Государственное автономное профессиональное образовательное учреждение Свердловской области

ЕКАТЕРИНБУРГСКИЙ МОНТАЖНЫЙ КОЛЛЕДЖ

УТВЕРЖДАЮ
Зам. директора по УР
_____
Л. С. Хоринова
«31» августа 2020 г.

Методические рекомендации для выполнения практических работ по МДК 02.04 «Информационные системы на базе интернет технологий»

Рассмотрен на заседании методического объединения информационных технологий

Пр. №1 от «28»августа 2020 г.

Руководитель МО Софьина Н.А.

Екатеринбург

2020

Чистяков Валерий Николаевич

ЛАБОРАТОРНАЯ РАБОТА № 1. СОЗДАНИЕ СЕРВЕРНЫХ СЦЕНАРИЕВ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИИ PHP

Цель изучение основных конструкций языка PHP.

Теоретические вопросы

Конструкции языка PHP, операторы присваивания, операторы вывода.

Основные конструкции языка PHP, операторы сравнения..

Основные конструкции языка PHP. Функции пользователя.

Операторы циклов while и for. Организация циклических вычислительных процессов.

Работа с массивами. Создание и обработка простых (индексированных) массивов и ассоциированных массивов. Использование циклов для работы с массивами

Задание № 1. Изучить конструкции языка PHP, операторы присваивания, операторы вывода. Подготовить в Блокноте или в любом текстовом редакторе программу, выполняющую следующие действия:

- создать три переменные с названием товаров ($product1, $product2, $product3) и соответствующие им переменные с ценой товаров ($price1, $price2, $price3), вывести их на экран;
- рассчитать и вывести среднюю цену товара.

Примерный вид вывода на экран результата работы программы представлен на рисунке.

Протестировать программу с различными значениями переменных.

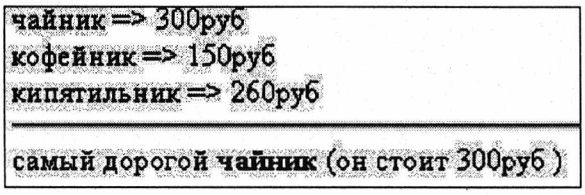
Оформить вывод данных о товарах в виде таблицы. Например, как показано на рисунке.

Для оформления таблицы поместить тэги таблицы в оператор вывода (echo или print). Новый вариант программы сохранить в файле с другим именем.

Использовать для табличного вывода HTML блоки. Для вывода переменных в тэги необходимо включить фрагменты программы. Сохранить файл.

Задание № 2. Изучить условные инструкции if, else, elseif. Подготовить программу для определения самого дорогого из трех товаров. За основу взять файл из задания 1. Сравнить цены

товаров и вывести наименование и цену самого дорогого товара. Сопроводить вывод результата соответствующим сообщением.



Сравнить цену первого товара с ценами второго и третьего товаров. Если она окажется больше сформировать вспомогательную переменную, например $max_prise, равную цене первого товара и $max_ product, равную наименованию первого товара. В противном случае сравнить ценывторого и третьего товаров (использовать конструкцию elseif и else) и записать во вспомогательные переменные соответствующие данные. Вывести вспомогательные переменные. Протестировать программу с различными значениями переменных.

Определить товар с минимальной ценой. Решить задачу, методом "вытеснения", используя только конструкцию if. Во вспомогательные переменные $max_prise и $max_ product сразу записать данные о первом товаре. Последовательно сравнить цены второго и третьего товаров со значением, записанным в переменной $max_prise (конструкция if). Если цена окажется меньше значения записанного в переменной $max_prise, переопределить переменные $max_prise и $max_ product. Протестировать программу с различными значениями переменных.

Задание № 3. Изучить материалы о работе с функциями. Оформить решение задачи Задания 2 с помощью функции, определяющей товар с максимальной ценой. Функция должна иметь шесть формальных входных параметров: три переменные, хранящие наименования товаров, и три переменные, задающие их стоимость. Вывод искомых данных производить внутри функции. После описания функции вызвать ее не менее трех раз с различными значениями фактических параметров.

Подготовить файл, обеспечивающий проверку правильности ввода пароля. Действия по проверке пароля должны выполняться с помощью пользовательской функции с одним входным аргументом. Функция должна сравнивать пароль, заданный внутри функции, с паролем,переданный ей через аргумент. Результат сравнения вывести в виде текста: "Пароль верный" или "Ошибка в пароле". Вывод сообщения должен производиться внутри тела функции. Протестировать программу с различными значениями пароля.

Модифицировать программу так, чтобы вызов функции выполнялся в операторе вывода. Например, если имя функции control($p), то ее вызов: print control("1234"). Внутри тела функции использовать инструкцию return.

Задание № 4. Изучить материалы, относящиеся к организации циклов в PHP. Подготовить текст программы для решения следующей задачи. Пусть стоимость товара равна 100 р. в начале текущего года. Процент инфляции в этом году по прогнозам составит 10 %. В последующие годы прогнозируется увеличение процента инфляции на 3,5 % в год. С помощью циклической программы вывести прогнозируемую стоимость товара к концу текущего года и в последующие годы. Прекратить расчеты, как только стоимость товара превысит 150 р. Использовать цикл while.

Решить ту же задачу с помощью цикла for. Вывести прогнозируемую стоимость товара к концу текущего года и в последующие 5 лет. Вывод оформить в виде таблицы ГОД => ЦЕНА => ИНФЛЯЦИЯ.

Модифицировать файл для решения следующей задачи. Пусть при достижении стоимости товара 170 р., инфляция начнет снижаться каждый год на 3,5 %. Спрогнозировать стоимость товара через 10 лет.

Задание № 5. Изучить основы работы с массивами. Подготовить текст программы, выполняющей следующие действия:

- создать список (индексированный массив), состоящий из пяти наименований товаров с помощью функции array();
- добавить еще не менее двух элементов массива с помощью идентификатора массива;
- определить количество элементов массив, используя функцию count(), и вывести названия товаров в цикле for.

Протестировать работу программы с различным количеством элементов массива.

Модифицировать программу, добавив сортировку массива в алфавитном порядке наименований товаров (использовать функцию sort). Вывести на экран исходный массив и результат сортировки.

Задание № 6. Подготовить программу для обработки ассоциативного массива. Программа должна обеспечивать следующее:

- создать ассоциативный массив: ТОВАР => ЦЕНА, где название товара – это ключ (индекс) массива, а цена – значения элементов массива;
- массив должен содержать не мене пяти элементов, три из них задать с помощью функции array(), а остальные задать непосредственно в операторе присваивания;
- вывести товары и их цены, используя оператор цикла foreach().

Протестировать работу программы с различным количеством элементов массива, добавив их любым способом.

Модифицировать программу для решения следующих задач:

- подсчитать количество товаров и их суммарную стоимость;
- отсортировать массив:

– в порядке убывания (возрастания) цены товара и вывести на экран. Использовать функции asort() и arsort().

– выполнить сортировку массива так, чтобы товары (ключи) расположились в алфавитном порядке для чего использовать функции ksort() или krsort().

Задание № 7. Создайте php-скрипт, выводящий страницу с форматиро-ванной средствами разметки HTML информацией о вас как о разработчике.

Задание № 8. Создайте php-скрипт, генерирующий страницу с табли-цей основных цветов HTML. Указания: интенсивности красно-го, зеленого и синего цветов принимают шестнадцатеричные значения 00, 33, 66, 99, CC, FF. Для преобразования между деся-тичными и шестнадцатеричными числовыми значениями ис-пользуйте стандартные функции dechex, hexdec.

Задание № 9. Реализуйте скрипт, генерирующий и выводящий в брау-зер случайные числа до тех пор, пока их сумма не станет больше или равна заданного значения $n. Указание: для генерации псевдослучайного целого числа, принадлежащего диапазону [$min,$max], используйте стандартную функцию rand($min,$max).

Задание № 10. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 2. ОБРАБОТКА ДАННЫХ НА ФОРМЕ

Цель: изучение основных конструкций языка PHP для работы с формами.

Теоретические вопросы

Работа с формами.

Обработка данных, введенных пользователем через поля формы.

Задание № 1. Изучить материалы, содержащие теоретические сведения об организации работы с формами в языке PHP.

Задание № 2. Подготовить текст программы, выполняющей следующие действия:

создать html-документ, содержащий форму с полями Ф.И.О., Адрес, Email, Пароль и передать введенные данные для обработки php-программе –для вывода данных на экран.

Протестировать работу программы. Решить ту же задачу, но с помощью одного файла. Проверить работоспособность программы.

Задание № 3. Подготовить программу для решения аналогичной задачи, но проверяющей пароль пользователя, вводимый через поле формы. Значение правильного пароля задается внутри текста программы. Сохранить текст программы в файле и протестировать ее.

Модифицировать программу так, чтобы в случае ввода правильного пароля, происходил переход на другой файл с текстом поздравления.

Задание № 4. Подготовить файл для отправки электронного письма. Файл должен содержать форму, в которой расположить 4 элемента с соответствующими комментариями:

- текстовое поле (text) с именем to;
- текстовое поле (text) с именем subject;
- текстовую область (textarea) с именем message;
- кнопка (submit) с именем mail_ok.

Данные из формы передать методом POST скрипту, с функцией отправки сообщения и проверкой правильности отправки письма.

Пояснения к программе отправления электронного письма

Для простоты обработки данных, полученных из формы, назовем соответствующие переменные: $to, $subject и $message. Затем информацию из этих переменных будем использовать для отправки письма на адрес e-mail, указанный в переменной $mail.

Отправка письма производится с помощью функции mail():

bool mail (string $to , string $subject , string $message)

Функция возвращает значение TRUE если почта отправлена и FALSE в противном случае. Так как при работе с локальным хостингом оправка письма не производится, проверку правильности передачи письма можно выполнить с помощью оператора If и вывести соответствующее сообщение на экран.

Пример программы для отправки электронного письма

```
<?
$go=$_POST['mail_ok'];
if(!$go)
{
?>

HTML-код формы для написания письма

<?}
else
{
$to=$_POST['to'];
$subject=$_POST['subject'];
$message=$_POST['message'];
$mail=mail($to,$subject,$message);
if ($mail==TRUE)
  {echo "Письмо отправлено";}
else
  {echo "Не удалось отправить";}
}
?>
```

Задание № 5. Написать программу-калькулятор, которая позволит пользователю передать два числа и указать арифметическую операцию, выполняемую над ними.

Задание № 6. Реализовать ввод и обработку анкеты пользователя. Форма анкеты заполняется на одной странице, скрипт-обработчик, реализованный в отдельном файле, проверяет правильность заполнения всех полей и делает вывод на основе представленной информации (например, вычисляет количество языков программирования, которые знает пользователь, определяет его возраст в годах по введенной дате рождения и т.п.).

Задание № 7. Реализовать тест из 3–4 вопросов с несколькими вариантами ответа на каждый вопрос, предусмотреть начисление баллов за выбранные пользователем варианты ответа. В конце тестирования, в зависимости от количества набранных баллов, вывести резюме по тесту.

