Introduction to Interactive Drill Down Reports on the Web

Michael G. Sadof, Bedford NH
Louis T. Semidey, San Francisco CA

ABSTRACT

Presenting data on the web can be an intimidating project for those who are not familiar with the technology. Luckily, SAS provides users with a method of presenting dynamic reports on a web server utilizing basic SAS syntax and even legacy reports. This workshop will walk you through several methods of utilizing Proc Report and will teach you some HTML code to present interactive reports on your web server. This technique requires you have several BI components running on your server but does not require advanced knowledge of SAS BI or HTML. In this workshop, we will start with the basics and develop a roadmap for producing dynamic reports with Stored Processes without using OLAP cubes or the OLAP Server. In the SAS BI environment, there are many ways of displaying data on the web. In this workshop, we will describe several of those techniques and create a project that utilizes them. The workshop will start by defining the necessary BI environment. We will continue by developing several simple reports at a detail and summary level, then enhancing with ODS and HTML to facilitate static drill downs and navigation. Then we will enhance the process by employing the SAS Stored Process Web Application to create dynamic reports. Finally we will introduce the use of AJAX (Asynchronous JavaScript and XML) for pages that do not require a full page refresh. This technique enables the development of rich internet applications. While the examples and code presented in the workshop will be simple the techniques can be leveraged and extrapolated to solve many real world presentation needs.

BI ARCHITECTURE

At the core of the SAS Business Intelligence (BI) Platform is the SAS Metadata Server. The metadata server basically controls all access to the metadata including users, libraries, reports and Stored Processes. Most of the metadata is stored in a set of SAS files known as the 'Foundation Repository'. The repository contains library definitions, report definitions, user logons, database passwords and much more. Our work here will be limited to identifying a library, defining a prompt and defining a Stored Process.

In the SAS Intelligence platform the engines that actually run SAS are called servers. There are many types of servers in BI one of which is the Stored Process Server which is designed to execute Stored Processes. A Stored Process is a SAS program that has been created in a certain way so that it can then be executed on the 'Stored Process Server' and send the results back to a web browser. We will be defining a stored process that 'streams' the data back to a web browser by utilizing the Output Delivery System (ODS). Figure 1 depicts the various layers in a typical BI installation. From the diagram it is seen that the middle tier or web server tier acts as a middle layer to communicate between the web browser and the server layer. In practicality once the layers are installed communication is handled by several very simple commands and the user only need know a simple set of syntax.

In order to run a report from a web browser the browser must communicate with the web server. In this case we will communicate with the web browser (middle tier) with the HTML or JavaScript language. We will formulate commands that can be interpreted properly by the Stored Process Server which in turn will process the SAS commands and subsequently deliver the report back to us in our web browser.

SAS has supplied several applications that run in a web browser. One of these applications is the Information Delivery Portal which can act as a portal for various types of web services. Another, which we shall utilize in this tutorial, is the 'SAS Stored Process Web Application'.

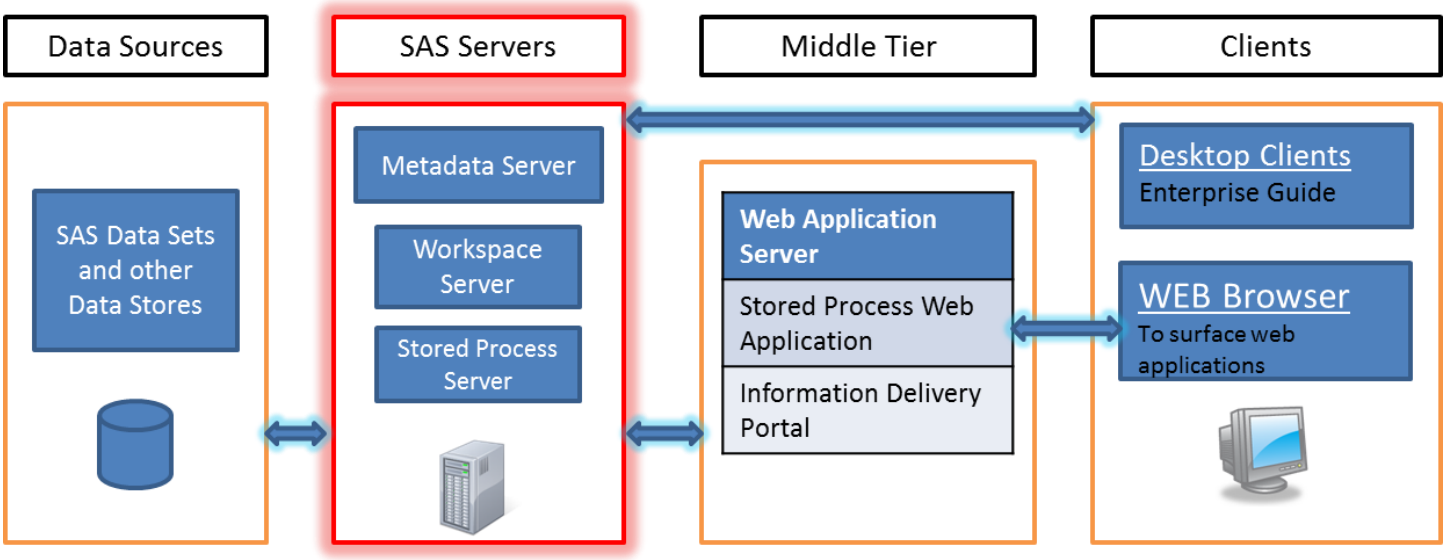


Figure 1. SAS BI Platform Architecture

SAS STORED PROCESS WEB APPLICATION

The syntax (actually just the URL) to execute this application from your web browser will depend upon which Web Application Server is installed at your site. Pre 9.4 version utilized WebSphere, WebLogic, or more commonly JBoss, but the current 9.4 version bundles the webserver in with the SAS installation and calls it SAS Web Server. These applications are installed at your site acting as a container and middle man to accept your commands and 'serve-up' your results--hence the name Web Server. The only thing important in this demonstration will be the URL. To demonstrate this we will logon to our SAS installation and fire up a web server session.

Please note that the examples in this written version of the paper were conducted on my own server and the addresses and locations will be different than yours. The sample screen prints may indicate different URL's and port numbers then you will see in this workshop or at your own site. Explanation of how to logon to this workshop's SAS installation can be found on a supplemental sheet. While SAS 9.3 and 9.4 BI installations are quite different under the hood the syntax remains the same and examples in this paper will work on either 9.3 or 9.4.

WEB SERVER DEMONSTRATION

In this workshop we will be signing on directly to the SAS server. At your site you will most probably log in to the client and access your SAS server with an IP address or DNS name. If you are logged in directly to the server you can refer to it by the name 'localhost'.

1. Logon to your local workstation
2. Use Remote Desktop to logon to the SAS Server (see supplement)
3. Open a browser (Internet Explorer) window
4. Type this command in the address bar

   http://localhost:80

   Note at your site you will most probably substitute your DNS name for localhost and different port number if necessary. See Appendix
   For instance at my site the SAS Server's DNS name is Dell-2950.mgs.local so I would logon as
   http://Dell-2950.mgs.local:81

5. You will see a screen similar to this one indicating that the SAS Web Application Server is running. This is an easy way to determine if the Webserver is up and running.

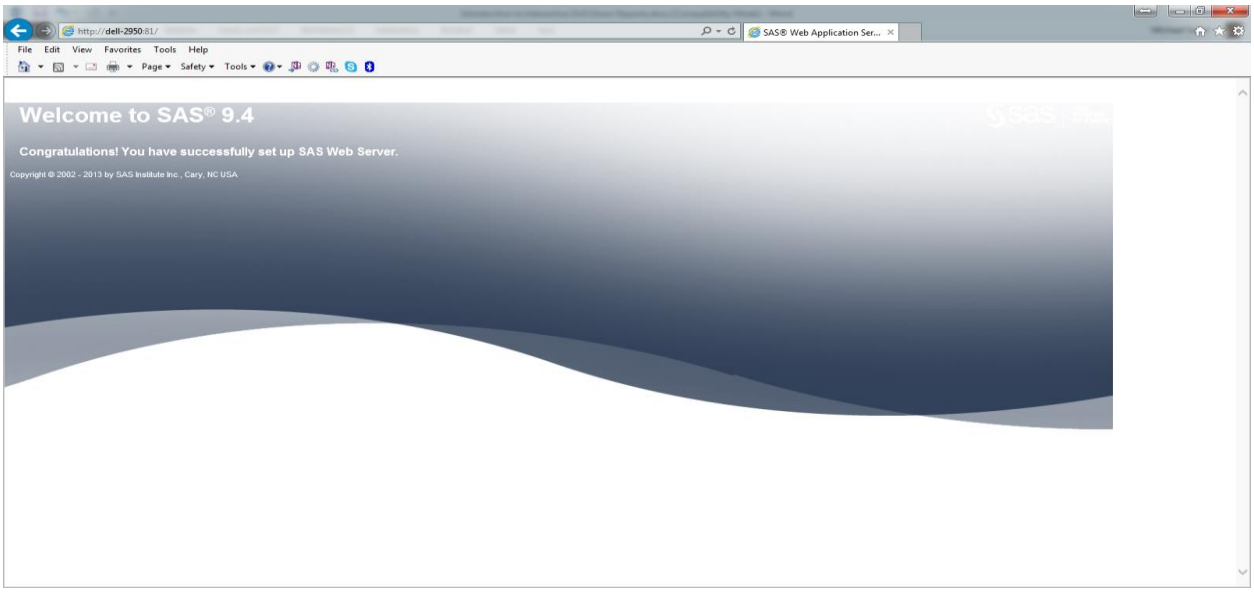


Figure 2. SAS Web Server Opening Screen

6. Next type this command into the browsers address bar to logon to the SAS Web Server Stored Process Application

   http://localhost:80/SASStoredProcess/do

7. This will bring up the familiar SAS LOGON Screen where you will enter your username and logon supplied on supplemental sheet.

