Paper TS01

To ODS RTF and Beyond, Third Edition

David Shannon, Amadeus Software, UK

ABSTRACT

The ability to send output to Microsoft Word was the most commonly requested function during the development of the Output Delivery System. Since version 8.1 ODS RTF has been formally available. The Rich Text Format (RTF) specification is a method of encoding formatted text and graphics which is read by Microsoft Word and several other text editors. So now that we have the ability to send output to Word, how can we apply it? What are the advantages over traditional fixed space output? How can we go beyond the documentation to extract some of Microsoft Word's features?

This paper will address these questions demonstrating how to define and apply generic appearance to a project and consider the advantages of RTF output to both the programmer and end user. Furthermore it will be shown how to go beyond the documentation by embedding RTF codes to enhance your reports. Finally this paper addresses some practical considerations when generating and distributing RTF files.

INTRODUCTION

For several years as both a statistician and SAS programmer one of the most consistent problems I have experienced is the distribution of reports and graphics generated from the SAS system. Whilst sending a text file to a recipient is simple enough, for that person to view the text in the intended format requires them to know the page settings, font and font sizes then configure their software before viewing and printing.

Although there are several methods of importing traditional listing output into Microsoft Word via either DDE from within the SAS system or using Microsoft Word itself, this is a cumbersome task requiring additional processing and time.

Sending reports directly from your reporting procedure or reporting data step to a Word document eliminates these problems and keeps the reporting process encapsulated in a single software environment for manageability.

This third revision of the paper discusses features available in SAS 9.1.3.

THE MECHANICS OF RTF

WHAT IS "RTF"?

A Rich Text Format (RTF) file is a text file using defined control words and symbols that preserve the formatting of text. Since SAS 8.2 the RTF 1.6 specification used creates Word 2000 documents.

TO ODS RTF

Just two additional lines of code are needed to generate a simple RTF file. Consider the following:

```
ods rtf;

title "Listing of CLASS Data";
proc print data=sashelp.class;
run;

ods rtf close;
```

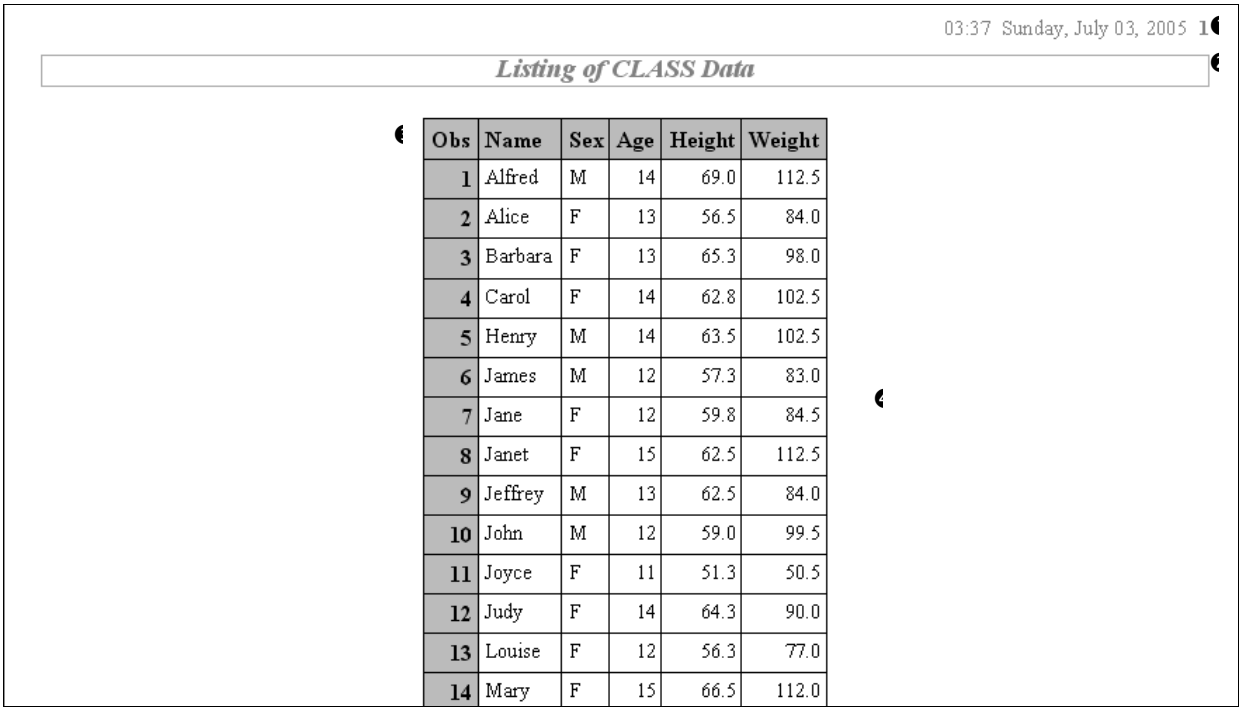
The first ODS statement opens the RTF destination; SAS 9 generates an automatic file name when none is specified, here the file SASRTF.RTF is created. Beware, it is only when the ODS RTF CLOSE statements are passed that the RTF file is saved to disk.