Задание № 8. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 3. ОРГАНИЗАЦИЯ ФАЙЛОВОГО ВВОДА-ВЫВОДА

Цель: изучение основных конструкций языка PHP для организации файлового ввода-вывода.

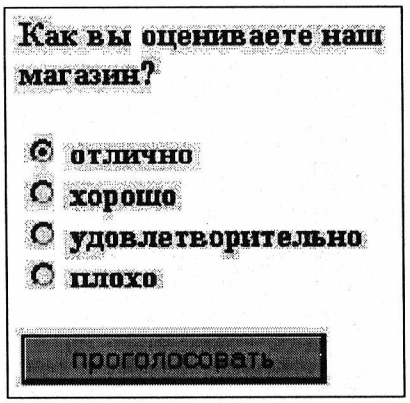
Теоретические вопросы

Организация файлового ввода-вывода в языке PHP.

Обработка данных, хранящихся в файлах.

Задание № 1. Изучить материалы, содержащие теоретические сведения об организации файлового ввода-вывода в языке PHP.

Задание № 2. Создать программу для проведения опроса – голосования по оценке какого-то товара или мероприятия. Создать форму для голосования с вопросом "Как вы оцениваете наш магазин?" и вариантами ответов в виде radio-button.



Подготовить текст программы, обеспечивающей следующие действия. По нажатию кнопки "проголосовать" нужно в соответствии с выбранной оценкой:

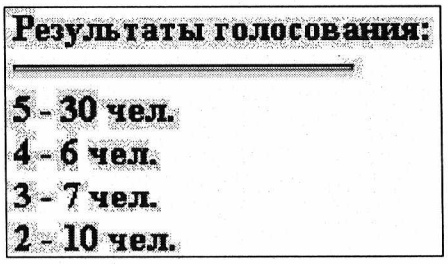
– открыть необходимый файл,

– прочитать записанное в файле число,

– увеличить его на единицу,

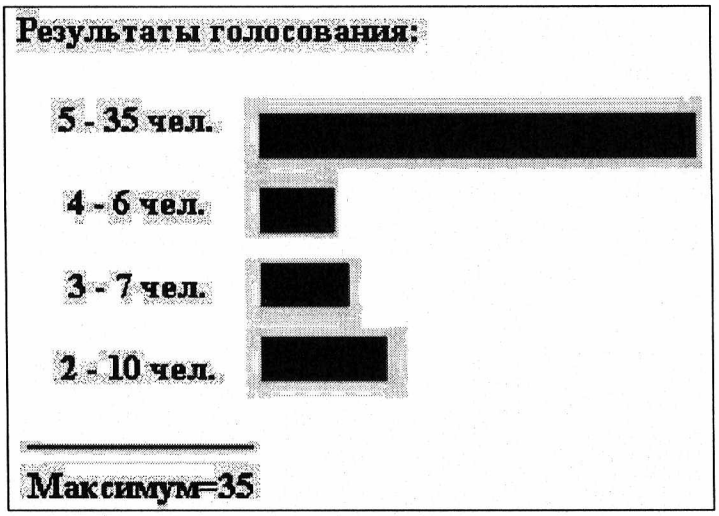
– и перезаписать результат в этот же файл.

Вывести результаты голосования.



Протестировать работу программы не менее десяти раз, просмотреть содержимое файлов.

Модифицировать программу так, чтобы результаты голосования выводились в виде диаграммы.



Создать два вспомогательных файла. Первый должен обеспечивать создание текстовых файлов 2.txt, 3.txt,4.txt, 5.txt и запись в них числа 0. Второй – удаление этих файлов. Произвести несколько раз тестирование процесса голосования.

Рекомендации по составлению программы

1. Создать файлы для хранения информации: 5.txt, 4.txt, 3.txt и 2.txt с первоначальным значением 0 в каждом файле. В дальнейшем в них будут записываться значения счетчиков ответов при голосовании.

2. Написать фрагмент программы, обеспечивающий вывод формы. Значения параметров поля формы указать цифрой (5, 4, 3, 2), совпадающей с именем файла. Например, для первой строки формы с отметкой отлично поле формы может выглядеть так:

<input type="radio" name="vote" value="5" checked > отлично

Заметим, что только это поле "отмечено" (checked), чтобы стимулировать принятие пользователем желаемого решения.

3. Составить программу обработки переданных данных (например, методом POST). Ниже приведен фрагмент программы для реализации обработки файлов (чтения и записи нового значения) :

```
if (@$_POST['vote'])
{
// если параметр vote передаётся методом POST, значит нажата кнопка проголосовать
$file=$_POST['vote'].".txt";
// в переменной vote содержится число 2, 3, 4, или 5. Наши файлы имеют такие же
// названия, значит мы можем использовать эти значения длявыбора файла,
// сформировав таким образом его имя
$f=fopen($file,"r");
// открываем файл для чтения
$votes=fread($f,100);
// записываем в переменную $votes старое количество голосов fclose($f);
// закрываем файл
$votes++;
// увеличиваем на единицу количество голосов
$f=fopen($file,"w");
// открываем файл для записи
fwrite($f,$votes);
// записываем в файл новое количество голосов
fclose($f);
// закрываем файл
```

3. Дописать фрагмент программы для считывания информации из каждого файла и вывода результата.

4. Для вывода диаграммы можно воспользоваться тэгом горизонтальной линии <hr> с параметрами. Например:

```
?>
<hr align="left" color="#FF0000" size="20" width="<?=$vline?>">
<? Программный код
```

Значение параметра width, отвечающего за ширину линии, здесь заданы фрагментом php-скрипта – упрощенная форма вывода переменной. Само значение переменной $vline должно быть связано с переменной $votes масштабным коэффициентом, который следует подобрать самостоятельно.

Задание № 3. Реализовать приложение для работы с телефонным справочником, данные которого хранятся в текстовом файле. Предусмотреть возможность сохранения нескольких номеров телефона для одного абонента.

Задание № 4. Реализовать приложение для проверки доступности сервера, адрес которого введен пользователем в форму.

Задание № 5. Реализовать приложение для получения и отображения информации, полученной с удаленного сервера (например, прогноза погоды, ленты заголовков новостей).

Задание № 6. Реализовать приложение-файло-обменник, позволяющее посетителям страницы загружать фай-лы, пока не исчерпан указанный в настройках приложения об-щий дисковый лимит, а также просматривать список всех за-груженных файлов и удалять их. Предусмотреть список разрешенных типов файлов.

Задание № 7. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 4. ОРГАНИЗАЦИЯ ПОДДЕРЖКИ БАЗЫ ДАННЫХ В PHP

Цель: получение навыков работы с базами данных в PHP.

Теоретические вопросы

Знакомство с возможностями языка PHP для работы с базами данных сервера MySQL.

Выполнение основных операций по подсоединению к базе данных, созданию и заполнению таблиц, просмотру и поиску информации в таблицах.

Знакомство с возможностями языка PHP для работы со связанными таблицами базы данных.

Создание запросов к нескольким связанным таблицам.

Задание № 1. Изучить материалы, содержащие теоретические сведения об организации работы с базами данных в языке PHP.

Задание № 2. Создать свою базу данных, например, baza1.

Просмотреть результаты работы MySQL, проанализировать выполненный SQL-запрос, получить его PHP-код. Выполнить тестирование примера по подключению к базе данных в PHP. Здесь и далее имя хоста будет localhost, имя пользователя – root, параметр пароль нужно оставить пустым. Таким образом, строка $p = mysql_connect("имя хоста", "имя пользователя","пароль") будет выглядеть так:

```
$p = mysql_connect("localhost", "root","");
```

А строка выбора базы данных:

```
mysql_select_db("baza1") or die("NO BASE");
```

Научиться создавать таблицы в базе данных средствами предложенного интерфейса. Познакомится с описанием характеристик полей при создании таблицы. Проанализировать описание полей при создании таблицы telephones:

```
create table telephones(id INT AUTO_INCREMENT PRIMARY KEY,
surname VARCHAR(20),
email VARCHAR(20),
tel VARCHAR(20));
```

Создать таблицу telephones средствами предложенного интерфейса. 4.3.Заполнить таблицу двумя записями, научиться просматривать и редактировать записи. Удалить таблицу.

Создать таблицу с помощью программы PHP. Выполнить программу. Дополнить таблицу несколькими записями через предложенный интерфейс. Просмотреть таблицу telephones. Пополнить таблицу программными средствами PHP.

Создать собственную форму для заполнения таблицы. Наполнить таблицу данным, среди которых должны быть несколько записей с одинаковыми фамилиями. Научиться выводить данные из таблиц программными средствами.

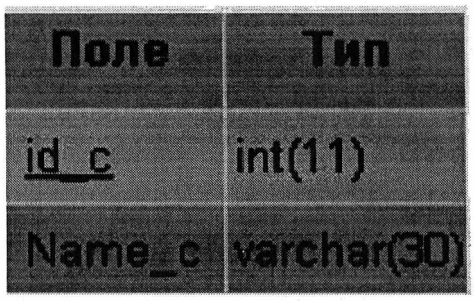
Создать таблицу с товарами products (поля: идентификатор, название, цена, описание) и программно заполнить ее пятью – шестью товарами. Обеспечить средства интерфейса для работы с товарами (заполнение и просмотр таблицы).

Задание № 3. Средствами предложенного интерфейса для работы с базами данных создать таблицу для хранения данных о странах – производителях товаров в Вашем Интернет магазине. Например, таблицу country, которая содержит поля:

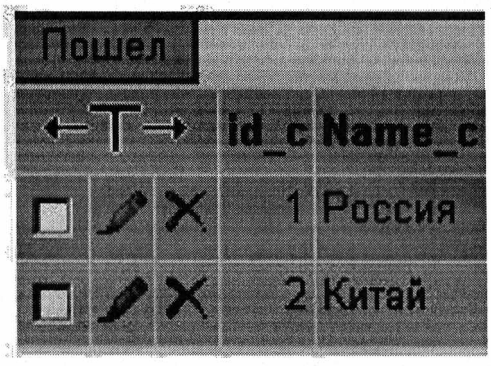
id_c – идентификатор страны, и

name_c – название страны.

Типы полей выбрать самостоятельно. Пример таблицы приведен на рисунке.



Заполнить таблицу двумя – тремя записями.



В таблицу products добавить еще одно поле – идентификатор страны – производителя. Тип данных этого поля должен совпадать с типом поля id_c таблицы country.

Создать таблицу telephones средствами предложенного интерфейса. 4.3.Заполнить таблицу двумя записями, научиться просматривать и редактировать записи. Удалить таблицу.

Создать таблицу с помощью программы PHP. Выполнить программу. Дополнить таблицу несколькими записями через предложенный интерфейс. Просмотреть таблицу telephones. Пополнить таблицу программными средствами PHP.

Создать собственную форму для заполнения таблицы. Наполнить таблицу данным, среди которых должны быть несколько записей с одинаковыми фамилиями. Научиться выводить данные из таблиц программными средствами.

