

Creating Interactive HTML Output With SAS

Chun Fan and Alex Ushveridze
Capella University, Minneapolis, MN

ABSTRACT

Presenting the output of SAS data processing results in an interactive web-based form would allow technically non-experienced people to easily interact with it and instantly get answers to their specific high-level questions. This paper discusses a method for creating such interactive output within SAS environment. The goal can be achieved by embedding JavaScript file inside ODS HTML. As an example we consider creation of sortable interactive tables with embedded sortable graphics.

INTRODUCTION

One of the biggest advantages of outputting SAS data processing results in the form of HTML files is the possibility of adding interactivity to that output.

Conceptually, the idea is very simple: by using SAS language it is easy to create any external text file with any textual contents. In particular, this could be an HTML file. In order to add interactivity to that file it is sufficient to use any scripting language like JavaScript – which again is nothing else but a text. Of course, JavaScript is known to be rather slow, however for not very intense calculations it is just great. A big advantage of using JavaScript is its platform neutrality and pure client-side nature – i.e. complete independence on any server-side and other supporting programs – the only thing one needs to run the JavaScript-based programs is a web-browser – and no matter which one.

To practically create an HTML /JavaScript-based interactive output within SAS it is sufficient to create an external text file by using the `_NULL_` data-step and then fill-up this file line-by-line with the corresponding HTML and/or JavaScript code by using PUT statement. The (oversimplified) sample code is shown below:

```
data _null_;  
  file "C:\myfile.html";  
  put "..some text 1.. ";  
  put "..some text 1.. ";  
  put "..some text 1.. ";  
  . . . . .  
run;
```

For referencing data and metadata obtained in previous SAS calculations one can effectively use SAS macro language.

This way of creating HTML /JavaScript-based interactive SAS output is rather universal and very flexible. It allows one to design any imaginable GUI. At the same time it requires some efforts in writing explicit HTML or JavaScript code which may produce some inconveniences for people not familiar with these programming languages.

The goal of this paper is to demonstrate that to some extent the interactive web-based output can be created by using the existing SAS ODS HTML commands. In fact, the way the ODS HTML works is similar to what we have described above. However, many details of HTML coding are hidden behind the ODS HTML commands.

As an example, we consider creation of sortable interactive tables with embedded sortable graphics. The users can click on the column headers to sort either the numeric or graphical results. The sortability of a graphical data is a nice feature by itself and it considerably simplifies the high level analysis of data tables.

EXAMPLE DATA SET

For demonstration, we consider a SAS dataset with three variables: Predictor, Importance and Shape. This is a typical dataset appearing as an the output of predictive modeling work and containing some basic properties of predictors.

Predictor	Importance	Shape
variable1	6.70	4321
variable2	4.50	1234
variable3	6.20	4312
variable4	5.30	3241
variable5	2.30	2341
variable6	5.00	4312
variable7	1.20	4231
variable8	4.70	1432
variable9	0.90	4231
...		

One of these properties is the Importance which shows how well does that predictor predict a target. The importance is traditionally well represented by a number – and nothing is wrong with that, but we want a little bit more: to make reading of the table easier we want to colorize the table cells containing these numeric values: the higher importance, the darker color.

Another property is Shape, which is a symbolic representation of the relationship between target and predictor. Roughly speaking, if Shape is 1234, we have an increasing graph, if Shape is 4321, we have a decreasing graph, and if Shape is 1342, we have a slightly skewed bell-shaped graph. Again, these numbers do their work rather well but we want a little bit more – we want to have instead some real tiny graphs – we hope that this will make the output results better understandable for people preferring visual display.

DESIRED INTERACTIVE OUTPUT

Here is the list of desired features to include in the interactive output:

1. to make it a web-based output;
2. to replace numeric values of Shape variable with tiny graphs, at the same time hide the text content;
3. to colorize cells representing the Importance variable
4. to make entire table sortable, for both numeric and graphical values.

HOW IT WORKS

CREATING WEB-BASED OUTPUT. This can be done easily and quickly by using ODS HTML in SAS. The general form of ODS HTML for SAS 9.1 is as follows:

EXAMPLE CODE

```
ods html3 file='c:\temp\output.html' ;
proc report data=originaldata nowd;
run;
ods html3 close;
```

CREATING TINY GRAPHS. This stage can be done in four steps.

Step 1. Creating SAS dataset (table) with 24 columns and 4 rows which represent different shapes in both textual (variable names) and numeric (variable values) form. The example of this dataset named 'shape' is

shown below:

1234	1423	2134	...
1	1	2	...
2	4	1	...
3	2	3	...
4	3	4	...

EXAMPLE CODE

```

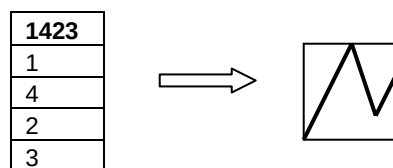
data transformeddata;
  set originaldata;
  a=int(shape/1000);           /*get the first number*/
  b=int(shape/100)-a*10;       /*get the second number*/
  c=int(shape/10)-a*100-b*10;  /*get the third number*/
  d=mod(shape,10);            /*get the last number*/
run;

proc sort data= transformeddata out=outdata nodupkey;   /*get rid
  by shape;                                           of duplicates */
run;

proc transpose data=outdata (keep=shape a b c d) out=shapes;
  id shape;
  var a b c d;
run;

```

Step 2. Use SAS MACRO to plot 24 combinations of “shape” values , and export the plots in .gif format. For example, for “shape” of “1423”, plot “1423” on vertical axis as shown below.



