующую кнопку. Таким

образом, функция будет выполнена только в том случае если произойдет событие,

описанное атрибутом onClick в контейнере INPUT.

Изначально предполагалось, что программа просмотра, которая поддерживает

JavaScript будет исполняться в среде многопотоковых операционных систем.

В этом случае к моменту порождения нового потока старый уже завершится

и setTimeout будет порождать последовательно исполняемые потоки. Однако,

из-за неаккуратной реализации JavaScript, команда setTimeout стала просто

"пожирать" ресурсы компьютера. Системы Windows 3.x вообще не

являются многопоточными и в них просто происходит переполнение стека при

выполнении данного скрипта, а как быстро "упадет" система зависит

от объема исполняемой по setTimeout функции. Самое лучше, что можно посоветовать,

если вдруг на экране появляется прокрутка - поскорее покинуть такую страницу.

Справедливости ради следует отметить, что прокрутку можно организовать

и другими способами, поэтому ее появление не обязательно должно вызывать

крах системы или крах программы просмотра.

JavaScript - это не единственный язык управления сценариями просмотра

документов. Имеется другая версия аналогичного языка - VBScript на основе

Visual Basic. Кроме того, управлять сценарием просмотра можно и из Java-апплетов,

что, конечно, сложнее, но зато более надежно и безопасно. При этом программист

получает возможность наследования и прочие атрибуты объектно-ориентированного

программирования. В конце концов, для создания фреймов и окон можно использовать

атрибуты соответствующих контейнеров HTML, которые позволяют делить рабочую

область экрана на фрагменты, перекрывать объекты и восстанавливать первоначальный

вид страницы. В Navigator 3.0 появились новые типы объектов и функции над

ними.

Массивы

Первый тип новых объектов, которые появились в JavaScript 1.1 - это

массивы. Тип "Array" введен в JavaScript 1.1 для возможности

манипулирования самыми разными объектами, отображаемыми навигатором. Это

- список всех гипертекстовых ссылок данной страницы, список всех картинок

на данной странице, список всех апплетов данной страницы, список всех элементов

формы и т.п. Пользователь может создать и свой собственный массив, используя

конструктор Array(). Делается это следующим образом:

new\_array = new Array()

new\_array5 = new Array(5)

colors = new Array

("red", "white", "blue")

Размерность массива может динамически изменяться, требуется лишь сначала

определить массив, а потом присвоить одному из его элементов значение.

Как только это значение будет присвоено, изменится и размерность массива:

colors = new Array()

colors[5] = "red"

В данном случае массив будет состоять из 6 элементов, первым элементом

считается элемент с индексом 0.

Для массивов определены три метода: join, reverse, sort. Join объединяет

элементы массива в строку символов, в качестве аргумента в этом методе

задается разделитель:

colors = new Array

("red", "white", "blue")

string = acolors.join("+")

В результате выполнения операции присваивания значения строке символов

string получим следующую строку:

string = "red"+"white"+"blue"

Метод reverse изменяет порядок элементов массива на обратный, а метод

sort отсортировывает их в порядке возрастания.

У массивов есть два свойства: length и prototype. Length определяет

число элементов массива. Если нужно выполнить некоторую рутинную операцию

над всеми элементами массива, то можно воспользоваться циклом типа:

color = new Array

("red", "white", "blue")

n = 0

while(n != colors.length)

{... операторы тела цикла

...}

Свойство prototype позволяет добавить свойства к объектам массива. Однако

наиболее часто в программах на JavaScript используются встроенные массивы,

главным образом графические образы (Images) и гипертекстовые ссылки(Links).