It's that simple. So end of discussion then?

Perhaps not, if we look at the output created there could be many elements of the default appearance we wish to

customise, perhaps enhance specific areas of the report dynamically, or to conform with our companies guidelines, or just to adjust the page settings of the document.

At this point we consider how the output generated is stored inside Word. Taking the partial output from the example above (Figure 1), we can see that a Word table is divided into rows and columns to store the printed output, where each cell contains one element of the report.



03:37 Sunday, July 03, 2005 1

Listing of CLASS Data

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83.0
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84.0
10	John	M	12	59.0	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90.0
13	Louise	F	12	56.3	77.0
14	Mary	F	15	66.5	112.0

Figure 1: Default ODS RTF Output

Default ODS RTF output includes the date and page number (❶) inserted as fields into a text box in the page header. By default, titles and footnotes (❷) appear in header and footer sections of the document, respectively. Although not visible, a bookmark is always included in the top left most cell of the output (❸) for each procedures output. Bookmarks take a default name of IDX and then IDX1 where the numeric suffix increments for each new output generated in the document.

The body of output generated is stored in a Word table (❹), where each datum is contained in its own cell in the table.

The default RTF style shades the cells of the header row grey, and makes the header row text bold. All the cell borders are drawn. When more than one procedure's output is generated in the same file a section break is added to the document. This also allows the titles to change between outputs.

WHAT ARE THE ADVANTAGES OF USING RTF?

Compared to the traditional listing output the advantages seem quite obvious. However the use of an alternative format for your output may require changes to your standard working practices and therefore you should be able to justify its use.

By using ODS and standard SAS system reporting procedures the programmer no longer needs to write additional code to post process output. In particular distributing graphics in a format which is easily viewed and printed has previously been cumbersome. SAS/GRAPH output can also be sent directly to a Word document. For example:

```
ods rtf file="graph1.rtf";
goptions reset=all;
proc gplot data=sashelp.class;
  plot weight*height=sex;
run;quit;
ods rtf close;
```

Advantages to the reader are most obviously the appearance. Whilst also pleasing to the eye, the ability to change fonts, font sizes, even colouring and shading can be used to enhance data within the report, improve readability and provide an overall professional presentation. As Microsoft Word and other software packages that read Word files are widely used, the end user has a tool to view both text and graphical reports produced from within SAS software.

DEFINING A GENERIC APPEARANCE FOR A PROJECT

Whilst we can use the style options within reporting procedures REPORT, TABULATE and PRINT to define exactly how our Word document will look, it is more efficient and easier to apply generic changes if we define and store a style using PROC TEMPLATE.

A generic style controls all the output generated, but individual outputs can override this generic style by using STYLE statement options at procedure level.

