ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

Государственное образовательное учреждение

высшего профессионального образования

"Глазовский государственный педагогический институт

имени В.Г. Короленко"

Факультет социальных и информационных технологий

Кафедра информатики

Выпускная квалификационная работа

**Создание динамических сайтов средствами Dreamweaver**

Выполнила

студентка группы 951 О.А. Богданова

Руководитель работы,

старший преподаватель

кафедры информатики Д.А. Горбушин

Глазов 2007

Содержание

Введение

Глава I. Интерфейс среды Dreamweaver. Обзор популярных интернет-технологий создания динамических сайтов

1.1 Интерфейс среды Dreamweaver

1.2 PHP

1.3 ASP

1.4 XML

Выводы по Главе I

Глава II. Методика выбора оптимального средства разработки

2.1 Выделение критериев сравнения популярных интернет-технологий

Выводы по Главе II

Глава III. Разработка динамических веб-страниц на базе различных технологий средствами Dreamweaver

3.1 Записная книжка на базе РНР

3.2 Электронный прайс-лист на базе XML

3.3 Тест на базе ASP

Выводы по Главе III

Заключение

Библиографический список

## Введение

World Wide Web - глобальная компьютерная сеть - на сегодняшний день содержит миллионы сайтов, на которых размещена всевозможная информация. Люди получают доступ к этой информации посредством использования технологии Internet. Для навигации в WWW используются специальные программы - веб-браузеры, которые существенно облегчают путешествие по бескрайним просторам WWW.Вся информация в веб-браузере отображается в виде веб-страниц, которые являются основным элементом байтов WWW.

Можно выделить следующие классификации веб-документов: по возможности изменения содержания и структуры - статические (трудоемкое и неудобное обновление) и динамические (более простое и быстрое обновление); по наличию обратной связи - пассивные (пользователь имеет возможность лишь просматривать информацию) и активные (пользователь имеет возможность обмениваться данными с сервером, участвовать в интерактивном диалоге). В данной работе интерес для нас представляют активные динамические веб-документы.

Создать качественную веб-страницу непросто, для этого потребуются не только навыки дизайнера, но и опыт программирования. Для облегчения этих задач существуют специальные программы. Наиболее известным и эффективным инструментом создания веб-сайтов является продукт компании Macromedia - Dreamweaver MX. Это мощная среда, предназначенная для проектирования, разработки и администрирования профессиональных веб-сайтов и приложений. Dreamweaver MX позволяет создавать сайты на базе различных веб-технологий: HTML, XHTML, XML, ColdFusion, ASP, ASP.net, JSP и PHP.

**Цель** нашей работы - исследование возможностей применения Dreamweaver для реализации веб-технологий: ASP, PHP и XML при разработке динамических сайтов.

**Гипотеза** основана на предположении, что использование единой интегрированной среды разработки, в которой возможна реализация различных веб-технологий, оптимизирует процесс построения динамических сайтов.

**Объект** работы - технологии разработки динамических сайтов.

**Предмет** работы - методы построения динамических сайтов средствами Dreamweaver.

Для построения сайта, написания сценариев для него можно использовать текстовые редакторы, вручную набирая код, специальные приложения для реализации отдельных технологий (например, Rapid PHP Editor), программы создания веб-страниц (например, FrontPage). Но только среда Dreamweaver позволяет создавать динамические веб-документы, соответствующие международным стандартам, интегрировать программный код в HTML-код, использовать при создании одного документа сразу несколько технологий. Поэтому работа, посвященная разработке динамических сайтов средствами Dreamweaver, является **актуальной**.

**Новизна** данной работы заключается в том, что была рассмотрена технология создания динамических сайтов на базе различных технологий в рамках одного инструмента.

Цель, гипотеза, объект и предмет определяют следующие **задачи**:

обзор и изучение литературы по исследуемой проблеме;

ознакомление с технологиями ASP, PHP и XML в рамках Dreamweaver;

выделение критериев сравнения технологий;

подготовка методики анализа прикладных задач с целью выбора оптимальной технологии для их реализации;

применение разработанной методики при решении конкретных прикладных задач.

Наша работа состоит из введения, трех глав, заключения и списка литературы.

Научная и теоретическая значимость исследования. В работе были изложены теоретические основы сайтостроения средствами разных технологий, реализуемых в рамках одного инструмента.

Практическая значимость исследования. Разработанная методика позволит легко и точно выбрать оптимальную технологию при создании сайта.

На защиту выносятся следующие **положения**:

использование различных технологий в рамках одного инструмента является рациональным и эффективным;

даже в рамках одного инструмента актуален выбор технологии, дающей оптимальный результат.

## Глава I. Интерфейс среды Dreamweaver. Обзор популярных интернет-технологий создания динамических сайтов

## 1.1 Интерфейс среды Dreamweaver

Мощная профессиональная среда Dreamweaver обладает всеми необходимыми средствами для генерации страниц HTML любой сложности и масштаба. Она обеспечивает режим визуального проектирования (WYSIWYG или What You See Is What You Get - "что ты видишь, то ты и получишь"), отличается очень чистой работой с исходным текстом веб-документов, обладает встроенными средствами поддержки больших сетевых проектов. Ни в одном из существующих в наше время редакторов концепция WYSIWYG не реализована полностью. [12] Программа Dreamweaver подошла к декларируемому идеалу ближе конкурентов. Прямая работа с кодами не исключена полностью, но сведена к разумному минимуму. Программа не только обладает мощным арсеналом средств визуального проектирования, но и способна отображать веб-страницы почти как специализированные программы просмотра: Microsoft Internet Explorer или Netscape Navigator. [1]

Macromedia Dreamweaver MX - одна из самых мощных программ, поддерживающих все современные стандарты Интернета и невероятно облегчающая выполнение даже самых сложных задач. Кроме того, она содержит в своем составе развитую систему подсказки и интерактивных уроков, позволяющих начинающему пользователю быстро приступить к работе. [17]

Интерфейс программы по сравнению с предыдущими версиями претерпел изменения. Новый стиль заметно облегчает работу с программой. Если в предыдущих версиях программы приходилось постоянно переключаться между окнами, то теперь можно одновременно работать и с кодом, и с конечным видом сайта, что существенно облегчает работу. [13]

При загрузке Dreamweaver появляется стартовое окно (рис.1), позволяющее выбрать тип нового создаваемого документа (HTML, ColdFusion, PHP и др.), либо создать документ по готовым образцам (CSS Style Sheets, Framesets и др.), а также открыть недавно использовавшиеся документы. Кроме этого имеются ссылки на интернет-ресурсы: сайт Dreamweaver MX Exchange, обзор программы Dreamweaver MX и справочная информация по Dreamweaver MX.

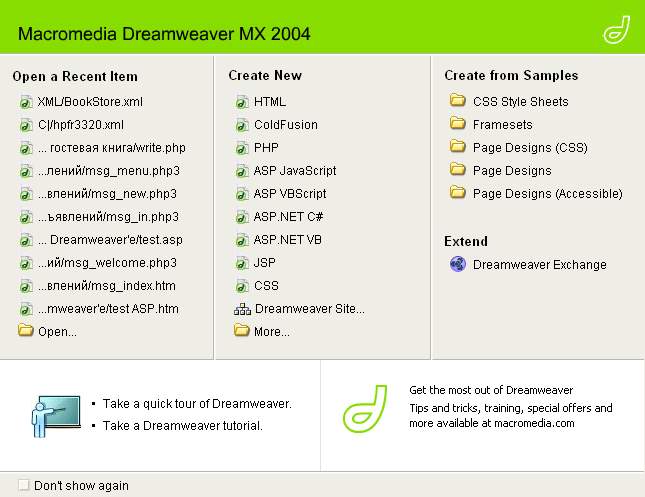


Рис.1. Стартовое окно Dreamweaver MX

Рабочая область Dreamweaver содержит следующие окна и панели:

окно документа (рис.2), позволяющее просматривать и редактировать код документа (Code), просматривать внешний вид будущего документа (Design) или одновременно код и внешний вид (Split). Также имеется возможность просмотра документа в интернет-браузере, проверки ошибок и т.д.;

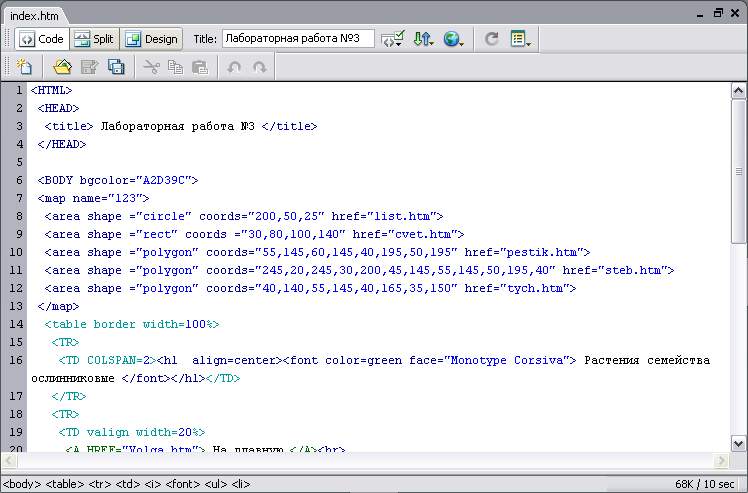


Рис.2. Окно документа

Стандартное раскрывающееся меню (File, Edit, View, Insert, Modify, Text, Commands, Site, Window, Help) (рис.3);



Рис.3. Стандартное раскрывающееся меню

Панель Properties (Свойства) (рис.4), позволяющая менять и добавлять свойства выделенного фрагмента кода;

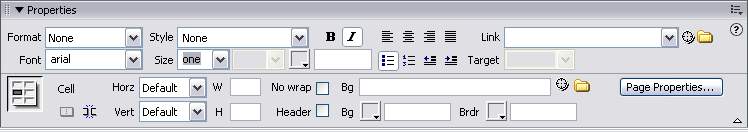


Рис.4. Панель Properties

Панель Insert (Вставка) (рис.5), включающая следующие вкладки: Common (Основные) - вставка гиперссылок, таблиц, рисунков, комментариев и т.д., а также Tag Chooser (Выбор тега); Layout (Разметка) - использование ячеек и таблиц для разметки документа перед добавлением содержимого; Forms (Формы) - добавление различных видов форм; Text (Текст) - форматирование текста и Font Tag Editor (Редактор тегов текста); HTML - добавление горизонтальной линии, элементов таблицы, фреймов, свойств head, скриптов; Application (Приложения) - работа с информационными структурами; Flash elements (элементы Flash) - добавление Flash-роликов, Favorites (Избранное) - возможность добавления на отдельную вкладку наиболее часто используемых объектов.



Рис.5. Панель Insert

Группа панелей: Design (содержит стили CSS), Code (содержит справку по тегам, объектам и функциям различных технологий), Application (содержит информацию о базах данных, компонентах и т.д.), Files (Диспетчер файлов) (Рис.6).