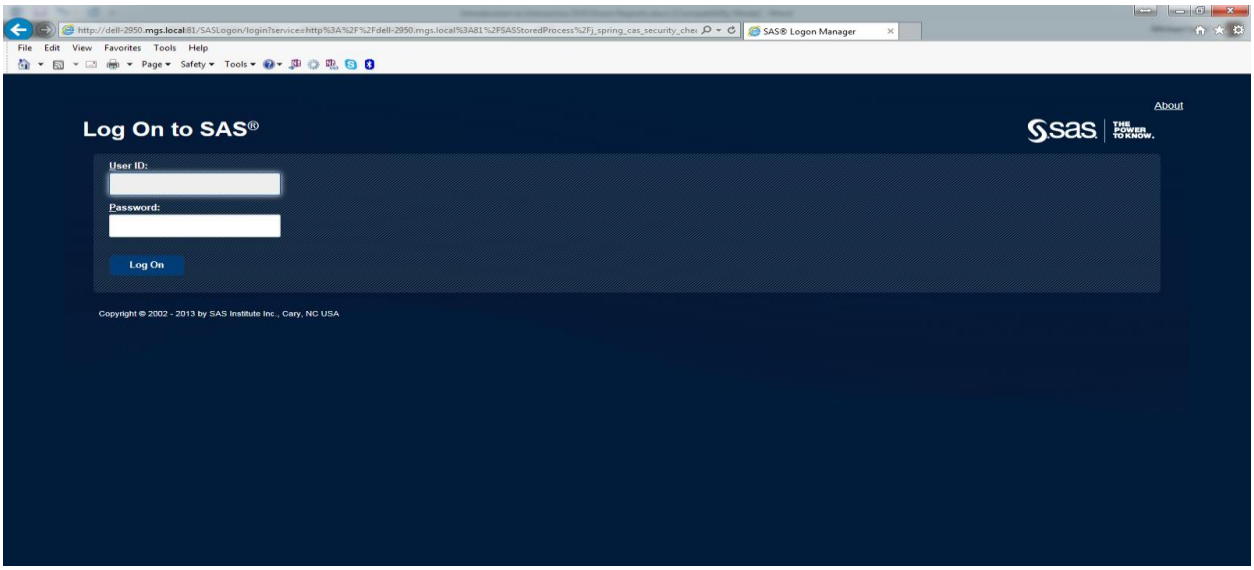


Figure 3. SAS Logon Screen

8. Next you will see the SAS Stored Process Web Application (Figure 3) where you may run a sample Stored Process. In our workshop we will be adding Stored Processes that can be surfaced through this application

9. Run the Sample: Shoe Sales by Region Stored Process by double clicking its name and explore by clicking some links. This will demonstrate how a SAS program in the background can be run from the web browser and can call other programs

10. Our next steps are to build a simple application like this and review ways to improve it.

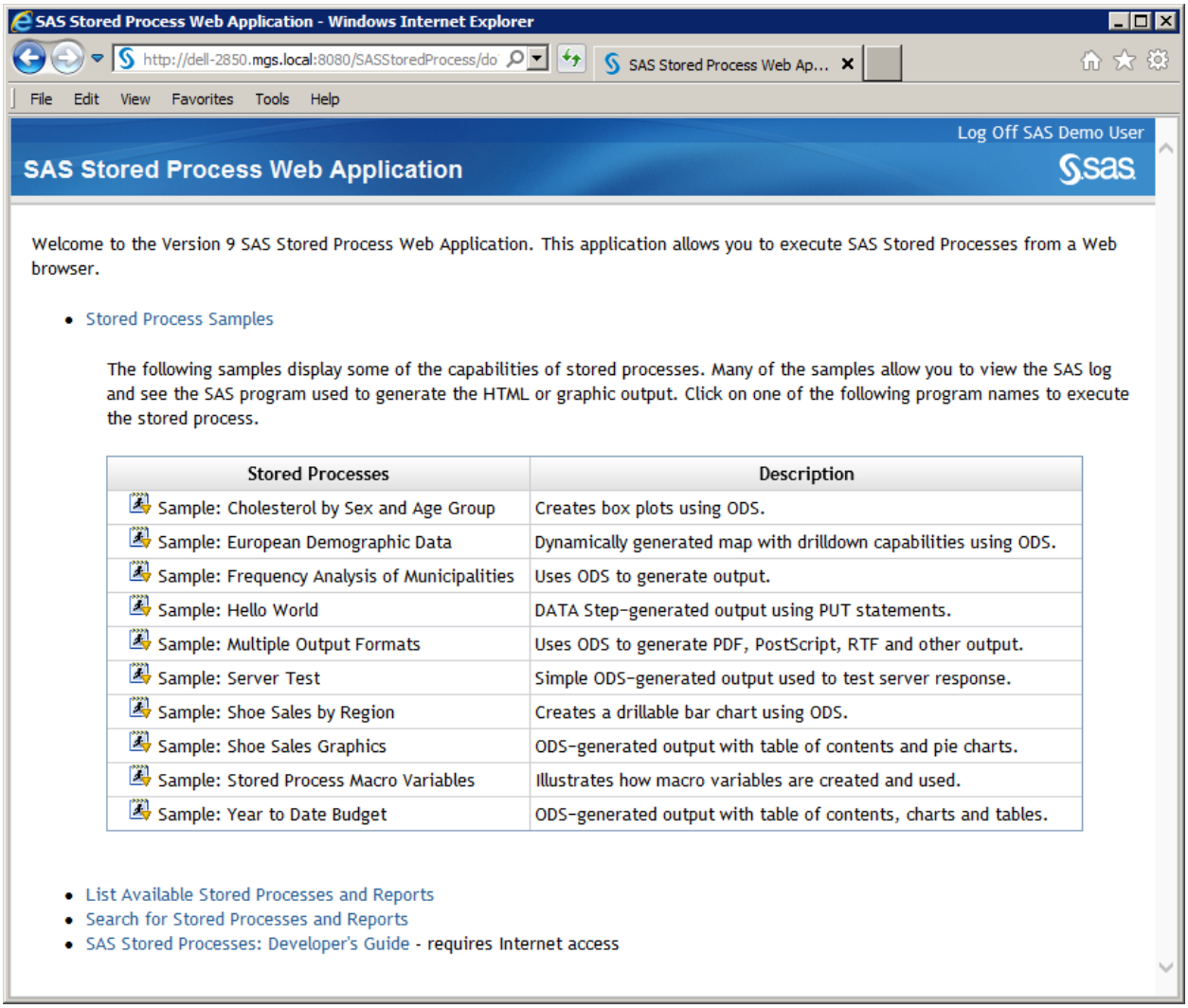


Figure 4. SAS Stored Process Web Application Sample Reports

SIMPLE PROC REPORTS FOR DEMONSTRATION

The technique we will be exploring today involves using ODS to create HTML pages that have embedded anchor tags which will use the HREF property to drill down to other reports.

- Proc Report
- ODS
- HTML Pages
- Anchor Tags
- HREF Property

Example1 - Demonstration of Proc Report

Example 1 creates several simple Proc Reports from the SASHELP data sets. It will get us familiar with the environment, setup and Proc Report.

1. Open up an Enterprise Guide Session
2. Open Example1 (location of programs on supplemental sheet)
3. Make certain that the output is set to HTML and SAS Report (Figure 4).
    a. Click on Toolbar Tools
    b. Click Options
    c. Click on Results General
    d. Make sure SAS Report and HTML are checked.
    e. Close window by clicking OK

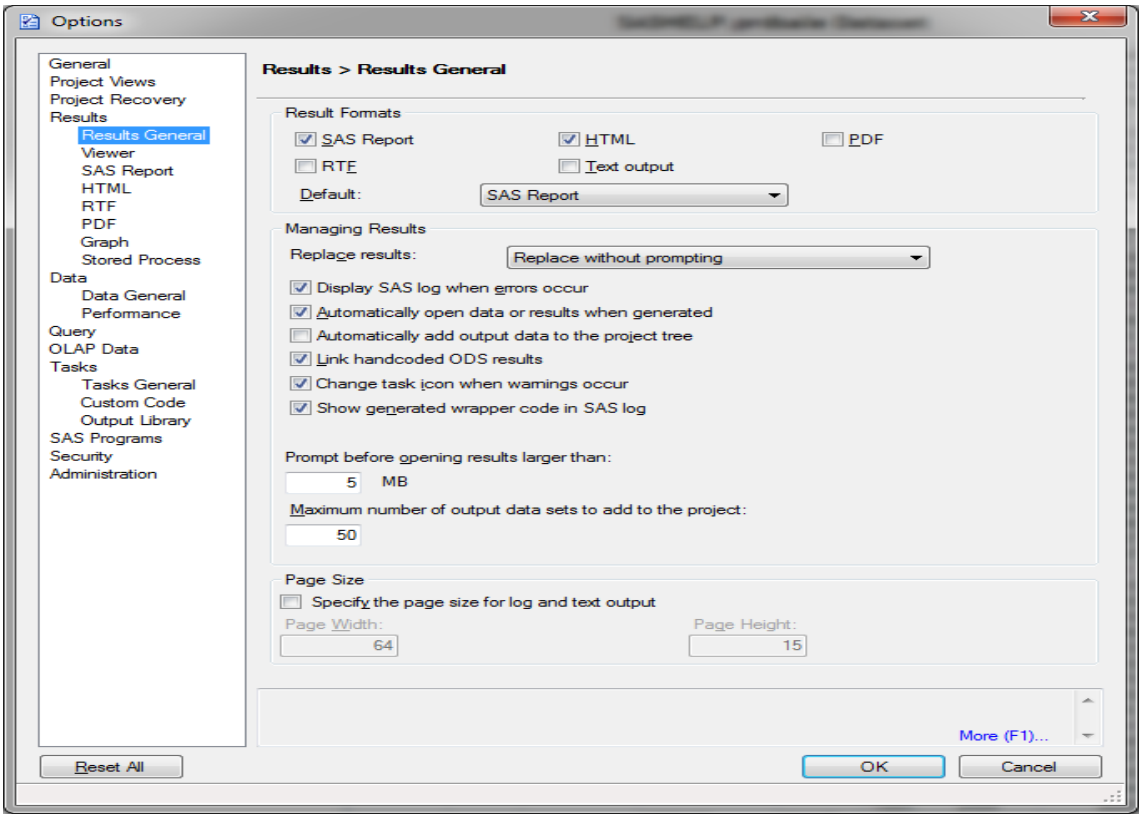


Figure 5. EG Options

4. Open the first program example labeled **`'Example 1.sas'`** (file/program/open)
5. Run each report separately and observe results
6. Highlight the first Proc Report labeled Example1a. This is a basic listing
    a. Right Click