Графика

До появления версии Navigator 3.0 в JavaScript использовались только

встроенные объекты типа Image. В новой версии языка появился конструктор

для этого типа объектов:

new\_image = new Image()

new\_image = new Image (width,height)

Часто для реализации мультипликации создают массив графических объектов,

которые потом прокручивают один за другим:

img\_array = new Array()

img\_array[0] = new Image(50,100)

img\_array[1] = new Image(50,100)

....

img\_array[99] = new Image(50,100)

У объекта Image существует 10 свойств, из которых самым важным является

src. Для присваивания конкретных картинок элементам массива img\_array следует

воспользоваться следующей последовательностью команд:

img\_array[0].src = "image1.gif"

img\_array[1].src = "image2.gif"

...

img\_array[99].src = "image100.gif"

В данном случае можно было воспользоваться и циклом для присвоения имен,

так как они могут быть составлены из констант и значений индексной переменной.

В новой версии языка объект типа Image можно поименовать в HTML таге

IMG, а затем обращаться к нему по имени. При этом следует учитывать, что

если Image используется внутри формы, то он является свойством этой формы.

Это значит, что для следующего графического объекта должны применяться

разные составные имена:

<img name=car src=car.gif> <-

document.car.src = "car1.gif"

<form name=kuku>

<img name=car src=car.gif>

</form>

document.kuku.car.src = "car1.gif"

Однако наиболее часто в примерах использования скриптов можно встретить

обращение к Image по индексу в массиве всех графических объектов данной

страницы. Если наш объект, например, является вторым Image на странице,

то будь он внутри формы или за ее пределами к нему всегда можно обратиться

по индексу:

document.images[1].src = "car1.gif"

Расширяя пример с массивом Image, создадим документ, в который будет

встроена мультипликация, определенная нашим массивом:

Пример 8.

<HTML>

<HEAD>

<SCRIPT>

function multi\_pulti()

{

img\_array = new Array()

img\_array[0] = new

Image(50, 100)

....

img\_array[99] = new

Image(50, 100)

img\_array[0].src =

"image1.gif"

...

img\_array[99].src =

"image100.gif"

n=0

while(n==0)

{

document.images[0].src =

img\_array[0].src

...

}

}

</SCRIPT>

</head>

<body onLoad="multi\_pulti()">

<img src=image1.gif>

</body>

</html>

Довольно часто используют не мультипликацию, а выбор картинки через

OPTION - еще один новый объект JavaScript. При этом можно через поле формы

SELECT менять не только саму картинку, но и гипертекстовую ссылку, которая

может быть связана с этой картинкой. На гипертекстовую ссылку также можно

ссылаться по индексу:

document.links[index].href = kuku.html

Данный прием оправдан и с точки зрения интерфейса навигатора. При использовании

такого сорта массивов ссылок не требуется их длительное перечисление и

листание страницы в рабочей области навигатора - можно просто выбрать ссылку

из падающего меню.

Другой способ для сокращения числа нажатий на клавиши - аппарат событий.

В том же объекте OPTION можно использовать событие onChange, что делает

необязательным нажатие кнопок типа submit. В этом случае достаточно будет

просто выбрать альтернативу и перейти к новой странице сразу после выбора.

Стеки гипертекстовых ссылок

В язык теперь введен новый тип объектов Area - это элемент контейнера

MAP, который определяет client-site imagemap. Собственно, главное достоинство

такого объекта состоит в том, что гипертекстовые ссылки, определенные в

AREA стали доступны для переопределения. Они появляются в массиве обычных

ссылок страницы и можно как получить значение URL, так и переопределить

его. К объекту AREA нельзя обратиться по имени, а можно использовать только

индекс массива гипертекстовых ссылок документа.

В контексте стека гипертекстовых ссылок интересно рассмотреть еще одну

возможность JavaScript, связанную с переходом по гипертекстовой ссылке

вообще. В обычном случае параметр HREF контейнера A должен иметь какое-нибудь

значение. Если, например, по событию onClick необходимо открыть новое окно

и в старом сохранить отображенный документ, то его URL следует указывать

в качестве значения HREF. В противном случае, в старое окно будет загружена

пустая страница, если HREF = " ".