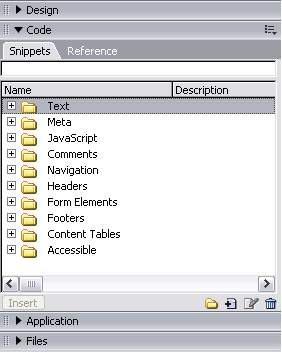


Рис.6. Группа панелей Design, Code, Application, Files

## 1.2 PHP

1. Общие сведения. PHP - это язык обработки гипертекста (HTML), используемый на стороне сервера (server side scripting language), конструкции которого вставляются в HTML-текст. Его цель - позволить максимально быстро создавать динамически генерируемые web-страницы. PHP является открытым программным продуктом, что означает его бесплатность и возможность создавать свои собственные расширения языка. [10]

PHP является самым молодым, перспективным и быстроразвивающимся из языков программирования для Интернет, доля его использования по сравнению с другими языками быстро растет. [3]

PHP позволяет отделить HTML-текст от выполняемой части, за счет чего можно добиться значительного снижения затрат времени на разработку проекта. Во многих случаях удается отделить программную часть проекта от разработки страниц на HTML, что облегчает работу и дизайнеру, и программисту. [18]

2. Основные характеристики.

совместимость с различными серверами и платформами;

поддержка технологий COM, XML, Java, CORBA, WDDX, Macromedia Flash;

поддержка взаимодействия с различными сервисами посредством соответствующих протоколов (LDAP, SNMP, IMAP, NNTP, POP3, HTTP и др.);

развитая функциональность для работы с сетевыми соединениями;

наличие функций для работы с текстовыми данными любых форматов, включая XML;

поддержка свыше 20 баз данных, развитая функциональность для работы с ними;

возможность создания полноценных объектно-ориентированных приложений;

сравнительно простой синтаксис и удобство в практическом использовании;

бесплатность;

открытость кода, благодаря которой можно создавать собственные расширения языка. [25]

3. Синтаксис. PHP-скрипты встраиваются в HTML-код с помощью открывающего тега <? php и закрывающего тега? >. Такого вида специальные теги позволяют переключаться между режимами HTML и PHP. Программа на РНР - это набор команд (инструкций). Обработчику программы необходимо как-то отличать одну команду от другой. Для этого используются специальные символы - разделители. В PHP инструкции разделяются так же, как и в Cи или Perl, - каждое выражение заканчивается точкой с запятой.

Закрывающий тег? > также подразумевает конец инструкции, поэтому перед ним точку с запятой не ставят. [5]

Часто при написании программ возникает необходимость делать какие-либо комментарии к коду, которые никак не влияют на сам код, а только поясняют его. PHP поддерживает несколько видов комментариев: в стиле Cи, C++ и оболочки Unix. Символы // и # обозначают начало однострочных комментариев, /\* и \*/ - соответственно начало и конец многострочных комментариев. [11]

4. Переменные и константы. Важным элементом каждого языка являются переменные, константы и операторы, применяемые к этим переменным и константам.

Переменные. Переменная в PHP обозначается знаком доллара, за которым следует ее имя (например: $my\_var). Имя переменной чувствительно к регистру, т.е. переменные $my\_var и $My\_var различны.

Имена переменных соответствуют тем же правилам, что и остальные наименования в PHP: правильное имя переменной должно начинаться с буквы или символа подчеркивания с последующими в любом количестве буквами, цифрами или символами подчеркивания.

В PHP 3 переменные всегда присваивались по значению. То есть когда вы присваиваете выражение переменной, все значения оригинального выражения копируется в эту переменную. PHP 4, кроме этого, предлагает еще один способ присвоения значений переменным: присвоение по ссылке. Для того чтобы присвоить значение переменной по ссылке, это значение должно иметь имя, т.е. оно должно быть представлено какой-либо переменной. Чтобы указать, что значение одной переменной присваивается другой переменной по ссылке, нужно перед именем первой переменной поставить знак амперсанд &. [7]

Константы. Для хранения постоянных величин, т.е. таких величин, значение которых не меняется в ходе выполнения скрипта, используются константы. Такими величинами могут быть математические константы, пароли, пути к файлам и т.п. Основное отличие константы от переменной состоит в том, что ей нельзя присвоить значение больше одного раза и ее значение нельзя аннулировать после ее объявления. Кроме того, у константы нет приставки в виде знака доллара и ее нельзя определить простым присваиванием значения. Для определения константы существует специальная функция define (). Ее синтаксис таков:

define ("Имя\_константы", "Значение\_константы", [Нечувствительность\_к\_регистру])

По умолчанию имена констант чувствительны к регистру. Существует соглашение, по которому имена констант всегда пишутся в верхнем регистре. Получить значение константы можно, указав ее имя. В отличие от переменных, не нужно предварять имя константы символом $. Кроме того, для получения значения константы можно использовать функцию constant () с именем константы в качестве параметра.

Кроме переменных, объявляемых пользователем, в PHP существует ряд констант, определяемых самим интерпретатором. Например, константа FILE хранит имя файла программы (и путь к нему), которая выполняется в данный момент, FUNCTION содержит имя функции, CLASS - имя класса, PHP\_VERSION - версия интерпретатора PHP. [8]

5. Управляющие конструкции. Управляющие конструкции в РНР представлены условными операторами, циклами, операторами передачи управления, операторами включения.

Условные операторы:

if. Это один из самых важных операторов многих языков, включая PHP. Он позволяет выполнять фрагменты кода в зависимости от условия. Структура оператора: if (выражение) блок\_выполнения;

else. Оператор else расширяет if на случай, если проверяемое в if выражение является неверным, и позволяет выполнить какие-либо действия при таких условиях. Структура оператора: if (выражение) блок\_выполнения else блок\_выполнения1;

elseif. Еще один способ расширения условного оператора if. elseif - это комбинация else и if. Как и else, он расширяет if для выполнения различных действий в том случае, если условие, проверяемое в if, неверно. Но в отличие от else, альтернативные действия будут выполнены, только если elseif-условие является верным. Структура оператора:

if (выражение) блок\_выполнения elseif (выражение1) блок\_выполнения1 ...

else блок\_выполнения N

switch. Еще одна конструкция, позволяющая проверять условие и выполнять в зависимости от этого различные действия. В зависимости от того, какое значение имеет переменная, switch переключается между различными блоками действия. switch очень похож на оператор if... elseif... else или набор операторов if. Структура оператора:

switch (выражение или переменная) { case значение1: блок\_действий1 break;

case значение2: блок\_действий2 break;

...

default:

блок\_действий\_по\_умолчанию }

Циклы:

while. Простой цикл, предписывающий PHP выполнять команды блока\_выполнения до тех пор, пока выражение вычисляется как True. Значение выражения проверяется каждый раз в начале цикла, так что, даже если его значение изменилось в процессе выполнения блока\_выполнения, цикл не будет остановлен до конца итерации (т.е. пока все команды блока\_выполнения не будут исполнены). Структура цикла: while (выражение) { блок\_выполнения };

do... while. Очень похож на цикл while, с той лишь разницей, что истинность выражения проверяется в конце цикла, а не в начале. Благодаря этому блок\_выполнения цикл do... while гарантированно выполняется хотя бы один раз. Структура цикла: do {блок\_выполнения} while (выражение);

for. Структура цикла: for (выражение1; выражение2; выражение3) {блок\_выполнения}

foreach. Предназначен исключительно для работы с массивами. Структура: foreach ($array as $value) {блок\_выполнения}.

Операторы передачи управления:

break. Заканчивает выполнение текущего цикла, может использоваться с числовым аргументом, который говорит, работу скольких управляющих структур, содержащих его, нужно завершить;

continue. Позволяет пропустить дальнейшие инструкции из блока\_выполнения любого цикла и продолжить выполнение с нового круга.

Операторы включения:

include. Позволяет включать код, содержащийся в указанном файле, и выполнять его столько раз, сколько программа встречает этот оператор. Включение может производиться любым из перечисленных способов: include 'имя\_файла'; include $file\_name; include ("имя\_файла");

require. Действует примерно так же, как и include. Основное отличие require и include заключается в том, как они реагируют на возникновение ошибки: include выдает предупреждение, и работа скрипта продолжается; ошибка в require вызывает фатальную ошибку работы скрипта и прекращает его выполнение. [23]

6. Обработка запросов. Формы. Любой запрос клиента к серверу должен начинаться с указания метода. Метод сообщает о цели запроса клиента. Протокол HTTP поддерживает достаточно много методов, но реально используются только три: POST, GET и HEAD. Метод GET позволяет получить любые данные, идентифицированные с помощью URL в запросе ресурса. Если URL указывает на программу, то возвращается результат работы программы, а не ее текст (если, конечно, текст не есть результат ее работы). Дополнительная информация, необходимая для обработки запроса, встраивается в сам запрос (в строку статуса). При использовании метода GET в поле тела ресурса возвращается собственно затребованная информация (текст HTML-документа, например).

Метод HEAD аналогичен методу GET, только не возвращает тело ресурса и не имеет условного аналога. Метод HEAD используют для получения информации о ресурсе. Это может пригодиться, например, при решении задачи тестирования гипертекстовых ссылок.

Метод POST разработан для передачи на сервер такой информации, как аннотации ресурсов, новостные и почтовые сообщения, данные для добавления в базу данных, т.е. для передачи информации большого объема и достаточно важной. В отличие от методов GET и HEAD, в POST передается тело ресурса, которое и является информацией, получаемой из полей форм или других источников ввода. [4]

Формы предназначены для того, чтобы получать от пользователя информацию. В них можно вводить текст или выбирать подходящие варианты из списка. Данные, записанные в форму, отправляются для обработки специальной программе (например, скрипту на PHP) на сервере. В зависимости от введенных пользователем данных эта программа может формировать различные web-страницы, отправлять запросы к базе данных, запускать различные приложения и т.п. [6]

Для обращения к переменным, переданным с помощью HTTP-запросов существует специальный массив - $\_REQUEST. Он содержит данные, переданные методами POST и GET, а также с помощью HTTP cookies. Это суперглобальный ассоциативный массив, т.е. его значения можно получить в любом месте программы, используя в качестве ключа имя соответствующей переменной (элемента формы). [26]

Приведем пример использования массивов $\_POST и $\_GET. Допустим, нам нужно обработать форму, содержащую элементы ввода с именами first\_name, last\_name, kurs. Данные были переданы методом POST, и данные, переданные другими методами, мы обрабатывать не хотим. Это можно сделать следующим образом:

<? php $str = "Здравствуйте, ". $\_POST ["first\_name"]." ". $\_POST ["last\_name"]. "! <br>";

$str. = "Вы выбрали для изучения курс по ".

$\_POST ["kurs"];

echo $str;

? >

Тогда на экране браузера, если мы ввели имя "Иван", фамилию "Петров" и выбрали среди всех курсов курс по PHP, увидим сообщение:

Здравствуйте, Иван Петров! Вы выбрали для изучения курс по PHP

7. Функции. В программировании, как и в математике, функция есть отображение множества ее аргументов на множество ее значений. То есть функция для каждого набора значений аргумента возвращает какие-то значения, являющиеся результатом ее работы.

