Paper TU-08

Losing the Fat: One Application's Journey to the Internet

Jeff Lessenberry, Jeff Lessenberry Consulting Group, Simpsonville, SC

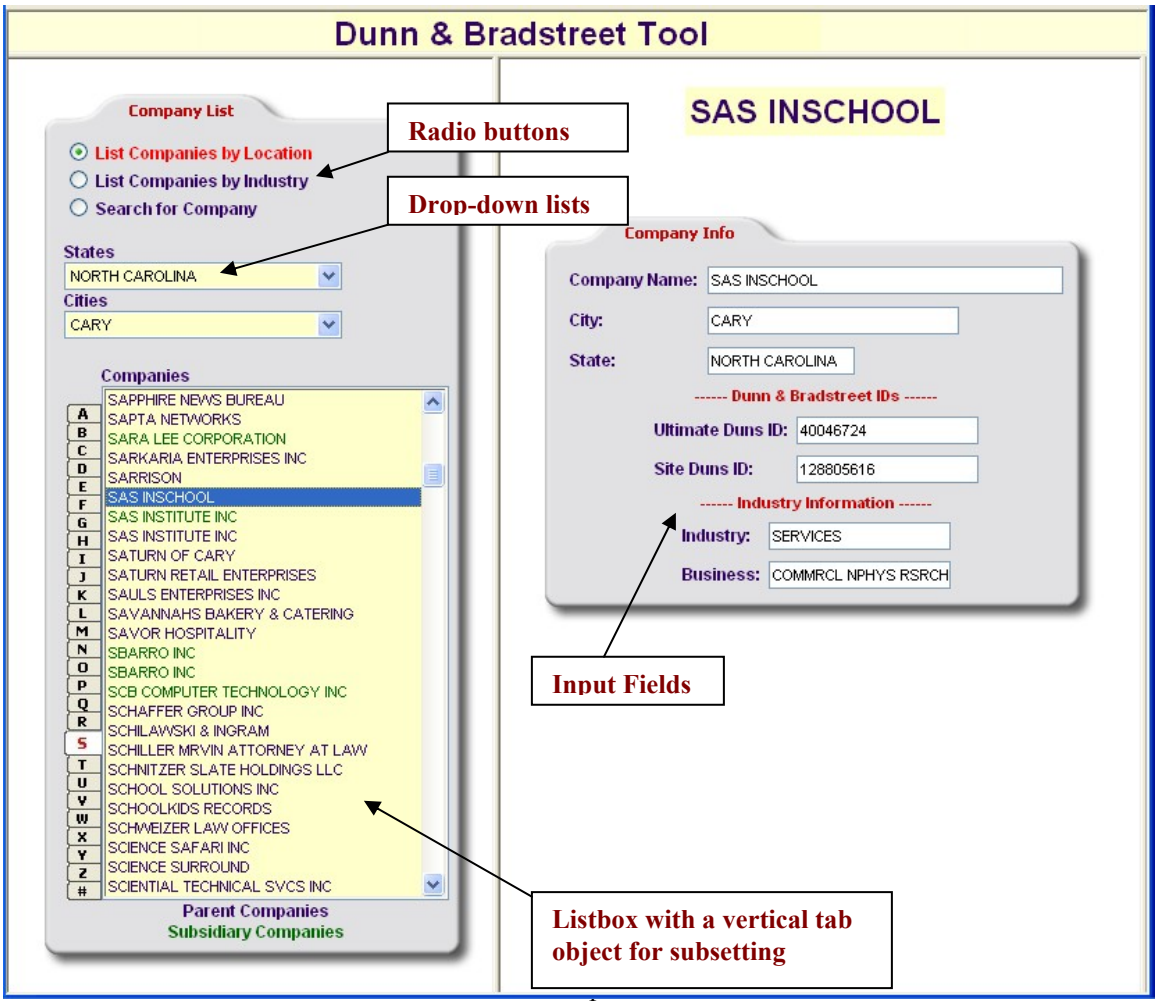
ABSTRACT

This paper will discuss the processes necessary to convert a SAS/FRAME application to an intranet-based application. Until a few years ago, I was content with coding fat client applications utilizing SAS/FRAME. I was happy with my frame objects and SCL until my manager approached me with a new web project. The web project used Java Server Pages as the interface and the SAS/IOM application server as the backend access to the data. Although I had no training in web development, I was really excited about the project. There are some basic techniques which have surfaced during my learning curve that make the development process easier. This paper will discuss those techniques and present an example application.