b. Run Selection on SASApp

7. Observe the results produced.
8. Highlight and run the second Proc Report (Example 1b) and notice how much more improved the results look with groups.
9. Observe that the use of an across variable will produce a much more readable but still voluminous result.
10. In the third example (Example 1c) a new computed column (COUNTRY_NAME) is added to enable the customization of sub headers. Observe also that a COMPUTE section was also added. We will use this type of compute section to create our drill down links in later examples.
11. The next report (Example 1d) will use the NOPRINT option to eliminate duplicate columns and we use the computed column rather than the original country grouping. This is where we will eventually place our drill down links

Subsequent examples will be based upon the technique utilized in the Proc Report code from Example 1d.

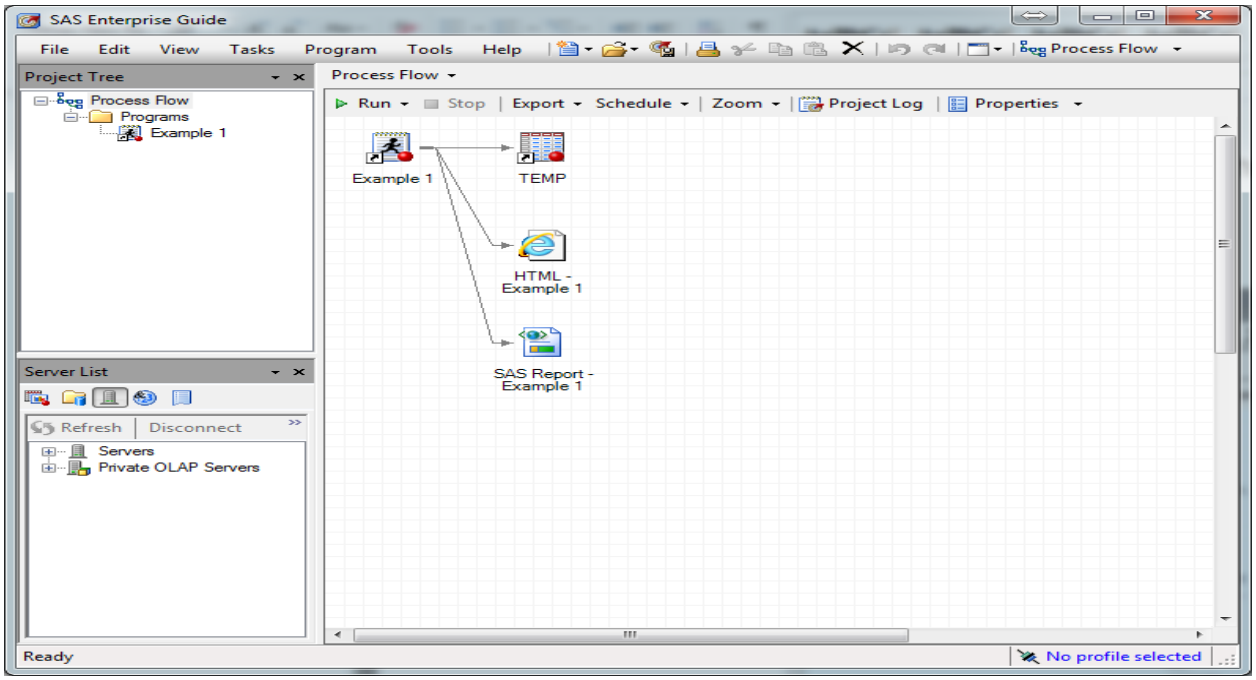


Figure 6. Sample EG Screen for Example 1

Example 2 - Introduce ODS

Example 2 will build upon the basic Proc Report from Example1d which will:

- Create detail and summary reports
- The summary report will be for all countries in the data set
- The detail reports will be for only one country

1. Open the program **'Example 2.sas'**. You will note it is very similar to Example 1d with the following additions:
   a. ODS statements
   b. Contains detail and summary reports
2. The summary report (Example 2a) should only contain the following columns:

```
Columns Country Country_Name Region Division Year , (Actual Predict);
```

11. The detail report (Example 2b) should only contain the following columns:

```
Columns Country Country_Name Region Division Prodtype product Year , (Actual
Predict);
```

12. The detail portion of the report contains a where clause to select only one country.

```
Data = sashelp.prdsale (where= (Country = "&country." ));
```

12. To run this program we have added a macro assignment (%LET) statement at the beginning of the program. You could also accomplish this with an EG prompt.

```
%LET COUNTRY = xxxxxx;  ** where xxxxxx is country name **;
```

13. The next enhancement is to add the ODS statements. These ODS statements should be placed directly after the option statement. Use the appropriate **body=** and **path=** clauses for each program summary or detail.

```
%global odspath;
%LET odspath=xxx;  * use valid path for your machine;
ods listing close;
ods html path="&odspath."
     body="&country._Rep2_detail.html";     /*use for detail */
  /* body="Report2_Summary.html"; */        /*use for summary*/
```

Demonstration (please run)

Run Example 2 – Detail report for each county in the sample dataset (CANADA, GERMANY, U.S.A.) This will create HTML files on the server that will be used in subsequent examples. You will note that the output in HTML is closer to what we would like but there are no drill-down links as yet.

Example 3 - Introduction of HTML Anchor Tags and HREF property

We will be surfacing the HTML version of these reports in our web browser. Happily SAS has built the HTML code for us. We just need to add a few tags which will create clickable links in our SAS-created HTML document.

The HTML Tag we will be using is the anchor tag which is surrounded by the **<A>** anchor open and **</A>** anchor close tags.

```
<A          > </A>
```

The HREF property will identify the URL. Insert the link to our HTML file within the quotes after HREF will enable the drill down. This needs to be in URL notation.

```
<A HREF=" " > </A>
```

The text before the anchor close tag </A> will be displayed on the HTML page.

```
<A HREF=" " >link text here without quotes</A>
```

We can then add the hint as a title property:

```
<A HREF=" " TITLE=" " > link text here without quotes</A>
```

That is the Anchor Tag in a nutshell, but obviously we need to a bit more work to get exactly what we want in the correct places and since we will be using macros the macro quoting functions come into play.

Open program **'Example3 – Navigation.sas'** and explore the Navigation Example. This example shows how to build clickable links within your title statement. Also introduced is the ODSESCAPE character that enables us to put SAS style specifications into the Title statement. It enables the use of various font sizes and colors and justification and most of the ODS style features. Run the "Example 3 – Navigation" program and observe that in the HTML output window you can see exactly what will be seen in your web browser. In the SAS Report window, which cannot interpret HTML code, you will see how the statement is built. This is not only instructive but very useful in debugging your code as the HREF's sometimes become very complicated. If you have run Example 2 correctly then these links should be clickable in the HTML version in your browser window. The anchor tag should look like this where 'XX' is a valid location on your hard disk like **"c:\"** or **"c:\temp"**. **Report2_Summary.html** is the name of the html file for the summary report. Other links in the output are displayed with similar structure.

```
<A HREF="XX\Report2_Summary.html" Title="Hint with double quotes" >Displayed
Link Text</A>
```

1

At this point we have most of the building blocks needed.

- Summary and Detail Proc Reports
- Computed Variables to hold links
- Anchor Tags and Navigation Bars

When an HTML anchor tag is placed on the report and the report is rendered to HTML it becomes a clickable link as in the navigation bar. All we need to do now is place the 'link' for the detail report into the compute block in the summary report so we can drill down to the detail from the summary.