В новой версии JavaScript введена функция void, точнее, тип void, который

означает отсутствие какого либо значения. Если необходимо выполнить некоторые

действия при выборе гипертекстовой ссылки, но при этом не перегружать текущие

страницы, то в параметре HREF можно указать конструкцию:

<A HREF = "javascript: void(0)">kuku</A>

Таким приемом часто пользуются при программировании событий, связанных

с проходом манипулятора мышь через поле гипертекстовой ссылки.

Фреймы и окна

При работе с фреймами и окнами в предыдущих версиях JavaScript постоянно

приходилось отслеживать последовательность открытия окон и фреймов для

того, чтобы аккуратно их потом закрывать. На некоторых неточностях работы

с окнами были основаны так называемые mail-bombs. Суть этих "подарков"

заключалась в том, что если пользователь по почте принимает документ, состоящий

только из одной команды:

window.close(),

то система, не спрашивая пользователя, закрывала текущее окно, а в этот

момент таким окном является окно электронной почты. Теперь, перед тем как

что-либо закрыть, система будет спрашивать разрешения. Правда, если в момент

получения команды на закрытие окна на экране имеется только одно окно навигатора,

то система его закроет без каких-либо комментариев.

Работа с окнами и фреймами на этом не исчерпывается. В систему введено

новое свойство opener, которое определено для текущего окна или фрейма,

а методы blur и focus распространены на работу с окнами. Свойство opener

определяет окно документа, вызвавшего открытие окна текущего документа.

Свойство определено для любого окна и фрейма. Если необходимо выполнить

некоторые функции по отношению к окну, открывшему данное окно, то можно

использовать выражение типа:

window.opener.[method]

Например, если требуется закрыть окно - предшественник, то можно просто

выполнить close:

window.opener.close()

Точно таким же способом можно изменить содержание этого окна при помощи

методов write или writeln.

Можно менять и другие свойства объектов в окне - предшественнике. Следующий

пример взят из дополнений к спецификации JavaScript:

window.opener.document.bgColor = СcyanТ

В данном случае для окна - предшественника определен светло-голубой

цвет в качестве цвета фона.

Но самым замечательным является то, что предшественника можно менять.

Это значит, что автор получает возможность открывать и закрывать окна не

в строго иерархической последовательности, а в произвольном порядке. Управление

многооконным интерфейсом в этом случае становится более гибким:

window.opener = new\_window

window.opener = null

Первый пример переназначает окно-предшественник, в то время как второй

вообще защищает предшественника от каких-либо действий.

Все что было сказано об окнах распространяется и на фреймы, которые

являются просто частным случаем окна. Фрейм часто может быть порожден путем

разбиения другого фрейма на части. Если при этом потребуется обратиться

к окну фрейма - предшественника, то свойство opener в этом случае незаменимо.

Кроме обращения к различным свойствам окон и фреймов было расширено

действие методов blur и focus с фреймов до окон. Теперь не только фрейм,

но и окно может быть сделано текущим с использованием метода focus или,

наоборот, переведено в фон при помощи метода blur. В ряде случаев, например,

при порождении нескольких страниц обращение к этим функциям бывает довольно

полезным.

Наследование кода скриптов различными страницами

У контейнера SCRIPT появился атрибут SRC. Это дает возможность авторам

страниц строить своеобразную библиотеку функций, к которым можно обращаться

из любой страницы, имеющей ссылку на такую библиотеку. При этом вовсе не

обязательно размещать саму библиотеку на том же сервере, где размещены

и гипертекстовые страницы Web. Можно использовать и чужие функции на языке

JavaScript.

В атрибуте SRC применяется обычный URL. Внутри файла скриптов не используются

таги SCRIPT - это обычный файл с определениями функций и переменных.

