Paper BI03-2013

Building a dynamic SAS HTML report With JavaScript and SAS® Tagsets

Mike Libassi, Elsevier Corporation, Dayton, Ohio

ABSTRACT

Generating dynamic reports with SAS HTML Tagsets allows the end-user to manipulate the output to meet their needs. When we add a dynamic framework to the front-end of report generation (such as running the report for a selected date or date range) we add even greater flexibility to the report. This paper covers the process used in building this framework and contains the following sections:

- Example report page components – Review the HTML page components that can be used in this process.
- Example report output - Shows example result of the SAS Tagsets format.
- Summary of data and execution process – A summary of the steps used with data selection and how it's passed to SAS for processing.
- Framework – This section covers the details of the HTML, JavaScript and SAS code used in report generation.

INTRODUCTION

The SAS Tagsets framework allows for a dynamic and organized report output. Adding the use of HTML and JavaScript with the Tagsets framework allows for dynamic query and results. This empowers report end users in both report generation and the output manipulation. This can be achieved with a novice level of HTML, JavaScript and SAS experience.

EXAMPLE REPORT PAGE COMPONENTS

Query webpage components are added to allow the end user to select the data for the report to process on. Figure 1 is an example of an HTML date selector with a JavaScript link that allows a user to select a date for the report.

Figure 1

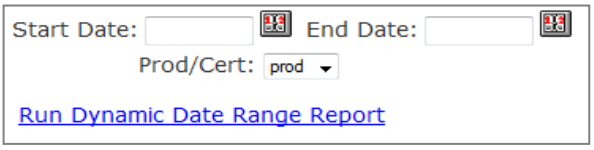


Figure 2

Other HTML elements can be added as needed. For example Figure 2 uses a start and end date along with a selection for the environment to run the report against. The data from these selections are passed into SAS on report generation when clicking the "*Run … report*" link on the page. JavaScript is used for the link to generate the URL that calls to SAS to generate the report.

EXAMPLE REPORT OUTPUT

The SAS Tagsets add many features to the report output, such as:

- Filtering
- Sorting
- Tabs
- Row highlighting

BKAPI Throttling for 25JUN13 (Click Column Header to Sort)

CSV version

eHour	Gadget_Name	Throttle_Type	apikey	requestId	response_Mean	APIKey_Throttled_Mean	APILevel_Th
(all)	(all)	(all)	1182 9	(all)	(all)	(all)	(all)
10	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.282	0.000248	
11	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.292	0.000268	
9	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.316	0.000272	
4	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.196	0.000256	
5	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.220	0.000224	
12	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.230	0.000206	
8	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.269	0.000266	
13	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.211	0.000233	
7	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.221	0.000255	
14	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.176	0.000247	
6	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.170	0.000225	
15	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.150	0.000258	
3	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.142	0.000259	
16	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.138	0.000172	
2	Protein Viewer	Unlimited	1182 9	ArticleRetrievalAPI	1.123	0.000233	

Figure 3

The tagsets output in Figure 3 allows the end-user to sort and filter the table results. SAS Tagsets have several options presented in previous works and on the SAS website (support.sas.com/rnd/base/ods/odsmarkup/tableeditor/).

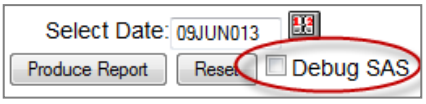
SUMMARY OF DATA AND EXECUTION PROCESS

The process in generating dynamic output for a selected input (example here is the selected date) is as follows:

1. Date is selected on report request page and the date selected is set to variable xdate.
2. Submit link is clicked using JavaScript. This generates a URL for sasweb/cgi-bin/broker with the variable passed (xdate) in and executes a SAS script.
3. The SAS script uses the Tagsets framework to produce a dynamic output.

The example here is using JavaScript to generate a date. Other HTML form elements (see www.w3schools.com/html/html_forms.asp for examples) can be utilized (like check boxes) to customize report parameters that are passed into SAS. An example is using a checkbox to set DEBUG=0 to DEBUG=131 when the checkbox is selected.

Example: `<input id="_DEBUG" name="_DEBUG" type="checkbox" value="131">Debug SAS`



The framework for sending a selected date and generating the tagset's output is as follows.

