Министерство образования и науки РФ

ФГБОУ ВПО «Тамбовский государственный технический университет»

Кафедра «САПР»

Лабораторная работа №6

по дисциплине «Интернет технологии»

«AJAX технология»

Выполнил: студент гр. . Ю.

Проверил: А. А.

Тамбов, 20

**Цель работы:** реализация ajax модели для динамичного обновления содержимого страницы сайта.

**Задание:**

Разработать web-приложение, реализующее концепцию AJAX.

Основные условия:

* реализация связи: html-страница ↔ php-скрипт ↔ база данных;
* в качестве ответа от сервера использовать следующие его формы: текст, XML, JSON (сделать три варианта исходной задачи);
* возможно использование материальной базы предыдущих работ.

Решение:

Все три варианта ответов сервера сводим на одну страницу, код в таблице 1.

|  |
| --- |
| Таблица 1. HTML код страницы |
| <!doctype html>  <html>  <head>  <meta http-equiv=**"Content-Type"** content=**"text/html; charset=windows-1251"**>  <title>**Поиск по базе**</title>  <script src=**"json2.js"**></script>  <script language=**"javascript"** src=**"ajax\_framework.js"**></script>  <script language=**"javascript"** src=**"ajax\_xml.js"**></script>  <script language=**"javascript"** src=**"ajax\_son.js"**></script>  </head>  <body bgcolor=**"Papayawhip"** >  <table bgcolor=red border=1 cellpadding=2>  <tr>  <th><h2>**Общий поиск**</h2></th><th><h2>**Персональный поиск**</h2></th>  </tr>    <tr><td rowspan=2>  <form id=**"searchForm"** name=**"searchForm"** method=**"post"** action=**"javascript:insertTask();"**>  <div class=**"searchInput"**>  <input name=**"searchq"** type=**"text"** id=**"searchq"** size=**"30"** onkeyup=**"javascript:searchNameq()"**/>  <input type=**"button"** name=**"submitSearch"** id=**"submitSearch"** value=**"Search"** onclick=**"javascript:searchNameq()"**/>  </div>  </form>    <h3>**Итог:**</h3>  <div id=**"msg"**>**Например "кола"**</div>  <div id=**"search-result"**>**--:**</div>  </td>    <td valign =top>  <form id=**"sForm"** name=**"sForm"** method=**"post"** action=**"javascript:insert();"**>  <div class=**"searchInput"**>  <input name=**"xml"** type=**"text"** id=**"xml"** size=**"30"** onkeyup=**"javascript:search()"**/>  <input type=**"button"** name=**"submitSearch"** id=**"submitSearch"** value=**"Search"** onclick=**"javascript:search()"**/>  </div>  </form>    <h3>**Итог:**</h3>  <div id=**"messaga"**>**Например "Vasa"**</div>  <div id=**"sresult"**>**--:**</div>  </td>  </tr>    <tr valign=top><td valign=top>  <form id=**"FormKaras"** name=**"FormKaras"** method=**"post"**>  <div>  <input name=**"son"** type=**"text"** id=**"son"** size=**"30"** onkeyup=**"javascript:Poisk()"**/>  <input type=**"button"** name=**"sub"** id=**"sub"** value=**"Poisk"** onclick=**"javascript:Poisk()"**/>  </div>  </form>    <h3>**Итог:**</h3>  <div id=**"spros"**>**Например "ГазМяс"**</div>  <div id=**"resultat"**>**--:**</div>  </td></tr>    </table>  </body>  </html> |

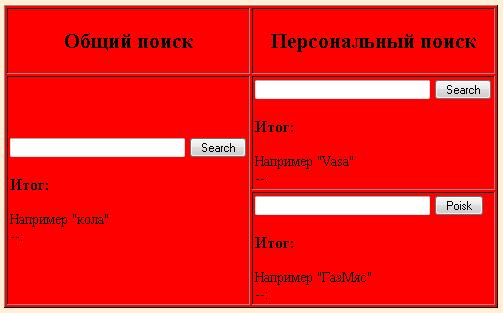
Размещаем в общем случае одно текстовое поле через input, рядом кнопочку для подстраховки, хотя это и лишне, ну и целевой блок посредством div id для вывода в него результатов. Визуально страница примет вид как на рисуноке 1.