Demonstration (please run)

1. Run **'Example 3 – Navigation.sas'**.
2. Go to **C:\HOW\Sadof\HTML_Output** in windows explorer and click on **C:\HOW\Sadof\HTML_Output\Navigation.html**

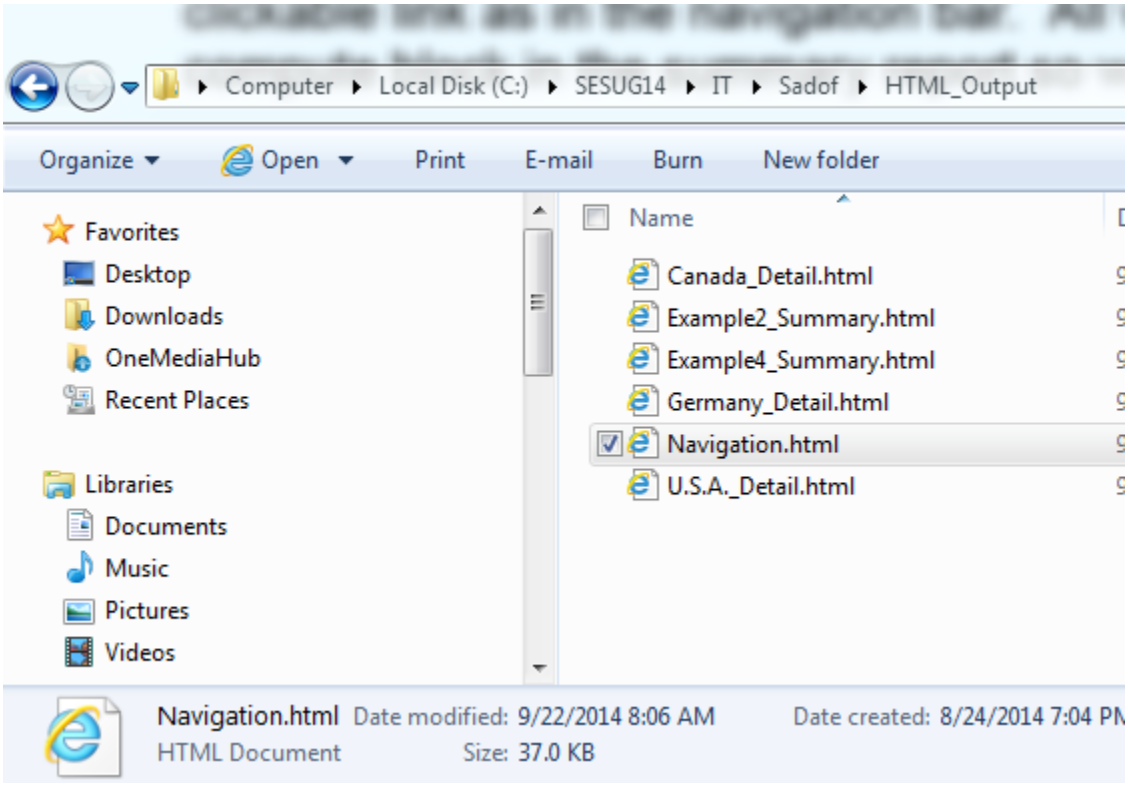


Figure 7. Use Explorer to surface HTML output.

3. You should be able to 'Drill Down' to the detail reports we have created in Example 2.
4. If you get the explorer message as shown below you can turn it off.
5. In IE go to Tools/Internet Options/Advanced and scroll down to turn on "Allow Active content to run in files on My Computer"
6. See Appendix for additional Internet Explorer (IE) settings.

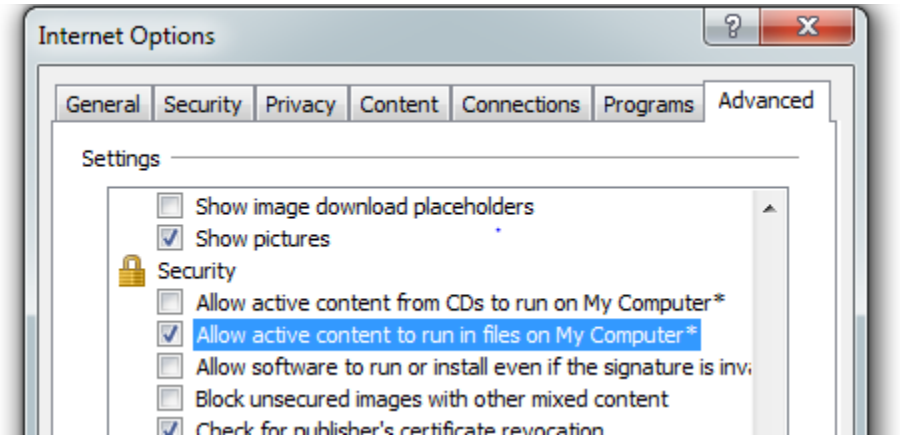


Figure 8. Allow Active Content.

Example 4 - The Drill Down

The detail and summary reports for Example4 are derived directly from Example 2. The detail report is essentially the same as Example 2 and the summary report introduces the "HREF's to the compute block so we can drill down to the detail report.

Summary Report:
The compute block for example 2 looks like this

```
compute Country_Name / char Length=50;
Country_Name=country;
      if _BREAK_='COUNTRY' then country_name=trim(Country)||" Total";
            else if _BREAK_='_RBREAK_' then Country_Name='Grand Total';
endcomp;
```

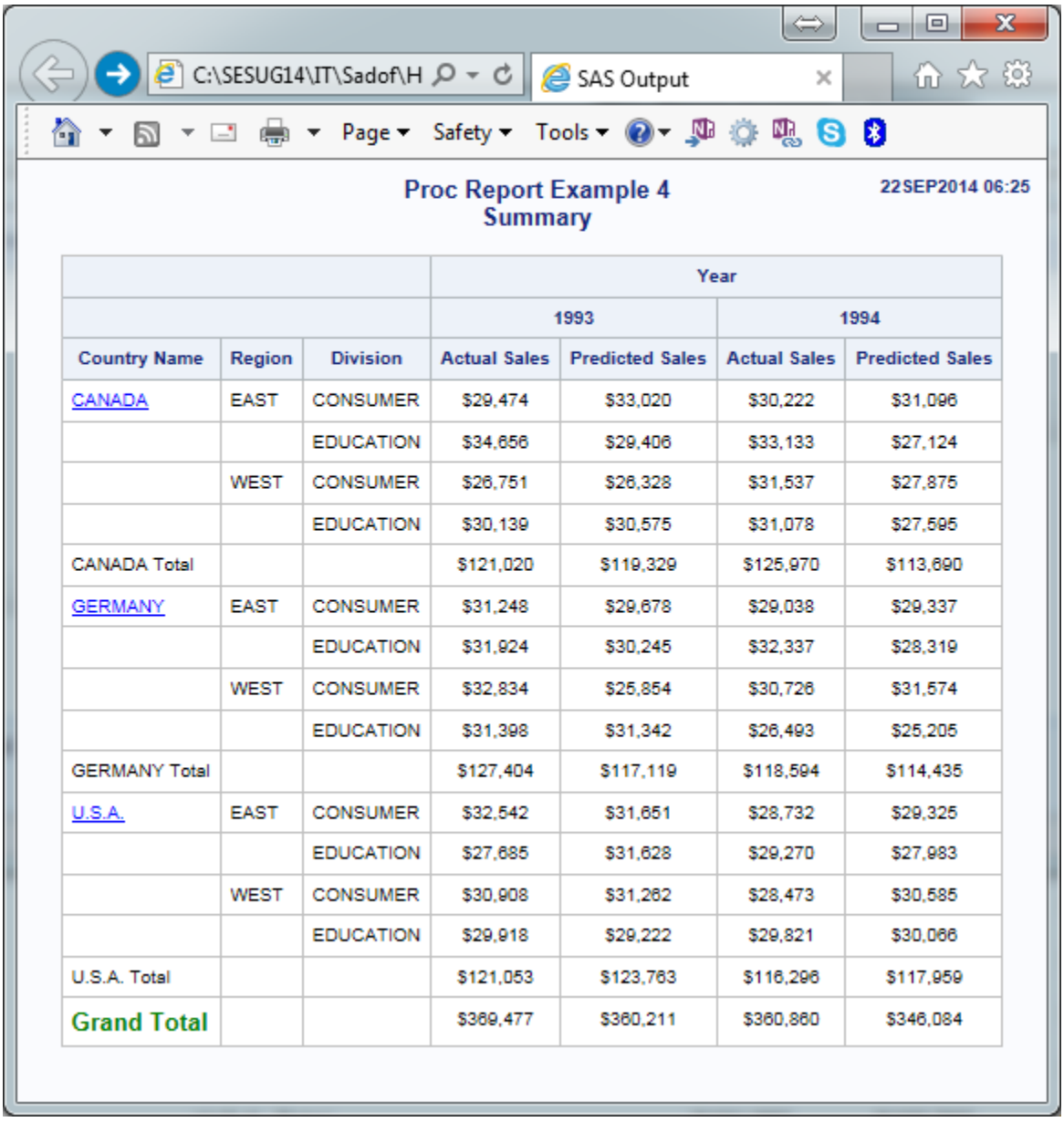
A few minor changes to the compute block will place the anchor tag and HREF directly on your report as the computed variable "Country_Name". I have also added a style statement to distinguish the "grand total" line in another color. You will notice the use of the concatenation symbols || to build the variable "**Country_Name**" as a string of characters with the appropriate quotes. Two quotes in a row **("")** has the effect of actually adding the quote to the string as we need for the HREF property of the tag. Spaces before or after the concatenation symbols || are ignored as are the line breaks. But be careful of spaces elsewhere in the text. Remember we want the name of the detail file to evaluate as **"Canada_Rep2_Detail.html"** so the TRIM(country) gives us the country name as retrieved from the group column.

```
compute Country_Name / char Length=150;* Need a variable with long length *;

*** Here is where we build the URL String for the Drill Down ***;
COUNTRY_name="<A HREF=""&odspath." || trim(country) || "_Detail.html""
Title=""This is the Hint"">" || trim(Country) || "</A>";

** Here is where we handle the report breaks **;
IF _BREAK_='COUNTRY' then Country_Name=trim(Country)||" Total";

IF _BREAK_='_RBREAK_' then DO;
      Country_Name='Grand Total';
      call define(_col_,'style', 'style=[FONT_WEIGHT=BOLD foreground=Green
font_size=3]');
      end;

endcomp;
```

Detail Report:
Remove the %LET for the countries. These variables will be passed through the HREF links.