INTRODUCTION

Evolution: a process of continuous change from a lower, simpler, or worse to a higher, more complex, or better state. This is a perfect definition for the world of application development. How many developers remember the ISPF screens from the mainframe days? I remember how overwhelmed I felt when I created my first SAS/FRAME screen, but the great objects and colors made it all seem worthwhile. While most users really liked the look and feel of the screens, the cost of putting SAS on every desktop made most budgets cry for mercy. The search for a cheaper interface had begun and so had the evolution from fat-client to thin-client applications. The web browser would be the vehicle to drive the next generation of GUI applications where frame objects and SCL would be replaced by HTML and Java. This paper will focus on the transition of a fat-client screen to a thinner version of itself.

THE WEB VERSION



As you can see from the application screen shot, I have duplicated many of the commonly used SAS/FRAME objects with the new web application. This application also builds all lists dynamically from SAS datasets via an IOM server connection. Here is an example of connection code for the IOM server running on my pc.

```
<%@taglib uri="http://www.sas.com/taglib/sasads" prefix="sasads"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<HTML>
<sasads:Connection id="con" scope="session" persistedName="iomtest"
 serverArchitecture="IOM" initialStatement="%include 'c:\sas8\autoexec.sas';run;"
 accessMethod="IOM" />

<%
    // Create a new factory
    com.sas.rmi.Rocf rocf = new com.sas.rmi.Rocf();
                session.setAttribute("rocf", rocf);
%>
<title>Dunn and Bradstreet Tool</title>
<frameset>
        <frameset  rows="30,*" >
             <frame NORESIZE name="Title" src="./dnb_top_title.jsp"
                    frameborder = "1" scrolling="NO" >
             <frameset cols="350,*,1"  >
                 <frame NORESIZE name="Selection" src="./dnb1b.jsp"
                        frameborder = "1"  scrolling="NO">
                 <frame NORESIZE name="Results" src="./v2_options_wait.html"
                        frameborder = "1"  scrolling="NO" >
                 <frame NORESIZE name="Runspace" src="about:blank"
                        frameborder = "1" >
                 </frameset>
         </frameset>
</frameset>
</HTML>
```

You will notice that I have set up four different frames within the web browser. The TITLE section (in green) is setup as a stand-alone frame. I normally have one title JSP that can be used with multiple frames by passing a variable to the screen. I also use the frame to do any user tracking for that application. The SELECTION section (in blue) is where the user can make all subsetting before displaying the final results. The RESULTS section (in red) displays the results and also passes messages to the user while processing is taking place. The RUNSPACE section (in plum) is used to run SAS code. This section is hidden because it is only 1 pixel in size. I like to have this space so that I can run SAS code without refreshing the screen. Think of it as a remote submit. Some people may say this is unnecessary, but I like to use it. By the way, most of those who don't care for this section were never SAS/FRAME developers.

THE RADIO BUTTONS

The radio buttons in this application are used to navigate between different subsetting options. In this case, I have set up a different JSPs for each subsetting option. I have tried different methods of hiding and unhiding objects but this just seems easier and more readable to the developer who comes after me. Here is the code used to do this:


```
<TR hspace="20" class='mainLabel'>
    <INPUT TYPE="radio" NAME="assign_select" CHECKED VALUE="a">
 <font color="#ff0000">List Companies by Location</font><BR>
    <INPUT TYPE="radio" NAME="assign_select"
                    onclick="return assign_select_onclick(1)" VALUE="b">
 List Companies by Industry<BR>
    <INPUT TYPE="radio" NAME="assign_select"
                    onclick="return assign_select_onclick(2)" VALUE="c">
 Search for Company<BR>
</TR>
```

Set the value when clicked

Set Type =”radio”

Set the radio button

Javascript to run when clicked

The first item is the currently checked button. The button text is also set to a non-standard color to make it stand out to the user as the current subsetting option. Also notice that the current option does not run the javascript because the radio button should only change screens if a user picks an option other than the current one. Here is the javascript used with the radio buttons to control the subsetting option screens:

```
function assign_select_onclick(assselIndex)
{
   var returnValue = true;
   MIndex = assselIndex;

   var f = document.forms[0];

   assign_select = myForm.assign_select[MIndex].value

   top.Selection.location="./dnb1" +assign_select+".jsp";
   top.Results.location="./v2_options_wait.html";
}
```