У каждой функции может быть, как мы уже говорили, список аргументов. С помощью этих аргументов в функцию передается различная информация (например, значение числа, факториал которого надо подсчитать). Каждый аргумент представляет собой переменную или константу. [10]

С помощью аргументов данные в функцию можно передавать тремя различными способами. Это передача аргументов по значению (используется по умолчанию), по ссылке и задание значения аргументов по умолчанию.

Когда аргумент передается в функцию по значению, изменение значения аргумента внутри функции не влияет на его значение вне функции. Чтобы позволить функции изменять ее аргументы, их нужно передавать по ссылке. Для этого в определении функции перед именем аргумента следует написать знак амперсанд "&". [18]

В РНР существует несколько встроенных функций: echo (), print (), date (), include. Все, кроме date (), являются языковыми конструкциями. Они входят в ядро PHP и не требуют никаких дополнительных настроек и модулей. Функция date () тоже входит в состав ядра PHP и не требует настроек. Но есть и функции, для работы с которыми нужно установить различные библиотеки и подключить соответствующий модуль. Например, для использования функций работы с базой данных MySql следует скомпилировать PHP с поддержкой этого расширения. В последнее время наиболее распространенные расширения и соответственно их функции изначально включают в состав PHP так, чтобы с ними можно работать без каких бы то ни было дополнительных настроек интерпретатора. [10]

8. Классы и объекты. Объект - это структурированная переменная, содержащая всю информацию о некотором физическом предмете или реализуемом в программе понятии, класс - это описание таких объектов и действий, которые можно с ними выполнять. [4]

Класс определяется с помощью следующего синтаксиса:

class Имя\_класса{ var $имя\_свойства;

/\*список свойств\*/ function имя\_метода () { /\* определение метода \*/ } /\*список методов\*/ }

Имя класса должно удовлетворять правилам именования объектов в языке PHP, но есть ряд имен, которые зарезервированы разработчиками для своих целей. В первую очередь это имена, начинающиеся с символа подчеркивания "\_". Для создания классов и функций нельзя использовать такие имена. Кроме того, зарезервировано имя stdClass, поскольку оно используется внутри движка PHP.

Переменная нужного типа создается из класса с помощью оператора new. Создав объект, мы можем применять к нему все методы и получать все свойства, определенные в описании класса. Для этого используют такой синтаксис: $имя\_объекта->название\_свойства или $имя\_объекта->название\_метода (список аргументов). Перед названием свойства или метода знака $ не ставят.

<? php $art = new Articles;

// создаем объект $art echo ($art - >title);

// выводим название объекта $art $another\_art = new Articles;

// создаем объект $another\_art $another\_art->show\_article ();

// вызываем метод для отображения объекта в браузер ? >

Каждый из объектов класса имеет одни и те же свойства и методы. Так, у объекта $art и у объекта $another\_art есть свойства title, description, author и методы Articles (), show\_article (). Но это два разных объекта. Представим себе объект как директорию в файловой системе, а его характеристики - как файлы в этой директории. Очевидно, что в каждой директории могут находиться одинаковые файлы, но тем не менее они считаются различными, поскольку хранятся в разных директориях. Точно так же свойства и методы считаются различными, если они применяются к разным объектам. Чтобы получить нужный файл из директории верхнего уровня, мы пишем полный путь к этому файлу. При работе с классами нужно указывать полное имя функции, которую мы хотим вызвать. Директорией верхнего уровня в PHP будет пространство глобальных переменных, а путь указывается с помощью разделителя - >. Таким образом, имена $art->title и $another\_art->title обозначают две разные переменные.

9. Работа с файлами. В PHP не существует функции, предназначенной именно для создания файлов. Большинство функций работают с уже существующими файлами в файловой системе сервера. Чтобы создать самый обычный файл, нужно воспользоваться функцией, которая открывает локальный или удаленный файл. Называется эта функция fopen (). Она связывает данный файл с потоком управления программы. Причем связывание бывает различным в зависимости от того, что мы хотим делать с этим файлом: читать его, записывать в него данные или делать и то и другое. [18] Синтаксис этой функции такой:

resource fopen (имя\_файла, тип\_доступа [, use\_include\_path])

В результате работы эта функция возвращает указатель (типа ресурс) на открытый ею файл. В качестве параметров этой функции передаются: имя файла, который нужно открыть, тип доступа к файлу (определяется тем, что мы собираемся делать с ним) и, возможно, параметр, определяющий, искать ли указанный файл в include\_path.

Значения, принимаемые параметром тип\_доступа:

r - открывает файл только для чтения; устанавливает указатель позиции в файле на начало файла.

r+ - открывает файл для чтения и записи; устанавливает указатель файла на его начало.

w - открывает файл только для записи; устанавливает указатель файла на его начало и усекает файл до нулевой длины. Если файл не существует, то пытается создать его.

w+ - открывает файл для чтения и записи; устанавливает указатель файла на его начало и усекает файл до нулевой длины. Если файл не существует, то пытается создать его.

a - открывает файл только для записи; устанавливает указатель файла в его конец. Если файл не существует, то пытается создать его.

a+ - открывает файл для чтения и записи; устанавливает указатель файла в его конец. Если файл не существует, то пытается создать его.

x - создает и открывает файл только для записи; помещает указатель файла на его начало. Если файл уже существует, то fopen () возвращает false и генерируется предупреждение. Если файл не существует, то делается попытка создать его. Этот тип доступа поддерживается начиная с версии PHP 4.3.2 и работает только с локальными файлами.

x+ - создает и открывает файл для чтения и записи; помещает указатель файла на его начало. Если файл уже существует, то fopen () возвращает false и генерируется предупреждение. Если файл не существует, то делается попытка создать его. Этот тип доступа поддерживается, начиная с версии PHP 4.3.2, и работает только с локальными файлами.

После выполнения необходимых действий с файлом, будь то чтение или запись данных или что-либо другое, соединение, установленное с этим файлом функцией fopen (), нужно закрыть. Для этого используют функцию fclose (). Синтаксис у нее следующий:

fclose (указатель на файл)

Для того, чтобы записать данные в файл, доступ к которому открыт функцией fopen (), можно использовать функцию fwrite (). Синтаксис у нее следующий:

int fwrite (указатель на файл, строка [, длина])

Эта функция записывает содержимое строки в файл, на который указывает указатель на файл. Если указан дополнительный аргумент длина, то запись заканчивается после того, как записано количество символов, равное значению этого аргумента, или когда будет достигнут конец строки.

Пусть в нашей рабочей директории нет файла my\_file.html. Создадим его и запишем в него строку текста:

<? php $h = fopen ("my\_file.html","w");

$text = "Этот текст запишем в файл";

if (fwrite ($h,$text)) echo "Запись прошла успешно";

else echo "Произошла ошибка при записи данных";

fclose ($h);

? >

В результате работы этого скрипта в браузере мы увидим сообщение о том, что запись прошла успешно, а в файле my\_file.html появится строка "Этот текст запишем в файл". Если бы этот файл существовал до того, как мы выполнили этот скрипт, все находящиеся в нем данные были бы удалены.

Если мы хотим прочитать данные из существующего файла, одной функции fopen () недостаточно. Она лишь возвращает указатель на открытый файл, но не считывает ни одной строки из этого файла. Поэтому для того, чтобы прочитать данные из файла, нужно воспользоваться одной из специальных функций: file, readfile, fread, fgets и т.п.

Функция fread осуществляет чтение данных из файла. Ее можно использовать и для чтения данных из бинарных файлов, не опасаясь их повреждения. Синтаксис fread ():

string fread (указатель на файл, длина)

В результате работы функция fread () возвращает строку со считанной из файла информацией.

С помощью функции fgets () можно считать из файла строку текста. Синтаксис этой функции практически такой же, как и у fread (), за исключением того, что длину считываемой строки указывать необязательно:

string fgets (указатель на файл [, длина])

В результате работы функция fgets () возвращает строку длиной (длина минус 1) байт из файла, на который указывает указатель на файл. Чтение заканчивается, если прочитано (длина минус 1) символов и встретился символ перевода строки или конец файла.

Функция readfile () считывает файл, имя которого передано ей в качестве параметра имя\_файла, и выводит его содержимое на экран.

Синтаксис:

int readfile (имя\_файла [, use\_include\_path])

Функция file () предназначена для считывания информации из файла в переменную типа массив. Синтаксис у нее такой же, как и у функции readfile (), за исключением того, что в результате работы она возвращает массив:

array file (имя\_файла [, use\_include\_path])

Для того, чтобы удалить файл с помощью языка PHP, нужно воспользоваться функцией unlink (). Синтаксис этой функции можно описать следующим образом: bool unlink (имя\_файла) [18]

## 1.3 ASP

1. Общие сведения. Active Server Pages (ASP) - это серверная среда для разработки и выполнения динамических интерактивных веб-приложений. Средства ASP позволяют объединить возможности HTML-страниц, команд сценариев и компонентов COM в интерактивных веб-страницах и мощных веб-приложениях, делают удобным и легким процесс их создания и изменения. [15]

Создание серверных сценариев с помощью ASP - быстрый и простой способ перейти к разработке сложных коммерческих веб-приложений. ASP предлагает универсальное решение задачи хранения в базе данных сведений, которые представлены HTML-формой, настройки веб-узлов с учетом особенностей посетителей и использования разнообразных возможностей обозревателей при работе с HTML-страницами. [21]

В ASP отсутствует ориентация на конкретный язык программирования, поэтому знакомства с любым языком сценариев (VBScript, JScript или PERL) будет достаточно для того, чтобы работать с Active Server Pages. Более того, на страницах ASP допускается использование любого языка сценариев, для которого был установлен COM-совместимый обработчик сценариев. В состав ASP входят обработчики сценариев VBScript и JScript, но имеется дополнительная возможность установки обработчиков для языков PERL, REXX и Python, которые могут быть получены от независимых разработчиков. [19]

Выполнение серверного сценария начинается после поступления от веб-обозревателя запроса на файл ". asp". После этого веб-сервер вызывает ASP для обработки указанного в запросе файла. Обработка этого файла происходит последовательно, от начала и до конца, при этом выполняются все содержащиеся в нем команды сценария, после чего файл отправляется на обозреватель в виде веб-страницы. [22]

Файл Active Server Pages (ASP) представляет собой текстовый файл с расширением ". asp". Этот файл может содержать:

текстовые данные;

теги языка HTML;

серверные сценарии.

2. Основные характеристики.

удобный способ объединение Server-Side Script c HTML;

скриптовый подход (интерпретируемый язык) - т.е. файл с исходным кодом ASP одновременно является его исполняемым файлом, что упрощает процессы разработки и поддержки;

концепция "Session" - переменные для каждого пользовательского соединения, как удачное решение вечной проблемы stateless-протокола HTTP;

возможность организации распределенной архитектуры на основе инфраструктуры COM (Component Object Model), DCOM, COM+. Дополнительные возможности, предоставляемые Microsoft Transaction Server (MTS) - такие, например, как контекст объектов, пул и т.д.;

удобный набор объектов-утилит: Server, Application, Request, Response, Session, ObjectContext$

обработка ошибок;

интеграция XML;

**включения на стороне сервера с атрибутом SRC;**

**шифрование сценариев ASP. [**15]

3. Синтаксис. Основной язык сценариев ASP представляет собой язык, используемый для обработки команд между разделителями <% и%>. По умолчанию основным языком сценариев является VBScript. В качестве основного языка сценариев можно использовать любой язык написания сценариев, для которого имеется обработчик сценариев. Основной язык сценариев может быть установлен для отдельных страниц или для всех страниц ASP-приложения.