THE FRAMEWORK

SAS TAGSETS IMPLEMENTATION

The tableeditor.tpl file is copied to the SAS server for use. The table editor package can be downloaded from SAS at support.sas.com/rnd/base/ods/odsmarkup/tableeditor/ (the page contains the ZIP file to download). It is recommended to download to a client system and uncompressed the file. Then review the readme.txt and index.html that is included in the zip file.

The Tagsets file is referenced (as a filename) as an http link to the .tpl file. SAS Example 1 shows where we reference the tableeditor.tpl as a file name "temp" from a URL reference. Temp is then used with the `%include` to allow for use in the SAS code.

```
filename temp url "http://<server>/cpm/prod/AMP/sas/prod/tableeditor.tpl";
ods path(prepend) work.templat(update);
%include temp;
```

SAS Example 1

Tagsets use an ODS reference with a `_webout` for the file (SAS Example 2). This directs the output to the web browser of the requesting client. (See SAS reference on ODS for further information at: support.sas.com/rnd/web/intrnet/dispatch/ods.html).

```
ods tagsets.tableeditor file=_webout
```

SAS Example 2

Tagsets have several options to modify the format and options of the output.

```
options(web_tabs="BKAPI Throttling Data Table"
       window_title="BKAPI Throttling"
       highlight_color="yellow"
       scrollbar_color="beige"
       title_style="normal"
       image_path="http://<server>/cpm/prod/AMP/html/amp_report_top.png"
       header_size="12"
       image_just="left"
       left_margin=".5"
       right_margin=".5"
       top_margin="1"
       bottom_margin="1"
       autofilter="yes"
       sort="yes"
       sort_arrow_color="green"
       ) style=statistical;
```

SAS Example 3

The Tagsets options in SAS Example 3 are few of the available options that can be used. See support.sas.com/rnd/base/ods/odsmarkup/tableeditor/ for a full list and choose options that work best for your report.

JAVASCRIPT FUNCTIONS

The front end of our process is the initial report page where the customer chooses a selected date to use for the report. This page uses JavaScript to generate a date string in SYSDATE9. format and the URL to run the SAS and generate the output.

```
function sdates()
{
  var defdate = new Date(new Date() - 86400000)  // Yesterday
  var defday = defdate.getDate();
  var defmonth  = defdate.getMonth();
  var defyear = defdate.getYear();
  monthstr="";
  switch(defmonth)
  {
       case 0: monthstr ="JAN"; break;
       case 1: monthstr ="FEB"; break;
       case 2: monthstr ="MAR"; break;
       case 3: monthstr ="APR"; break;
       case 4: monthstr ="MAY"; break;
       case 5: monthstr ="JUN"; break;
       case 6: monthstr ="JUL"; break;
       case 7: monthstr ="AUG"; break;
       case 8: monthstr ="SEP"; break;
       case 9: monthstr ="OCT"; break;
       case 10: monthstr ="NOV"; break;
       case 11: monthstr ="DEC"; break;
  }
if (defday < 10) defday = "0" + defday;
defyear = "0" + defdate.getYear()%100;
document.getElementById("XDATE").value = defday+monthstr+defyear;
}
```

JavaScript Example 1

In JavaScript Example 1 we set up a function to set the "xdate" variable. The third party date picker used (Figure 4) does not return a "MON" formatted month. To fix the format the date picker month, the variable "defmonth" is run through a switch statement (seen in JavaScript Example 1) to format the numerical month to a "MON" format. This will give us a DDMONYY formatted date that the SAS code will accept. When using a third-party date selector be sure to evaluate any formats returned to ensure it will be acceptable by your SAS script.