Рисунок Визуальное представление страницы

Первая часть из трех готова, далее идет серверная из написания php сценария. Так как сервер возвращает для каждого из трех вариантов разные ответы, то, дабы не смешивать все в салат, делаем для каждой задачи отдельный файл с кодом. В таблице 2 сведен вариант с текстом в качестве ответа серверной стороной.

|  |
| --- |
| Таблица 2. PHP код серверной страницы для текстового ответа |
| <?php    $host="localhost";  $user="root";  $pass=""; //установленный вами пароль  $db\_name="test";  $link=**mysql\_connect**($host,$user,$pass);  **mysql\_select\_db**($db\_name,$link);  **mysql\_query**("SET NAMES 'utf8'");  **print** "<table border=1 cellspacing=0 cellpadding=0 style='border-collapse:collapse;border:none;mso-border-alt:solid black .5pt;'>";  **print** "<tr bgcolor=red><td align=center>ID</td><td align=center>Name</td><td align=center>E-mail</td></tr>";  $i = 0;  $searchq = $\_GET['name'];//iconv("utf-8", "utf-8", $\_GET['name']);//  $getName = **mysql\_query**('SELECT \* FROM students WHERE name LIKE "%'.addslashes($searchq).'%"');  **while** ($row = **mysql\_fetch\_array**($getName))  {  **if**($i%2)  {  **echo** "<tr bgcolor=Wheat>";  **echo** "<td>";  **echo** $row['id'].'</td><td>'.$row['name'].'</td><td>'.$row['address'].'</td>';  **echo** "</tr>";  }  **else**  {  **echo** "<tr bgcolor=Cornsilk>";  **echo** "<td>";  **echo** $row['id'].'</td><td>'.$row['name'].'</td><td>'.$row['address'].'</td>';  **echo** "</tr>";  }  $i++;  }    **print** "</table>";  **mysql\_close**($link);    ?> |

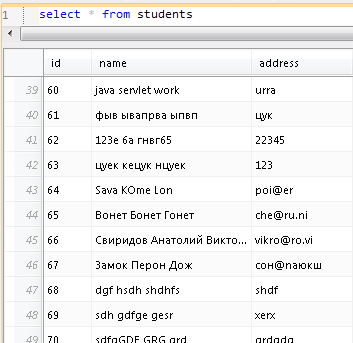
Тут подключаемся к базе данных (Рисунок 2) и делаем выборку, результат оформляем таблицей и отпрявляем ~~в путь~~ эхом.

Рисунок База данных students

Вариант с xml немного другой, так как необходимо формировать ответ с соблюдением правил разметки. Код приведен в таблице 3.

|  |
| --- |
| Таблица 3. PHP код серверной страницы для xml ответа |
| <?php  $host="localhost";  $user="root";  $pass=""; //установленный вами пароль  $db\_name="test";  $link=**mysql\_connect**($host,$user,$pass);  **mysql\_select\_db**($db\_name,$link);  **mysql\_query**("SET NAMES 'utf8'");  $searchq = $\_GET['name'];//iconv("utf-8", "utf-8", $\_GET['name']);//  $getName = **mysql\_query**('SELECT \* FROM students WHERE name LIKE "%'.addslashes($searchq).'%"');    header("Content-Type: text/xml");  **echo** "<?xml version=\"1.0\" encoding=\"utf-8\"?>";  **echo** "<books>";  **while** ($row = **mysql\_fetch\_array**($getName))  {  //Формируется структура дерева тегов:  **echo** '<books\_name>'.$row['name'].'</books\_name>';  **echo** '<num>'.$row['id'].'</num>';  **echo** '<pages>'.$row['address'].'</pages>';  }  **mysql\_close**($link);  //Формируется структура дерева тегов: //корень дерева тегов:  **echo** "</books>";  ?> |

Кроме уже упоминавшегося ранее подключения к базе, тут обязательно указывается формат того контента, который предпологается посылать в качестве ответа ( header("Content-Type: text/xml"); ). После этого отправляется структура документа в виде дерева с одним корнем и вложениями со значениями. У каждого открытого тега обязательным является его закрывающий.