В разделители ASP можно поместить любую инструкцию, выражение, процедуру или оператор, применение которых разрешено в используемом основном языке сценариев. [24]

В VBScript поддерживаются комментарии, начинающиеся с апострофа. В отличие от комментариев HTML, они удаляются при обработке сценария и не передаются в обозреватель. В JScript поддерживаются символы комментариев // Эти символы следует использовать на каждой строке комментария.

VBScript не учитывает регистр вводимых данных, JScript учитывает.

4. Переменные и константы. Переменная *-* это именованная область памяти компьютера, содержащая данные, например числа или текстовые строки. Данные, содержащиеся в переменной, называются значением переменной. Переменные предлагают способ хранения, извлечения и манипулирования данными с помощью имен, что помогает понять работу сценария.

Объявление переменной означает уведомление обработчика сценариев о том, что переменная с определенным именем существует.

VBScript не требует объявления переменных, но считается хорошим стилем написания сценариев объявлять все переменные перед их использованием. Чтобы объявить переменную в VBScript, используйте операторы **Dim, Public** или **Private**. Например: <% Dim UserName%>

Можно использовать оператор VBScript **Option Explicit** в файле. asp, чтобы требовать явного объявления переменных операторами **Dim**, **Private**, **Public** и **ReDim**. Оператор **Option Explicit** должен появиться после любой директивы ASP, но перед текстами HTML и командами сценария. Этот оператор влияет на команды ASP, написанные на VBScript; но не оказывает влияния на команды JScript.

В JScript для объявления переменных используется оператор **var**. Например:

<% var UserName%>. Обычно необходимо объявлять переменную в JScript, когда необходимо отличить переменную внутри функции от глобальной переменной, используемой вне функции. В этом случае если не отличать эти две переменные, JScript будет предполагать, что ссылаются исключительно на глобальную переменную. [20]

Константа - это имя, которое используется вместо числа или строки. Некоторые основные компоненты, поставляемые с ASP, например ActiveX Data Objects (ADO), определяют константы, которые можно использовать в сценарии. Компонент может объявить константы в библиотеке типов компонента - файле, содержащем сведения об объектах и типах, поддерживаемых компонентом COM. Объявив библиотеку типов в файле. asp, можно использовать константы в любом сценарии в том же файле. asp. Аналогично можно объявить библиотеку типов в файле Global. asр, чтобы использовать константы в любом файле. asp в приложении. Для объявления библиотеки типов используется тег <METADATA> в файле. asp или файле Global. asр. [19]

5. Инструкции. Подинструкцией в VBScript и других языках написания сценариев понимают синтаксически полную конструкцию, которой описывается один вид действия, объявления или определения. Примером часто используемой инструкции является условная инструкция **If... Then... Else**. [24]

Между различными частями инструкции может располагаться HTML-текст.

<%

Dim dtmHour

dtmHour = Hour (Now ())

If dtmHour < 12 Then

%>

Good Morning!

<% Else%>

Hello!

<% End If%>

Если заданное в сценарии условие выполняется (полдень еще не наступил), веб-сервер отправляет в обозреватель указанный в условии текст HTML ("Good Morning"). В противном случае в обозреватель отправляется строка ("Hello!"), указанная после **Else**. Подобное чередование HTML-текста и команд сценариев часто применяется при работе с инструкцией **If... Then... Else**.

6. Обработка запросов. Формы. Формы HTML, наиболее распространенный метод сбора сведений на основе веб, состоят из группы специальных тегов HTML, представляющих собой элементы пользовательского интерфейса на веб-странице. Текстовые поля, кнопки и флажки являются примерами элементов, позволяющих пользователям взаимодействовать с веб-страницей и отправлять сведения на веб-сервер.

Когда форма HTML создана, необходимо обработать сведения, введенные пользователем, то есть, отправить их файлу. asp для анализа и других действий. Снова изучите программу HTML, приведенную в предыдущем примере. Заметьте, что атрибут ACTION тега <FORM> ссылается на файл Profile. asp. Когда пользователь отправляет сведения HTML, обозреватель передает их методом POST файлу. asp на сервере, в данном случае - Profile. asp. Файл. asp может содержать сценарии, обрабатывающие сведения и взаимодействующие с другими сценариями, компонентами COM и ресурсами, например, с базой данных.

Существует три основных способа сбора сведений, введенных в формы HTML, с использованием ASP:

статический файл. htm, содержащий форму, значения из которой передаются в файл. asp;

файл. asp, создающий форму, отправляющую сведения в другой файл. asp;

файл. asp, создающий форму, отправляющую сведения самой себе, то есть, файлу. asp, содержащему форму. [19]

Объект ASP Request предоставляет два семейства, облегчающих получение сведений, введенных в форму и отправленных по запросу URL.

Семейство QueryString возвращает значения формы, переданные веб-серверу, в виде текста следующего после вопросительного знака в запрошенном адресе URL. Значения формы можно добавить к запрошенному адресу URL, используя либо метод HTTP GET, либо вручную добавив значения формы в адрес URL. [20]

Файл Profile. asp может содержать следующий сценарий обработки данных формы, с помощью которой были введены имя, фамилия и возраст пользователя:

Привет, <%= Request. QueryString ("FirstName")%> <%= Request. QueryString ("LastName")%>.

Вам <%= Request. QueryString ("Age")%> лет!

При использовании метода HTTP GET для передачи длинных и сложных значений формы на веб-сервер возможна потеря сведений. Некоторые веб-серверы ограничивают длину строки запроса адреса URL, поэтому длинные значения формы, переданные методом GET, могут быть обрезаны. Если требуется отправить большой объем сведений из формы на веб-сервер, следует использовать метод HTTP POST. Метод POST, отправляющий данные формы в теле запроса HTTP, может отправлять на сервер текст неограниченного объема. Для получения значений. отправленных методом POST, можно использовать семейство Form объекта ASP Request. [24]

Способ хранения значений в семействе Form похож на способ хранения значений в семействе QueryString. Например, если пользователь заполнил форму, введя длинный список имен, их можно получить с помощью следующего сценария:

<% lngTotal = Request. Form ("Food"). Count

For i = 1 To lngTotal

Response. Write Request. Form ("Food") (i) & "<BR>"

Next%>

Для отправки в обозреватель содержимого, обозначенного ограничителями ASP, или из процедуры, используется метод Write объекта Response. Например, в следующей инструкции показано, как отправлять пользователям разные приветствия в зависимости от того, посещали ли они данную страницу раньше:

<% If blnFirstTime Then

Response. Write "<H3 ALIGN=CENTER>Welcome to the Overview Page. </H3>"

Else

Response. Write "<H3 ALIGN=CENTER>Welcome Back to the Overview Page. </H3>"

End If%>

Метод Response. Write используют в том случае, если нежелательно разрывать инструкцию на части с помощью разделителей или необходимо собрать строку, которая должна быть отправлена в обозреватель.

7. Процедуры. Процедура является группой команд сценария, выполняющих определенную задачу и возвращающих значение. Можно определять собственные процедуры и вызывать их из сценария любое число раз.

Определение процедур можно поместить в тот же файл. asp, из которого они будут вызываться, или можно поместить наиболее часто используемые процедуры в общий файл. asp и использовать директиву #include, чтобы включить его в другие файлы. asp, вызывающие процедуры. Кроме того, можно реализовать те же функции в компоненте COM. [21]

Определение процедур должны быть заключены в теги <SCRIPT> и </SCRIPT> и соответствовать правилам объявленного языка сценариев. Для процедур на языках, отличающихся от основного языка сценариев, служит элемент <SCRIPT>. Для процедур на основном языке сценариев служат ограничители сценариев (<% и%>).

При использовании тега HTML <SCRIPT> для указания, что сценарий следует выполнять на сервере, необходимо использовать два атрибута. Синтаксис использования тега <SCRIPT>:

<SCRIPT LANGUAGE=JScript RUNAT=SERVER>

определение процедуры

</SCRIPT>

Атрибут RUNAT=SERVER указывает, что сценарий должен быть выполнен на веб-сервере. Если не установить этот атрибут, данный сценарий будет выполнен в обозревателе клиента. Атрибут LANGUAGE определяет язык сценариев, используемый в данном блоке сценария. Можно указать любой язык, для которого на сервере установлен обработчик сценариев. Для указания VBScript случит значение VBScript. Для указания JScript случит значение JScript. Если не указать атрибут LANGUAGE, будет считаться, что в блоке сценария использован основной язык сценариев. [22]

Команды в блоке сценариев должны составлять одну или несколько полных процедур на выбранном языке сценариев. Например, следующие команды определяют процедуру JScript MyFunction.

<HTML>

<SCRIPT LANGUAGE=JScript RUNAT=SERVER >

function MyFunction ()

{ Response. Write ("Вызвана процедура MyFunction (). ") }

</SCRIPT>

Для вызова процедуры ее имя следует включить в команду. При вызове процедур JScript из VBScript необходимо использовать скобки после имени процедуры; если процедуре не требуются аргументы, оставьте скобки пустыми. При вызове процедур VBScript или JScript из JScript после имени процедуры всего следует ставить скобки.

В VBScript для вызова процедуры также можно использовать ключевое слово Call. Однако, если вызываемой процедуре требуется аргументы, список аргументов должен быть заключен в скобки. Если ключевое слово Call опущено, также следует опустить скобки, в который заключен список аргументов. Если для вызова любой встроенной или пользовательской функции используется синтаксис Call, значение, возвращаемое функцией, теряется.

Чтобы передать весь массив процедуре VBScript, используется имя массива и следующие за ним пустые скобки; в JScript - пустые квадратные скобки. [21]

8. Компоненты и объекты. Компоненты COM являются основой построения настоящих веб-приложений, обладающих мощными возможностями. Компоненты реализуют функции, позволяющие решать с помощью сценариев конкретные задачи, такие как выполнение финансовых транзакций или проверка данных. Кроме того, в ASP имеется набор основных компонентов, значительно улучшающих создаваемые сценарии.

Компонент COM представляет собой повторно используемый блок инструкций. Компонент предназначен для построения программ и обеспечивает решение некоторой задачи или набора задач. Чтобы создать веб-приложение, компоненты объединяют друг с другом, в том числе посредством сетей. Компоненты COM обеспечивают решение общих задач, избавляя пользователей от необходимости самим создавать соответствующие программы. [20]

Компонент - это исполняемый код, содержащийся в библиотеке динамической компоновки (. dll) или исполняемом файле (. exe). Компоненты предоставляют один или несколько объектов, каждый из которых является законченным набором инструкций, выполняющим в рамках данного компонента определенные функции. Каждый объект имеет методы (запрограммированные процедуры) и свойства (атрибуты функционирования). Чтобы воспользоваться объектом, предоставленным компонентом, следует создать один экземпляр нужного объекта и назначить ему имя переменной. Для создания экземпляра объекта применяется метод ASP **Server. CreateObject** или тег HTML <OBJECT>. С помощью инструкции присвоения переменной, используемой в данном языке сценариев, экземпляру объекта присваивается некоторое имя. [15]

Методом называется действие, которое может быть выполнено над объектом или с помощью объекта. Синтаксис вызова метода: Объект. Метод параметры

Параметры изменяются в зависимости от метода.