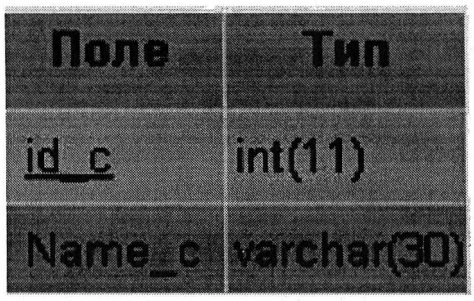
Создать таблицу с товарами products (поля: идентификатор, название, цена, описание) и программно заполнить ее пятью – шестью товарами. Обеспечить средства интерфейса для работы с товарами (заполнение и просмотр таблицы).

Задание № 3. Средствами предложенного интерфейса для работы с базами данных создать таблицу для хранения данных о странах – производителях товаров в Вашем Интернет магазине. Например, таблицу country, которая содержит поля:

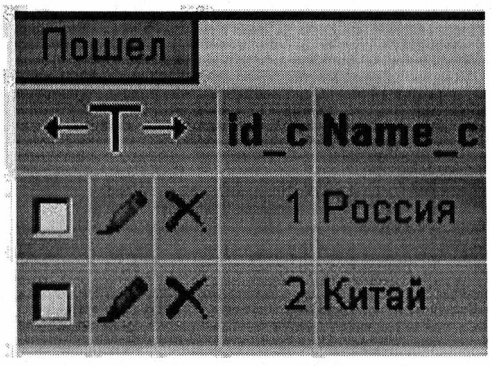
id_c – идентификатор страны, и

name_c – название страны.

Типы полей выбрать самостоятельно. Пример таблицы приведен на рисунке.



Заполнить таблицу двумя – тремя записями.



В таблицу products добавить еще одно поле – идентификатор страны – производителя. Тип данных этого поля должен совпадать с типом поля id_c таблицы country.

Задание № 6. Есть база: фильмы и страны, у одного фильма может быть много стран производителей, надо построить базу. Нерадивый программист удалял фильмы, но не подчистил таблицу связи. Надо найти и удалить все мусорные записи, которые остались в таблице связи.

Задание № 7. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 5. ОТСЛЕЖИВАНИЕ СЕАНСОВ (SESSION)

Цель: получение навыков регистрации и авторизации пользователей на сайте в PHP.

Теоретические вопросы

Понятие сессии.

Отслеживание сеансов.

Идентификация пользователей и процессов.

Понятие и назначение cookies-файлов.

Большинство сложных web-приложений позволяют регистрацию и авторизациюпользователей на сайте. Основой для управления механизмами авторизации служат сессии. Сессии позволяют создавать и использовать переменные, сохраняющие свое значение в течение всего времени работы пользователя с сайтом. При этом у каждого пользователя сайта данные переменные будут собственными, т.е. их область видимости (англ. variable scope) распространяется на все время нахождения на сайте конкретного пользователя, причем для каждого захода пользователя на ваш сайт эти переменные будут различными.

В основе всего механизма сессий лежит решение задачи об идентификации того, от кого именно пришел запрос на сервер. Если это будет точно известно, то уже не возникнет большой проблемы в том, чтобы предоставить скрипту информацию, относящуюся именно к этому конкретному пользователю.

Данная задача решается путем присвоения каждой сессии уникального идентификатора SID (англ. Session IDentifier), который создается в тот момент, когда пользователь заходит на сайт, и уничтожается, когда пользователь уходит с сайта. Он представляет собой строку из 32 символов, например, ac4f4a45bdc893434c95dcaffb1c1811. SID передается на сервер вместе с каждымзапросом клиента и возвращается обратно вместе с ответом сервера.

Алгоритм генерации SID позволяет гарантировать его уникальность, поэтому исключена возможность того, что две сессии будут иметь один и тот же идентификатор сессии. PHP может использовать два различных механизма в качестве транспортного средства для передачи SID:

- файлы cookies;
- дополнительный параметр URL-адреса.

Cookies (от англ. *cookie* – печенье) – это небольшие текстовые файлы с зашифрованной информацией, отправляемые веб-сервером и хранимые на компьютере пользователя в назначенной браузером для этой цели системной папке. Браузер всякий раз при попытке открыть страницу соответствующего сайта пересылает его cookie серверу в составе HTTP-запроса, приэтом SID сохраняется "внутри" браузера и остается незаметным для пользователя. Поддержка cookies – необязательное условие для браузера, она может отсутствовать или быть отключена. Все cookies, как минимум, имеют имя и значение, а отпра-вить их программно можно функцией bool setcookie (string $name [, string $value]), имеющей еще не-сколько необязательных параметров.

Прочитать имеющиеся для данной страницы cookies можно как элементы ассоциативного массива $_COOKIE.

Функции для удаления cookie не существует, это происходит либо по истечении срока ее хранения, либо по явному вызо-ву setcookie с указанием третьим параметром времени хранения, которое уже истекло: setcookie ('my_cookie','',time()–14*24*3600); Следует также учесть, что вначале сервер направляет cookie клиенту как часть отклика HTTP, потом клиент, если он готов принять cookie, возвращает ее серверу. Поэтому для проверки того, подключены ли cookies в браузере клиента, скрипту может понадобиться программная перезагрузка страницы.

```
<?php
 $my_cookie = '';
 if (!isset($_GET['step'])) { //Первый вызов
  setcookie ('my_cookie','ok');
  header ('Location: '.$_SERVER['PHP_SELF'].
'?step=1');//Перезагружаем с параметром step=1
 }
 else { //Это повторный вызов?
  $res=@$_COOKIE['my_cookie']=='ok'?
   'Yes':'No';

 setcookie ('my_cookie','',
  time()-14*24*3600);
 echo '<br>Cookie test: '.$res;
 }
?>
```

При проверке этого кода после включения/выключения в браузере поддержки cookies для некоторых обозревателей может понадобиться их перезапуск. Менее "красивый" альтернативный способ работы с SID – его передача через параметр URL-адреса. PHP имеет возможность автоматически добавлять SID ко всем ссылкам в генерируемых HTML страницах, поэтому вам, как правило, не нужно будет заботиться о том, чтобы добавлять этот идентификатор к каждой ссылке вручную. Если же вы по каким-либо причинам хотите сами передавать идентификатор сессии – вы всегда можете получить его из константы SID или из функции session_id(). Данный способ неудобен тем, что SID будет отображаться во всех URL-адресах приложения, что может создать, например, проблему с идентификацией URL-адресов поисковыми машинами. Отключить использование номера сессии в URL можно настройкой session.use_trans_sid = 0 в файле php.ini.На практике примерно у 99,5 % пользователей cookies включены и поддерживаются, поэтому данный способ можно считать не очень актуальным.

Для того чтобы иметь возможность использовать сессионные переменные в своей программе, необходимо сначала создать сессию: session_start(); Нужно вызывать эту функцию на каждой странице, где тре-буется использовать сессионные переменные. Для наглядности сессии можно задать имя с помощью функции session_name(имя_сессии). Делать это нужно еще до инициализации сессии. Получить имя текущей сессии можно с помощью этой же функции, вызванной без параметров: session_name().

Регистрация сессионных переменных производится путем вызова следующей функции:

session_register('var1','var2',...);

Зарегистрировать переменную также можно, просто зап-сав ее значение в ассоциативный массив $_SESSION:

$_SESSION['имя'] = 'значение';

В этом массиве хранятся и открыты для программного доступа все зарегистрированные (глобальные) переменные сессии. Получить идентификатор текущей сессии можно с помощью функции session_id(). Функция session_unregister(имя_переменной) удаляет указанную глобальную

переменную из текущей сессии. Для того чтобы сбросить значения всех переменных сессии,можно использовать функцию session_unset().

Уничтожить текущую сессию целиком можно функцией session_destroy(). Она не сбрасывает значения глобальных переменных сессии и не удаляет cookies, но уничтожает все данные, ассоциируемые с текущей сессией. Следует понимать, что использование механизма сессий не гарантирует полной безопасности системы. Во-первых, данные, передающиеся по открытому протоколу HTTP, могут быть перехвачены, во-вторых, так как переменные сессии глобальны, они сохраняются в зашифрованном виде в cookie-файлах на компьютере пользователя. Более надежным решением представляется хранение ценных данных, таких как логины и пароли, только в базе и с шифрованием паролей, а также использование в важных приложениях защищенных версий протоколов, таких как HTTPS. С учетом сказанного, добавим к возможностям примера 9 несложный механизм регистрации и авторизации пользователей с использованием сессий.

Часть кода, доступная только для авторизованных пользователей, может быть реализована в файле index.php и других модулях системы следующим образом:

```
$login1 = check_user();
if (!empty($login1)) {
  echo '<p>Вы вошли как '.$login1.
  '. <a href="logout.php">Выход</a></p>';
  //код для авторизованного пользователя
 }
 else {
  include "logform.php";
 }
```

Функция check_user, добавленная в модуль functions.php, просто проверяет, выполнялась ли авторизация, связанная с установкой определенной переменной сессии:

```
function check_user () {
 if (!isset($_SESSION['my_inside']))
  return '';
 else return $_SESSION['current_user'];
}
```

Включаемый модуль logform.php содержит код для вывода формы авторизации, отправки логина и пароля модулю login.php, непосредственно отвечающему за вход в систему, а также ссылки на модуль регистрации register.php:

```
<form method="post" action="login.php">
<table width=100% align=center border=0>
 <tr><td>Логин:</td>
 <td><input type=text size=32 maxlength=32
   name=login></td></tr>
 <tr><td>Пароль:</td>
 <td>
  <input type=password size=32 maxlength=32
   name=password></td></tr>
 <tr><td align=center>
  <a href="register.php">Регистрация</a></td>
  <td align=center>
   <input type=submit value="OK"></td></tr>
</table></form>
<?php /*другой код */ ?>
```

Модуль login.php сравнивает данные, полученные из формы входа, с логином и зашифрованным паролем, хранящимися в базе. При совпадении данных устанавливаются нужные переменные сессии и происходит перенаправление на главную страницу:

```
<?php
 $params = array('login','password');
 require_once ("functions.php");
 function login ($login1, $password1) {
  $sql = 'select login, password from '.
  'users where login="'.$login1.'" limit 0,1';
  $result = dbquery ($sql);
  if ($result and dbrows($result)>0) {
   $data = dbfetcha ($result);
   if ($data['password']==md5($password1)) {
    $_SESSION['my_inside']=1;
    $_SESSION['current_user']=$login1;
   }
  }
 }
 login ($login,$password);
 header('Location: index.php');
?>
```

Модуль logout.php, также вызываемый по ссылке из index.php, позволяет выйти из системы:

```
<?php
 require_once ("functions.php");
 $current_user=check_user();
 if (!empty($current_user)) {
  $_SESSION = array();
  session_destroy ();
 }
 header('Location: index.php');
?>
```