Другой возможностью работы с функциями стало введение нового объекта

Function, порождаемого конструктором Function:

new\_Function = new Function

(arg1, arg2, .., argn, function\_body)

Главное отличие от обычного декларирования функции заключается в том,

что в данном случае порождена переменная new\_Function, с которой можно

работать, как с любым другим объектом. При обычном определении функции

такой переменной не порождается.

Как любой объект, Function имеет свои свойства, но не имеет методов.

В качестве свойств функции выступают аргументы и возможность назначения

новых свойств через prototype.

При обращении к объектам страницы-родителя можно использовать свойство

opener окон и фреймов, что позволяет компенсировать отсутствие наследования

и аппарат глобальных переменных в JavaScript.

Java, JavaScript и модули расширения

В новой версии языка HTML есть возможность организовать взаимодействие

между Java апплетами и JavaScript-скриптами. Достигается это за счет использования

атрибута MAYSCRIPT в контейнере APPLET. Собственно, в JavaScript определен

объект типа APPLET, к которому можно обращаться либо по имени, либо по

индексу в массиве апплетов. У этого объекта имеется только одно свойство

- имя. Какие-либо другие свойства или методы для данного типа объектов

не определены.

Кроме апплетов JavaScript позволяет работать и с Plug-ins, представляющие

собой массив соответствующего типа, для которого определен ряд свойств.

Используя эти свойства, можно определить установленные plig-ins и их соответствия

MIME-типам. Назначить plug-ins или манипулировать ими нельзя.

Расширение возможностей использования графических объектов

К расширению возможностей следует отнести:

обтекание графики текстом;

функция client-site imagemap.

Обтекание достигается за счет использования атрибута ALIGN элемента

IMG. ALIGN может принимать значения: TOP, MIDDLE, BOTTOM, LEFT, RIGHT.

Поскольку рисунок, встраиваемый по IMG, рассматривается программами просмотра

как один большой символ, атрибут ALIGN определяет, где относительно других

символов текста в строке будет располагаться этот рисунок. Если задано

значение TOP, то картинка отображается как подстрочная - остальные символы

строки выравниваются по верхнему краю рисунка, при значении MIDDLE картинка

располагается как обычный символ и текст форматируется по центру. Если

задано значение BOTTOM, рисунок отображается как надстрочная буква, текст

выравнивается по его нижнему краю. Значения LEFT и RIGHT этого атрибута

используются для размещения картинки в левой или правой части страницы,

что приводит к обтеканию картинки текстом справа или слева соответственно.

Client-site imagemap - это способ избавиться от многообразия реализаций

стека графических гипертекстовых ссылок, которые применяются различными

HTTP - серверами. Способ организации client-site imagemap состоит из двух

частей: определения image как client-site imagemap и определения фрагментов

image как гипертекстовых ссылок. Приведем пример:

Пример 9.

<HTML>

<!-

Author: HTMLed User

Date: April 1, 1997

->

<HEAD>

<TITLE>Graphics</title>

</HEAD>

<BODY BGCOLOR = #FFbcbc>

<center>

<h4><img src = html.gif>Язык ги-

пертекстовой разметки HTML</h4>

<hr>

<h1>Графика в HTML</h1>

<hr>

<table border=2>

<tr><td><a href = graph1.htm>

Обои</a>

<td><a href = #img>IMG</a>

<td><a href = move.htm> GIF89a</a>

<td><a href = #map>client-site

maps</a>

</table>

<hr>

</center>

<a name = map>

<h2>Стеки графических ссылок</h2>

<MAP name = "mymap">

<AREA SHAPE = "RECT" COORDS =

"6,0,43,34" HREF = #html1>

<AREA SHAPE = "RECT" COORDS =

"305,0,342,34" HREF = #img>

<AREA SHAPE = "RECT" COORDS =

"80,0,120,34" HREF = #html1>

<AREA SHAPE = "RECT" COORDS =

"156,0,194,34" HREF = #img>

</MAP>

<p>

<img src = image.gif USEMAP = "#mymap"><br>

<hr>

<a name = img>

<h2>Обычный IMG</h2>

Обычно графика вставлялась в текст

документа как символ.<br>

<img src=htpdicon.gif><br>

Поэтому приходилось вставлять ее

между строк текста

<p>

<img src = htpdicon.gif align =

left hspace = 10>В HTML 2.0

Netscape Extensions появилась

возможность обтекания графики

текстом при применении атрибута

align:<br>

<img src = htpdicon.gif align =

left>

<hr>

<a name = html1>

Пример перехода по графической

ссылке.