Supposing it is desired to define a look which is very plain when printed on paper. PROC TEMPLATE is used to achieve this by inheriting a supplied style which closely resembles the requirement, and then edit its attributes to suit our needs.

The sample style below named “plain” replaces most fonts to be Verdana; available by default on Windows machines of recent operating system versions. The extensive colours list is required to avoid generating numerous warnings in the log when the style is applied. The edits to the BODY and TABLE elements of the styles also provide alterations, removing line drawing and maximising use of the page by removing space between cells.

```
proc template ;
  define style plain ; */ store=project.styles;
  parent = styles.printer;
  replace fonts /
    'TitleFont' = ("Verdana",10pt,Bold)
    'TitleFont2' = ("Verdana", 10pt)
    'StrongFont' = ("Verdana",10pt,Bold)
    'EmphasisFont' = ("Verdana",10pt)
    'BatchFixedFont' = ("Courier New",9pt)
    'FixedEmphasisFont' = ("Courier New",9pt)
    'FixedStrongFont' = ("Courier New",9pt)
    'FixedHeadingFont' = ("Courier New",9pt)
    'FixedFont' =("Courier New",9pt)
    'headingEmphasisFont' = ("Verdana",10pt,Bold)
    'headingFont' = ("Verdana",10pt,Bold)
    'docFont' =("Verdana",10pt);
  replace color_list /
    "bg" = white
    "fg" = black
    "fgH" = blue
    "bgH" = white
    "link" = blue ;
  replace colors
    "Minimal colours used" /
    'headerfgemph' = color_list('fg')
    'headerbgemph' = color_list('bgH')
    'headerfgstrong' = color_list('fgH')
    'headerbgstrong' = color_list('bgH')
    'headerfg' = color_list('fgH')
    'headerbg' = color_list('bgH')
    'datafgemph' = color_list('fg')
    'databgemph' = color_list('bg')
    'datafgstrong' = color_list('fg')
    'databgstrong' = color_list('bg')
    'datafg' = color_list('fg')
    'databg' = color_list('bg')
    'batchbg' = color_list('bg')
    'batchfg' = color_list('fg')
    'tableborder' = color_list('fg')
    'tablebg' = color_list('bg')
```

```
      'notefg' = color_list('fg')
      'notebg' = color_list('bg')
      'bylinefg' = color_list('fg')
      'bylinebg' = color_list('bg')
      'captionfg' = color_list('fg')
      'captionbg' = color_list('bg')
      'proctitlefg' = color_list('fg')
      'proctitlebg' = color_list('bg')
      'titlefg' = color_list('fg')
      'titlebg' = color_list('bg')
      'systitlefg' = color_list('fg')
      'systitlebg' = color_list('bg')
      'Conentryfg' = color_list('fg')
      'Confolderfg' = color_list('fg')
      'Contitlefg' = color_list('fg')
      'link2' = color_list('link')
      'link1' = color_list('link')
      'contentfg' = color_list('fg')
      'contentbg' = color_list('bg')
      'docfg' = color_list('fg')
      'docbg' = color_list('bg');
    style body from document /
      leftmargin=2.5cm
      rightmargin=2.5cm
      topmargin=2.5cm
      bottommargin=2.5cm;
    style Table from output /
      background = _undef_
      frame = hsides
      rules = groups
      cellpadding = 2pt
      cellspacing = 0pt
      borderwidth = 1pt;
    end;
  run;
```

Furthermore settings to define paper size can be controlled trough the global options statement, but your PC must know about the paper size specified. This is usually determined by the print drivers installed:

```
options papersize=A4;
```

APPLYING A GENERIC APPEARANCE TO A PROJECT

Style definitions are stored in an “item store” which is located in a SAS system library. This can then be made available to our SAS session with the ODS PATH statement.

```
ods path project.styles(read) sashelp.tmplmst(read);
```

To make the style available to multiple users simultaneously locate the item store in a SAS system library available to all users on the project and give read-only access to the styles. Specifying the default SAS item store after our customised item store tells the SAS system to use the default style attributes if there are any missing definitions in our customised style.