Наконец, модуль register.php предоставляет форму для регистрации, проверяет введенный логин на уникальность и добавляет новую запись о пользователе, если регистрация успешна:

```
<?php
 $params = array('login', 'password1',
 'password2');
 require_once ("functions.php");
 include "head.php";
 $error='';
 if (!empty($password1) and
      $password1!=$password2)
  $error.='Пароли не совпадают';
 if (!empty($login)) {
  $sql='select login from users '.
   'where login="'.$login.'" limit 0,1';
  $result=dbquery($sql);
  if ($result and dbrows($result)) {
   $data=dbfetcha($result);
   if ($data['login']==$login) {
    $error.='Такой логин уже есть';
   }
  }
 }

if (!empty($error)) {
 echo '<p>'.$error.
  ', <a href="register.php">попробуйте '.
  'еще раз</a></p>';
 include "foot.php";
 exit;
}
if (!empty($login) and !empty($password1)) {
 $sql='insert into users (login,password)'.
      ' values ("'.$login.'","'.
      md5($password1).'")';
 $result=dbquery($sql);
 if (!$result) { echo "Bad sql $sql"; }
 else {
  echo '<p>Регистрация успешна, войдите '.
   'в систему с <a href="index.php">'.
   'главной страницы</a></p>';
 }
}
else {
?>
 <form method="post">
 <table width=100% align=center border=0>
  <tr><td>Новый логин:</td>
<td><input type=text size=32 maxlength=32
     name=login></td></tr>
<tr><td>Новый пароль:</td>
<td><input type=password size=32 maxlength=32
     name=password1></td></tr>
<tr><td>Новый пароль еще раз:</td>
<td><input type=password size=32 maxlength=32
     name=password2></td></tr>
<tr><td colspan="2" align=center>
```

```
  <input type=submit value="OK"></td></tr>
</table></form>
<?php
 }
 include "foot.php";
?>
```

Предполагается, что включаемые файлы с именами head.php и foot.php содержат общие для всех файлов проекта дизайнерские "шапку" и "подвал" страницы. Рекомендуемые по итогам главы разделы стандартной справки: "Справочник языка" – "Предопределенные переменные" – "$_COOKIE", "Справочник функций" – "Функции для работы с сессиями".

Задание № 1. Изучить приведенный теоретический материал.

Задание № 2. Протестируйте приведенный пример.

Задание № 3. Написать программу сохранения персональных настроек пользователя (ник и фон страниц) с использованием функций управления сессией.

Задание № 4. Написать программу, которая применяет функции управления сессией для запоминания того, какие страницы уже посещались пользователем. Вывести список ссылок на все посещенные страницы.

Задание № 5. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 6. СОЗДАНИЕ ПРОЕКТА «РЕГИСТРАЦИЯ»

Цель: получение навыков регистрации и авторизации пользователей на сайте в PHP.

Теоретические вопросы

Понятие сессии.

Отслеживание сеансов.

Идентификация пользователей и процессов.

Понятие и назначение cookies-файлов.

Задание № 1. Добавьте к скрипту приведенного в лабораторной работе № 6 возможности регистрации и авторизации пользователя, обеспечьте возможность добавления сообщений только от зарегистрированных пользователей. Реализуйте авторизацию через cookie-файлы.

Задание № 2. Создать форму, с помощью которой пользователь может задать свой ник и выбрать цвет фона страниц сайта.

Задание № 3. Использовать cookie для того, чтобы приветствовать пользователя по имени на следующих страницах с заданным фоном.

Задание № 4. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 7. СОЗДАНИЕ ПРОЕКТА «ИНТЕРНЕТ-МАГАЗИН»

Цель: получение навыков создания проекта «Интернет-магазин» в PHP.

Теоретические вопросы

Понятие интернет-магазина.

Технологии создания интернет-магазина.

Задание № 1. Создать веб-сайт "Интернет-магазин". Необходимо создать элемент дизайна форму для заказа товара в интернет-магазине. Реализовать средствами языка написания сценариев PHP обработку заказов клиентов по каталогу. Для этого необходимо создать сценарий, который считывает информацию из формы для обработки. По результатам заказа на сайте необходимо вывести на экран содержимое корзины пользователя с рассчитанной суммой заказа. На сайте необходимо осуществить проверку вводимых пользователем данных в форму. При обработкезаказа следует использовать математические функции, а также различные условные операторы. Тематика:

1) книжный магазин научной литературы (заказ книг в определенной научной области);

2) магазин программных средств (заказ программ для ЭВМ);

3) магазин видео (покупка фильмов на DVD-дисках);

4) магазин компьютерных игр (заказ компьютерных игр на CD-дисках);

5) музыкальный магазин (заказ музыкальных компакт дисков);

6) цветочный магазин (заказ доставки цветов);

7) магазин парфюмерии и косметики (заказ косметики и парфюмерии по каталогу);

8) магазин одежды (заказ одежды по каталогу);

9) спортивный магазин (заказ спортивной формы и инвентаря);

10) магазин мебели (заказ мебели, элементов интерьера и товаров для дома).

При разработке веб-сайта связать между собой 10–15 веб-страниц. В исходном html-коде использовать комментарии каждого тега.

Задание № 2. Создать веб-сайт интернет-магазин. Реализовать средствами языка написания сценариев PHP обработку заказов клиентов по каталогу. Необходимо создать элемент дизайна форму и реализовать сохранение данных пользователя в текстовый файл. Организовать работу по открытию, просмотру и записи данных в файл. Помимо этого необходимо реализовать вывод содержимого файла на экран. Реализовать счетчик посещений. Обработку следует реализовать по тематике лабораторной работы № 3. При разработке веб-сайта связать между собой 10–15 веб-страниц. В исходном html-коде использовать комментарии каждого тега.

Задание № 3. Написать скрипт, позволяющий организовать интернет-магазин.

Список товаров хранится в базе данных на стороне сервера. Покупатель должен иметь возможность просмотреть все имеющиеся в наличии товары и сделать заказ. Покупатель должен иметь возможность сделать запрос, например, указав интервал цен, который его устраивает или какие-либо другие данные. До тех пор, пока покупатель выбирает отдельные товары, его заказ хранится на стороне клиента в виде cookie. После того как покупатель сформировал заказ, заказ отсылается на сторону сервера, где покупка товара учитывается в базе данных.

Варианты

1. В базе данных содержится информация о книгах: автор, название, изображение обложки, издательство, год выпуска, цена.

2. В базе данных содержится информация об автомобилях: модель, изображение автомобиля, год выпуска, тип кузова, мощность двигателя, цвет, цена.

3. В базе данных содержится информация о туристических поездках: страна, город, изображение городской достопримечательности, количество дней, дата поездки, класс отеля.

4. В базе данных содержится информация о журналах: название, изображение обложки, год выпуска, номер, издательство, число страниц, цена.

5. В базе данных содержится информация о местах в отеле: название отеля, класс номера, изображение номера, количество мест в номере, цена.

Задание № 4. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 8. СОСТАВЛЕНИЕ СХЕМ XML-ДОКУМЕНТОВ

Цель: ознакомиться с XML, научиться разбирать структуру XML-документа.

Теоретические вопросы

Понятие XML.

Правила синтаксиса XML– документа.

Структура XML-документа.

XML (Extensible Markup Language) – это язык разметки, описывающий целый класс объектов данных, называемых XML– документами. Этот язык используется в качестве средства для описания грамматики других языков и контроля за правильностью составления документов, т.е. сам по себе XML не содержит никаких тэгов, предназначенных для разметки, он просто определяет порядок их создания.

Сам процесс создания XML документа очень прост и требует от нас лишь базовых знаний HTML и понимания тех задач, которые мы хотим выполнить, используя XML в качестве языка разметки. Таким образом, у разработчиков появляется уникальная возможность определять собственные команды, позволяющие им наиболее эффективно определять данные, содержащиеся в документе. Автор документа создает его структуру, строит необходимые связи между элементами, используя те команды, которые удовлетворяют его требованиям, и добивается такого типа разметки, которое необходимо ему для выполнения операций просмотра, поиска, анализа документа.

XML позволяет осуществлять контроль за корректностью данных, хранящихся в документах, производить проверки иерархических соотношений внутри документа и устанавливать единый стандарт на структуру документов, содержимым которых могут быть самые различные данные. Это означает, что его можно использовать при построении сложных информационных систем, в которых очень важным является вопрос обмена информацией между различными приложениями, работающими в одной системе. Создавая структуру механизма обмена информации в самом начале работы над проектом, менеджер может избавить себя в будущем от многих проблем,связанных с несовместимостью используемых различными компонентами системы форматов данных.

Задание № 1. Создайте XML-документ.

index.xml

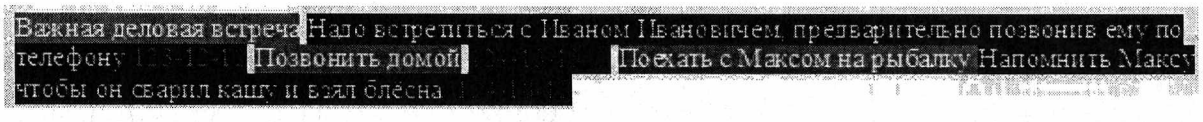
```
<?xml version="1.0" encoding="windows-1251" ?>
<notepad>
<note id="1" date="11/04/99" time="13:30">
<subject>Важная деловая встреча</subject>
<importance/>
<text>
        Надо встретиться с <person id="1625">Иваном

Ивановичем</person>, предварительно
        позвонив ему по телефону <tel>123-12-12</tel>
</text>
</note>
<note id="2" date="12/04/99" time="13:00">
<subject>Позвонить домой</subject>
<text>
<tel>124-13-13</tel>
</text>
</note>
...
<note id="3" date="13/04/99" time="5:00">
<subject>Поехать с Максом на рыбалку</subject>
<text>
        Напомнить Максу чтобы он сварил кашу и взял блёсна.
<tel>124-10-13</tel>
</text>
</note>

</notepad>
```

Создайте файл содержащий стиль оформления 1.css.

1.css

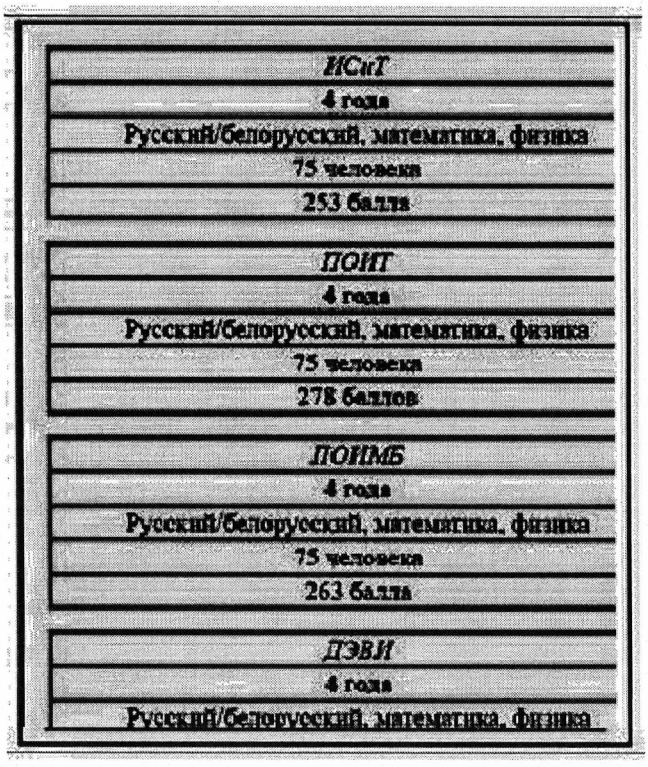
```
text { font-family: Verdana, Arial, Helvetica, sans-serif; font-size:
12px; font-style: normal; color: #FFFFFF; background-color: #202036}

subject { font-family: Arial;font-size: 12px; font-style: normal;
color: white; background-color: gray}

tel {color: yellow}
```

Результат



Задание № 2. Создайть XML-документ, который будет содержать информацию по вашей специальности в других университетах (университет, проходной балл, план набора, город, в котором размещен университет). При выполнении задания используйте css.

