

Paper 067-31

## Teaching Your RTF Tagset to Do Clever Tricks

Wayne Hester, SAS Institute Inc., Cary, NC

### ABSTRACT

Using the new RTF Tagset with the ODS Markup destination in SAS 9.2 enables many controls and customizations that the traditional ODS RTF destination does not support. Explaining and describing the controls and processes can be confusing and pedantic. Therefore, this paper presents a simple comparison between the default RTF Tagset and an additional experimental sample RTF Tagset. The sample, which is provided by SAS, shows how to customize a tagset.

### INTRODUCTION

Tagsets are SAS code that is processed by the PROC TEMPLATE statement. Do not confuse tagsets with Style or Table templates. PROC TEMPLATE enables tagsets to be pre-compiled and stored for quick access. The tagset code primarily consists of data and events. Events are triggered by information from the SAS executables. What is done with the information depends on the tagset. By switching out the tagset, the same SAS code can produce EXCEL, HTML, or RTF files.

This paper explains how to manipulate a tagset to produce different types of output. The primary concept involves using a rough type of inheritance. Inheritance enables a new tagset to be built on the structure of a currently functioning tagset. This paper also explains how to customize an RTF document by making simple modifications to the RTF Tagset, which is new in SAS 9.2. Throughout this paper, the production tagset is called the "RTF Tagset," and the experimental sample tagset is called the "RTF Sample Tagset."

### THE RTF TAGSET

Before you learn how to customize a tagset, it is necessary to briefly explain how the RTF Tagset creates an RTF document. This process consists of creating puddles of cells that collect in ponds of rows. The rows collect in lakes of panels. The panels collect in pages of the document body. At the end of the process, all the parts are assembled into an RTF document. Changing the output from the RTF Tagset involves changing the events that create and assemble the puddles and ponds. Using these groupings helps you fit the tables and panels from your output to a fixed page size.

The RTF Tagset output is different from the output that you get from the traditional RTF engine. The primary differences in the output between the two RTF tagsets involve how page breaks, titles, and footnotes are handled. The RTF Sample Tagset relies on Microsoft Word to make implicit page breaks. An *implicit page break* occurs when a table is too long to fit on a page. Titles and footnotes are placed in RTF instructions that enable Word to apply them to pages as they are needed.

In contrast, the RTF Tagset enables SAS to place titles and footnotes on the page in tables in the document outside the scope of Word. For this to happen, SAS software must be responsible for the implicit page breaks.

### EVENT SUBSTITUTION – NOT INHERITANCE

Tagsets use the events from the parent when the events do not exist in the tagsets. This is not true inheritance as inheritance is understood for languages such as Java and Python; it is a replacement of an event or global value. Any change in an event requires the event to be fully expressed in the new tagset.

The following discussion reviews the 10 tagset events that are modified to change the nature of the production RTF Tagset into the RTF Sample Tagset. Only a few of these events need special attention.

First, name your tagset so that it can be stored and recovered. If you use a name that is already in use, that tagset is replaced by the new tagset. The tagset name enables you to access the tagset in the SAS source code. In this instance, you use:

```
ODS tagsets.rtf_sample OUT="filename.rtf";

proc template;
  define tagset tagsets.rtf_sample;
```

Next, you specify the parent of your current tagset. If an event is requested and it cannot be found in the RTF Sample Tagset, you can use the tagset that is in the parent. In this instance, that is the RTF Tagset.

```
/* Inherit from the rtf.tpl */
parent=tagsets.rtf;
```

The UNIFORM and DEFAULT\_STYLE options are specified in the measured source library and need to be set in the new tagset. These options can be set in the tagset to avoid having to set them at every invocation of the RTF Sample Tagset. Because Word must be able to break a table at any point for implicit paging of long tables, all the rows in the table have the same column widths. The UNIFORM option ensures identical column widths, and is used by printer destinations such as PDF to achieve uniform tables across pages.