Such an ODS PATH statement is ideally located in a project autoexec file. Note, however, if we need to alter our customised style we must change the access to the item store to update mode:

```
ods path project.styles(update);
```

OPERATING SYSTEMS

ODS RTF is available on all operating systems, not just Windows. Note that the RTF is formatted for the environment it is created in. If you wish to create RTF in one environment, such as UNIX and view the RTF in another environment, such as Windows, the record separator may need to be altered. For example, if creating RTF

files on a mainframe for viewing under Windows then the appropriate record separator syntax for ASCII is:

```
ODS RTF RECORD_SEPARATOR='0D0A'x
```

Record separator takes a hexadecimal number.

ODS RTF FEATURES

FILE PROPERTIES

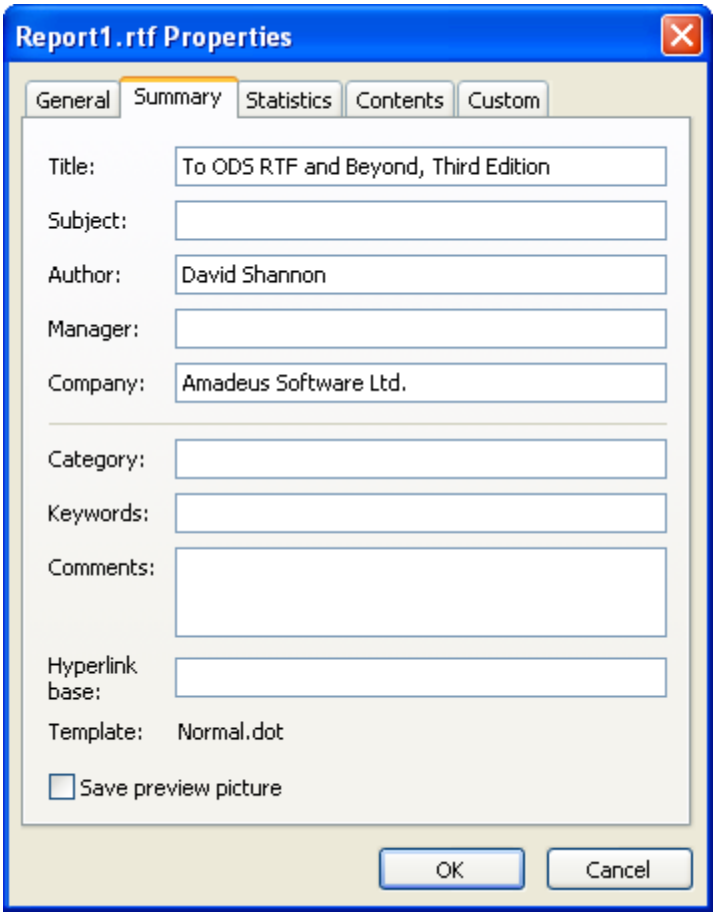
There are several options of the ODS RTF statement which we may use to enhance both the attributes of the document and the contents.

By default when using SAS 9.1.3 SP3 the RTF document properties of author and title are set as “SAS Version 9.1” and “V9.1 SAS System Output”, respectively. These options on the ODS statement shown below allow these values to be set programmatically.

We can set the document author and title properties on the ODS RTF statement as follows:

```
ods rtf author="David Shannon"
        title="To ODS RTF and Beyond, Third Edition"
        file="Report1.rtf";
```

Figure 2: Document Properties Controlled by ODS RTF



Once the output is generated the RTF file can be opened in Microsoft Word (Word 2003 shown). By selecting File -> Properties from within Word it can be seen that these properties have been set. See Figure 2 to the left.