Задание № 3. Создайте XML-документ с подключением css в соответствии с рисунком.



Задание № 4. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 9. ОТОБРАЖЕНИЕ XML-ДОКУМЕНТОВ РАЗЛИЧНЫМИ СПОСОБАМИ

Цель: ознакомиться со способами отображения XML-документов.

Теоретические вопросы

Понятие XSL-таблицы.

Отличия CSS и XSL.

Отображение XML-документа с помощью XSL.

Отображение XML-документа с помощью связывания данных.

Отображение XML-документа с помощью DOM.

Отображение XML -документ с помощью XSL

XSL является приложением XML, т.е. XSL-таблица представляет собой корректно сформированный XML-документ, который отвечает правилам XSL. Подобно любому XML-документу, XSL-таблица стилей содержит простой текст, и вы можете создать ее с помощью любого текстового редактора.

Можно связать XSL-таблицу стилей с XML-документом, включив в документ инструкцию по обработке xml-stylesheet, которая имеет следующую обобщенную форму записи:

<?xml-stylesheet type="text/xsl" href="3.xslt"?>

Таблица стилей при этом должна размещаться на том же домене, что и XML-документ, с которым вы ее связываете.

Если вы не связали XML-документ ни с CSS-таблицей, ни с XSL-таблицей стилей, браузер отобразит документ с помощью встроенной XSL-таблицы, которая используется по умолчанию. Эта таблица стилей отображает исходный XML-текст в виде дерева с возможностью свертывания/развертывания уровней.

В отличие от CSS, содержащей правила, XSL-таблица стилей включает один или несколько шаблонов, каждый из которых содержит информацию для отображения в определенной ветви элементов в XML-документе.

Каждая XSL-таблица стилей должна иметь элемент Документ, известный как корневой элемент, является XML-элементом верхнего уровня, который содержит все остальные элементы.

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheetversion="1.0" xmlns:xsl="http://www.w3.org/XSL/Transform">

Элемент Документ xsl:stylesheet служит не только хранилищем других элементов, но также идентифицирует документ как XSL-таблицу стилей. Все XSL-элементы принадлежат пространству имен xsl, т.е. вы предваряете имя каждого XSL-элемента префиксом xsl:, обозначающим пространство имен.

Элемент Документ xsl:stylesheet XSL-таблицы стилей должен содержать один или несколько шаблонов элементов, которые для краткости будем называть шаблонами.

<xsl:template match="/">

<!— дочерние элементы … -->

</xsl:template>

Браузер использует шаблон для отображения определенной ветви элементов в иерархии XML-документа, с которым вы связываете таблицу стилей. Атрибут match шаблона указывает на определенную ветвь.

Значение атрибута match носит название образца (pattern). Образец в данном примере ("/") представляет корневой элемент всего XML-документа.

Каждая XSL-таблица стилей должна содержать один и только один шаблон с атрибутом match, который имеет значение "/".

Корневой образец ("/") не представляет элемент Документ (или корневой элемент) XML-документа. Он представляет весь документ, для которого элемент Документ является дочерним (т.е. он аналогичен корневому узлу Document в объектной модели документа DOM)

Пример использования XSL-таблицы стилей для предоставления информации в виде таблицы:

<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">

<html>

<body>

<h2> Список специальностей факультета ИТ</h2>

<table border="1">

<tr bgcolor="#9acd32">

<th style="text-align:center">Специальность</th>

<th style="text-align:center">Срок обучения</th>

<th style="text-align:center">Предметы ЦТ</th>

<th style="text-align:center">План набора</th>

<th style="text-align:center">Проходной балл</th>

</tr>

<xsl:for-each select="FACULTY/SPECIALIZATION">

<tr>

```
<td><xsl:value-of select="NAME"></td>
<td><xsl:value-of select="TIME"></td>
<td><xsl:value-of select="EXAM"></td>
<td><xsl:value-of select="PAGES"></td>
<td><xsl:value-of select="PASSING"></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

Оператор пути в значении атрибута select относится к текущемуэлементу. Каждый контекст внутри XSL-таблицы стилей относится к текущему элементу. Если вы опустите атрибут select для XSL-элемента value-of, элемент будет осуществлять вывод текстового содержимого плюстекстовое содержимое всех дочерних элементов в текущий элемент.

Порядок элементов value-of в шаблоне определяет порядок, в котором браузер отображает эти элементы. Таким образом, даже из этой простой таблицы стилей вы можете понять, что XSL-таблица стилей является гораздо более гибкой, чем CSS, которая всегда отображает элементы в том порядке, в котором они следуют в документе.

Элемент for-each выполняет две основные задачи:

– осуществляет вывод блока элементов, содержащихся внутри элемента for-each, повторяя его для каждого XML-элемента в документе, отвечающего образцу, присвоенному атрибуту select элемента for-each;

– внутри элемента for-each задает текущий элемент, устанавливаемый атрибутом select элемента for-each.

Не нужно включать в XSL-шаблон элементы, представляющие элементы HTML или BODY, которые являются стандартными составными частями HTML-страницы, поскольку браузер сам эффективно их формирует.

Каждый из элементов, представляющих HTML-разметку, должен быть корректно сформированным XML-элементом, а также стандартным HTML-элементом. Не забывайте, что XSL-таблица стилей является XML-документом.

Задание № 1. Оформите задание лабораторной работы № 8 через подключение XSL.

Отображение XML-документа с помощью связывания данных

Метод связывания данных требует создания HTML-страницы, связывания с ней XML-документа и установления взаимодействий стандартных HTML-элементов на странице, таких как SPAN или TABLE, с элементами XML. В дальнейшем HTML-элементы автоматически отображают информацию из связанных с ними XML-элементов.

Два основных этапа при связывании данных:

1. Установка связи XML-документа с HTML-страницей, на которой вы хотите отобразить данные XML. Этот шаг обычно реализуется включением HTML элемента с именем XML в HTML-страницу. Например, следующий элемент на HTML-странице связывает XML-документ Book.xml со страницей:

<XML ID="dsoBook" SRC="Book.xml"></XML>

2. Сцепление HTML-элементов с XML-элементами. Когда вы сцепляете HTML-элементы с XML-элементом, HTML-элемент автоматически отображает содержимое XML-элемента. Например, следующий элемент SPAN на HTML-странице сцеплен с элементом AUTHOR связанного XML-документа:

<SPAN DATASRC="#dsoBook" DATAFLD="AUTHOR"></SPAN>

В результате HTML-элемент SPAN отображает содержимое XML-элемента AUTHOR.

Чтобы отобразить XML-документ на HTML-странице, вы должны установить его связь со страницей. Самый простой путь сделать это в Microsoft Internet Explorer 5 – включить в страницу HTML-элемент с именем XML, так называемый фрагмент данных. Можно использовать одну из двух различных форм записи для фрагмента данных.

В первой форме весь текст XML-документа помещается между начальным и конечным тегами XML.

Во второй форме записи HTML-элемент с именем XML остается пустым и содержит только URL XML-документа.

Когда браузер открывает HTML-страницу, его встроенный XML-процессор синтаксически анализирует XML-документ. Браузер также создает программный объект, который носит название Объект исходных данных (Data Source Object DSO), который хранит данные XML и обеспечивает доступ к этим данным. DSO хранит данные XML как набор записей, т.е. множество записей и их полей.

Когда вы сцепляете HTML-элемент с XML-элементом, DSO автоматически предоставляет значение XML-элемента и управляет всеми его свойствами. DSO также позволяет вам напрямую осуществлять доступ и манипулирование имеющимся набором записей посредством ряда методов, свойств и событий.

Если вы открываете XML-документ через фрагмент данных на HTML-странице, Internet Explorer 5 проверяет, является ли документ корректно сформированным, а также – если документ включает объявление типа документа – является ли он валидным. Однако в том случае, если документ содержит ошибку, Internet Explorer 5 просто не будет отображать данные XML, не выводя сообщение об ошибке.

Вы можете осуществлять сцепление HTML-элементов с XML-элементами двумя основными способами.

Табличное сцепление, что означает сцепление HTML-элемента TABLE с данными XML, так что в таблице автоматически отображается весь набор записей, принадлежащих XML-документу.

<TABLE DATASRC="#dsoInventory" BORDER="1" CELLPADDING="2">

<THEAD> <TH>Title</TH> <TH>Author</TH> </THEAD> <TR ALIGN="center"> <TD><SPAN DATAFLD="TITLE" STYLE="font-style:italic"></SPAN></TD> <TD><SPAN DATAFLD="AUTHOR"></SPAN></TD> </TR> </TABLE>

Если XML-документ содержит много записей, можно воспользоваться постраничный вывод, для этого необходимо:

Установите максимальное число записей, которое будет выводиться на странице с помощью атрибута DATAPAGESIZE элемента TABLE.

Присвойте уникальный идентификатор атрибуту ID элемента TABLE.

<TABLE ID="InventoryTable" DATASRC="#dsoInventory" DATAPAGESIZE="5">

Для перемещением между записями используются методы элемента TABLE такие как FirstPage, previousPage, nextPage, LastPage.

Для отображения иерархической структуры записей можно использовать вложенные таблицы. Например, разметка для вложенной таблицы может выглядеть следующим образом:

<TABLE DATASRC="#dsoInventory" DATAFLD="BOOK" BORDER=0 CELLSPACING=10>

Сцепление по отдельным записям , что означает сцепление не табличных элементов HTML (например, элементов SPAN) с XML-элементами таким образом, что за один раз отображается только одна запись.

<SPAN STYLE="font-weight:bold" DATASRC="#dsoBook" DATAFLD="TITLE"></SPAN>

Нужно учитывать, что HTML-элемент может отобразить за раз только одну запись DSO(объект исходных данных), ассоциированную с XML-документом. Для доступа к другим записям нужно воспользоваться методами для перемещения между записями moveFirst, movePrevious, moveNext, moveLast, move, принадлежащими объекту recordset DSO.<BUTTON ONCLICK="dsoInventory.recordset.moveFirst()"> |< First </BUTTON>Кроме представленных существует еще ряд других способов для связывания не табличных HTML-элементов. Это могут быть как индивидуальные HTML-элементы, используемые для связывания данных по одной записи, так и HTML-элементы, содержащиеся в сцепленной таблице HTML. Например:<A DATASRC="dsoInventory" DATAFLD="REVIEWS"> reviews</A>

Если необходимо отобразить атрибут элемента XML-документа, то следует учитывать, что DSO и элемент, и атрибут будет хранить как вложенные записи. Следовательно, набор записей превратится в иерархический набор, и для отображения вложенных записей необходимо будет воспользоваться вложенной таблицей. Чтобы иметь возможность отобразить как символьные данные, так и атрибут как вложенную запись, следует иметь в виду то обстоятельство, что DSO использует специальное имя $TEXT для обращения ко всем символьным данным элемента, не включая при этом значений атрибута, имя поля для которого будет совпадать с именем атрибута. Вы можете использовать имя $TEXT в качестве имени поля, чтобы связать ячейку таблицы с символьными данными, содержащимися в записи элемента. Например:

<TABLE DATASRC="#dsoInventory"DATAFLD="AUTHOR"> <TR> <TD><SPAN DATAFLD="$TEXT"></SPAN></TD> <TD><SPAN DATAFLD="Born"></SPAN></TD> </TR></TABLE>

Задание № 2. Установить взаимодействие стандартных HTML-элементов на странице, таких как SPAN и TABLE, с элементами XML.

Отображение XML-документа с помощью DOM

При написании сценария вы создаете HTML-страницу, связываете ее с XML-документом и имеете доступ к индивидуальным XML-элементам с помощью специально написанного кода сценария (JavaScript или Microsoft Visual Basic Scripting Edition [VBScript]). Браузер воспринимает XML-документ как объектную модель документа (Document Object Model – DOM), состоящую из большого набора объектов, свойств и команд. Написанный код позволяет осуществлять доступ, отображение и манипулирование XML-элементами.

В браузерах находятся встроенные библиотеки DOM. Для сценариев на стороне клиента доступно множество объектов для работы с XML-документом, самые важные из них, объекты XMLDOMDocument, XMLDOMNode, XMLDOMNodeList, XMLDOMParseError, представляющие интерфейс для доступа ко всему документу, отдельным его узлам и поддеревьям,

предоставляющие необходимую для отладки информацию о произошедших ошибках анализатора, соответственно.

Объект XMLDOMNode, реализующий базовый DOM интерфейс Node, предназначен для манипулирования с отдельным узлом дерева документа. Его свойства и методы позволяют получать и изменять полную информацию о текущем узле – его тип (dataType, nodeType, nodeTypeString), название (baseName, prefix, nodeName), его содержимое (attributes ,text,nodeValue, childNodes) и т.д.

При выполнении данной лабораторной работы могут быть полезны следующие свойства:

nodeName – Возвращает полное название (вместе с Namespace атрибутом) текущего узла в виде строки. Доступно только для чтения.

baseName – Возвращает название элемента без префикса Namespace. Только для чтения.

prefix – Возвращает Namespace префикс. Только для чтения.

dataType – Определяет тип содержимого текущего узла (описываемый схемами данных). Доступно для записи и чтения.

nodeType – Возвращает тип текущего узла. Только для чтения.

nodeTypeString – Возвращает тип узла в виде текста. Только для чтения.

attributes – Возвращает список атрибутов текущего узла в виде коллекции XMLDOMNamedNodeMap. Если атрибутов нет, то свойство length будет содержать нулевое значение. Для тех узлов, у которых не может быть атрибутов, возвращается null. Доступно только для чтения.

nodeValue – Возвращает содержимое текущего узла. Доступно для чтения и записи.

childNodes – Для тех узлов, которые имеют дочерние элементы, возвращает их список в виде XMLDOMNodeList. В том случае, если дочерних элементов нет, значение свойства length списка равно нулю. Только для чтения.

lastChild – Возвращает последний дочерний элемент или null, если таковых не имеется. Свойство доступно только для чтения.

firstChild – Возвращает первый дочерний элемент или null. Только для чтения.

nextSibling – Возвращает следующий дочерний элемент. Только для чтения.

previousSibling – Возвращает предыдущий дочерний элемент. Доступно только для чтения.

parentNode – Содержит ссылку на родительский элемент. В том случае, когда такого элемента нет, возвращает null. Доступно только для чтения.

Объект XMLDOMDocument представляет верхний уровень объектной иерархии и содержит методы для работы с документом: его загрузки (readyState, load(url), loadXML(xmlString), save(objTarget), abort() и т.д.), создания в нем элементов (createElement(tagName)), атрибутов (createAttribute (name)), комментариев (createComment(data)) и т.д. Многие свойства и методыэтого объекта реализованы также в рассмотренном выше классе Node, т.к. документ может быть рассмотрен как корневой узел с вложенными в него поддеревьями.

Объект XMLDOMNodeList представляет собой список узлов – поддеревья и содержит методы, при помощи которых можно организовать процедуру обхода дерева. Например:

length – число элементов списка узлов;

item(i) – выбор i-того элемента из списка. Возвращает объект XMLDOMNode;

nextNode() – выбор следующего элемента в списке, если такого элемента нет, то возвращает null. Первый вызов этого метода возвратит ссылку на первый элемент списка;

reset() – сброс внутреннего указателя текущего элемента.

Объект XMLDOMParserError позволяет получить всю необходимую информацию об ошибке, произошедшей в ходе разбора документа. Все свойства этого объекта доступны только для чтения. Основные свойства:

errorCode – содержит код возникшей ошибки либо 0, если таковой не случилось;

url – возвращает URL обрабатываемого документа;

filepos – возвращает смещение относительно начала файла фрагмента, в котором обнаружена ошибка;

line – содержит номер строки, содержащей ошибку;

linepos – позицию ошибки в строке, в которой была обнаружена ошибка;

reason – описание ошибки;

srcText – содержит полный текст строки, в которой произошла ошибка;

Задание № 3. Написать кода сценария (Microsoft Visual Basic Scripting Edition [VBScript]) для доступа к индивидуальным XML-элементам.

Задание № 4. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 10. РАЗРАБОТКА WEB-ПРИЛОЖЕНИЯ С ПОМОЩЬЮ XML

Цель: ознакомиться со способами разработки Web-приложений с помощью XML.

Теоретические вопросы

PHP запись в XML

Чтение и получение данных из XML.

Отображение XML-документа с помощью XSL.

Отображение XML-документа с помощью связывания данных.

Отображение XML-документа с помощью DOM.

Задание № 1. PHP запись в XML. Инструмент XMLWriter был создан специально для записи в XML формате. Один из важных его методов это startDocument(), который позволяет задать кодировку версии XML. Создайте XML следующего вида.

```
1 <?xml version="1.0"?>
2 <Customer>
3 <id>1</id>
4 <name>Марк</name>
5 <address>Санкт-Петербург</address>
6 </Customer>
```

Получить такой формат можно с помощью XMLWriter следующим образом:

```
<?php
//Простой пример
$xml = new XMLWriter(); //создаем новый экземпляр класса XMLWriter
$xml->openMemory(); //использование памяти для вывода строки
$xml->startDocument(); //установка версии XML в первом теге документа
$xml->startElement("Customer"); //создание корневого узла
```

```
$xml->writeElement("id", "1");
$xml->writeElement("name", "Марк"); //запись элемента
$xml->writeElement("address", "Санкт-Петербург");
$xml->endElement(); //закрытие корневого элемента
echo $xml->outputMemory(); //завершение записи в XML
?>
```

Рассмотри пример записи в XML. Например, задача поставлена – получить файл XML в следующем формате:

```
<xml version="1.0"?>
<purchase>
<customer>
<id>1</id>
<time>2013-04-19 10:56:03</time>
<total>$350</total>
</customer>
<customer>
<id>2</id>
<time>2013-04-23 13:43:41</time>
<total>$1456</total>
</customer>
</purchase>
```

Получить такой формат можно с помощью XMLWriter следующим образом:

```
<?php
$xml = new XMLWriter();
$xml->openMemory();
$xml->startDocument();
$xml->startElement("purchase");
$xml->startElement("customer"); //открытие элемента первого покупателя с ID = 1
$xml->writeElement("id", 1);
$xml->writeElement("time", "2013-04-19 10:56:03");
$xml->writeElement("total", "$350");
$xml->endElement(); //закрытие элемента первого покупателя с ID = 1
$xml->startElement("customer"); //Открытие элемента второго покупателя с ID = 2
$xml->writeElement("id", 2);
$xml->writeElement("time", "2013-04-23 13:43:41");
$xml->writeElement("total", "$1456");
$xml->endElement(); //закрытие элемента первого покупателя с ID = 2
$xml->endElement();
echo $xml->outputMemory();
?>
```

Задание № 2. Получить файл XML в следующем формате:

```
<?xml version="1.0"?>
<products>
<product pid="314">
<name>Яблоко</name>
<price>$1.00</price>
<discount>3%</discount>
</product>
<product pid="315">
<name>Манго</name>
<price>$0.90</price>
<discount>3%</discount>
</product>
</products>
```

Задание № 3. Чтение и получение данных из XML. Ниже будет приведен пример чтения и получения данных из XML с использованием классов XMLReader и SimpleXMLElement. Читать будем уже имеющийся XML объект, который мы создали в примере №1 при PHP записи в XML. XMLReader используется для получения заданного узла из XML.