Demonstration

1. Open and run the program **'Example 4 - Summary.sas'**.
2. You may not be able to click on the links from within EG so open up a windows explorer window and navigate to **c:\HOW\Sadof\HTML_Output** to view the output by clicking on **Example4_Summary.html**.
3. You will note that if you ran Example 2 correctly you will be able to drill down to detail from Proc Report. Note that once the detail report is surfaced you will have to use the browser back button to get back to the Summary Report in as much as we have not yet fully implemented the Navigation bar described in Example 3.



Proc Report Example 4
Summary

22SEP2014 06:25

			Year			
			1993		1994	
Country Name	Region	Division	Actual Sales	Predicted Sales	Actual Sales	Predicted Sales
CANADA	EAST	CONSUMER	$29,474	$33,020	$30,222	$31,096
		EDUCATION	$34,656	$29,406	$33,133	$27,124
	WEST	CONSUMER	$26,751	$26,328	$31,537	$27,875
		EDUCATION	$30,139	$30,575	$31,078	$27,595
CANADA Total			$121,020	$119,329	$125,970	$113,690
GERMANY	EAST	CONSUMER	$31,248	$29,678	$29,038	$29,337
		EDUCATION	$31,924	$30,245	$32,337	$28,319
	WEST	CONSUMER	$32,834	$25,854	$30,726	$31,574
		EDUCATION	$31,398	$31,342	$26,493	$25,205
GERMANY Total			$127,404	$117,119	$118,594	$114,435
U.S.A.	EAST	CONSUMER	$32,542	$31,651	$28,732	$29,325
		EDUCATION	$27,685	$31,628	$29,270	$27,983
	WEST	CONSUMER	$30,908	$31,262	$28,473	$30,585
		EDUCATION	$29,918	$29,222	$29,821	$30,066
U.S.A. Total			$121,053	$123,763	$116,296	$117,959
Grand Total			$369,477	$360,211	$360,860	$346,084

Figure 9. Output from Example 4 Summary Report

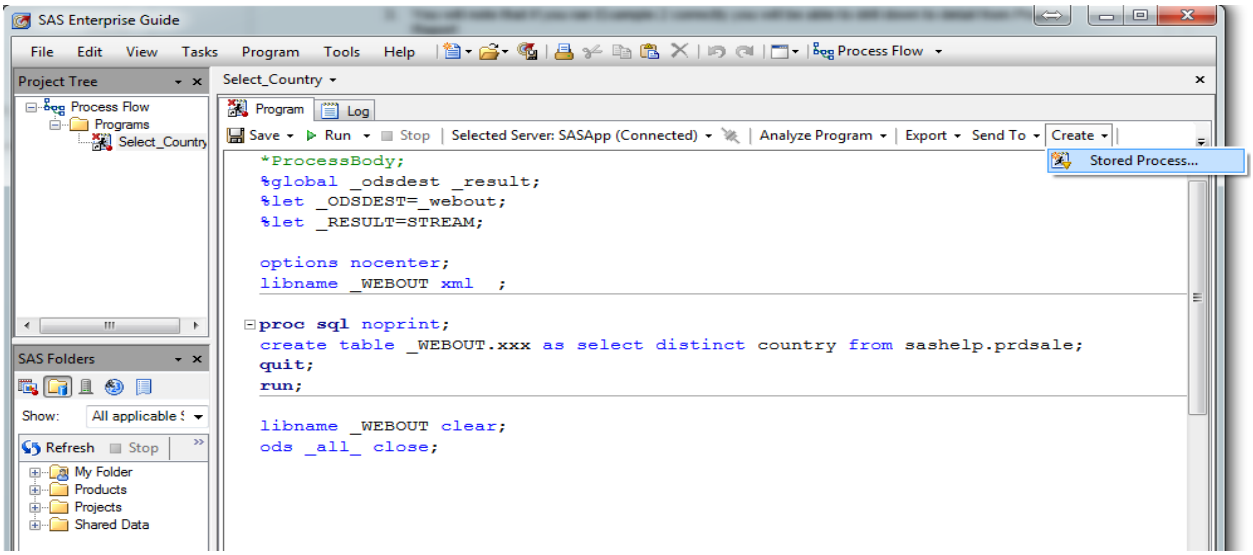
Notice the date and timestamp in title. We will use this in future examples to verify that a dynamic Stored Process is being used rather than a static HTML page.

In this example we have created series of static HTML pages with drill down capability. Please note that in this example the detail report have to be 'pre-created' before the drill downs will work and contain static data. Our dynamic example follows.

Example 5 - Build Stored Process

In Example 5 we will convert the static HTML reports to Stored Processes.
First let us create a very simple stored process that will produce XML output that we will use for the AJAX example. Please note that your metafolder names may vary depending upon installation.

1. Open the program **`'Select_Country.sas'`** .
2. This is a very simple program that selects a list of distinct countries from our "sashelp.prdsale" data set.
3. Note that we have added the `*processbody;` statement to indicate that this is the beginning of a stored process and assigned certain 'keyword' global macro variables.
4. The libname statement uses the XML engine to produce xml output.
5. The ODS destination is set to _webout a special destination used to stream output in a browser.
6. This program will not run correctly in the EG window.
7. To create the stored process we either
    a. Click on the create button or
    b. File/new/Stored Process



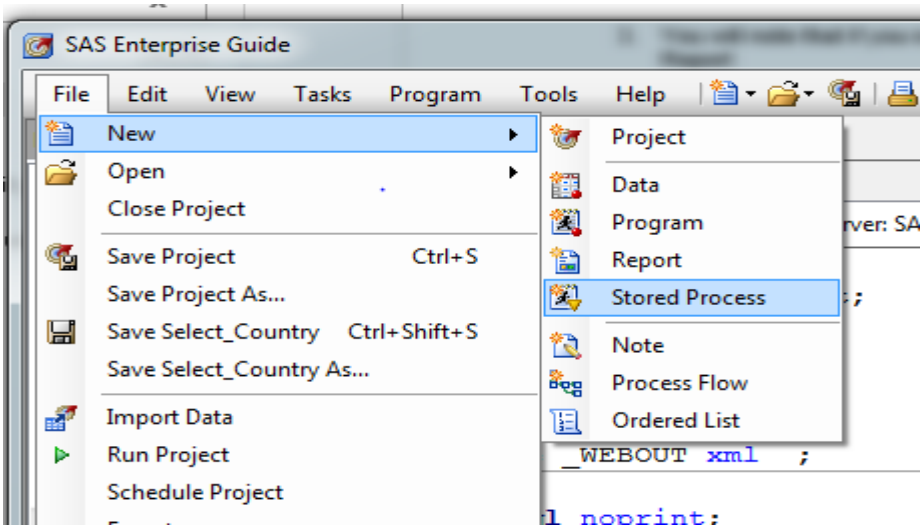


Figure 10. Create Stored Process in EG

8. Save the Stored Process code in metadata with the same name as the program. Use Figure 11 as a guide to set the parameters. Be sure to uncheck the items in the include code for section.

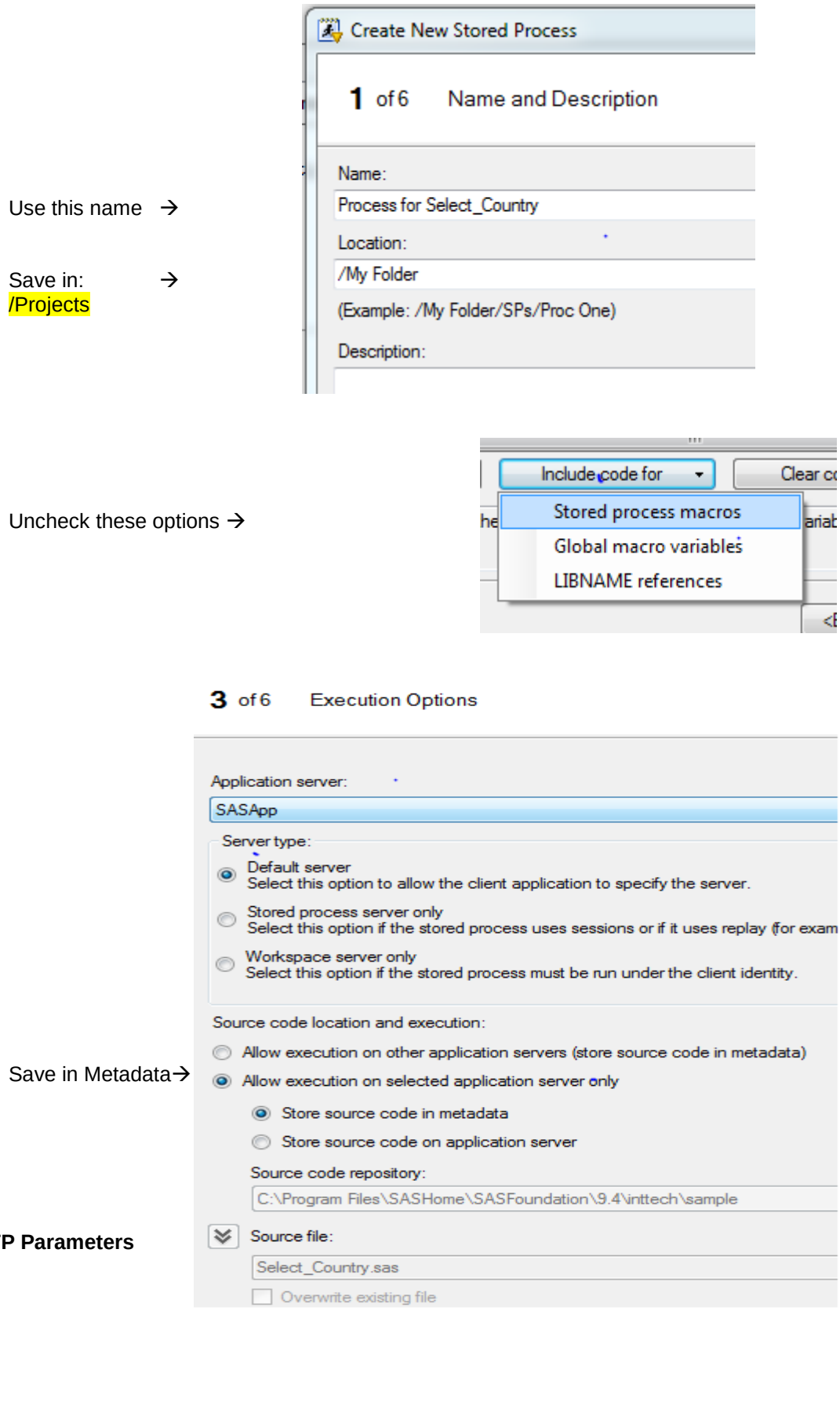


Figure 11. STP Parameters

Additional Note about Metadata locations:

The medatadata folder location varies from site to site. Generally it is better to create a separate folder for stored processes like **`/projects/pharma2015`** *but the user may choose to use the personal folders like* **`/My Folder/.`** *Throughout the remaining sections of this paper I use these specifications interchangeably. Note that it is easier to have a fixed location like* **`/Projects/Pharma2015`** *when trying to access from Javascript/HTML. We will use* **`/Projects/`** *in this tutorial for simplicity.*

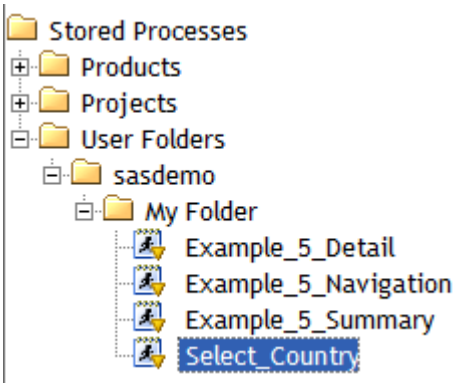
9. Once this stored process is created you can execute it from the Stored Process Web Application to test. You will not be able to execute in EG due to the _webout assignment we need for AJAX.
10. Go to browser window and enter the following in the navigation bar as we did in an earlier example and we should see the new stored processes that were just built.

`http://localhost:80/SASStoredProcess/do?`

11. Choose:

List Available Stored Processes and Reports

12. Drill down to Shared Stored Processes and you will see the stored process we just created **`'Select_Country'`**. Your username will show up rather than sasdemo.
13. Execute the program to test it. Available stored processes will look something like this.



14. The output will look like the following

15. Or possibly like this depending upon your browser

CANADA GERMANY U.S.A.

Example 5 – Add Stored Process links to create dynamic report

The heart of the drill down call is the next HREF statement we are about to build.

http://localhost:80/SASStoredProcess/guest?&_program=/Products/Example_5_Summary&rep_stat=Active&year=2014&_debug=LOG

The breakdown of the format of this URL which will call the SAS Stored Process Web Application to run your stored process is as follows:

http://	Start of URL	
localhost:80/	Server name and Port	localhost works for workshop since we are all signed on to our own virtual machine. You will have to use a valid DNS name or ip address
SASStoredProcess/	Name of SAS Application that will run our commands	Always remains same
guest?	Indicates guest user or regular logon which goes to the SAS logon manager. ? acts as a break between first part of URL and variable/value pairs that follow	We have set up the SAS Stored Process Web Application to accept guests without a password for demo purposes. See Appendix.
do?		
&_program	Variable to store name of program we are going to execute	Ampersand (&) and underscore(_) are required
Products/Example_5_Summary	Metadata path and name of program	Spaces are allowed but you can use %20 to substitute for spaces
Your%20Name%20Here/Example_5_Summary or possibly **Users/sasdemo/My Folder/Example_5_Summary**	No spaces in second example (if your metadata folder had spaces in the name)	In this example the name of the metadata directory has spaces in it
List of value pairs below		
&_program=/Products/Example_5_Summary	Stored Process program name	
&rep_stat=Active	These variables are passed as macro variables to the stored process as if we had a statement	

	like: %LET rep_stat=Active
&year=2015	Passed to STP
&_debug=LOG	Tells SAS to print LOG for debugging

Note: The name of the program is the metadata name of the program and not necessarily the name of the program on the file system which in this case would be "example_5_summary.sas". The name of the metadata folder varies between installations.

Exercise

Example_5_Detail

1. Open the program **'Example_5_Detail.sas'** .
2. You will note it is similar to Example 4 detail with the following changes:
   a. The ODS statements have been removed
   b. Title statement has been updated
   c. It is ready to be created into a stored process
3. Create a stored process from this code example and name it carefully as:
   Example_5_Detail
4. The name is important because we will need to call it from the HTML code.
5. Check the box to store the code in metadata.

Example_5_Navigation

1. Open the program **'Example_5_Navigation.sas'** (it is similar to Example 3 Navigation)
2. This is sample code to build a navigation bar at the top of any report and is used for illustrative purposes.
3. Create a stored process for this code and name it: **Example_5_Navigation**
4. Save it in the metatafolder: **/Products**

Example_5_Summary

1. Derived from Example 4
2. Modified title statements to contain links to store processes
3. Contains HREF links to stored processes in Proc Report
4. Changes are summarized below

Removed other **%LET** statements pointing to static HTML pages and added these **%LET** statements to point to stored processes rather than static html.

Added this **%LET** statement as shortcut to server name and port and URL

```
*Global Macro for server name and port;

%LET tsession=http://&_SRVNAME.:&_SRVPORT.&_URL.?;
```

These specifications for program names and metafolder names vary by installation and site. Macro names are used to facilitate changes. Set up your folder names here as necessary.

```
*Metadata Folder and Names;
%LET &metafolder=&_metafolder; * user folder *;
*%LET &metafolder=Products;    * Fixed folder *;
%LET detpgm=&metafolder.Example_5_Detail;
%LET navpgm=&metafolder.Example_5_Navigation;
%LET sumpgm=&metafolder.Example_5_Summary;
```

Title statements for navigation to point to the stored processes.

```
Title4 j=c h=2
      '<A HREF="' "%SUPERQ(TSESSION)" '&_ACTION=EXECUTE'
      '&_program=' "&sumpgm"
      '&repstat='  "Active"
      '&year='     "2013"
       ' Title="Hovering Hint" '
      '>Link to Stored Process Summary </A>'
  h=2 ' | ' h=2
      '<A HREF="'"%SUPERQ(TSESSION)"  '&_ACTION=EXECUTE'
      '&_program=' "&navpgm"
      '&repstat='  "Active"
      '&year='     "2013"
       ' Title="You can place text here for a Hint when Hovering" '
      '>Link to Navigation Stored Process</A>'
  h=2 ' | ' h=2
      'Place Holder for Other Reports'
  h=2 ' | ' h=2
      'Place Holder ';
```

Changed definition of link in Proc Report

```
COUNTRY_NAME='<A HREF="' || "%SUPERQ(TSESSION)" ||
       '&_program='|| "&detpgm" ||
       '&repstat=' || "Active" ||    '&year=' || "2013" ||
       '&country=' || trim(country) ||
       '&_debug='  || "LOG" ||'"'||"
          Title=""Detail Report"" >"||trim(x)||"</A>";
```

Open up the Stored Process Web Application and navigate through our newly created example.

Note that if you use this link you will be prompted for userid and password

http://localhost:80/SASStoredProcess/do?

But if you use this link you will be taken directly to the application because we set AllowGuest to be true in Management Console

http://localhost:80/SASStoredProcess/guest?

Example 6 - Simple HTML Example

The next example is a very simple HTML program that can be used to call a stored process. Illustrating how to create the < A HREF> anchor tags. Please note you will have to adjust for your particular server and user name.
The HREF for the select country stored process would look like this

```
<a
HREF="http://localhost:80/SASStoredProcess/guest?&_program=/Products/Select_C
ountry" Title='Select Country'>Select Country </a>
```

Example 7 - AJAX, Javascript, and HTML

Our next and final example will show how we can run these stored processes directly from an HTML page or website that you have created. It has provisions for prompting and even a live lookup to the data to

assist in prompting and provide valid choices. The example uses quite a bit of Javascript to provide an example for those who wish to build their own pages. However the primary point of this example is to show that we can execute a SAS Stored Process from an HTML web page and can return data without a complete page refresh to provide live feedback as to the user's choices.

Ajax is an acronym for **A**synchronous **J**avaScript **A**nd **X**ML. With Ajax, web applications can send data to, and retrieve data from, a server asynchronously (in the background) without interfering with the display and behavior of the existing page. This is a summary of the basic AJAX commands.\:

XML data can be retrieved using the XMLHttpRequest object:

```
'xmlhttp=new XMLHttpRequest();
```

And the call to the SAS Stored Process Web Application is accomplished using the open and send methods of the XMLHttpRequest Object where *mytest* represents the call to the STP.

```
xmlhttp.open("GET","http:/mytest/do?",true);
xmlhttp.send();
```

The example **'JAVA_AJAX.html'** utilizes this technique to retrieve the valid country names in a box for the user to choose from. At this point we will utilized the "Select_Country" stored process described and built previously.