Свойство является атрибутом, описывающим объект. Свойства определяют характеристики объекта, например его тип, или описывают состояние объекта, например включен объект или отключен. Синтаксис: Объект. Свойство

Иногда значение свойства можно прочитать или задать. Кроме того, некоторые объекты допускают добавление новых свойств.

9. Работа с файлами. Директивы включения дают возможность включения в файл содержимого другого файла перед обработкой его веб-сервером. В ASP этот механизм поддерживается только директивой #include. Для вставки файла в файл. asp используется следующий синтаксис:

<! - #include virtual | file ="имя\_файла" - ->

Ключевые слова virtual и file обозначают тип пути, используемый для включения файла, а имя\_файла представляет собой путь и имя файла, который требуется включить.

Включаемые файлы могут иметь любое расширение, но удобно давать им расширение, отличающее их от других файлов.

Ключевое слово virtual указывает, что путь начинается с виртуального каталога. Ключевое слово file указывает, что используется относительный путь. Относительный путь начинается с каталога, содержащего файл, в который включается другой файл. [22] Синтаксис включения файлов с атрибутом SRC приведен в таблице 1.

Таблица 1

Синтаксис включения файлов с атрибутом SRC

|  |  |  |
| --- | --- | --- |
| Тип пути | Синтаксис | Пример |
| Относительный | SRC="*путь\имя\_файла*" | SRC="Utilities\Test. asp" |
| Виртуальный | SRC="/*путь/имя\_файла*" | SRC="/MyScripts/Digital. asp" |
| Виртуальный | SRC="\*путь\имя\_файла*" | SRC="\RegApps\Process. asp" |

## 1.4 XML

1. Общие сведения. С появлением языка XML реализовалась идея о том, что любые данные должны сопровождаться собственным описанием. Широкое распространение, которое получил расширяемый язык разметки (Extensible Markup Language, XML) за время, прошедшее с 1998 года, когда первый стандарт XML был опубликован консорциумом W3C (World Wide Web Consortium), служит подтверждением того, что многие отрасли науки и промышленности созрели для серьезных изменений в области описания данных.

Одна из причин быстрого распространения XML заключается в том, что Интернет способствует знакомству широких слоев пользователей с HTML, стандартным языком разметки веб-страниц.html помог многим пользователям получить представление о том, как с помощью тегов в простой текст можно включать дополнительную информацию. Основываясь на этой идее, можно сделать следующий небольшой, хотя и нетривиальный, шаг - перейти к использованию тегов XML для описания данных. [9]

Наиболее очевидным преимуществом XML является возможность расширения этого языка для решения специфических для каждого пользователя задач. Эта возможность делает XML столь привлекательным для тех, кому приходится работать с различными типами данных в какой-либо специальной области. Другое, не столь очевидное преимущество заключается в том, что для любых расширений этого языка при проверке и обработке данных можно продолжать использовать инструментарий XML, если только при записи данных соблюдены правила языка.

Оказалось, что подобная гибкость - как раз то, что требуется для многих практических приложений. Компании, которые обмениваются информацией с другими компаниями через Интернет, все чаще и чаще отказываются от стандарта EDI (Electronic Data Interchange - электронный обмен данными) в пользу XML. Предполагается, что эти так называемые В2В (Business-to-business - бизнес для бизнеса) приложения содержат наибольшее количество данных в формате XML во всем Интернете. [14]

Расширяемый язык разметки (XML) является подклассом стандартного языка разметки (Standard Generalized Markup Language, SGML). XML был задуман как гибкий и в то же время формальный метаязык для использования в Интернете.

Метаязык (metalanguage) - это язык, предназначенный для описания других языков. Например, можно сказать, что словарь английского языка в совокупности с английской грамматикой образуют метаязык, описывающий английский язык. [29]

Назначение языка XML - описывать языки разметки. В языке разметки (markup language) для структурирования данных используются теги. Язык гипертекстовой разметки (Hypertext Markup Language, HTML), наиболее распространенный на сегодняшний день язык разметки, исходно был написан на SGML, но мог бы быть и в какой-то степени был написан и на XML.

В то время как HTML содержит фиксированный набор тегов, в XML теги вообще отсутствуют. Вместо этого XML позволяет программисту самому создать такой язык разметки, который в точности соответствует требованиям конкретного приложения.

В приложениях XML обычно используются следующие типы данных и вспомогательные функции:

сам файл XML, имеющий строго определенную структуру;

определение типа документа (Document Type Definition, DTD), где определяется структура файла XML (необязательный элемент);

таблицы стилей, содержащие информацию о том, как данные должны быть отформатированы при выводе (необязательный элемент);

процессор XML и различные служебные функции для манипулирования данными и переформатирования данных. [2]

Поскольку документы HTML создаются обычно в предположении, что их будут читать с помощью веб-браузера, многие теги HTML указывают, как следует форматировать данные, но не содержат никакой информации для обычного человека. Назначение XML заключается как раз в том, чтобы отделить данные, содержащиеся в документе, от кода, который задает формат отображения этих данных. Это свойство XML позволяет извлекать данные из документа автоматически, то есть с помощью программных средств. [14]

2. Области применения XML. XML можно применять как на стороне сервера, так и на стороне клиента.

XML на стороне клиента. На стороне клиента XML позволяет достичь такого уровня соответствия конкретным условиям представления данных, которого очень трудно или невозможно достичь с использованием HTML. Например, для таких устройств, как PDA (Personal Digital Assistant - "карманный" компьютер, предназначенный для выполнения некоторых специальных функций) или мобильный телефон, требуется, чтобы страницы были отформатированы совсем не так, как для стандартных веб-браузеров. Обычно, если даже имелась готовая страница, предназначенная для веб-браузера, для подобных устройств приходилось полностью ее переделывать, то есть фактически создавать новую версию этой страницы. Однако благодаря структурированным данным документа XML, в котором содержательные данные отделены от форматирующих указаний, все, что требуется сделать для приведения страницы в соответствие с каждым конкретным отображающем ее прибором, - это применить к имеющимся данным нужную таблицу стилей.

XML на стороне сервера. В наши дни XML оказывает очень большое влияние на организацию работы сервера. Один из способов применения XML на стороне сервера - передача сообщений (messaging), то есть обмен данными между приложениями или компьютерами. Чтобы приложения и компьютеры могли обмениваться информацией, для них должен быть определен единый формат сообщений. Основная причина, по которой XML прекрасно подходит для создания форматов передачи сообщений, - это его простота. XML подчиняется строго определенному стандарту, он не связан с какой-либо операционной системой или производителем, он совместим с большим количеством инструментальных средств и приложений, которые на протяжении многих лет разрабатывались для SGML. Требование строгого соответствия документов XML стандартам, установленным для правильно оформленных документов, гарантирует, что любой анализатор XML будет в состоянии прочесть и осмыслить любой документ XML. Кроме того, гораздо больше людей знакомы с языками разметки, чем с форматами сообщений, необходимыми для построения систем EDI. Благодаря XML формат сообщений может разработать любой, кто способен составить правильно оформленный документ XML.

Другой областью применения XML в веб-документах является определение метасодержимого. Метасодержимое, или сведения о содержимом, позволяет сделать работу поисковых машин гораздо эффективнее. [29]

3. Использование XML для хранения данных. XML подходит и для создания баз данных. В документе XML используется древовидная структура хранения данных. Хотя по большому счету хранение данных в виде документов XML не слишком эффективно, у такого способа хранения есть свои преимущества. Как и в отношении передачи сообщений, самым большим преимуществом является простота. Древовидная структура - интуитивно понятный и знакомый способ организации данных. Кроме того, почти любой тип древовидной структуры - от реляционных баз данных до объектно-ориентированных баз данных и иерархических структур - может быть представлен с помощью дерева данных XML. Другое существенное преимущество использования XML для хранения данных заключается в том, что XML поддерживает набор символов Unicode. Следовательно, любой символ любого алфавита мира можно включить в документы XML на "законном основании". [16]

4. Определение правильно оформленного документа XML. Весь текст XML-документа можно грубо разделить на две категории, символьные данные и разметку. К разметке (markup) относится все, что начинается с символа < и заканчивается символом > или начинается с символа & и заканчивается парой символов &;. Символьные данные (character data) - это все, что не является разметкой. Символьные данные можно подразделить еще на две категории: проверенные символьные данные (Parsed Character DATA, PCDATA) и непроверенные символьные данные (unparsed character data). Как следует из названия, данные PCDATA - это данные, проверенные анализатором XML.

Элемент, лишенный содержимого, называется пустым элементом (empty element). Открывающий и закрывающий теги пустого элемента можно объединить в один тег, поместив в его конец косую черту: <br/>. В XML также допускается запись пустого элемента с помощью открывающего и закрывающего тегов, например <br></br>.

Фактически существуют два типа пустых элементов: те, которые заранее определены как пустые и в принципе не могут иметь содержимого, и те, которые оказались пустыми случайно. Чтобы различить эти два типа элементов, рекомендуется использовать пару (открывающий тег, закрывающий тег) для тех элементов, которые не содержат данных, и один тег для тех элементов, которые определены как пустые.

У элементов могут быть атрибуты. Атрибут (attribute) - это пара имя-значение, расположенная в открывающем теге элемента.

В XML значения атрибутов должны быть заключены в одинарные или двойные кавычки. [27]

5. DTD и допустимость документа XML. DTD, или определение типа документа, - это способ явным образом определить структуру класса документов XML.

Основным базовым типом объявления в DTD является объявление элемента, <! ELEMENT>. Формат объявления элемента:

<! ELEMENT имя\_элемента правило>

Каждый элемент, который используется в документе XML, должен быть определен в DTD. Существует несколько правил, которым нужно следовать при именовании элементов:

имена элементов не должны содержать символа < или >;

имя элемента должно начинаться с буквы или символа подчеркивания. После первой буквы в имени элемента может содержаться любое количество букв, цифр, дефисов, точек или символов подчеркивания;

имена элементов не могут начинаться с последовательности xml (в любой комбинации верхнего или нижнего регистров);

двоеточия запрещены, они применяются только в пространствах имен. [14]

6. Использование XSL. Расширяемый язык таблиц стилей (Extensible Stylesheet Language, XSL) - это язык для конструирования таблиц стилей. Таблицы стилей XSL используются для описания внешнего вида документов XML, предназначенных для чтения людьми. [29]

Применение таблицы стилей к документу процессор таблиц стилей осуществляет в два этапа. Первый этап - трансформация дерева (tree transformation). Трансформация дерева позволяет упорядочивать элементы, а также перемещать данные XML и выполнять вычисления с этими данными. Второй этап - форматирование (formatting). Форматирование фактически представляет собой процесс задания стиля отображения данных, размеров шрифтов, разрывов страниц и т.п.