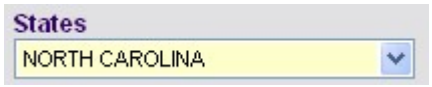
Gets the button number selected

Pulls the value assigned to that button

Runs the subset screen selected

THE DROP-DOWN LISTS

The Drop-Down list is basically listboxes with only one value showing at a time. It is populated from a SAS dataset. Only unique values are shown in this drop-down list.




```
<TR hspace="20" class='mainLabel' >
  States<BR>
    <sasads:Submit  id="submit1a" connection="con" display="none">
            proc summary data=climb.location_summary nway;
                 class state_name ;
                 id scode;
                 output out=all_states(drop=_type_ _freq_);
            run;
    </sasads:Submit>
    <sasads:DataSet id="dsn_sList"      connection="con"
              dataSet="work.all_states"     scope="page" />
    <select name="form_sList" onChange="s_change()" size=1 width="95"
           style="width: 200px;" class='listField'>
<%
   String[] state = (String[])dsn_sList.getFormattedColumn
                    (dsn_sList.getColumnIndex("state_name"));
   String[] scode = (String[])dsn_sList.getFormattedColumn
                    (dsn_sList.getColumnIndex("scode"));

   if (state_pick == "") {
      state_pick = scode[0];
      session.setAttribute("state_pick", state_pick);
                        }
   for (int i=0; i< state.length; i++) {
%>
   <option value="<%=scode[i].trim()%>"
   <% if (scode[i].equals(state_pick)) { %> selected <% } %>
   ><%= state[i].trim() %></option>
<%
   }
      dsn_sList.close();
%>
    </select>
</TR>
```

Create an input dataset for the list

Open dataset

Setup form listbox Note: Size=1

Pull the variable values from DS

Determine if there is a selected value

Write values to HTML page

Close Dataset

When a selection is made from the pull-down list, a section of javascript is run. The javascript pulls the value of the selected item and refreshes the screen with that value. Here is the javascript:

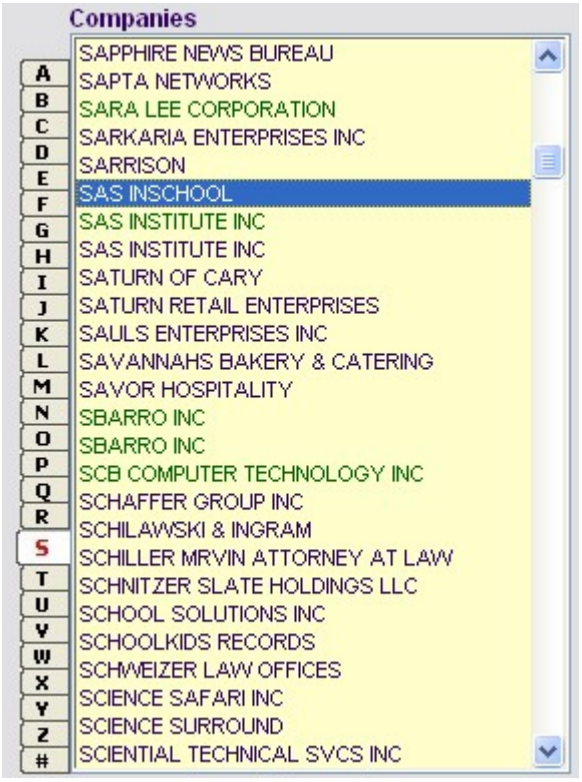
```
function s_change()
  {
        var f = document.forms[0];
        var idx2 = f.form_sList.selectedIndex;
        var sselect = "";
        sselect = f.form_sList.options[idx2].value;


        top.Selection.location="./dnb1a.jsp?state_pick=" + sselect;
        top.Results.location="./v2_options_wait.html";
  }
```

Gets the value of the selected item from the list

Refresh the screen with the selected value

THE LISTBOX WITH A VERTICAL TAB OBJECT

The list box is setup the same as the drop-down box except the size option is set higher than 1. The listbox also sets the text color based on the value of a variable. The example listbox only contains variable values of the selected alphabet tab.



The vertical alphabet tab is used to subset the data values shown in the listbox. Unlike the tab object in SAS/FRAME, this vertical alphabet tab is actually made up of 27 different letter images. Each letter has two images; an unselected image and a selected image.