Because this tagset is not known to SAS internally, a default style is not applied. We can apply a default style in the tagset. All other global settings are inherited from the tagset `rtf.tpl`.

```
uniform=yes;
default_style="styles.rtf";
```

### THE PAGEBREAK EVENT

The PageBreak Event is the most complex event in this tagset. The PageBreak Event gives the tagset some of its key features. In this event, you will change how page breaks are performed. The PageBreak Event is called when an *explicit page* break is requested, for example, at the beginning of a procedure. The underlying source code surfaces the same information to both versions and the tagset determines what to do with it.

Start the PageBreak Event by using:

```
define event pagebreak;
```

Some events might have START and FINISH sections that allow the underlying executable code to perform work between the parts. Other events might need to be called between the start and the finish.

```
start:
```

In the RTF Sample Tagset and the RTF Tagset, the names of all streams end with the suffix `_tbl`. A *stream* is an indefinitely-sized repository. When the repository gets larger, it can be stored to disk. All tables are stored in the stream `body_tbl` until the document is assembled and closed.

```
/* TEXT is loaded by using the startpage value "ON", "OFF", or "NOW" */
open body_tbl;
```

The internal macro `$STARTPAGE` is set as an option in the RTF Tagset. The dollar sign (\$) marks the option as a macro. The macro `$STARTPAGE` is set by using the ODS `STARTPAGE` option. The macro `$FIRST_PAGE` is also set in the RTF Tagset; this macro indicates the first page in the document.

```
put "\pard\par" /if cmp("OFF", $startpage);

do /if $first_page;
  put "\sectd\linex0\endnhere";
else;
  put "\pard\sect\sectd\linex0\endnhere";
done;
```

The uppercase words are attributes that are passed in from the executable that is calling the event. These attributes are not case sensitive, but it is easier to read the tagset by using a consistent case. Any tagset that contains the PageBreak Event gets these attributes. Based on the attribute, the tagset writes out RTF code. The RTF specifications are the text that's written out by using PUT statements.

```

/* Value passes in orientation. */
do / if VALUE;
  put "\pgwsxn" WIDTH "\pghsxn" HEIGHT "\lndscpsxn" / if
    cmp("LANDSCAPE", VALUE);
  put "\pgwsxn" WIDTH "\pghsxn" HEIGHT / if cmp("PORTRAIT", VALUE);
done;

/* STARTPAGE value */
put "\sbknone" /if cmp("OFF", $startpage);
put CR;
unset $first_page ;

put CR;
/* headery is topmargin plus the value computed for
  page number and date instructions. The bordertopwidth
  is borrowed for passing this value */
eval $headery TOPMARGIN + BORDERTOPWIDTH;

put "\headery" $headery "\footery" BOTTOMMARGIN;
put "\marglsxn" LEFTMARGIN "\margrsxn" RIGHTMARGIN;
put "\margtsxn" TOPMARGIN "\margbsxn" BOTTOMMARGIN CR;

do /if exists(PAGE_COLUMNS);
  put "\sbkcol\cols" PAGE_COLUMNS;
  set $pagecols PAGE_COLUMNS;
done;

/* header values repeat on implied page breaks */
put "{\header\pard\plain\qr\pvmrg\phmrg\posxr\posy0{" CR;

```

Most of what is being written is done with text strings that contain RTF specifications and values and macros. There are also odd-looking names such as \$\$pageno\_tbl. The double dollar signs (\$\$) are needed when a stream is written to another stream by using a PUT statement. The stream \$\$pageno\_tbl collects the page number and date information. Each stream is built by as many events as needed and exists until they are cleared with an UNSET statement.

```

Put $$pageno_tbl;

close;

```

The preceding CLOSE statement closes the start of the PageBreak Event. Later, the event is called with a request to issue the finish. Streams that have collected the titles (\$\$titles\_tbl) and footnotes (\$\$footnotes\_tbl) are used.

```

finish:
  open body_tbl;

```

The title and footnote information is joined by using the \header (see above example) and \footer (see below) RTF specifications. Word uses these specifications to place titles and footnotes in the document where it determines the correct locations to be.

```

  put $$titles_tbl;
  put "}}" CR;
  put "{\footer{" CR;
  put $$footnotes_tbl;
  put "}}" CR;
  close;
  set $startpage $previous_startpage /if cmp("NOW", $startpage);
end;

```

Here is a quick review of the syntax of the PageBreak Event. Attributes such as TOPMARGIN and BORDERTOPWIDTH are passed to the tagset from the executable code. The stream \$\$titles\_tbl is built to hold the current titles. The internal macro \$FIRSTPAGE is created in the tagset. The remaining code contains simple tagset decisions about what to do with the information.