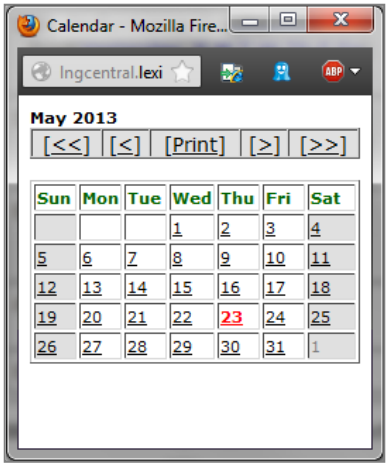


Figure 4

This date selected is assigned to the JavaScript variable "startDate". Then set to the SAS variable "XDATE" in a JavaScript function that generates the URL (see JavaScript Example 2) that is sent to SAS web broker.

```
<SCRIPT language=JavaScript> function RunDynamicT()
{
 var startDate = document.getElementById('XDATE').value;
 window.location = "http://<server>/cpm/sasweb/cgi-
bin/broker?XDATE="+startDate+"&ProduceReport=Produce+Report&_SERVICE=es_smapi&_PROGR
AM=code.bkapi_throttle_day.sas&_DEBUG=0&_SERVICE=es_smapi";
} </SCRIPT>
```

JavaScript Example 2

The function RunDynamicT (JavaScript Example 2) also has some variables that SAS uses; like the SAS code to run (PROGRAM=) and the SAS service name (SERVICE=) and debug level (DEBUG=). Be sure to set these for your environment as needed.

HTML JAVASCRIPT IMPLEMENTATION

The JavaScript in Example 3 is used for producing the title and date selection seen here in Figure 5.

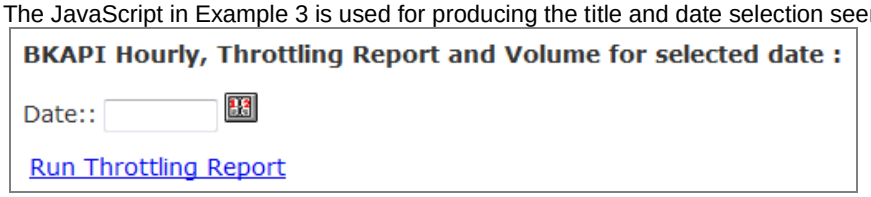


Figure 5

```
<p><b><font size="3">BKAPI Throttling Report, for selected date:</font></b><br><br>
<tr>
<td align="right" width="100"><font size="3">Date:</font> </td>
<td align="left"><input id="XDATE" name="XDATE" type="text" size="10">
<a href="javascript:show_calendar('GoToSection2.XDATE');"
   onmouseOver="window.status='Date Picker'; return true;"
   onMouseOut="window.status=''; return true;">
<img src="http://<server>/es_smapi/html/images/show-calendar.gif"
width="20" height="18" border="0"></a> <font size="3">
```

JavaScript Example 3

As mentioned the date picker is a third party component, from Softtricks.com (www.softricks.com/products/calendar/index.html) and needed the further modifications to match the date input requirements for the SAS script as seen in the JavaScript Example 1.

The HTML page makes use of HTML forms. The `xdate` variable in JavaScript Example 3 is wrapped in a form named `GoToSection2`. HTML forms contain input elements like checkboxes, text fields, radio-buttons and submit buttons in a container like:

```
<form>
.
input elements
.
</form>
```

See www.w3schools.com/html/html_forms.asp for examples and tutorials on HTML forms.

The code for the "Run Throttling Report" link seen in Figure 5 is below in JavaScript Example 4. The link refers back to the "RunDynamicT" JavaScript function declared at the top of the HTML page (seen in JavaScript Example 2).

```
<a href="javascript:RunDynamicT();"<font size="3">Run Throttling Report
</font></a></p>
```

JavaScript Example 4

The "`javascript:RunDynamicT();`" calls the RunDynamicT() function (JavaScript Example 2) resulting in the generated URL, with the selected and reformatted date, to the SAS web service.

Here is a sample Generated URL with the date 09MAY13 used for the XDATE:

```
http://<server>/sasweb/cgi-bin/broker
?XDATE=09MAY13&ProduceReport=Produce+Report&_SERVICE=es_smapi&_PROGRAM=code.bkapi_thro
ttle_day.sas&_DEBUG=0&_SERVICE=es_smapi
```

An important note is to pass the DEBUG=0 to the broker when using the SAS Tagsets. We have seen some unexpected results in the output when this was left out.

CONCLUSION

Adding a dynamic report generator web page for SAS is easy to use with current standard HTML and JavaScript. When added with the output options of SAS Tagsets we create a dynamic report system that adds value to the report user. Some adjustment may be needed for both the variables selected on the webpage and the end Tagsets results however in the end the user is empowered to select what data to fetch and manipulate the end results to meet their needs.