UnSelected 

Selected 

All of the letter images are set to unselected except for the tab which has been selected. If no tab has been selected, the A tab will be selected as a default. Here is the code for the initial tab setup.

```
String isub1 = "off";
if (isubpick.equals("1")) {isub1 = "on";}
session.setAttribute("isub1", isub1);
```

Since the Dunn & Bradstreet data is so big, I decided to make the alphabet tabs show up as a default.

Here is the code used to create the listbox and alphabet vertical tab selector: The data has also been split into a dataset for every letter and a dataset for numerics.

```
<tr class='mainLabel'><BR>
    <sasads:Submit  id="submit1b" connection="con" >
            %let subf1 = A; %let subf10 = J; %let subf19 = S;
            %let subf2 = B; %let subf11 = K; %let subf20 = T;
            %let subf3 = C; %let subf12 = L; %let subf21 = U;
            %let subf4 = D; %let subf13 = M; %let subf22 = V;
            %let subf5 = E; %let subf14 = N; %let subf23 = W;
            %let subf6 = F; %let subf15 = O; %let subf24 = X;
            %let subf7 = G; %let subf16 = P; %let subf25 = Y;
            %let subf8 = H; %let subf17 = Q; %let subf26 = Z;
            %let subf9 = I; %let subf18 = R; %let subf27 = NUM;
```

Setup macro variables to alphabet tabs

```
                    data _null_;
                         length fwhere $ 500 fletter $ 3;
                         fletter = "&subf<%=isubpick%>";
                         call symput('fletter',fletter);
                    run;
                    data cut_company;
                         set climb.dnb_&fletter(where=(scode = <%=state_pick%> and
                                                       ccode = <%=city_pick%> ));
                    run;
                    proc sort data=cut_company;
                         by prospect_name;
                    run;
          </sasads:Submit>

<td vspace="0" cellpadding="0" cellspacing="0" border="0" valign="bottom"
align="right" hspace="5" >
   <a onclick="fletter_select(1)"><img src="./images/subset_1_<%=isub1%>.jpg"></a><BR>
   <a onclick="fletter_select(2)"><img src="./images/subset_2_<%=isub2%>.jpg"></a><BR>
   <a onclick="fletter_select(3)"><img src="./images/subset_3_<%=isub3%>.jpg"></a><BR>
   <a onclick="fletter_select(4)"><img src="./images/subset_4_<%=isub4%>.jpg"></a><BR>
   <a onclick="fletter_select(5)"><img src="./images/subset_5_<%=isub5%>.jpg"></a><BR>
   <a onclick="fletter_select(6)"><img src="./images/subset_6_<%=isub6%>.jpg"></a><BR>
   <a onclick="fletter_select(7)"><img src="./images/subset_7_<%=isub7%>.jpg"></a><BR>
   <a onclick="fletter_select(8)"><img src="./images/subset_8_<%=isub8%>.jpg"></a><BR>
   <a onclick="fletter_select(9)"><img src="./images/subset_9_<%=isub9%>.jpg"></a><BR>
   <a onclick="fletter_select(10)"><img  src="./images/subset_10_<%=isub10%>.jpg"></a><BR>
   <a onclick="fletter_select(11)"><img src="./images/subset_11_<%=isub11%>.jpg"></a><BR>
   <a onclick="fletter_select(12)"><img src="./images/subset_12_<%=isub12%>.jpg"></a><BR>
   <a onclick="fletter_select(13)"><img src="./images/subset_13_<%=isub13%>.jpg"></a><BR>
   <a onclick="fletter_select(14)"><img src="./images/subset_14_<%=isub14%>.jpg"></a><BR>
   <a onclick="fletter_select(15)"><img src="./images/subset_15_<%=isub15%>.jpg"></a><BR>
   <a onclick="fletter_select(16)"><img src="./images/subset_16_<%=isub16%>.jpg"></a><BR>
   <a onclick="fletter_select(17)"><img src="./images/subset_17_<%=isub17%>.jpg"></a><BR>
   <a onclick="fletter_select(18)"><img src="./images/subset_18_<%=isub18%>.jpg"></a><BR>
   <a onclick="fletter_select(19)"><img src="./images/subset_19_<%=isub19%>.jpg"></a><BR>
   <a onclick="fletter_select(20)"><img src="./images/subset_20_<%=isub20%>.jpg"></a><BR>
   <a onclick="fletter_select(21)"><img src="./images/subset_21_<%=isub21%>.jpg"></a><BR>
   <a onclick="fletter_select(22)"><img src="./images/subset_22_<%=isub22%>.jpg"></a><BR>
   <a onclick="fletter_select(23)"><img src="./images/subset_23_<%=isub23%>.jpg"></a><BR>
   <a onclick="fletter_select(24)"><img src="./images/subset_24_<%=isub24%>.jpg"></a><BR>
   <a onclick="fletter_select(25)"><img src="./images/subset_25_<%=isub25%>.jpg"></a><BR>
   <a onclick="fletter_select(26)"><img src="./images/subset_26_<%=isub26%>.jpg"></a><BR>
   <a onclick="fletter_select(27)"><img src="./images/subset_27_<%=isub27%>.jpg"></a><BR>
   </td>');

       <sasads:DataSet id="compList" connection="con"
                  dataSet="work.cut_company" scope="page" />
   <td class='mainLabel'>Companies
       <select name="company" onChange="comp_change()" size=25 width="95"
              style="width: 250px;" class='listField'>
   <%
          String[] c_name = (String[])compList.getFormattedColumn
                            (compList.getColumnIndex("prospect_name"));
          String[] c_dunsid = (String[])compList.getFormattedColumn
                              (compList.getColumnIndex("duns_id"));
          String[] c_udunsid = (String[])compList.getFormattedColumn
                               (compList.getColumnIndex("ultimate_duns_id"));

          for (int i=0; i< c_name.length; i++) {
   %>
              <option value="<%=c_dunsid[i].trim()%>"
              <% if (c_dunsid[i].equals(c_udunsid[i])) { %>
                 STYLE="color: #330066;font-weight:bold"
              <% } else { %>
                 STYLE="color: #006600"
```