BOOKMARKS

We can make use of is Word’s bookmark facility. When the SAS system creates each new table of output the upper left cell contains a bookmark named IDX. Graphic objects generated by the SAS system are also book marked. We can change this for each new table created with the following syntax:

```
ods rtf bookmark="report_1";
```

PAGE BREAKS

The startpage option can be used to control where and when Word places page breaks:

```
ods startpage= now | never | yes | no;
```

Value	Action
Now	Forces a page break now
No | Never | Off	Prevents all page breaks
Yes | On	Puts a page break between all outputs

The STARTPAGE option can be used either on its own ODS statement or with other ODS RTF options. For example:

```
ods rtf file="report.rtf" startpage=no;
  <procedure>
  <procedure>
ods rtf startpage=now;
  <procedure>
ods rtf close;
```

We can make use of these options to combine a summary table and graphic on the same physical page:

```
ods rtf file="report1.rtf" startpage=no style=plain;

proc means data=sashelp.class nonobs maxdec=1 n mean std min max;
  var height weight;
run;

goptions reset=all device=actximg ;
proc gplot data=sashelp.class ;
  plot weight * height;
run;
quit;

ods rtf close;
```

There are options we can use to control where titles and footnotes appear. Normally titles and footnotes appear as part of the graphic. However by setting the options NOGTITLE and NOGFOOTNOTE on the ODS RTF statement our titles and footnotes, respectively, will now appear in the header and footer areas of the document.

From the example code above, the document appears as follows to combine a table and graphic on the same physical page as demonstrated Figure 3 below:

The MEANS Procedure

Variable	N	Mean	Std Dev	Minimum	Maximum
Height	19	62.3	5.1	51.3	72.0
Weight	19	100.0	22.8	50.5	150.0

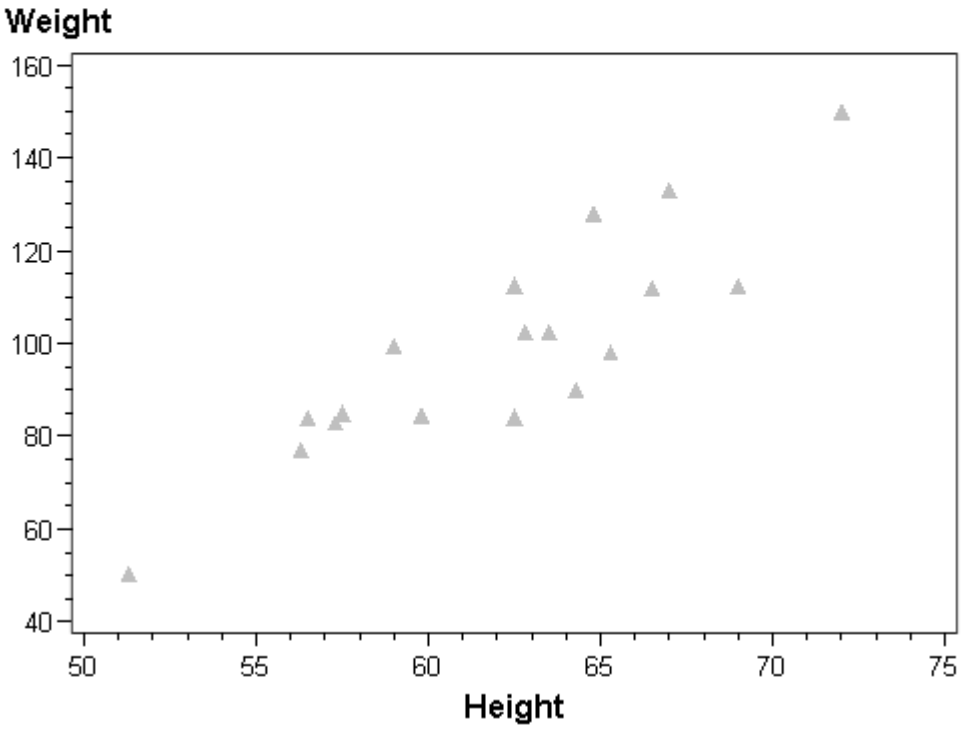


Figure 3: Avoiding Page Breaks between Outputs

SIMULTANEOUSLY CREATING MULTIPLE RTF FILES

There are situations when we may wish to send output to multiple RTF files simultaneously. This is achieves by assigning an ID to the open destination.

```
ods rtf(1) file="report1.rtf";
ods rtf(2) file="report2.rtf";
```