EXAMPLE CODE

```

%macro plot;
%local vars;
%let f = %sysfunc(open(transposed));
%if &f %then %do;
  %do i = 2 %to %sysfunc(attrn(&f,nvars));
    %let vars = %sysfunc(varname(&f,&i));
    /*export the plot in gif format in specified name and place*/
    filename grafout "c:\temp\&vars..gif";
    goptions reset=all device=gif hsize=0.20in vsize=0.20in
      gsfname=grafout gsfmode=replace noaxis;
    title;
    proc gplot data=transposed;
      plot &vars * _name_ / vaxis=axis1 haxis=axis2;

```

```

        axis1 label=none value=none major=none minor=none;
        axis2 label=none value=none major=none minor=none;
        symbol v=none i=join w=2;
run;

quit;

filename grafout clear;

%end;
%let f = %sysfunc(close(&f));
%end;
%mend;
%plot;

```

Step 3. Put shape values into .gif format using PROC FORMAT.

EXAMPLE CODE

```

proc format;
  value graph
    1234='_1234.gif'
    1243='_1243.gif'
    1324='_1324.gif'
    1342='_1342.gif'
    1423='_1423.gif'
    1432='_1432.gif'
    2134='_2134.gif'
    2143='_2143.gif'
    2314='_2314.gif'
    2341='_2341.gif'
    2413='_2413.gif'
    2431='_2431.gif'
    3124='_3124.gif'
    3142='_3142.gif'
    3214='_3214.gif'
    3241='_3241.gif'
    3412='_3412.gif'
    3421='_3421.gif'
    4123='_4123.gif'
    4132='_4132.gif'
    4213='_4213.gif'
    4231='_4231.gif'
    4312='_4312.gif'
    4321='_4321.gif'
    other = 'ne.gif';

```

Step 4. Set PREIMAGE= attribute in PROC REPORT like in this portion of code:

```
define shape /style(column)={ preimage=graph. }
```

Note: Full version of PROC REPORT code will be given in the last section.

HIDING THE TEXT. This can be done by modifying STYLE(COLUMN) = attribute in PROC REPORT as follows:

```
define shape /style(column)={ foreground=#D3D3D3}
```

Note: Full version of PROC REPORT code will be given in the last section.

COLORIZING CELLS REPRESENTING VARIABLES. This can also be achieved by formatting. For example, in this case, background color for the variable Importance will be changed based on their value ranges. First of all , we format background colors as follows.

EXAMPLE CODE

```
proc format;
  value bgi
    0-0.9='#FFFFFF'
    1-1.9='#FFE3E3'
    2-2.9='#FFC6C6'
    3-3.9='#FFAAAA'
    4-4.9='#FF8E8E'
    5-5.9='#FF7171'
    6-high='#FF5555';
run;
```

Then in PROC REPORT procedure, the style of those variables should be defined as the specified formats.

EXAMPLE CODE

```
proc report data= originaldata nowd ;
  define Importance / sum style(column)={background=bgi.} ;
run;
```

MAKING TABLE SORTABLE. This can be achieved by embedding an external JavaScript file sortable.js inside ODS HTML. To make the table sortable, there are three steps:

Step 1. Download sortable.js file from the JavaScript library (for more detail see ref. [1])

Step 2. Embed JavaScript code inside ODS HTML within the HEADTEXT= attribute, and include sortable.js by putting a link to it in the head of your page.

Step 3. Add the sorting behavior to the table tag by giving it a class of "sortable" within the TAGATTR= attribute.

EXAMPLE CODE

```
ods html3 file='c:\temp\output.html'
  headtext='<SCRIPT LANGUAGE=JAVASCRIPT TYPE="TEXT/JAVASCRIPT"
  SRC="c:\temp\sortable.js"></script>' ;
proc report data= originaldata nowd
  style(report)={tagattr=" class=""sortable"" "};
column Predictor Importance Shape;
define Predictor / order order=data 'Predictor';
define Importance / sum style(column)={background=bgi. foreground=black};
define Shape / style(column)={ preimage=graph. foreground=#D3D3D3} ;
run;
ods html3 close;
```

CONCLUSION

This paper offers a method for creating an interactive output within SAS environment . The goal is achieved by embedding JavaScript file inside ODS HTML. The file can be modified to add other events, and you can combine multiple behaviors within the table. It is simple to implement and really affordable. As an example we consider creation of sortable interactive tables with embedded sortable graphics. More interactive functions like filtering, printing, saving or exporting files may be added in the future.

REFERENCES

- [1] Stuart Langridge. 2007. SortTable version 2. <http://www.kryogenix.org/code/browser/sorttable/>
- [2] SAS Institute Inc. 2004. SAS® 9.1 Language Reference: Dictionary, Volumes 1, 2, and 3. Cary, NC: SAS Institute Inc.

CONTACT INFORMATION

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

Chun Fan
Capella University
225 S 6th Street 9th floor
Minneapolis, MN 55402
612 977 4273
Chun.Fan@capella.edu

Alex Ushveridze
Capella University
225 S 6th Street 9th floor
Minneapolis, MN 55402
612 977 5669
Alex.Ushveridze@capella.edu

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.
Other brand and product names are trademarks of their respective companies.