This program will create an XML stream (not a file) with a list of the counties in our PRDSALES dataset on sashelp. This example and stored process was created previously (**Select_Country.sas**). The strange formation of the libname statement is to put the XML out to a stream and not a fixed file. Notice we define the "_WEBOUT" library with the XML engine, then write some randomly named table. Here, the table is named: "table" and the SAS library "_Webout". The Javascript in **'JAVA_AJAX.html'** will execute the stored process and read that XML data as an input stream and populate a text box.

The example presented here were derived in part from a great tutorial website:

http://www.w3schools.com/ajax/default.asp

http://www.w3schools.com/ajax/ajax_xmlhttprequest_send.asp

Link to SAS Documentation of _webout.table statement:

http://www.sas.com/offices/europe/uk/support/sas-hints-tips/stored.html

Examine the HTML **'JAVA_AJAX.html'** with notepad. The beginning of the code starts with <script type="text/javascript"> which represents the javascript and javascript functions portion and ends with </SCRIPT>. The HTML portion begins midway down at <! Start Building Page->. The first division has a navigation bar similar to what we built in Example 5. The anchor tag and HREF property to create a link to a stored process looks as we would expect:

```
<a
HREF="http://localhost:80/SASStoredProcess/guest?&_program=/Products/Example_5_Navigation&year
=2013&repstat=html" Title='Hard Coded Link to Navigation STP'>Navigation Bar1</a>
```

The second division has some JavaScript to create functions that will utilize the XMLHttpRequest object to retrieve data from the STP called "Select_Country". The rest of the HTML creates a small table and several examples of input for us to use. Upon completion of input and pressing the Submit button the HTML page sends its request to the SAS Stored Process Server. Notice the use of "POST" rather than "GET". SAS seems to run faster with POST but works with GET as well.

The third division displays the selections and uses a submit button to submit request.

Javascript functions are employed to eliminate as much as possible the need to enter URL's in various places within the page and as an example for you to explore various types of entry.

APPENDIX

This section describes how to configure the SAS Stored Process Web Application to allow guest use without passing a username and password. This is a link to the SAS 9.3 documents that describe these procedures.

http://support.sas.com/docume`ntation/cdl/en/stpug/62758/HTML/default/viewer.htm#n0ax7wadghvldwn1bbarfuly6adl.htm

This is the 9.4 link:

http://support.sas.com/documentation/cdl/en/stpug/62758/HTML/default/viewer.htm#webappcfg.htm

Starting with SAS 9.2, users can run stored processes without having to log on. A guest user name can be defined to run stored processes under a fixed account. The guest user name and password are specified in the SAS Stored Process Web Application initialization parameters.

To enable guest access, the SAS Stored Process Web Application initialization parameter AllowGuest must be set to true. Use the Configuration Manager in SAS Management Console to set this parameter.

Expand the Configuration Manager group on the Plug-ins tab in SAS Management Console.
Right-click the Stored Process Web App 9.3 node and select Properties.
In the Properties dialog box, click the Advanced tab.
Double-click the property value for the AllowGuest property, and change the value to true in order to grant guest access to the Anonymous Web User.

After you modify the advanced properties for the SAS Stored Process Web Application in the Configuration Manager, you must stop the Web application server, restart SAS Remote Services, and then start the Web application server.

You may access the application with this type of link as we have done in our application.
If ALLOWGUEST is true, then the following command will bypass the logon screen and take you directly to the application using the ‘webanon’ username and password

http://yourserver.com:80/SASStoredProcess/guest?

Here are some screen shots from the Management Console.

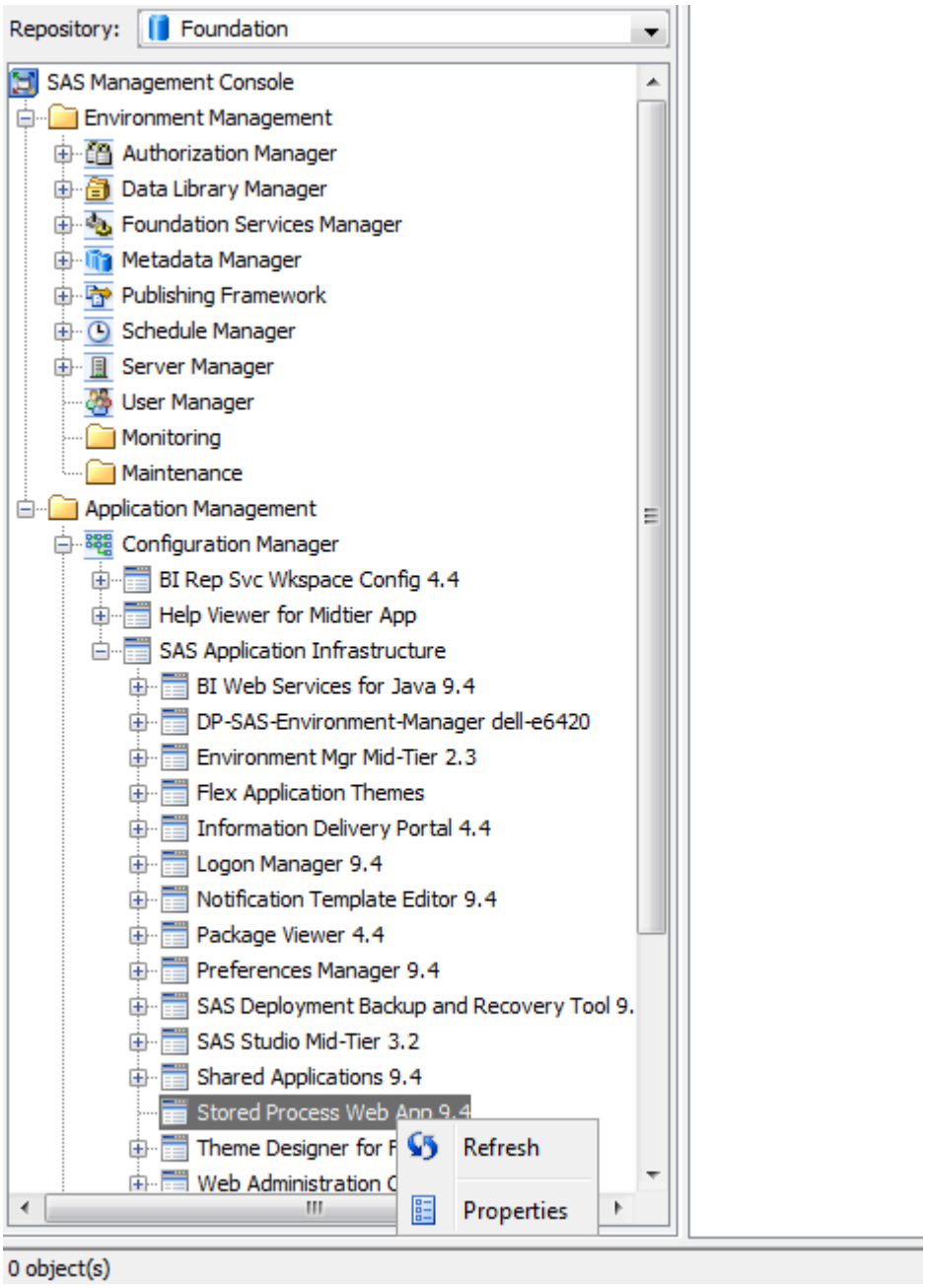


Figure 9. Stored Process Web Application – Configuration Management

Right Click and Choose Properties
On Internal Connection Tab you will see setting for hostname and port number
Then on advanced Tab you will see settings for guest
Add an entry for AllowGuest and set value to be true