#### THE STARTPAGE EVENT

The StartPage Event is much simpler than the PageBreak Event . The StartPage Event consists of setting flags for use with the STARTPAGE option. This event removes explicit page breaks and is specific to the STARTPAGE option. STARTPAGE is a very useful and involved option, but its specifics are beyond the scope of this paper.

```
define event startpage;
  start:
    unset $startpage / if cmp("ON", TEXT);

    do / if cmp("NOW", TEXT);
      set $previous_startpage $startpage;
      set $startpage "NOW";
    done;

    do / if cmp("OFF", TEXT);
      set $startpage "OFF";
      open body_tbl;
```

In addition to setting the flags for the paging events, the StartPage Event writes RTF specifications to the body stream to ensure a physical space with the next table when explicit page breaks are turned off.

```
      put "{\pard\par}" CR;
      close;
      break;
    done;

end;
```

#### THE TITLES AND FOOTNOTES EVENTS

The Titles and Footnotes Events publish the internal streams that hold the title and footnote information to the document body.

In the RTF Sample Tagset, Word repeats the titles and footnotes on implicit page breaks when the table is longer than a page. For this instance, these events are not needed, but you need to make sure that the parent's copy is not called either. The empty PUT statement is needed for these events to compile with PROC TEMPLATE.

```
define event footnotes;
  start:
    put;
end;

define event titles;
  start:
    put;
end;
```

#### THE PANEL\_START AND PANEL\_END EVENTS

A broad use of panels helps control inappropriate breaks in tables. Panels are built before they are placed on the document page. Each panel is bracketed by Panel\_Start and Panel\_End events. The underlying source library calls the Panel\_Start Event. The Panel\_Start Event does not exist in the parent RTF Tagset, but it does exist in the RTF Sample Tagset, where it is used to clear streams and flags.

```

define event panel_start;
  unset $table_head_done;
  unset $$table_head;
  unset $$table_body;
  unset $$table_foot;
end;

```

The Panel\_End Event is short and simple. The body is populated in increments of panels. When more than one table occurs on a page, a space is placed between them.

The only new part to this event is the insertion of a spacing and context break. The insertion only occurs for "normal" panels. The macro \$CURRENT\_TBL contains the name of a stream that re-directs the output. These alternate streams are quite useful but do not need the spacing.

```

define event panel_end;
  start:
  trigger table_end start;
  open body_tbl;

  /* There are many dummy streams created at start-up. */
  /* Do not force a context break for them. */
  put "{\pard\par}" CR /if ^ $current_tbl;
  close;
end;

```

#### THE TABLE\_END EVENT

The Table\_End Event is very similar to the parent; however, here are some tricks that might be useful in other contexts.

```

define event table_end;
  start:

```

Bylines are captured but are published only when the panel fits the page. There is little "bait-and-switch" to ensure this.

```

/* Catch the byline. It is saved off but only occurs in some streams. */
break / if cmp($current_tbl, "byline_tbl");
open body_tbl;

```

Because Word repeats the header of the table for you, you only have to generate it once. The internal flag \$table\_head\_done determines when to insert the header section of the table. It is re-set in the Panel\_Start Event.

```

  put $$byline_tbl /if ^ $current_tbl;
  put $$table_head /if ^ $table_head_done;

  /* Only one header per table because Word handles the page breaks. */
  set $table_head_done "DONE";
  put $$table_body;
  put $$table_foot;
  unset $$byline_tbl;
  unset $$table_head;
  unset $$table_body;
  unset $$table_foot;
  close;
end;