Спецификация XSL содержит три различных языка, предназначенных для осуществления этих двух задач:

XML Path Language (XPath) - язык для создания ссылок на различные части документа XML;

XSL Transformations (XSLT) - язык, предназначенный для генерации дерева документа;

Extensible Stylesheet Language (XSL) - XSLT плюс описание набора объектов форматирования и свойств форматирования. [28]

Например, мы имеем файл ex01. xml, содержащий следующий код:

<? xml version="1.0" encoding="WINDOWS-1251"? >

<tutorial>

<title>"Война и мир"</title>

<author>Лев Толстой</author>

</tutorial>

Добавим в него строку <? xml-stylesheet type='text/xsl' href='ex01. xsl'? > и создадим XSL-файл ex01. xsl, содержащий следующий код:

<xsl: stylesheet version="1.0" xmlns: xsl="http://www.w3.org/TR/WD-xsl">

<xsl: template match="/">

<p><strong><xsl: value-of select=" // author"/></strong></p>

<p><xsl: value-of select=" // title"/></p>

</xsl: template>

</xsl: stylesheet>

7. Требования к правильно оформленному документу XML.

Каждый элемент должен иметь открывающий и закрывающий теги, за исключением пустых элементов, для которых предусмотрен специальный синтаксис пустого элемента;

Имя открывающего тега должно совпадать с именем закрывающего тега. Язык XML чувствителен к регистру;

Элементы должны быть правильно вложены друг в друга;

Имена элементов не должны содержать символа < или > и должны начинаться с буквы или символа подчеркивания;

Имена элементов не могут начинаться с символов xml (в любой комбинации верхнего и нижнего регистров);

В именах элементов не должно содержаться двоеточия, за исключением пространств имен;

Атрибуты не должны появляться более одного раза в открывающем теге или в теге пустого элемента;

Значения атрибутов должны быть заключены в кавычки;

В значениях атрибутов не должны содержаться прямые или непрямые ссылки на внешние сущности;

Текст, подставляемый вместо любой сущности, на которую имеется прямая или косвенная ссылка, являющаяся значением некоторого атрибута, не должен содержать символа < (это не относится к сущности &lt;); [2]

## Выводы по Главе I

Macromedia Dreamweaver MX - одна из самых мощных программ, поддерживающих все современные стандарты Интернета и невероятно облегчающая выполнение даже самых сложных задач.

Интерфейс программы по сравнению с предыдущими версиями претерпел изменения. Новый стиль заметно облегчает работу с программой. Рабочая область Dreamweaver содержит следующие окна и панели: окно документа, стандартное раскрывающееся меню, панель Properties, панель Insert, группу панелей Design, Code, Application, Files для удобства разработчика.

Нами были рассмотрены технологии РНР, ASP и XML. Для описания технологий РНР и ASP были отмечены следующие положения: общие сведения о технологии, общие характеристики, синтаксис, переменные и константы, управляющие конструкции, обработка запросов, функции, классы и объекты, работа с файлами. Технология XML была рассмотрена отдельно, так как не является технологией создания серверных сценариев.

PHP - это язык обработки гипертекста, используемый на стороне сервера, конструкции которого вставляются в HTML-текст. Его цель - позволить максимально быстро создавать динамически генерируемые web-страницы.

РНР имеет много возможностей (совместимость с различными серверами и платформами, поддержка различных технологий, использование различных протоколов, наличие функций для работы с текстовыми данными любых форматов, включая XML, работа с базами данных, простой синтаксис и т.д.), но несмотря на это существует недостаток - бесплатность. Для создателей рядовых сайтов это плюс, но для серьезных компаний, дорожащих своей репутацией, это минус. Технология практически идеальна для разработки развлекательного или коммерческого интернет-проекта (например, интернет-магазина).

Active Server Pages (ASP) - это серверная среда для разработки и выполнения динамических интерактивных веб-приложений. Средства ASP позволяют объединить возможности HTML-страниц, команд сценариев и компонентов COM в интерактивных веб-страницах и мощных веб-приложениях, делают удобным и легким процесс их создания и изменения.

ASP также имеет большое количество возможностей (удобный способ объединение Server-Side Script c HTML, скриптовый подход, концепция "Session", использование компонентов COM, удобный набор объектов-утилит, обработка ошибок, интеграция XML и др.), но уступает РНР по быстроте обработки сценариев (РНР не использует COM как основу работы), совместимостью с различными операционными системами (ASP рассчитан на Windows), работе с СУБД (ASP ориентирован прежде всего на СУБД от Microsoft). Встроенный в ASP-страницы SQL усложняет код и делает его непереносимым на другой источник данных. ASP можно однозначно порекомендовать для разработки сложных приложений на платформе MS Windows (IIS) с использованием технологий COM, DCOM, ActiveX и т.д.

Расширяемый язык разметки (XML) является подклассом стандартного языка разметки (SGML). XML был задуман как гибкий и в то же время формальный метаязык для использования в Интернете. Его назначение - описывать языки разметки. XML можно применять как на стороне сервера, так и на стороне клиента.

XML подходит и для создания баз данных. В документе XML используется древовидная структура хранения данных. Хотя по большому счету хранение данных в виде документов XML не слишком эффективно, у такого способа хранения есть свои преимущества. Как и в отношении передачи сообщений, самым большим преимуществом является простота.

## Глава II. Методика выбора оптимального средства разработки

Исследовав популярные технологии разработки динамических сайтов - ASP, PHP и XML - мы можем приступить к выделению критериев, характеризующих эти технологии с целью разработки на их основе методики выбора оптимального средства создания сайта в зависимости от конкретных прикладных задач.

## 2.1 Выделение критериев сравнения популярных интернет-технологий

Разработка каждого сайта диктует конкретные требования в зависимости от прикладных задач: структура, вид работы с информацией (получение, хранение, передача, обработка), а также от характеристик самой технологии. Исходя из этих требований, мы можем выделить критерии сравнения следующих интернет-технологий: ASP, PHP и XML.

**Простота**. Легкость освоения, простой и понятный синтаксис. Предварительное знакомство с языками VBScript или JScript гарантирует легкое освоение ASP; с языками С и Perl - технологии РНР. В XML теги отсутствуют, вместо этого XML позволяет программисту самому создать такой язык разметки, который соответствует требованиям конкретного приложения.

**Гибкость**. Легкая расширяемость приложений по мере необходимости. Присутствует во всех трех технологиях.

**Компактность**. Отсутствие лишних тегов, наличие встроенных функций для уменьшения кода. Технология ASP является более компактной по сравнению с PHP. В XML содержатся только теги описания данных; теги разметки и различных сценариев исключены.

**Совместимость**. Совместимость с другими технологиями, возможность интеграции в различные языки программирования. Все три технологии совместимы друг с другом и прочими технологиями.

**Эффективность**. Быстрота обработки сценариев. "Движок" PHP является транслирующим интерпретатором, что позволяет обрабатывать сценарии с достаточно высокой скоростью. ASP использует COM как основу работы, поэтому имеет несколько меньшую скорость. В XML сценарии отсутствуют.

**Работа с базами данных**. Возможность извлечения информации из баз данных, записи в базу, хранения данных в базе и т.д. PHP работает с огромным количеством СУБД, в то время как ASP ориентирован прежде всего на СУБД от Microsoft. В XML имеется возможность как занесения данных в базу из xml-файла, так и формирование из базы данных xml-файла.

**Работа с файлами**. Возможность вставки файла, операции с файлами данных (хранение, извлечение, запись информации). В технологиях ASP и PHP присутствуют одинаково эффективные возможности работы с файлами. В XML имеется возможность интеграции двоичных файлов в xml-документ.

**Обработка данных**. Возможность структурирования, сортировки, поиска информации и т.д. Технологии ASP и PHP позволяют производить все виды обработки данных. В XML можно производить сортировку и поиск данных с помощью языка таблиц стилей XSL.

**Использование HTML-форм**. Сбор и отправка данных, полученных с помощью форм, на сервер для последующей обработки. Технологии ASP и PHP поддерживают использование HTML-форм, в XML этой возможности нет.

**Обмен данными**. Возможность обмена данными с другими приложениями. Имеет место в технологиях ASP и PHP. Имеется возможность обмена данными xml-файлами между собой.

**Наличие прямой обратной связи**. Наличие прямой, без посредничества, связи между пользователем и сервером. Присутствует в ASP и PHP, отсутствует в XML.

**Использование таблиц стилей**. Возможность создания таблиц стилей для описания внешнего вида документов. В HTML (куда интегрируются ASP и PHP) это CSS, в XML это XSL.

**Использование СОМ**. Компоненты COM обеспечивают решение общих задач, избавляя пользователей от необходимости самим создавать соответствующие программы. Присутствует в ASP и PHP, отсутствует в XML.

**Создание серверных сценариев**. Основа любого языка для создания динамических сайтов. Присутствует в ASP и PHP, отсутствует в XML.

**Описание данных**. Важная функция, позволяющая представлять данные в едином формате, единым способом записи. Отсутствует в ASP и PHP, присутствует в XML.

**Наличие встроенных функций и объектов**. Уменьшение кода, облегчение работы разработчика. В технологии ASP имеется более обширный набор встроенных функций и объектов по сравнению с PHP. В XML они отсутствуют.

**Написание процедур**. Возможность написания собственных функций для упрощения работы. Присутствует в ASP и PHP, отсутствует в XML.

**Использование сеансов (сессий).** Используя сессии, можно создавать приложения, способные автоматически "узнавать" каждого посетителя узла и собирать относящиеся к нему сведения. Присутствует в ASP и PHP, отсутствует в XML.

**Обработка ошибок**. Важная функция, позволяющая избавиться от ошибок в коде на начальных этапах разработки. Присутствует в ASP и PHP, отсутствует в XML.

**Поддержка различных протоколов**. Возможность поддержки взаимодействия с различными сервисами посредством соответствующих протоколов (протокол управления доступом к директориям LDAP, протокол работы с сетевым оборудованием SNMP, протоколы передачи сообщений IMAP, NNTP и POP3, протокол передачи гипертекста HTTP и т.д.). PHP поддерживает намного больше различных протоколов по сравнению с ASP. В XML поддержка протоколов отсутствует.

**Обеспечение безопасности**. Обеспечение сохранности конфиденциальной информации, ограничение доступа пользователей к некоторым файлам и каталогам, невозможность просмотра исходного текста сценариев в браузере. Присутствует в ASP и PHP, отсутствует в XML.

**Интеграция программного кода в HTML-код**. При разработке приложений удобно внедрять текст сценария в HTML-код. Присутствует в ASP и PHP, отсутствует в XML

**Поддержка различных кодировок**. Вне зависимости от того, в какой стране и на каком языке был разработан сайт, данные в нем будут отображаться корректно. Все три технологии поддерживают различные кодировки.

**2.2 Методика выбора оптимального средства разработки динамических сайтов**

Опираясь на выделенные нами критерии сравнения интернет-технологий, мы можем разработать методические рекомендации в зависимости от конкретного технического задания, то есть набора требований к содержанию и функциям разрабатываемого сайта (табл.2).