Последним идет вариант ответа через json объект (Таблица 4).

|  |
| --- |
| Таблица 4. PHP код серверной страницы для json ответа |
| <?php  **require\_once**('JSON.php');  $json = **new** Services\_JSON();  $host="localhost";  $user="root";  $pass=""; //установленный вами пароль  $db\_name="test";  $link=**mysql\_connect**($host,$user,$pass);  **mysql\_select\_db**($db\_name,$link);  **mysql\_query**("SET NAMES 'utf8'");  $searchq = $\_GET['name'];//iconv("utf-8", "utf-8", $\_GET['name']);//  $getName = **mysql\_query**('SELECT \* FROM students WHERE name LIKE "%'.addslashes($searchq).'%"');    $names = **array**();  $ids = **array**();  $address = **array**();    **while** ($row = **mysql\_fetch\_array**($getName))  {  **array\_push**($ids, $row['id']);  **array\_push**($names, $row['name']);  **array\_push**($address, $row['address']);  }    **mysql\_close**($link);  $res\_arr = **array**('num'=>$ids, 'pages'=>$address,'inbooks'=>$names);  $output = $json->encode($res\_arr);  **print**($output);  ?> |

Все ответы, снятые с базы, пакуются в одномерные массивы методом array\_push(массив, значение), далее, в строке $res\_arr = array('num'=>$ids, 'pages'=>$address,'inbooks'=>$names) формируется массив вида «ключ→значение». Следующей строкой еще раз ~~скотчем~~ запаковываем массив в json-объект и посылаем на вывод. Два нюанса, первый – это строка require\_once('JSON.php'); которая говорит «в этом файле есть метод Services\_JSON()», чем мы пользуемся на следующей строке.

Ответ сервера надо обработать, этим занимается js скрипт, их тоже три (Таблица 5).

|  |
| --- |
| Таблица 5. JS для обработки текстового ответа |
| //создать обьект XMLHttpRequest  **function** createObject**()**  **{**  **var** request\_type**;**  **var** browser **=** navigator**.**appName**;**  **if(**browser **==** "Microsoft Internet Explorer"**)**  **{**  request\_type **=** **new** ActiveXObject**(**"Microsoft.XMLHTTP"**);**  **}**  **else**  **{**  request\_type **=** **new** XMLHttpRequest**();**  **}**  **return** request\_type**;**  **}**  **var** http **=** createObject**();**  //Сформировать запрос на сервер\_php  **function** searchNameq**()**  **{** //http://habrahabr.ru/post/17640/  //некодируем поисковую строку для вывода "Запрос:"  searchq **=** document**.**getElementById**(**'searchq'**).**value**;**  document**.**getElementById**(**'msg'**).**style**.**display **=** "block"**;**  document**.**getElementById**(**'msg'**).**innerHTML **=** "Итог для запроса <strong>" **+** searchq**+**""**;**  //Для дальнейшей работы запрос кодируем в последовательность %ESCAPE%  searchq **=** encodeURI**(**document**.**getElementById**(**'searchq'**).**value**);**    //чтобы браузер не кэшировал страницу:  nocache **=** Math**.**random**();**    http**.**open**(**'get'**,** 'in-search.php?name='**+**searchq**+**'&nocache = '**+**nocache**);**  http**.**onreadystatechange **=** searchNameqReply**;**  //null - всё уже в http.open(get)  http**.**send**(**null**);**  **}**  //Вывести ответ на страницу  //http.responseText;//http.responseXML;//JSON.parse(response);  **function** searchNameqReply**()**  **{**  **if(**http**.**readyState **==** 4**)**  **{**  **var** response **=** http**.**responseText**;**  document**.**getElementById**(**'search-result'**).**innerHTML **=** response**;**  **}**    **}** |

Функция createObject() создаст объект для отправки запросов в асинхронных приложениях, без обновления страницы. Функция searchNameq() формирует пасылку, внутри которой есть параметр name (нужен для выборки) и окончание в виде рандомного числа, видимо для предания псевдо-уникальности каждого запроса. Затем подписываемся на прием сообщений со стороны сервера, занимаем ожидательную позицию. Если ожидание оправдалось, то http.readyState == 4 сработает, иначе при передаче или приёме произошел сбой, либо ответ настолько большой, что еще недошел и т.п. Остаётся вывести принятый ответ на форму в конкретный div блок.

