

## Using Java Server Pages to Move from Page to Page

One Java server page can create a form that the client can fill out and submit to a second Java server page. This makes moving from page to page easier to program. The html code stays in the JSP and is not part of a servlet.

The following example adds a column to the database above and then presents the client with a second form to be filled out. This form calls another JSP, which in turn works with a second Java bean. The first file is the original html file.

### addColumnn.html

```
<html>
<head>
<title> Add a Column </title>
</head>

<body>
<h3>Enter name of new column.</h3>
<form method = "get" action=" ../market/addcolumnn.jsp">
  <input name="colName" type="text" value = "" size = "10" /> Column Name<br />
  <p><input type="submit" value="Send"></p>
</form></body></html>
```

It is used to add a new column to the database using the JSP, addcolumn.jsp. This file follows next.

### addcolumn.jsp

```
<html>
<head><title> Add Column JSP. </title></head>

<body bgcolor="white">
<font size=4 color="blue">
<%! String [][] tableData;
  int rows, columns, row, col;
  String colName;
%>
<jsp:useBean id="addColumnBean" scope="session" class="produce.AddColumnBean" />
<jsp:setProperty name="addColumnBean" property="*" />

<% addColumnBean.processRequest(); %>

<h4>Produce Table</h4>
<% tableData = addColumnBean.getTableData ();
  rows = addColumnBean.getRows ();
  columns = addColumnBean.getColumns ();
  colName = request.getParameter ("colName");
%>
<form method = "get" action = " ../market/enterData.jsp">
<table border='1' bordercolor='blue' cellspacing='10'>
  <tr><td>Name</td><td><% out.println (colName); %></td></tr>
  <% for (int row = 1; row < rows; row ++)
```

```

        {
            String name = tableData [row][3];
            String id = tableData [row][1]; %>
            <tr>
                <td> <% out.print (name); %></td>
            <td> <input name = <%out.println (""+id+"");%> type = "text" value = "" size = "10" /></td>
            </tr>
        <% } %>
    </table>
    <p><input type="submit" value="Send" /></p>
</form>
</font></body></html>

```

Notice that it creates a new form whose action attribute is the file, enterData.jsp. It also uses the table created by the DisplayBean shown before. The Java bean that goes with addcolumn.jsp follows:

#### AddColumnBean.java

```

package produce;

import java.sql.*;
import java.io.*;

// AddColumnBean adds a new column to the table.
public class AddColumnBean
{
    private Connection con;
    private String colName;
    private String [][] tableData;
    private int rows;
    private int columns;

    public String getColName () {return colName;}
    public String [][] getTableData () {return tableData;}
    public int getRows () {return rows;}
    public int getColumns () {return columns;}

    public void setColName (String c) {colName = c;}
    public void setTableData (String [][] t) {tableData = t;}
    public void setRows (int r) {rows = r;}
    public void setCols (int c) {columns = c;}

    public void processRequest ()
    {
        try
        {
            // Get a jdbc-odbc bridge and connect to addresses.mdb.
            Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
            con = DriverManager.getConnection ("jdbc:odbc:produce");

```

```

        alterTable ();
        getTable ();
        con.close ();
    } catch (ClassNotFoundException e){System.out.println ("Class Not Found exception.\n");}
    catch (SQLException e){System.out.println ("SQL Exception\n");}
} // process

// alterTable adds a new column to the database with the name given by the request parameter.
public void alterTable ()
{
    try
    {
        Statement stmt = con.createStatement ();
        String query =
            "Alter Table ProduceTable Add " + colName + " varchar (10)";
        int success = stmt.executeUpdate (query);
        if (success == 0) System.out.println ("Alter error.");
        else System.out.println ("Column inserted.");
        stmt.close ();
    } catch (SQLException es) {System.out.println ("SQL Exception");}
} // alterTable

/* getTable creates a two dimensional table containing the database data.*/
public void getTable ()
{
    try
    {
        Statement stmt = con.createStatement ();
        String query = "Select * From ProduceTable";
        ResultSet rs = stmt.executeQuery (query);
        ResultSetMetaData metaData = rs.getMetaData ();
        columns = metaData.getColumnCount ();
        tableData = new String [20][columns+1];
        for (int count = 1; count <= columns; count ++)
            tableData [0][count] = metaData.getColumnName (count);
        rows = 1;
        while (rs.next ())
        {
            for (int col = 1; col<= columns; col ++)
                tableData [rows][col] = rs.getString (col);
            rows ++;
        }
        stmt.close ();
    } catch (SQLException es) {System.out.println ("SQL Exception");}
} // getTable

} // class AddColumnBean

```

The file, addcolumn.jsp creates a form with action enterData.jsp. This file follows:

## enterData.jsp

```
<html>
<head><title> Enter Data JSP. </title></head>

<body bgcolor="white">
<font size=4 color="blue">

<%! String [][] tableData;
      int rows, columns, row, col;
%>
<jsp:useBean id="enterDataBean" scope="session" class="produce.EnterDataBean" />
<jsp:setProperty name="enterDataBean" property="*" />

<% enterDataBean.processRequest(request); %>

<h4>Produce Table</h4>
<%
      tableData = enterDataBean.getTableData ();
      rows = enterDataBean.getRows ();
      columns = enterDataBean.getColumns ();
%>

<table border='1' bordercolor='blue' cellspacing='10'>
  <% for (int row = 0; row < rows; row ++ ) { %>
    <tr>
      <% for (int col = 1; col <= columns; col ++ ) { %>
        <td> <% out.println (tableData [row][col]); %></td>
      <% } %>
    </tr>
  <% } %>
</table>
<p><a href=" ../market/index.html">Return</a></p>
</font>
</body></html>
```