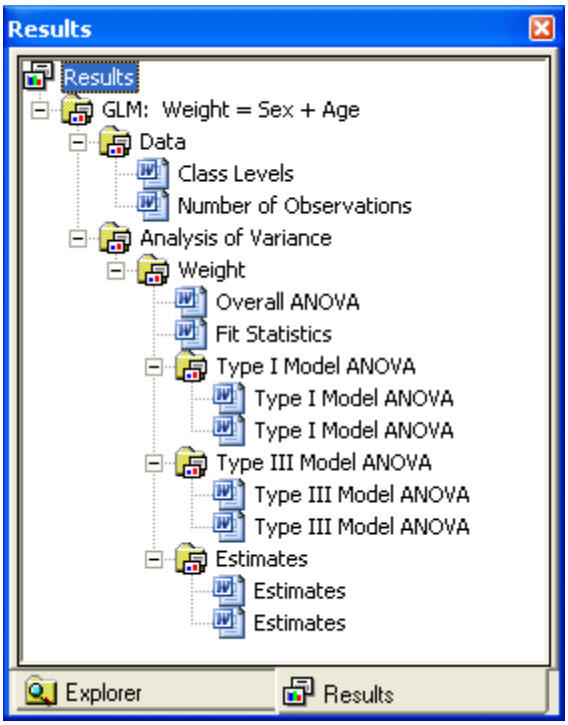
When creating multiple files we can also select or exclude output objects to either or both of these files or change the appearance by using the style option on the ODS statement.

For example, consider the output generated by PROC GLM, we may only be interested in specific parts of the results for conveying information, however we may want to retain all the output for documentation. This can be achieved as follows:

```
ods listing close;
ods noproctitle;

title1 h=11pt "Weight = Sex + Age";

ods rtf(1) file="full glm output.rtf" style=minimal;

ods rtf(2) file="partial glm output.rtf" style=plain;
```

```
ods rtf(2) select modelanova estimates;

proc glm data=sashelp.class;
  class sex;
  model weight = sex age;
  estimate 'Male vs. Female' sex 1 -1;
run;
quit;

ods _all_ close;
```

Figure 4: Simultaneously Creating Multiple Documents with ODS RTF



When the results window is expanded (Figure 4) we can see the objects created in two Word documents have two icons:

ESCAPE SEQUENCES

So far all examples of customising text with styles etc. are applied to a complete cell or table. Additionally it is possible to format a section of text within a cell by embedding the appropriate RTF code. Some of these codes are now available via escape sequences which remove the ambiguity of embedding raw RTF codes.

```
data a;
  length text $300;
  text="We can force a soft return here: \-2n Then continue "||
       "on with our text as normal.  Or we can add further "||
       "formatting such as \{super superscript text} or \{sub "||
       "subscript text} like that.  \S={font_style=italic}There"||
       "\S={} are of course\S={font_weight=bold} several\S={} "||
       "other text formats."  ;
run;

ods escapechar='\';

ods listing close;
ods rtf file="escape sequence.rtf" style=plain;
proc report data=a noheader nofs style={asis=yes frame=box rules=none};
  column text;
  define text / style={cellwidth=10cm};
```

```
run;
ods rtf close;
```

This produces the output as shown in Figure 5 below.

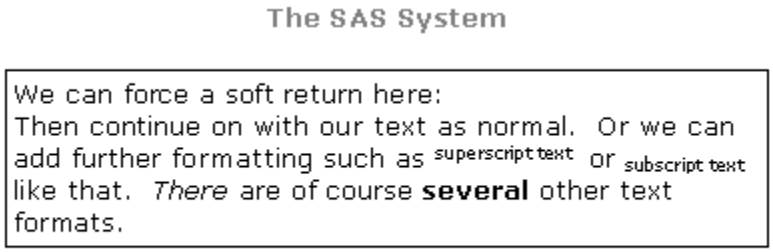


Figure 5: Escape Sequences

PAGE NUMBERING

Since ODS RTF was introduced, the ability to add Word's Page X of Y formatted page numbering has been well documented. This was achieved by embedding the appropriate RTF control words into SAS code that rendered the PAGE and NUMPAGES Word fields in our output documents.