<hr>

</BODY>

</HTML>

В примере 9 графический образ, определенный в контейнере <IMG SRC

=... USEMAP = ...>, является объектом, который разбит на прямоугольники.

С каждым прямоугольником связана гипертекстовая ссылка. Эти ссылки определены

в контейнере <MAP ...>... </MAP>, а связь между контейнерами

IMG и MAP осуществляется через контейнер MAP.

Расширение возможностей создания таблиц

Для организации таблиц используется элемент TABLE, который является

контейнером для других элементов описания таблицы. Атрибут BORDER определяет

разделяющие линии между графами таблицы, обычно они бывают трехмерные или

простые. Сама форма этих линий задается в файле конфигурации программы

просмотра.

Элементы TH (Table Header) и TD (Table Data) используются в примере

10 не только для описания граф таблицы, но и для объединения граф при помощи

атрибутов ROWSPAN -пропуск строки и COLSPAN - пропуск столбца. Цифра в

этих атрибутах определяет количество последовательно расположенных граф

таблицы, объединенных в одну.

Пример 10.

<HTML>

<HEAD>

<title>Versions</title>

</HEAD>

<BODY BGCOLOR = #FFbcbc>

<center>

<hr>

<h1>Версии HTML</h1>

<hr>

<table border = 2>

<tr><TD colspan = 3

align=center>HTML 1.0

<tr><td colspan = 3 align =

center> HTML 2.0

<tr><td>HTML+<TD>Netscape

Extensions <td> Microsoft

Extensions

<tr><td align = center>HTML

3.0<BR> (Project) <br>Arena<td align = center>

Navigator 2.0 <td align =

center> Explorer

<tr><td colspan = 3 align =

center>????? HTML3.0 ?????

</table>

</center>

</BODY>

</HTML>

И так, приведенные расширения стандарта языка HTML не описывают всего многообразия возможностей, которые доступны из большинства программ просмотра. Однако, это наиболее распространенные способы форматирования страниц в WWW. Язык

HTML продолжает развиваться, а его нынешний стандарт пока еще не только

далек от идеала, но даже не приобрел какой-либо законченной формы.

### 3. Публикация сайта в Интернет

Мы готовы опубликовать свой сайт в Интернет. Что для этого нужно?

### Домен и хостинг

Во-первых, доменное имя. Доменное имя или домен – символическое имя, под которым Ваш сайт будет известен в сети. Получить его можно либо, купив у регистратора, либо бесплатно, но в виде поддомена от одного из бесплатных хостингов.  
  
Во-вторых, хостинг. Хостинг– услуга размещения сайта на серверах, имеющих выход в Интернет. В настоящее время компаний, предоставляющих эту услугу огромное количество. Остается только выбрать подходящую. К критериям выбора относятся:

* Цена - есть как платные, так и бесплатные хостинги.
* Объем предоставляемого трафика и дискового пространства
* Поддержка тех или иных средств (PHP, MySQL, возможность развертывания CMS, наличие дополнительных скриптов и т.д.)

Чаще всего, компании, предоставляющие хостинг, являются и регистраторами доменов, и наоборот. Так что приобрести эти 2 услуги труда не составляет. Оплата для большинства компаний возможна через электронные платежные системы типа Webmoney, Яндекс.Деньги и другие.