The form looks like the following:

**Produce Table**

Name	Color
Grapes	purple
Carrots	orange
Pear	yellow
Beans	green
Bananas	yellow
Broccoli	green
Apples	red
Peas	green
Oranges	oranges

The Java bean that works with this file is

EnterDataBean.java

// EnterDataBean enters data into the new column in the database.

```
package produce;
```

```
import java.sql.*;
import java.io.*;
import javax.servlet.http.*;
```

// EnterDataBean adds the data for the new column.

```
public class EnterDataBean
{
    private Connection con;
    private String [][] tableData;
    private int rows;
    private int columns;
```

```

public String [][] getTableData () {return tableData;}
public int getRows () {return rows;}
public int getColumns () {return columns;}

//public void setTableData (String [][] t) {tableData = t;}
//public void setRows (int r) {rows = r;}
//public void setCols (int c) {columns = c;}

public void processRequest (HttpServletRequest request)
{
    try
    {
        // Get a jdbc-odbc bridge and connect to addresses.mdb.
        Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
        con = DriverManager.getConnection ("jdbc:odbc:produce");
        enterData (request);
        setTableData ();
        con.close ();
    } catch (ClassNotFoundException e){System.out.println ("Class Not Found exception.\n");}
    catch (SQLException e){System.out.println ("SQL Exception");}
} // processRequest

// enterData enters the new data.
public void enterData (HttpServletRequest request)
{
    int row = 1;
    String id, quantity, colName;
    try
    {
        Statement stmt = con.createStatement ();
        String query = "Select * From ProduceTable";
        ResultSet rs = stmt.executeQuery (query);
        ResultSetMetaData metaData = rs.getMetaData ();
        int columns = metaData.getColumnCount ();
        colName = metaData.getColumnName (columns);

        while (rs.next ())
        {
            id = rs.getString ("ID");
            quantity = request.getParameter (id);
            stmt = con.createStatement ();
            query =
            "Update ProduceTable SET "+colName+" = " + quantity + " Where ID = " + id + """;
            int success = stmt.executeUpdate (query);
            if (success == 0) System.out.println ("Update Enter error.");
            stmt.close ();
            row ++;
        }
    } catch (SQLException es) {System.out.println ("SQL Exception - Enter Scores");}
} // enterData

```

```

public void setTableData ()
{
    try
    {
        Statement stmt = con.createStatement ();
        String query = "Select * From ProduceTable";
        ResultSet rs = stmt.executeQuery (query);
        ResultSetMetaData metaData = rs.getMetaData ();
        columns = metaData.getColumnCount ();
        tableData = new String [20][columns+1];
        for (int count = 1; count <= columns; count ++)
            tableData [0][count] = metaData.getColumnName (count);
        rows = 1;
        while (rs.next ())
        {
            for (int col = 1; col<= columns; col ++)
                tableData [rows][col] = rs.getString (col);
            rows ++;
        }
    } catch (SQLException e){System.out.println ("SQL Exception");}
} // setTableData
} // class EnterDataBean

```

## Include Files

We can include html files in JSP files. This allows us to add the same heading to all the files. An example of a file that we can include is called welcome.html and is shown below.

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head> <title>Welcome File</title> </head>

<body>
<font size="5" color="blue">
Welcome to the Seaport Deli

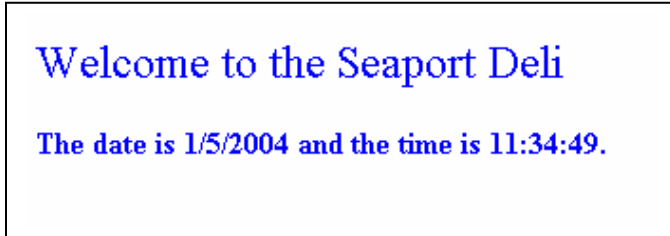
<h4>
<script language="Javascript">
<!--Example that writes the current date on the page>
    date = new Date ()
    month = date.getMonth ()
    month = month + 1    // Month numbers begin with 0.
    day = date.getDate ()
    year = date.getYear ()
    document.write ("The date is ", month)
    document.write ("/", day)
    document.write ("/", year)
    hour = date.getHours ()
    if (hour > 12) hour = hour - 12    // Hours are given in military time.
    minutes = date.getMinutes ()
    seconds = date.getSeconds ()

```

```

        document.write (" and the time is ", hour, ":", minutes, ":", seconds, ".")
    //-->
</script></h4>
</font></body></html>

```



This file can be included in all files in the application. An example, find.jsp, follows.

```

<html>
<head><title> Find Produce JSP. </title></head>

<body bgcolor="white">
<font size=4 color="blue">

<center> <%@ include file = "date-time.html"%> </center>

<jsp:useBean id="findBean" scope="session" class="produce.FindBean" />
<jsp:useBean id="connBean" scope="session" class="produce.ConnectionBean" />
<jsp:setProperty name="findBean" property="*" />

<% findBean.processRequest(); %>
<% if (findBean.getFound () { %>
<h4>Produce Table</h4>
<table border = "1" cellspacing = "5">
    <tr>
        <td><% out.println (findBean.getId()); %></td>
        <td><% out.println (findBean.getType()); %></td>
        <td><% out.println (findBean.getName()); %></td>
        <td><% out.println (findBean.getVariety()); %></td>
        <td><% out.println (findBean.getPrice()); %></td>
    </tr>
</table>
<% } else { %>
    <h4>The product is not in the database.</h4>
<% } %>
<hr>
<p><a href=" ../market">Return</a></p>
</font>
</body>
</html>

```