APPENDIX

EXAMPLE SAS

```
/* ------------------------------------------------------------------
 * BKAPI Dynamic Daily report
 * Program Name: ampdynamic.sas
 * Purpose:      Generate AMP summary pref and vol report for a selected
 *               date.
 * Author:       Mike Libassi
 * Date Created: 15AUG2012
 * Input data: xxxxxxxxxxxxx
 *             xxxxxxxxxxxxx
 *             xdate - Date Selected passed in from web request
 * Output Dir: SAS Web
 * Output files: none
 *
*------------------------------------------------------------------- */

libname amp server=xxxxxxxxxxxxx;
libname UsageWS server=xxxxxxxxxxxxx;

PROC SQL;
 create table work.gadgetThrottle as select
         APITHS.edate,
         APITHS.ehour,
         AG.gadget_name,
         AG.throttle_type,
         APITHS.apikey,
         APITHS.requestid,
         APITHS.response_mean,
         APITHS.apikey_throttled_mean,
         APITHS.apilevel_throttled_mean,
         APITHS.request_count,
         APITHS.requestfailed
from usagews.apithrottledhourlystat as APITHS
 left join amp.ampgadgets as AG on (APITHS.apikey = AG.api_key)
 where APITHS.edate = "&xdate"d;
quit;


filename temp url "http://<server>/cpm/prod/AMP/sas/prod/tableeditor.tpl";
ods path(prepend) work.templat(update);
%include temp;

ods tagsets.tableeditor file=_webout
options(web_tabs="BKAPI Throttling Data Table"
       window_title="BKAPI Throttling"
       highlight_color="yellow"
       scrollbar_color="beige"
       title_style="normal"
       image_path="http://<server>/cpm/prod/AMP/html/amp_report_top.png"
       header_size="12"
       image_just="left"
       left_margin=".5"
       right_margin=".5"
       top_margin="1"
       bottom_margin="1"
       autofilter="yes"
       sort="yes"
       sort_arrow_color="green"
       ) style=statistical;

/*  ----------------------Data Table - 1st tab ----------------------- */
TITLE;
TITLE2 HEIGHT=4 "BKAPI Throttling for &xdate (Click Column Header to Sort)";
```

```
TITLE3 HEIGHT=4 "<a href=//<server>/AMP/reports/throt/bkapithrot-&xdate..csv>CSV
version</a>";
FOOTNOTE HEIGHT=4 "Generated on %SYSFUNC(DATE(), EURDFDE9.) at %SYSFUNC(TIME(),
TIMEAMPM8.)";
PROC PRINT DATA=work.gadgetThrottle NOOBS;
        VAR ehour gadget_name throttle_type apikey requestid response_mean
apikey_throttled_mean api level_throttled_mean request_count requestfailed ;
RUN;

ods tagsets.tableeditor close;

/*  ---------------------- CSV file ------------------------------- */
x rm /cpm/prod/AMP/reports/throt/bkapithrot*.csv;

filename report2 "/cpm/prod/AMP/reports/throt/bkapithrot-&xdate..csv" new;
 ods csv close;
 ods csv body=report2;

PROC PRINT DATA=work.gadgetThrottle NOOBS;
        VAR ehour gadget_name throttle_type apikey requestid response_mean
apikey_throttled_mean api level_throttled_mean request_count requestfailed ;
RUN;
RUN; QUIT;

ods csv close;

/* ------------------------------------------------------------------
   end of code.
   ------------------------------------------------------------------ */
```

RECOMMENDED READING

- Creating a Data Grid Like VB.NET Chevell Parker, SAS Institute: http://support.sas.com/rnd/base/ods/odsmarkup/tableeditor/
- HTML Forms and Input. w3schools.com: http://www.w3schools.com/html/html_forms.asp
- SAS reference on ODS: http://support.sas.com/rnd/web/intrnet/dispatch/ods.html

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name:	Mike Libassi
Enterprise:	Elsevier
E-mail:	mike.libassi@elsevier.com
Twitter:	@mikelibassi
Linkedin:	www.linkedin.com/in/mikelibassi/