```

**THE IMPLICIT\_PAGEBREAK EVENT**

The Implicit\_Pagebreak Event is very simple for the RTF Sample Tagset. The same event in the RTF Tagset needs to perform much more work. The interesting action here is the optional invocation of the START section in the Table\_End Event. Using a "trigger" keyword is the way to call an event that is internal to a tagset.

```

/* There are no implicit page breaks performed by the tagset. */
/* The tagset needs to insert the table_body that has been created. */
define event implicit_pagebreak;
  start:
    do / if ! cmp("table_head", $section_tbl) and ^ VALUE;
      trigger table_end start;
    else;
      unset $$table_head;
      unset $$table_body;
      unset $$table_foot;
    done;

end;

```

**THE PARSKIP AND PARAGRAPH EVENTS**

The Parskip and Paragraph Events are very similar to the events in the parent RTF Tagset. The Parskip and Paragraph Events are used when there are requests for space in the document. The Parskip Event is used by the measured library and the Paragraph Event is used by it's source library parent, the XML library.

The slight change in syntax means the entire event needs to be moved to this tagset location. Because Word handles the page fits, you do not use a measured table to space the tables. Instead, the rtf {\pard\par} is used. The output is written to the row stream so that it can be flushed away if necessary.

```

/* The \pard\par syntax does more than just force a space. It is */
/* often needed to prevent contexts from colliding and getting */
/* confused. */
define event parskip;
  start:
    do /if $current_tbl;
      open $current_tbl;
    else;
      open row_tbl;
    done;

    put "{\pard\par}" CR;
    close;

end;

define event paragraph;
  start:
    do /if $current_tbl;
      open $current_tbl;
    else;
      open row_tbl;
    done;
    put "{\pard\par}" CR;
    close;

end;

```

This end statement matches the "define tagset tagsets.rtf\_sample;" statement.

```
end;  
  
run;
```

## SUMMARY

You should never have to write a complete tagset. By re-writing several events and, possibly, adding options and functions, significant changes in output can be accomplished. With this brief overview of actions and vocabulary, the online HTML version of the RTF Sample Tagset will show how quickly an existing tagset can be re-shaped.

Although most of the source code for the RTF Sample Tagset is duplicated in this paper, you can download the complete tagset, the SAS source code, and two RTF output files from the SAS Customer Support Web site at <http://support.sas.com/saspresents>. See Hester, Wayne. "Teaching Your RTF Tagset to Do Clever Tricks." SAS Presentations at SUGI 31. March 2006.

## BIBLIOGRAPHY

Gebhart, Eric. A. 2005. "ODS Markup: The SAS Reports You've Always Dreamed Of." *Proceedings of the Thirtieth Annual SAS Users Group International Conference*, Philadelphia, PA. available at <http://www2.sas.com/proceedings/sugi30/085-30.pdf>.

Gebhart, Eric. A. 2005. "ODS Tagsets and EXCEL, an Excellent Combination." available at [http://support.sas.com/rnd/base/topics/odsmarkup/Tagsets\\_&\\_Excel.ppt](http://support.sas.com/rnd/base/topics/odsmarkup/Tagsets_&_Excel.ppt).

Gebhart, Eric. A. 2005. "Tagset Spelunking and Cartography: Debugging and Exploring Tagsets with Battery-Powered Headlamps." *Proceedings of the Thirtieth Annual SAS Users Group International Conference*, Philadelphia, PA. available at <http://www2.sas.com/proceedings/sugi30/014-30.pdf>.

Gebhart, Eric. A. 2002. "ODS Markup: The Power of Choice and Change" *Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference*, Orlando, FL. Available <http://www2.sas.com/proceedings/sugi27/p003-27.pdf>.

## CONTACT INFORMATION

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

FW Hester  
SAS Institute Inc.  
SAS Campus Drive  
Cary, NC 27513  
Phone: (919) 677-8000  
[Wayne.Hester@sas.com](mailto:Wayne.Hester@sas.com)

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.