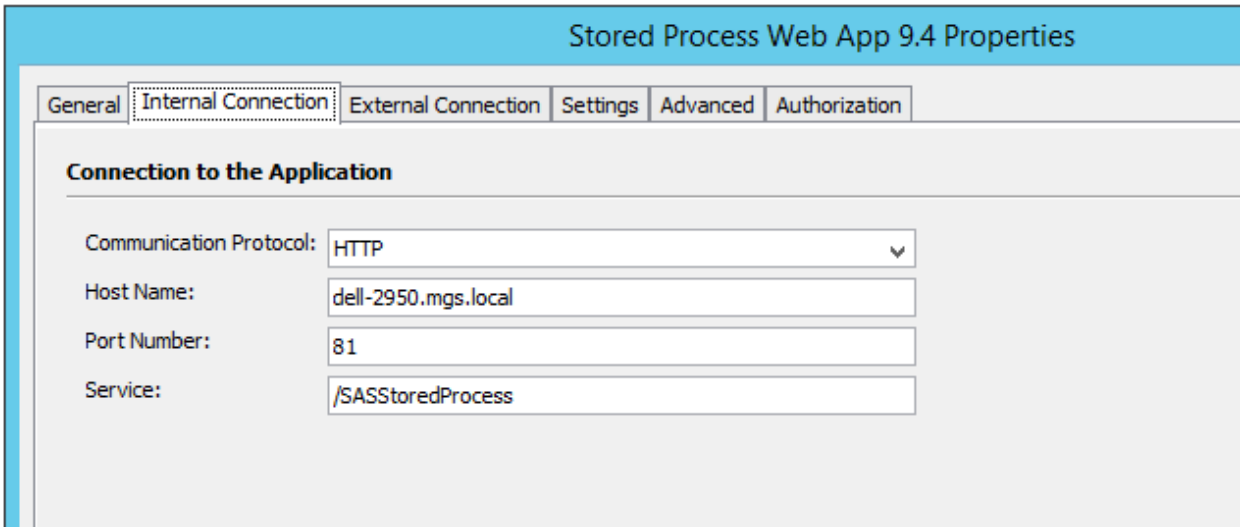


Figure 10. Properties of the Stored Process Web App 9.4 – Host and Port Number

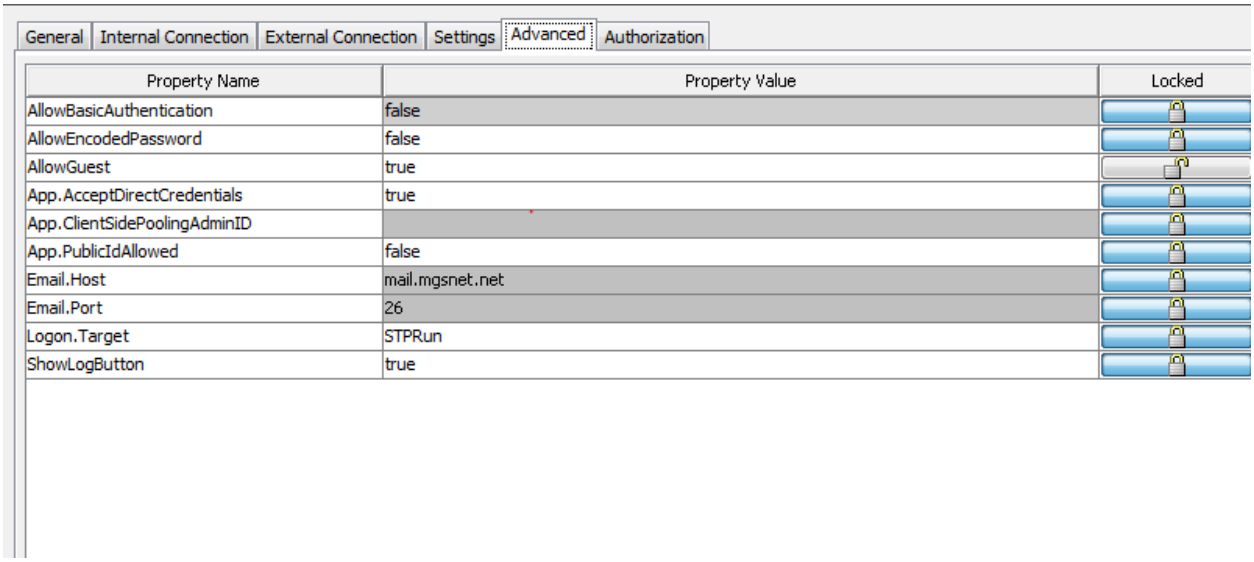


Figure 11. Setting of AllowGuest property and adding GuestPassword and GuestUsername if desireable

If you have trouble getting the AJAX to return into the window you may have to set your Internet Explorer to include your SAS Mid-Tier Server (Webserver) to be in your trusted zone,

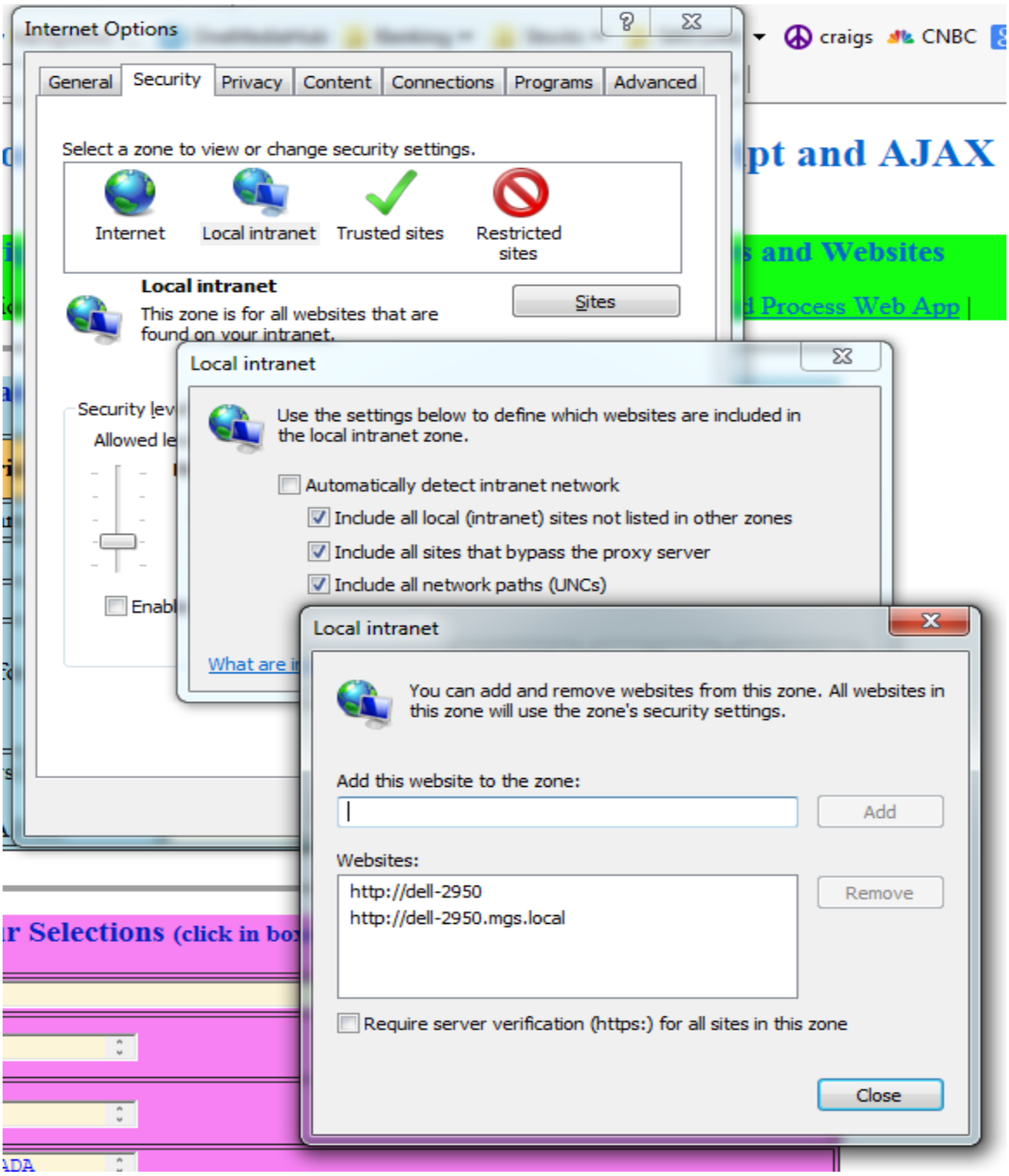


Figure 12. Setting Trusted Zone for local intranet operations in Internet Explorer

CONCLUSION

By combining Proc Report's capability of utilizing HTML anchor tags and the SAS Stored Process Web Application's ability to run a stored process we can develop dynamic drill down reports that can be surfaced on a web page. This technique involves the combination of ODS and HTML. Relatively simple HTML can provide the user with a prompting framework. More advanced HTML, JavaScript, and Ajax can provide the user with a dynamic prompting.

REFERENCES

Churchill, Alan*, "AJAX and SAS®: Smooth Web Applications"*, SAS Global Forum 2009

Carpenter, Art. *Carpenter's Complete Guide to the SAS® Proc Report Procedure*, SAS Institute, Cary, NC. 2007

Fallon, Jeffery A. Fallon, "*Using AJAX and SAS® Stored Processes to Create Dynamic Search-Suggest Functionality"*, SAS Global Forum 2010

Hamilton, Jack. "*Using the Compute Block in Proc Report",* Proceedings of the 13th Annual Western Users of SAS Software, 2005

Haworth, Lauren E., Zender, Cynthia L., and Burlew, Michele M. *Output Deliver System: The Basics and Beyond.* SAS Institute, Cary, NC. 2009

Mason, Phillip, "*Using SAS® Stored Processes with JavaScript and Flash to Build Web-Based Applications"*, SAS Global Forum 2011

SAS Institute Inc. *SAS® 9.3 Stored Processes: Developer's Guide*. SAS Institute Inc., Cary, NC. 2011

http://www.w3schools.com/ - Reference for Javascript and XML online

http://www.w3schools.com/ajax/ajax_xmlhttprequest_send.asp

http://www.sas.com/offices/europe/uk/support/sas-hints-tips/stored.html - SAS UK Stored Process Hints and Tips

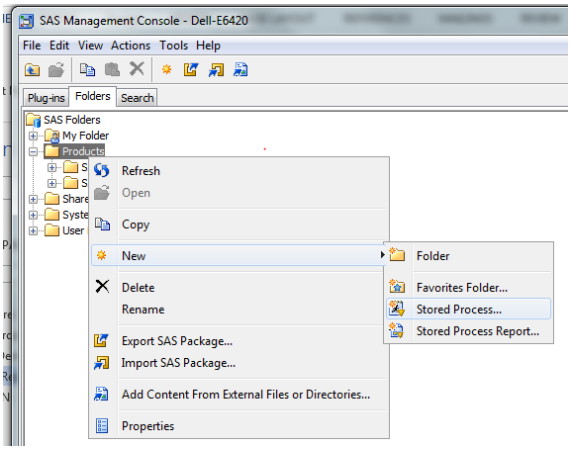
CONTACT INFORMATION

Comments, questions, and additions are welcomed.
Contact the authors at:

Michael G. Sadof
MGS Associates, Inc.
Bedford, NH
mgs@mgsnet.net

Louis T. Semidey
The Semidey Group
San Francisco, CA
ltsemidey@aol.com

TRADEMARKS


SAS Management Console - Dell-E6420
File Edit View Actions Tools Help
Plug-ins Folders Search
SAS Folders
My Folder
Products
Refresh
Open
Copy
New
Delete
Rename
Export SAS Package...
Import SAS Package...
Add Content From External Files or Directories...
Properties
Folder
Favorites Folder...
Stored Process...
Stored Process Report...