Create an input dataset for the list

Setup all tabs on screen. Change image if currently selected tab letter

Open dataset

Setup form listbox Note: Size=25

Pull the variable values from DS

Write values to listbox in HTML page. Change text color by data values

```
            <% } %>
              <% if (c_dunsid[i].equals(comppick)) { %>
                 selected
              <% } %>
               ><%= c_name[i].trim() %>
            </option>
 <%
       compList.close();
 %>
    </select>
</td>
</tr>
```

Set selected if it is the current selection

Close Dataset

When a selection is made from the listbox, a javascript is executed. Here is the javascript code:

```
function comp_change()
{
      var f = document.forms[0];
      var idx = f.company.selectedIndex;
      var comselect = "";
      var compname = "";

      compselect = f.company.options[idx].value;
      compname = f.company.options[idx].text;

      top.Results.location="./dnb2.jsp?comp_listid=" + compselect;
}
```

Gets the value of the selected item from the list

Display the results for the selected company

When an alphabet tab is selected, a javascript is executed. Here is the javascript code:

```
function fletter_select(fselect)
 {
      var fletter_num = fselect;
      var f = document.forms[0];

       var idx2 = f.form_sList.selectedIndex;
       var sselect = "";
       sselect = f.form_sList.options[idx2].value;

       var idx = f.form_cList.selectedIndex;
       var cselect = "";
       cselect = f.form_cList.options[idx].value;

       top.Selection.location="./dnb1a.jsp?state_pick=" + sselect +
                           "&city_pick=" + cselect +
                           "&isubpick=" + fletter_num ;
       top.Results.location="./v2_options_wait.html";

 }
```

Gets the value of the alphabet tab

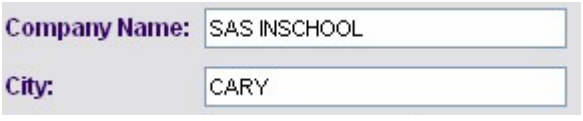
Gets the value of the selected state from the list

Gets the value of the selected city from the list

Refresh the screen with the selected value

THE INPUT FIELDS

The input fields are probably one of the most used objects. It can be used to either display 'read only' information or fields for inputting data. The data in these fields are being populated from a previously created SAS dataset.




```
<td align='left' class='mainLabel'>Company Name:</td>
<td><input readonly type='text' name='keyword1'
                  value="<%=prospect_name[0]%>" size='45'
                  class='listField2'></td>
```

Input box label

Input box

CONCLUSION

I hope this paper has shed light on how to convert some of your fat-client applications to thin-client applications. I believe SAS/FRAME developers have a unique advantage over those who have developed only web applications. It is up to us to embrace the new technology instead of being afraid of it.

ACKNOWLEDGMENTS

CONTACT INFORMATION

Jeff Lessenberry
Jeff Lessenberry Consulting Group
1906 Jonesville Road
Simpsonville, SC 29681-4231
864-967-4433
jlcg@charter.net