Код сценария для xml немного отличается от текстового варианта (Таблица 6).

|  |
| --- |
| Таблица 6. JS для обработки xml ответа |
| **function** searchReply**()**  **{**//http://www.ibm.com/developerworks/ru/library/wa-ajaxintro8/  **if(**http**.**readyState **==** 4**)**  **{**  **var** response **=** http**.**responseXML**;**  **var** name **=** response**.**getElementsByTagName**(**"books\_name"**);**//тег внутри php кода!  **var** id **=** response**.**getElementsByTagName**(**"num"**);**//тег внутри php кода!  **var** address **=** response**.**getElementsByTagName**(**"pages"**);**//тег внутри php кода!  //firstChild.nodeValue - проход по дереву тегов xml    **for** **(var** i**=**0**;** i **<** name**.**length**;** i**++)**  txtName **=** name**[**i**].**firstChild**.**nodeValue**;**    **for** **(var** i**=**0**;** i **<** id**.**length**;** i**++)**  txtId **=** id**[**i**].**firstChild**.**nodeValue**;**    **for** **(var** i**=**0**;** i **<** address**.**length**;** i**++)**  txtAddress **=** address**[**i**].**firstChild**.**nodeValue**;**    document**.**getElementById**(**'sresult'**).**innerHTML **=** txtId**+**" "**+**txtName**+**" "**+**txtAddress**;**  **}else** document**.**getElementById**(**'sresult'**).**innerHTML **=** "Ошибка статуса http.readyState == 4"**;**  **}** |

Отличие заключается в том, что принятый ответ нужно распаковать.

*Свойство http.responseXML содержит объектную модель документа XML. К DOM XML применяются те же методы и используются те же свойства элементов, что и в объектной модели HTML.*

Поэтому распаковываем ориентируясь на имя тега getElementsByTagName(), а значения вытаскиваем обходя полученные теги firstChild.nodeValue.

Для json код сценария на создание объекта и отправки параметров на сервер идентичны текстовому варианту кода. В таблицу 7 внесен метод обработки ответа сервера.

|  |
| --- |
| Таблица 7. JS для обработки json ответа |
| **function** searchOtvet**()**  **{**//http://www.ibm.com/developerworks/ru/library/wa-ajaxintro8/  **if(**http**.**readyState **==** 4**)**  **{**  **var** response **=** http**.**responseText**;**  **var** jsonData **=** JSON**.**parse**(**response**);**  //id[]:  **for** **(var** i**=**0**;** i**<**jsonData**.**num**.**length**;** i**++)**  txtID **=** jsonData**.**num**[**i**];**  //name[]:  **for** **(var** i**=**0**;** i**<**jsonData**.**inbooks**.**length**;** i**++)**  txtNAME **=** jsonData**.**inbooks**[**i**];**  //address[]:  **for** **(var** i**=**0**;** i**<**jsonData**.**pages**.**length**;** i**++)**  txtADRES **=** jsonData**.**pages**[**i**];**    document**.**getElementById**(**'resultat'**).**innerHTML **=** txtID**+**" "**+**txtNAME**+**" "**+**txtADRES**;**  **}else** document**.**getElementById**(**'resultat'**).**innerHTML **=** "Ошибка статуса http.readyState == 4"**;**  **}** |

*JSON.parse(response) - получает ответ сервера в текстовом виде и преобразует в объект JavaScript . Для использования данной функции необходимо подключить библиотеку json.js*

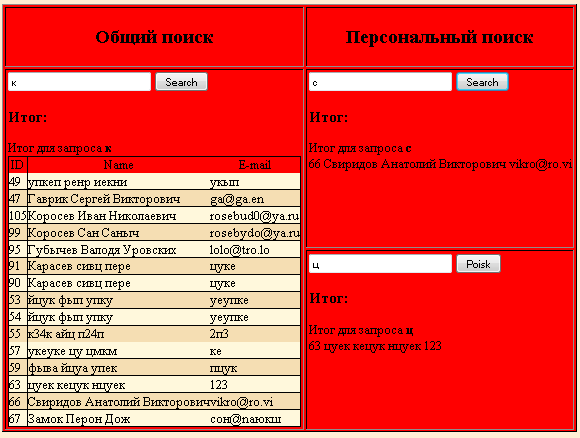
Результат на рисунке 3.

Рисунок Результат выполнения поиска