Таблица 2

Выбор технологии в зависимости от технического задания

|  |  |  |
| --- | --- | --- |
| В чем существует необходимость | да\нет | Что выбрать |
| Обратная связь | да | PHP, ASP |
| нет | XML |
| Экономическая эффективность | да | ASP, PHP, XML |
|  | нет | **-** |
| Полноценный интерфейс | да | PHP, ASP |
|  | нет | XML |
| Возможность изменения контента в ответ на некие действия посетителя | да | ASP, PHP |
|  | нет | XML |
| Регистрация и авторизация посетителей сайта | да | PHP, ASP |
|  | нет | XML |
| Безопасность сайта и баз данных | да | ASP, PHP |
|  | нет | XML |
| Использование интерактивных модулей - опросов, голосований с возможностью формирование новых и хранения старых | да | PHP, ASP |
|  | нет | XML |
| Простая модификация сайта (добавление/удаление страниц) | да | ASP, PHP, XML |
|  | нет | **-** |
| Отслеживание статистики посещений сайта | да | PHP, ASP |
|  | нет | XML |
| Организация электронных каталогов товаров и услуг, вывод прайс-листов  и т.д. | да | ASP, PHP, XML |
|  | нет | **-** |
| Высокий уровень интеграции элементов информационной системы, построенных с помощью определенной технологии | да | РНР, ASP |
|  | нет | XML |
| Простота разработки | да | РНР, ASP |
|  | нет | XML |

Мы можем сделать вывод, что для разработки сайтов, не содержащих сценариев и не имеющих возможности обратной связи, следует использовать технологию XML; для разработки несложного, однородного по форме содержания сайта - технологию РНР; для разработки несложного, разнородного по форме содержания сайта - технологию ASP; для разработки сложной разнородной системы следует задействовать все три технологии.

После получения технического задания разработчик может обратиться к приведенным выше методическим рекомендациям и, ответив на вопросы из таблицы, выбрать в результате оптимальное средство разработки сайта. Если задача разнородна, и в результатах наблюдается большой разброс, то следует использовать гибридный проект, то есть с использованием нескольких технологий.

## Выводы по Главе II

Мы выделили следующие критерии сравнения технологий PHP, ASP и XML: простота, гибкость, компактность, совместимость, эффективность, работа с базами данных, работа с файлами, обработка данных, использование HTML-форм, обмен данными, наличие прямой обратной связи, использование СОМ, описание данных, наличие встроенных функций и объектов, использование сеансов, поддержка различных протоколов, обеспечение безопасности, поддержка различных кодировок и т.д. Опираясь на выделенные критерии, мы разработали методические рекомендации в зависимости от конкретного технического задания. Использование этих рекомендаций поможет разработчику выбрать оптимальное средство создания сайта. Выбор технологии зависит от сложности и формы содержания сайта - для разработки сайтов, не содержащих сценариев и не имеющих возможности обратной связи, следует использовать технологию XML; для разработки несложного, однородного по форме содержания сайта - технологию РНР; для разработки несложного, разнородного по форме содержания сайта - технологию ASP; для разработки сложной разнородной системы следует задействовать все три технологии.

## Глава III. Разработка динамических веб-страниц на базе различных технологий средствами Dreamweaver

Нами были поставлены следующие задачи: разработка теста (несколько вопросов с возможностью выбора единственного ответа на каждый из них), записной книжки (возможность ввода пользователем информации и вывод ее на этой же странице) и электронного прайс-листа (на примере книжного магазина: информация о книгах - код, название, автор, цена и количество). Подразумевалось создание небольших заготовок с помощью Dreamweaver, не содержащих длинного, сложного кода - исключительно с целью демонстрации применения определенной технологии в зависимости от задачи. Выбор делался на основании разработанных ранее методических рекомендаций.

Для разработки теста была выбрана технология ASP, для разработки записной книжки - технология РНР, для разработки электронного прайс-листа - технология XML.

## 3.1 Записная книжка на базе РНР

Структура записной книжки: HTML-форма для отправки пользователем данных на сервер (файл index.html), серверный сценарий для обработки полученной информации (board. php) и файл для хранения данных (notes. txt). Пользователь вводит данные, они поступают на сервер, записываются в файл и затем выводятся на HTML-страницу. Новая введенная пользователем информация добавляется в файл данных и также выводится на HTML-страницу.

Создание формы в Dreamweaver - Insert > Form > Form. Затем требуется заполнить нужные поля (рис.7).

Аналогично происходит добавление элементов управления в форму - Insert > Form > TextArea (вставка текстового поля) и Insert > Form > Button (вставка кнопки). Так же заполняем нужные поля.

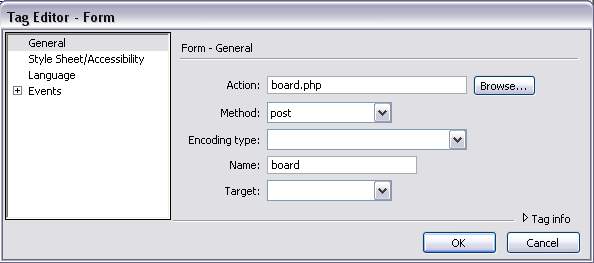


Рис.7. Создание формы

Конечный код формы:

<form action="board. php" method="post" name="board">

<textarea name="message" cols="30" rows=""></textarea><br><br>

<input name="write" type="submit" value="Написать">

</form>

Файл notebook. php тоже содержит код формы, а также теги php:

<? php

$date = date ("Y-m-d H: i: s");

// запись в переменную date текущей даты и времени

$s = $date. "<br>". $message. "<hr>\r\n";

// запись в переменную s даты, данных, введенных пользователем в форму, и горизонтальной линии. \r\n - переход на новую строку

$f = @fopen ("notes. txt","a+");

// открытие файла notes. txt для чтения и записи

fwrite ($f,$s);

// добавление содержимого переменной s в файл

fclose ($f);

// закрытие файла

include ("notes. txt");? >

// вывод содержимого файла на HTML-страницу

Конечный вид страницы представлен на рис.8.

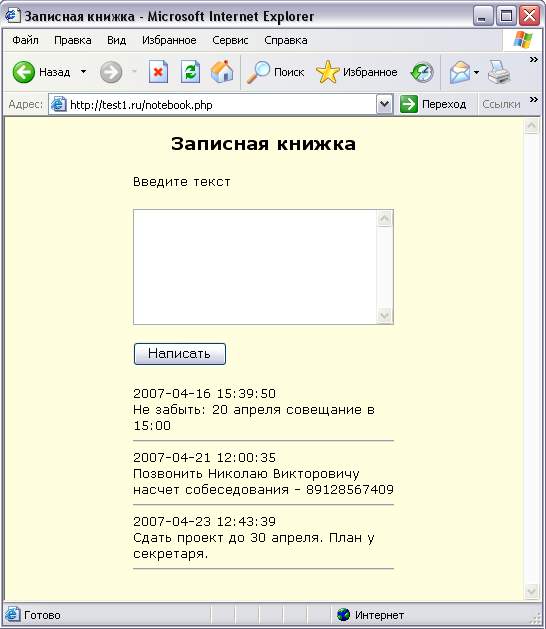


Рис.8. Записная книжка

## 3.2 Электронный прайс-лист на базе XML

В Dreamweaver отсутствует механизм автоматического создания документа xml, поэтому написание кода проходило "вручную", основываясь на теоретических знаниях. Каждый элемент (в нашем случае - книгу) описываем с помощью одинакового набора тегов - book, code (код), title (название), author (автор), price (цена) и quantity (количество) (рис.9). Для того чтобы позже была возможность применения таблицы стилей XSL, введены дополнительные теги: booksCaption, booksCode, booksTitle, booksAuthor, booksPrice, booksQuantity.

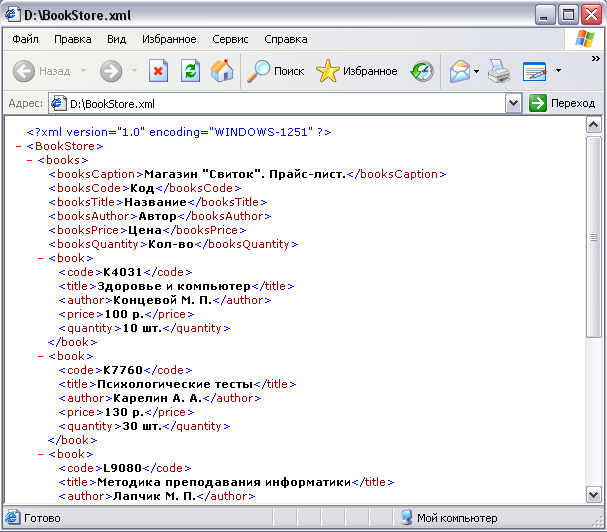


Рис.9. Вид документа без применения XSL

Для применения к документу xml таблицы стилей XSL в файле BookStore. xml прописывается следующая строка:

<? xml-stylesheet type="text/xsl" href="BookStore. xsl"? >.

Файл BookStore. xsl содержит следующий код:

<xsl: stylesheet version="1.0" xmlns: xsl="http://www.w3.org/TR/WD-xsl">

<xsl: template match="/">

<body bgcolor="#F8F8FF">

<font face="Verdana">

<h4><font color="#191970">

<xsl: value-of select=" // booksCaption"/>

</font></h4>

<table border="1">

<tr bgcolor="#B0C4DE">

<td align="center"><strong><xsl: value-of select=" // booksCode"/></strong></td>

<td align="center"><strong><xsl: value-of select=" // booksTitle"/></strong></td>

<td align="center"><strong><xsl: value-of select=" // booksAuthor"/></strong></td>

<td align="center"><strong><xsl: value-of select=" // booksPrice"/></strong></td>

<td align="center"><strong><xsl: value-of select=" // booksQuantity"/></strong></td></tr>

<xsl: for-each select="BookStore/books/book" order-by="number (price)">

<tr bgcolor="#F5F5F5">

<td><xsl: value-of select="code"/></td>

<td><xsl: value-of select="title"/></td>

<td><xsl: value-of select="author"/></td>

<td><xsl: value-of select="price"/></td>

<td><xsl: value-of select="quantity"/></td></tr>

</xsl: for-each>

</table>

</font>

</body>

</xsl: template>

</xsl: stylesheet>

Вид документа с применением таблицы стилей представлен на рис.10.

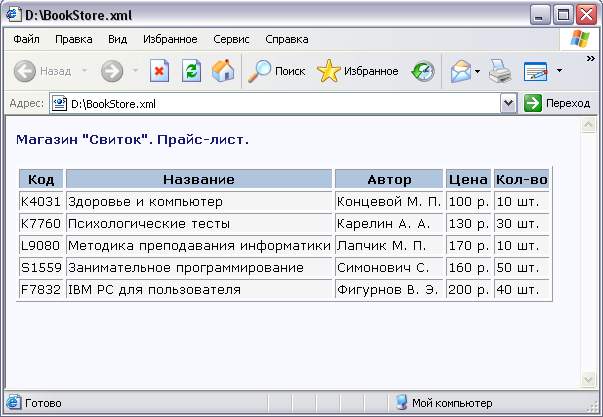


Рис.10. Вид документа с применением XSL

## 3.3 Тест на базе ASP

Мы выяснили, что Dreamweaver не дает возможности создания asp-страниц с активностью на стороне сервера, поэтому было решено создать asp-страницу, содержащую скрипт, написанный на языке VBScript и выполняющийся на стороне клиента. Скрипт заключается в теги <script> и </script>.