```
// Чтение XML формата
$rxml = new XMLReader(); //Создание элемента для чтения
$rxml->xml($nXML); //Загрузка XML, $nXML – строка в формате XML
//Переместиться к первому элементу customer
while($rxml->read() && $rxml->name !== 'customer');
$amountSpent = 0;
//Получим значение поля total у второго узла дерева
while($rxml->name === 'customer'){
 //Чтение текущего дочернего через SimpleXMLElement
 $node = new SimpleXMLElement($rxml->readOuterXML());
 //Проверяем, номер элемента, если он равен 2 то это искомый элемент
 if($node->id == 2){
 $amountSpent = $node->total;
 break;
 }
 //Переместиться к следующему элементу customer
 $rxml->next('customer');
}
echo $amountSpent;
```

Задание № 4. Добавление узлов дерева в существующий XML. Используя XMLWriter и SimpleXMLElement, добавим узел дерева в существующий XML:

```
//Добавим новый узел в имеющийся XML
$sXML = new SimpleXMLElement($nXML2); // загрузка в XML
$newchild = $sXML->addChild("product");
//Добавление параметров записи
$newchild->addAttribute("pid", 328);
$newchild->addChild("name", "Банан");
$newchild->addChild("price", "$3.00");
$newchild->addChild("discount", "0.3%");
```

```
echo $sXML->asXML();
```

Задание № 5. Перезапись элементов существующего XML. Перезапись существующего узла дерева XML может быть реализована следующим образом:

```
/**
 * Перезапись узлов дерева XML формата.
 */

$productId = 314;
$parent = new DomDocument;
// создаем новый элемент дома product
$parent_node = $parent->createElement('product');

//Добавляем атрибут
$attribute = $parent->createAttribute("pid");
//устанавливаем значение
$attribute->value = $productId;
$parent_node->appendChild($attribute);

// Добавляем дочерний элементы
$parent_node->appendChild($parent->createElement('name', "Яблоко"));
$parent_node->appendChild($parent->createElement('price', "$2.00"));
$parent_node->appendChild($parent->createElement('discount', "1%"));
//Вставляем созданные элементы в создаваемый 'product'
$parent->appendChild($parent_node);
// Загружаем оригинальный XML формат
$dom = new DomDocument;
$dom->loadXML($nXML);
// Находим имеющийся элемент с pid = 314
$xpath = new DOMXpath($dom);
$nodelist = $xpath->query("/products/product[@pid={$productId}]");

$oldnode = $nodelist->item(0);

// Импортируем созданый ранее элемент в текущее дерево
$newnode = $dom->importNode($parent->documentElement, true);
// заменяем страрый элемент на новый
$oldnode->parentNode->replaceChild($newnode, $oldnode);
// сохраняем XML
echo $dom->saveXML();
```

Задание № 6. Удаление элементов XML. Пример удаления одного из продуктов:

```
 //Удаление продукта Манго
//загрузим оригинальный формат XML
```

```
$productId = 315;
$dom = new DomDocument;
$dom->loadXML($nXML2);
// Найдем элемент который необходимо удалить
$xpath = new DOMXpath($dom);
$nodelist = $xpath->query("/products/product[@pid={$productId}]");
$oldnode = $nodelist->item(0);
// Удаляем элемент
$oldnode->parentNode->removeChild($oldnode);
echo $dom->saveXML();
```

Задание № 7. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 11. ИСПОЛЬЗОВАНИЕ ЯЗЫКА СЦЕНАРИЕВ JAVASCRIPT ПРИ СОЗДАНИИ WEB-САЙТА

Цель: ознакомиться с приёмами разработки сценариев на языке JavaScript.

Теоретические вопросы

Переменные JavaScript.

Операторы JavaScript.

Обработка строк.

Обработка массивов.

Задание № 1. Добавьте в пример три страницы. Две страницы должны отображать информацию о магазинах *Посуда* и *Мебель*. Третья страница – главная в сайте сети магазинов *ВСЁДЛЯ ДОМА*. На ней должны быть ссылки на страницы магазинов, входящих в сеть:

```
html><!— СКРИПТ загружается из файла primJs.js--> <HEAD>
<TITLE>Сеть</title>
</head>
<body>
<SCRIPT language="JavaScript" src="PrimJs.js">
</script>
<H2 align=center style="color:green">Магазин "ПОДАРКИ"</h2>
Адрес: Лесная ул., д.2<P>
Транспорт: трамваи 7, 23, автобусы 56, 93
</body>
</html>
// Файл primJs.js
a="background-color:#00ffff; color:#ff00ff;"
a+="font-size:24pt; font-family:'Times New Roman'"
naim='Сеть магазинов "ВСЁ ДЛЯ ДОМА"'
var da=new Date()
d=da.getDate()+"."+(da.getMonth()+1)+"."+da.getFullYear()
document.write('<P      align=center      style= "'+a+'">'+
```

naim+'</p><P>Сегодня '+d+'</p>')

Задание № 2. Создайте страницу с изображением и подписью под ним. При щелчке поподписи, она должна менять свой цвет. Щелчок по изображению должен вызывать замену изображения и подписи. Функция для обработки события должна вызываться из родительского по отношению к изображению и подписи объекта.

Задание № 3. Создайте сайт из двух страниц. Первая страница имеет заголовок Заказ мебели. На ней расположены два поля со списками (теги <SELECT>), поле (<INPUT>) и кнопка (<SUBMIT>). Из первого поля со списком пользователь выбирает изделие (шкаф, стол, сервант и т.д.). Из второго поля со спискомпользователь выбирает материал (дуб,орех, бук). В третье поле нужно ввести количество заказываемых изделий. После ввода данных необходимо проверить, все ли данные введены. Если обнаружена ошибка, то нужно вывести сообщение и предложить её исправить. Правильно введённые данные нужно отправить на веб-сервер. Вторая страница содержит написанный на PHP скрипт, с помощью которого формируется следующее сообщение:

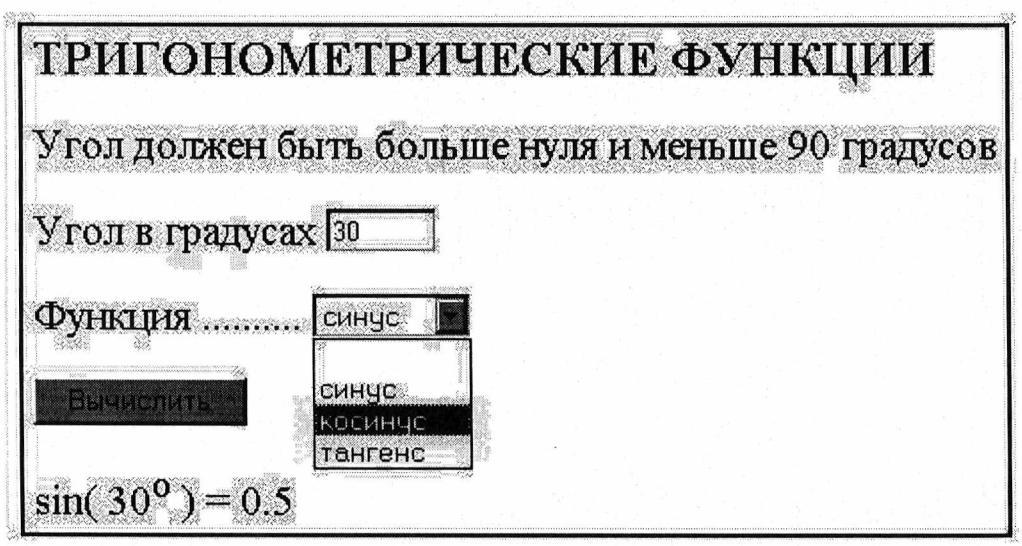
Ваш заказ принят

Заказано изделие – название заказанного изделия

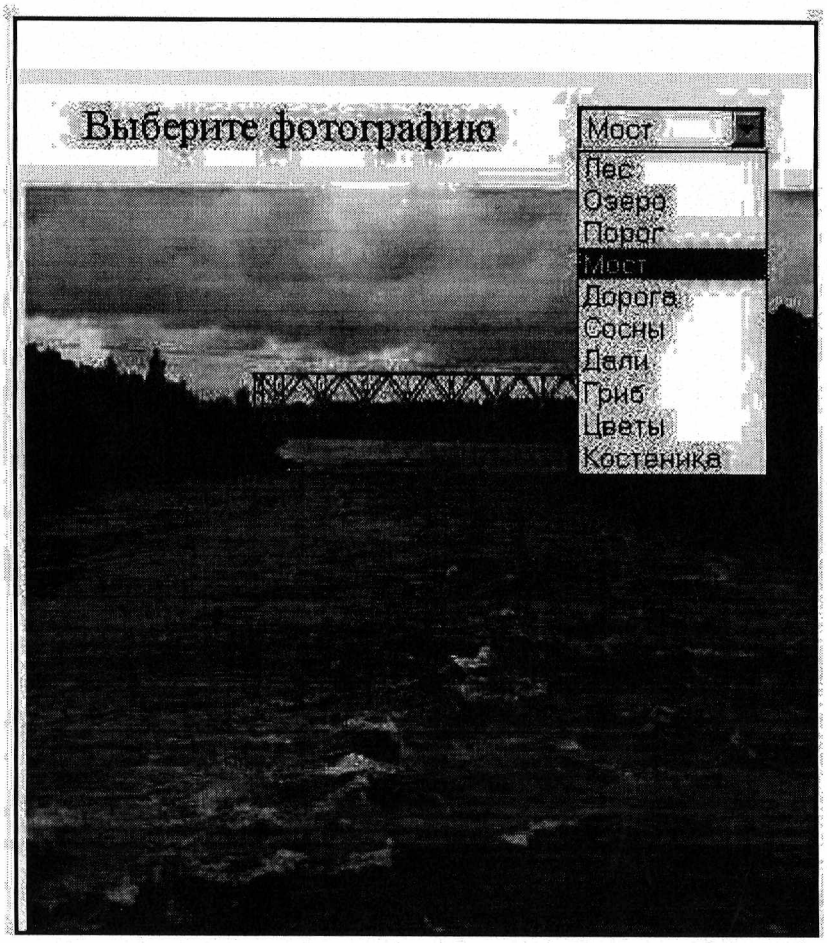
Материал – заказанный материал

Количество – заказанное количество

Задание № 4. Создайте страницу для вычисления тригонометрических функций. Вводимые пользователем данные должны проверяться немедленно после ввода и после нажатия кнопки *Вычислить*:



Задание № 5. Создайте страницу, на которой пользователь может просматривать фотографии, выбирая их названия из поля со списком (тег <SELECT>):



Задание № 6. Создайте страницу, на которой строится эллипс с задаваемыми пользователем размерами большой и малой полуосей:

Задание № 7. Напишите сценарий перемещения цветного квадрата по кругу. Траекторию удобно описывать параметрическими уравнениями:

y=R*sin(t) ,

где: R– радиус круга, 0 <= t <= 2

Квадратом может служить контейнер <DIV> … с цветным фоном.

Задание № 8. Напишите сценарий, который определяет имя браузера. Если загрузка произошла в любом браузере кроме Internet Explorer, то пользователь должен быть предупреждёно том, что страница правильно отображается только в браузере Internet Explorer. Поместите написанный сценарий в отдельный файл с расширением js. Скопируйте в свой каталог пример, который правильно отображается только в браузере Internet Explorer. Вставьте в скопированный

пример тег <SCRIPT ... > для загрузки файла со скриптом. Проверьте пример в браузерах Internet Explorer и Mozilla.

Задание № 9. Создайте сайт, состоящий из двух страниц. Сайт служит для вывода таблицы значений тригонометрической функции (sin, cos или tg) в заданном диапазоне и с заданным шагом. На первой странице пользователь задаёт исходные данные, а на второй получает соответствующую таблицу. Окно с новой страницей должно открываться методом open(). Исходные данные должны проверяться сразу после ввода и после нажатия кнопки Вычислить. Таблица должна иметь следующий вид:

Угол		sin
в градусах	в радианах	
30	0.5236	0.5
32	0.5585	0.5299
34	0.5934	0.5592
36	0.6283	0.5878

Задание № 10. Создайте страницу для учёта поступления товаров. Пользователь может менять в таблице количество и цену выбранного товара, вводя новые значения в поля, расположенные под таблицей. Введёнными значениями заменяются соответствующие данные в таблице и автоматически подсчитывается суммарная стоимость всех товаров:

Задание № 11. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 12. ПРИМЕНЕНИЕ ТЕХНОЛОГИИ AJAX

Цель: ознакомиться с технологией AJAX.

Теоретические вопросы

Назначение AJAX.

Особенности AJAX-приложений.

Объект XMLHttpRequest

Создание Ajax-приложения

AJAX (Asynchronous JavaScript and XML) – это концепция использования нескольких смежных технологий, ориентированная на разработку высокоинтерактивных приложения, быстро

реагирующих на действия пользователя, выполняющих большую часть работы на стороне клиента и взаимодействующих с сервером посредством внеполосных обращений.

AJAX применяется для разработки веб-приложений, к которым предъявляются следующие требования:

Приложение должно передавать пользователям свежие данные, полученные с сервера.

Новые данные должны интегрироваться в существующуюстраницу без ее полного обновления.

Для работы с такими приложениями в браузере необходимо, чтобы он соответствовал требованиям:

– поддержка посредников (для внеполосных вызовов HTTP). Обычно реализуется в форме объекта XmlHttpRequest%

– поддержка обновляемой модели DOM.

Объект XmlHttpRequest представляет собой компактную объектную модель для отправки сценарием обращений HTTP в обход браузера. Клиентский код сценария не может влиять на процесс размещения запроса и результат отправки запроса. XmlHttpRequest позволяет сценарию отправлять HTTP запросы и обрабатывать полученные ответы.

Задание № 1. Изучите пример системы, имитирующей работу сервиса GoogleSuggest на основе AJAX. Предполагается, что пользователь может вводить в текстовое поле формы название автомобильной марки, получая при этом динамически список вариантов названий, соответствующих уже введенным символам, без перезагрузки страницы. Начнем с веб-страницы:

```
<html>
<head>
<script src="chint.js"></script>
</head>
<body>
<form>
First Name:
<input type="text" id="txt1" onkeyup="showHint(this.value)">
</form>
<p>Suggestions: <span id="txtHint"></span></p>
</body>
</html>
```

Как видно из кода, при наступлении события onkeyup (отжатие клавиши) вызывается обработчик showHint().В файле chint.js имеется следующий код обработчика:

```
var xmlHttp;
function showHint(str)
{
if (str.length==0)
 {
 document.getElementById("txtHint").innerHTML="";
 return;
 }
xmlHttp=GetXmlHttpObject();
if (xmlHttp==null)
 {
 alert ("Your browser does not support AJAX!");
 return;
 }
var url = "gethint.php";
url = url + "?q=" + str;
url = url + "&sid=" + Math.random();
xmlHttp.onreadystatechange = stateChanged;
xmlHttp.open("GET", url, true);
xmlHttp.send(null);
}
```

```
function GetXmlHttpObject()
{
var xmlHttp=null;
try
 {
 // Firefox, Opera 8.0+, Safari
 xmlHttp = new XMLHttpRequest();
 }
catch (e)
 {
 // Internet Explorer
 try
  {
  xmlHttp=new ActiveXObject("Msxml2.XMLHTTP");
  }
 catch (e)
  {
  xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
 }
return xmlHttp;
}
```

Из кода видно, что каждый раз, когда вводится символ, вызываетсяфункция-обработчик. Если при этом содержимое текстового поляформы непустое (str.length > 0), функция выполняет следующие действия:

Формируется url для отправки веб-серверу.

Добавляется значение параметра q, равное содержимомутекстового поля, к url.

Добавляется к url случайное число для предотвращениякеширования.

Создается объект XMLHTTP, при этом указывается функция (stateChanged) подлежащая исполнению при наступлениисобытия ввода символа.

Открывается объект XMLHTTP с указанным значением url.

Отправляется HTTP запрос веб-серверу.

Если поле ввода пустое, происходит очистка содержимого раздела txtHint на веб-странице.

Ключевым моментом в данной системе является использование объекта XMLHttpRequest. Данный объект по-разному создается в различных браузерах. Так, Internet Explorer для этого использует ActiveXObject, в то время как остальные браузеры используют встроенный в JavaScript объект XML Http Request.

Для поддержки работы системы в разных браузерах использован оператор "try-catch". Сначала делается попытка создать объект XMLHttpRequest для браузеров Firefox, Opera или Safari:xmlHttp = new XMLHttpRequest().В случае неудачи, делается следующая попытка создания объекта для Internet Explorer: xmlHttp = new ActiveXObject("Msxml2.XMLHTTP"). Если это также не удается, то делается попытка создания объектауже для Internet Explorer: xmlHttp = new ActiveXObject("Microsoft.XMLHTTP").В случае, если ни одна из этих попыток не принесла успеха, выдается сообщение об отсутствии поддержки AJAX браузером.

```
<?php
header("Cache-Control: no-cache, must-revalidate");
 // Прошедшая дата
header("Expires: Mon, 1 Sep 2008 07:30:00 GMT");
// Инициализация массива названий
$a[]="Audi";
$a[]="BMW";
$a[]="Buick";
$a[]="Chevrolet";
$a[]="Citroen";
$a[]="Dodge";
$a[]="Ferrari";
$a[]="Fiat";
$a[]="Ford";
$a[]="Honda";
$a[]="Hyundai";
$a[]="Cherokee";
$a[]="Cherry";
$a[]="Lada";
$a[]="Lamborghini";
$a[]="Lincoln";
$a[]="Mazda";
$a[]="Mercedes";
$a[]="Mitsubishi";
$a[]="Nissan";
$a[]="Opel";
$a[]="Peugeot";
$a[]="Plymoth";
$a[]="Pontiac";
$a[]="Renault";
$a[]="Rover";
$a[]="Saab";
$a[]="Subaru";
$a[]="Suzuki";
$a[]="Toyota";
$a[]="Volkswagen";
$a[]="Volvo";
//получение параметра q из URL
$q = $_GET["q"];
//поиск соответствий из массива если длина q > 0
if (strlen($q) > 0)
{
 $hint = "";
 for($i = 0; $i<count($a); $i++)
 {
 if (strtolower($q) == strtolower(substr($a[$i],0,strlen($q))))
  {
  if ($hint == "")
   {
   $hint=$a[$i];
   }
  else
   {
   $hint=$hint." , ".$a[$i];
   }
  }
 }
}
// Возврат строки "нет вариантов" если соответствий не найдено
// либо найденное соответствие
if ($hint == "")
{
$response = "no suggestion";
}
else
{
$response = $hint;
}
//вывод результата
echo $response;
```

Задание № 2. Разработайте код выбора марки автомобиля с использованием Ajax.

Задание № 3. Разработайте код выбора страны, региона, города с использованием Ajax.

Задание № 4. Разработайте код отправки файла с использованием Ajax.

Задание № 5. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 13. ИСПОЛЬЗОВАНИЕ БИБЛИОТЕКИ JQUERY

Цель: ознакомиться с библиотекой jQuery.

Теоретические вопросы

Назначение jQuery.

Работа с библиотекой jQuery.

jQuery синтаксис.

jQuery селекторы.

Задание № 1. Выполнить пример.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>
<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("p").hide();
        });
    });
</script>
</head>

<body>
<h2>This is a heading</h2>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body>
</html>
```

Проверить работу метода hile(). Попробовать заменить библиотеку jquery-latest.js на другие две, описанные выше, сравнить результаты работы.

Задание № 2. Выполнить пример.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("#test").hide();
        });
    });
</script>
</head>

<body>
<h2>This is a heading 2</h2>
<p>This is a paragraph.</p>
<p id="test">This is another paragraph.</p>
<button>Click me</button>
</body>

</html>
```

Проанализировать отличие его от примера в задании 1.

Задание № 3. Выполнить пример.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("#but1").click(function () {
            $("#test").hide();
        });
        $("#but2").click(function () {
            $("#test").show();
        });
    });
</script>
</head>

<body>
<h2>This is a heading 2</h2>
<p>This is a paragraph.</p>
<p id="test">This is another paragraph.</p>
<button id="but1">Click me</button>
<button id="but2">Click me for show</button>
</body>

</html>
```

Проанализировать отличие его от примера в задании 2.

Задание № 4. Выполнить пример.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("button").click(function () {
            $(".test").hide();
        });
    });
</script>
</head>
<body>

<h2 class="test">This is a heading</h2>
<p class="test">This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body>
</html>
```

Проанализировать отличие его от примера в задании 3.

Задание № 5. Выполнить пример.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("p").dblclick(function () {
            $(this).hide();
        });
    });
</script>
</head>
<body>

<p>If you double-click on me, I will disappear.</p>
<p>Click me away!</p>
<p>Click me too!</p>

</body>
</html>
```

Задание № 6. Выполнить пример.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>
<script>
    $(document).ready(function () {
        $("input").focus(function () {
            $(this).css("background-color", "#cccccc");
        });
        $("input").blur(function () {
            $(this).css("background-color", "#ffffff");
        });
    });
</script>
</head>
<body>

Name: <input type="text" name="fullname"><br>
Email: <input type="text" name="email">

</body>
</html>
```

Задание № 7. Выполнить пример.

```
<!DOCTYPE html>
<html>
<head>
<script src="http://code.jquery.com/jquery-latest.js">
</script>

<script>
    $(document).ready(function () {
        $("button").click(function () {
            $("p").toggle();
        });
    });
</script>
</head>
<body>

<button>Toggle</button>
<p>This is a paragraph with little content.</p>
<p>This is another small paragraph.</p>
</body>
</html>
```

Задание № 8. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 14. ИСПОЛЬЗОВАНИЕ ФРЕЙМВОРКА ДЛЯ СОЗДАНИЯ САЙТА

Цель: ознакомиться с библиотекой jQuery.

Теоретические вопросы

Концепция фреймворков.

Обзор современных фреймворков.

Сравнение популярных фреймворков.

Выбор оптимального фреймворка для разработки сайта.

Задание № 1. Выберите и обоснуйте выбор фреймворка.

Задание № 2. Разработать сайт-каталога одежды и обуви, который будет обладать следующими особенностями:

1. Товары разделены по категориям, с возможностью создания подкатегорий.
2. Удобная административная панелью.
3. Многоязычность, поддержка русского, румынского и английского языков.
4. Поддержка высоких нагрузок (кэширование).

Задание № 3. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 15. СОЗДАНИЕ САЙТА НА CMS

Цель: ознакомиться с CMS.

Теоретические вопросы

Понятие CMS.

Принципы работы CMS.

Обзор современных CMS.

Сравнительный анализ CMS.

Задание № 1. Выберите и обоснуйте выбор CMS.

Задание № 2. Изучите выбранную CMS.

Задание № 3. Установите необходимое программное обеспечение.

Задание № 4. Спроектируйте сайт

Задание № 5. Создайте меню и страницы сайта.

Задание № 6. Оформить отчет.

ЛАБОРАТОРНАЯ РАБОТА № 16. АДМИНИСТРИРОВАНИЕ САЙТА

Цель: ознакомиться с технологиями администрирования сайта.

Теоретические вопросы

Задачи администрирования сайта.

Настройка сайта.

Обеспечение безопасности сайта.

Оптимизация сайта.