### Размещение сайта на сервере

Предположим, домен и хостинг мы приобрели. Что дальше? Нам необходимо разместить все файлы нашего сайта на сервере у хостера. Как правило, нам предоставляется несколько вариантов для этого.  
  
Обычно это загрузка содержимого по *HTTP-протоколу*, с использованием панели управления хостера или по *FTP с помощью ftp-клиента*.  
  
Второй вариант намного быстрее и проще. Одним из лучших ftp-клиентов является встроенный в [Total Commander](http://www.internet-technologies.ru/?url=http%3A%2F%2Fwincmd.ru).  
  
Как только устанавливается связь с сервером ftp (хостер обязательно сообщит адрес, логин и пароль входа), пространство сервера станет доступным в виде логического устройства (как и обычные диски Вашего компьютера) на одной из двух панелей. Запускаем процесс копирования, ждем его окончания.  
  
Все, наш сайт опубликован в Интернете. Можете набрать в адресной строке браузера его имя и полюбоваться результатом.

### 7. Простое решение по созданию сайта

Все приведенные выше инструкции, о том, как создать сайт, достаточно подробные и понятные многим пользователям, однако, что делать тем, кто не понимает или не имеет времени и желания разбираться в этом? Именно для таких пользователей создан специальный сервис, позволяющий значительно упростить данную задачу и создать сайт всего за два клика. Этот сервис называется [**SetUp**](http://www.internet-technologies.ru/setup) ([подробная инструкция](http://www.internet-technologies.ru/how-to-create-a-website-with-setup.html)). Все, что нужно - зайти на указанный сайт, зарегистрироваться и нажать "*создать сайт*", выбрать дизайн из семи тысяч вариантов и указать *название-сайта.setup.ru* и далее - "*создать*". Осталось только наполнить сайт информацией и после этого можно принимать поздравления - у вас получилось создать свой сайт **бесплатно и быстро**! Все очень просто и доступно - вы можете убедитесь в этом сами!  
  
Другим подобным конструктором является [**Wix**](http://www.internet-technologies.ru/wix) ([пошаговая инструкция](http://www.internet-technologies.ru/how-to-create-a-website-with-wix.html)). В использовании он совсем не сложнее Setup'а, но в свою очередь обладает целым рядом конкурентных преимуществ. Рекомендуем вам попробовать оба сервиса и выбрать подходящий под ваши цели.  
  
Хотим выразить Вам благодарность, что прочитали (а еще лучше, если попробовали выполнить наши рекомендации) нашу инструкцию до конца и надеемся, что она была для вас полезной. Мы не преследовали цель одной статьей научить Вас делать крутые и навороченные сайты - это просто невозможно, но хотели осветить данный процесс, чтобы в случае каких-то неясностей вы понимали к какому из разделов это относится и могли самостоятельно найти нужную информацию. В этом вам так же должны помочь указанные нами ссылки.  
  
Мы рассчитываем, что у вас возникнут вопросы при создании своего первого сайта, а может и не первого, поэтому милости просим на наш замечательный [**форум вебмастеров**](http://3wforums.ru/).  
  
После того как Вы создали свой первый сайт и немного его раскрутили, Вам стоит определиться с тем, каким образом будет проводится монетизация сайта. Предлагаем вам ознакомиться с наиболее эффективными [способами заработка на сайте](http://www.internet-technologies.ru/how-to-make-money-on-site.html).

***Библиографический список:***

***1.*** http://www.delajsam.ru/create/

***2.*** http://www.delajsam.ru/freesite/

***3.*** «Как сделать сайт в оффлайне от и до? Реально.» Олег Титов.

***4.*** Язык HTML, JAVASCRIPT и WWW. Ирина Кузина.

***5.*** http://www.internet-technologies.ru/how-to-create-a-website.html#devTemplateSite

***6.*** http://www.internet-technologies.ru/how-to-create-a-website.html#publicSite

***7.*** http://www.internet-technologies.ru/how-to-create-a-website.html#publicSite