Файл Test. asp содержит форму для сбора данных и скрипт для их обработки. Создание формы в Dreamweaver - Insert > Form > Form. Вставка кнопки - Insert > Form > Button, вставка переключателей - Insert > Form > Radio Button. Код кнопки:

<input type = "button" value="Проверка" onclick="valid ()">

При нажатии на кнопку происходит вызов функции valid ().

Создаем группу переключателей для каждого вопроса, задаем им уникальное имя и команду при нажатии - это требуется для работы скрипта.

Фрагмент кода формы:

<b>1. Сколько лет длилась столетняя война? </b>

<input type="radio" name="zad1" onclick="zad1\_1 ()"> 100

<input type="radio" name="zad1" onclick="zad1\_2 ()"> 116

<input type="radio" name="zad1" onclick="zad1\_3 ()"> 99

При нажатии на переключатель вызывается функция, обозначенная после onclick.

При написании скрипта сначала задаем переменные и обнуляем их:

<script Language = VBScript>

dim ot1, ot2, ot3, ot4, ot5

ot1=0: ot2=0: ot3=0: ot4=0: ot5=0

Затем для каждого вопроса задаем значение - правильный (1) и неправильный (0) ответ:

sub zad1\_1 ()

ot1=0

end sub

sub zad1\_2 ()

ot1=1

end sub

sub zad1\_3 ()

ot1=0

end sub

Для описания функции valid () задаем переменную ot для хранения результатов теста

sub valid ()

ot=""

и обрабатываем каждый ответ

if ot1=1 then

ot=ot+"Ответ 1 правильный"

else

ot=ot+"Ответ 1 неправильный"

end if

ot=ot+chr (13)

Далее выводим сообщение с результатами:

MsgBox ot, 0, "Результаты"

window. navigate ("Test. asp")

end sub

Конечный вид страницы представлен на рис.11.

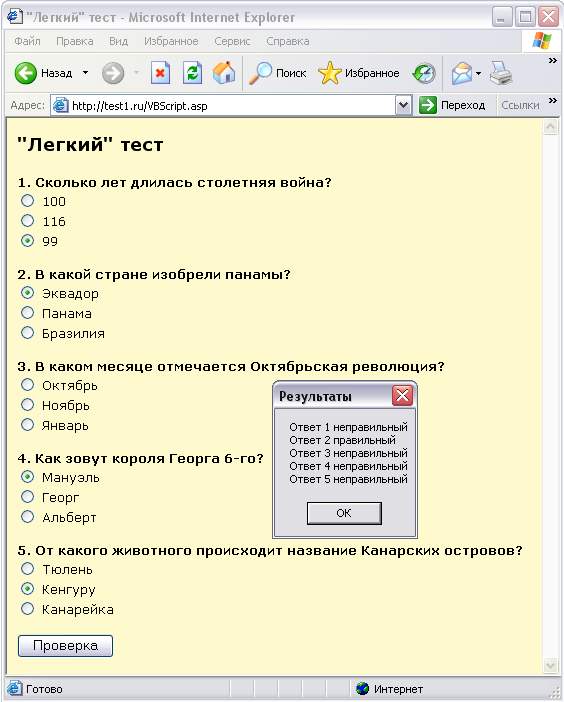


Рис.11. Тест.

## Выводы по Главе III

Нами были поставлены следующие задачи: разработка записной книжки, теста и электронного прайс-листа. С помощью Dreamweaver были созданы заготовки, реализующие эти задачи, на базе технологий РНР, XML и ASP. Выбор в пользу того или иного средства разработки делался на основании разработанных ранее методических рекомендаций.

Были приведены следующие данные по каждой заготовке: структура документа, процесс разработки страниц, фрагменты программного кода с комментариями и конечный вид страниц.

## Заключение

Macromedia Dreamweaver MX - одна из самых мощных программ, поддерживающих все современные стандарты Интернета и невероятно облегчающая выполнение даже самых сложных задач. Dreamweaver MX позволяет создавать сайты на базе различных веб-технологий: HTML, XHTML, XML, ColdFusion, ASP, ASP.net, JSP и PHP.

PHP - это язык обработки гипертекста, используемый на стороне сервера, конструкции которого вставляются в HTML-текст. Его цель - позволить максимально быстро создавать динамически генерируемые web-страницы. РНР имеет много возможностей (совместимость с различными серверами и платформами, поддержка различных технологий, использование различных протоколов, наличие функций для работы с текстовыми данными любых форматов, включая XML, работа с базами данных, простой синтаксис и т.д.), но несмотря на это существует недостаток - бесплатность. Для создателей рядовых сайтов это плюс, но для серьезных компаний, дорожащих своей репутацией, это минус. Технология практически идеальна для разработки развлекательного или коммерческого интернет-проекта (например, интернет-магазина).

Active Server Pages (ASP) - это серверная среда для разработки и выполнения динамических интерактивных веб-приложений. Средства ASP позволяют объединить возможности HTML-страниц, команд сценариев и компонентов COM в интерактивных веб-страницах и мощных веб-приложениях, делают удобным и легким процесс их создания и изменения.

ASP также имеет большое количество возможностей (удобный способ объединение Server-Side Script c HTML, скриптовый подход, концепция "Session", использование компонентов COM, удобный набор объектов-утилит, обработка ошибок, интеграция XML и др.), но уступает РНР по быстроте обработки сценариев (РНР не использует COM как основу работы), совместимостью с различными операционными системами (ASP рассчитан на Windows), работе с СУБД (ASP ориентирован прежде всего на СУБД от Microsoft). Встроенный в ASP-страницы SQL усложняет код и делает его непереносимым на другой источник данных. ASP можно однозначно порекомендовать для разработки сложных приложений на платформе MS Windows (IIS) с использованием технологий COM, DCOM, ActiveX и т.д.

Расширяемый язык разметки (XML) является подклассом стандартного языка разметки (SGML). XML был задуман как гибкий и в то же время формальный метаязык для использования в Интернете. Его назначение - описывать языки разметки. XML можно применять как на стороне сервера, так и на стороне клиента. XML подходит и для создания баз данных. В документе XML используется древовидная структура хранения данных. Хотя по большому счету хранение данных в виде документов XML не слишком эффективно, у такого способа хранения есть свои преимущества. Как и в отношении передачи сообщений, самым большим преимуществом является простота.

Мы выделили критерии сравнения технологий PHP, ASP, XML и, опираясь на них, разработали методические рекомендации в зависимости от конкретного технического задания. Выбор технологии зависит от сложности и формы содержания сайта - для разработки сайтов, не содержащих сценариев и не имеющих возможности обратной связи, следует использовать технологию XML; для разработки несложного, однородного по форме содержания сайта - технологию РНР; для разработки несложного, разнородного по форме содержания сайта - технологию ASP; для разработки сложной разнородной системы следует задействовать все три технологии.

С помощью Dreamweaver были созданы заготовки, реализующие различные задачи (записная книжка, тест, электронный прайс-лист), на базе технологий РНР, XML и ASP. Выбор в пользу того или иного средства разработки делался на основании разработанных ранее методических рекомендаций. По каждой заготовке были приведены следующие данные: структура документа, процесс разработки страниц, фрагменты программного кода с комментариями и конечный вид страниц.

Нами были рассмотрены возможности применения мощного и эффективного инструмента разработки динамических сайтов для реализации технологий ASP, PHP и XML. В работе были изложены теоретические основы сайтостроения средствами разных технологий, реализуемых в рамках одного инструмента. Была разработана методика, которая впоследствии может применяться разработчиками для выбора оптимального средства создания динамических веб-сайтов в зависимости от прикладных задач.

## Библиографический список

1. Бардзелл Д. Macromedia Dreamweaver MX 2004 с ASP, ColdFusion и PHP. Из первых рук. - М.: ЭКОМ, 2006. - 235 с.
2. Бумфрей Ф., Диренцо О. XML. Новые перспективы. - М.: ДМК-Пресс, 2000. - 272 с.
3. Веллинг Л., Томсон Л. Разработка Web-приложений с помощью PHP и MySQL. - М.: Вильямс, 2005. - 429 с.
4. Дунаев В.В. Самоучитель PHP. - СПб.: Питер, 2007. - 160 с.
5. Дунаев В.В. Самоучитель. Сценарии для Web-сайта. PHP и JavaScript. - СПб.: BHV-Санкт-Петербург, 2006. - 279 с.
6. Зольников Д.С. PHP 5. Как самостоятельно создать сайт любой сложности. - М.: НТ Пресс, 2005. - 109 с.
7. Кузнецов М.В., Симдянов И.В. PHP 5. Практика создания web-сайта. - СПб.: BHV-Санкт-Петербург, 2005. - 401 с.
8. Куссуль Н.Н., Шелестов А.Ю. Использование PHP. Самоучитель. - М.: Вильямс, 2006. - 117 с.
9. Кэгл К. XML. - М.: ЛОРИ, 2006. - 425 с.
10. Леонтьев Б.К. PHP 5.0 для начинающих, или как создать динамичный Web-сайт. - М.: Новый издательский дом, 2006. - 109 с.
11. Мазуркевич А., Еловой Д. PHP: настольная книга программиста. - М.: Новое знание, 2006. - 242 с.
12. Мак Т., Вест Р. Dreamweaver MX 2004. Шаг за шагом. Самоучитель. - М.: ЭКОМ, 2006. - 312 с.
13. Молер Дж., Боуэн К. Dreamweaver MX 2004. Руководство Web-дизайнера. - М.: ЭКСМО, 2005. - 67 с.
14. Старыгин А. XML Разработка WEB-приложений. - СПб.: BHV-Санкт-Петербург, 2003. - 199 с.
15. Фландерс Й. ASP - взгляд изнутри. Для программистов. - М.: ДМК-Пресс, 2001. - 135 с.
16. Хабибуллин И.Ш. Самоучитель XML. - СПб.: BHV-Санкт-Петербург, 2003. - 111 с.
17. Хестер Н. Создание Web-страниц в Dreamweaver. - М.: НТ Пресс, 2005. - 104 с.
18. Шкрыль А.А. PHP - это просто. Программируем для Web-сайта. - СПб.: BHV-Санкт-Петербург, 2006. - 161 с.
19. http://library. mnwhost.ru/webdev/asp/asp. php
20. http://gun. cs. nstu.ru/IISHelp/iis/htm/asp/iiwanew. htm
21. http://www.stfw.ru/page. php? id=3182
22. http://web. wbz.ru/index. php? page=lesns&hid=2&pid=2
23. http://php. segmenta.ru/manual/ru/intro-whatcando. php
24. http://webzona.org/article/a-3.html
25. http://stud. h16.ru/education/php/
26. http://www.phpclub.net/manrus/
27. http://www.sytchev.ru/xml/xml
28. http://www.uni-vologda. ac.ru/students/tav/xsl. htm
29. http://www.citforum.ru/internet/xslt/xslt. shtml
30. http://argon.com.ru/internet/webmastering/webtech