A disadvantage is that the code looks cumbersome and complication; but whilst it can easily be wrapped up into a single macro call, SAS 9 has made life simpler still.

The escape sequence {PAGEOF} allows programmers to readily make Microsoft Word perform page numbering in the output document; additionally the escape sequences {THISPAGE} and {LASTPAGE} allow custom page descriptions to be created. :

The following simple program demonstrates this:

```
ods rtf style=plain;
ods escapechar="^";

* Simplest mechanism;
proc print data=sashelp.class;
  title "Page ^{pageof}" ;
run;

* Cusomise your page numbering text;
proc print data=sashelp.class ;
  title "This is page ^{thispage} from a total of ^{lastpage} pages" ;
run;

ods rtf close;
```

The outputs shown below are generated from the above proc prints, respectively:

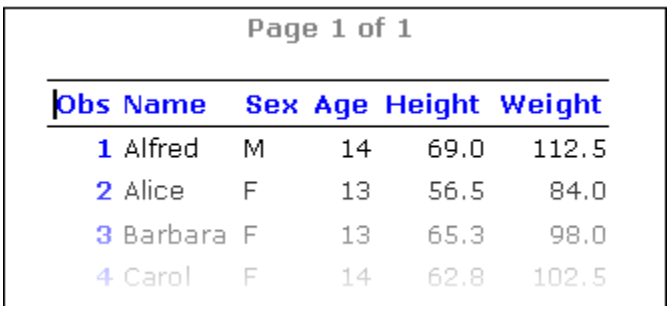


Figure 6: Using the {PAGEOF} Escape Sequence in SAS 9

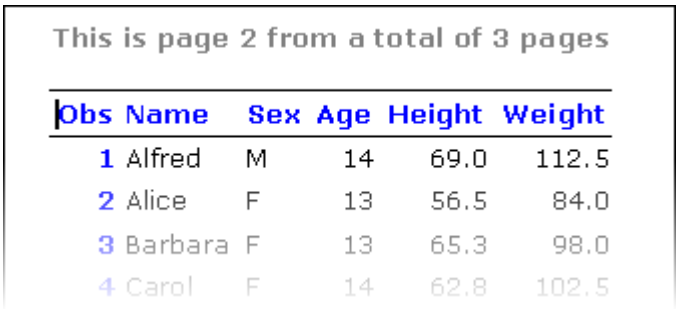


Figure 7: Using the {THISPAGE} and {LASTPAGE} Escape Sequences in SAS 9

For users of SAS V8.2, the original method still works equally well by inserting the following footnote statement into your program:

```
footnote1 J=R '{Page \field {\*\fldinst PAGE \\*MERGEFORMAT}} {of \field
{\*\fldinst NUMPAGES  \\*MERGEFORMAT} }';
```

The start page number can be controlled by using the pgnstartN RTF field, where the suffixed N is the desired start number.

The following sets the start page number of 2:

```
footnote1 J=R '{Page \field {\*\fldinst {\pgnstart2 PAGE} \\*MERGEFORMAT}}';
```

DECIMAL ALIGNMENT

A commonly requested feature is to align decimal values in reports. SAS 9 allows this to be achieved with a new column justification. In addition to left, centred and right there is now also decimal. Consider the following proc report code, noting the justification on the definition statement for RESULT1:

The code shown below (with the input data set available!) produced the following output:

```
ods rtf style=plain;
title "Decimal Alignment in SAS 9";
proc report data=weight nofs noheader;
  column  sex _name_ result1;
  define sex       / group f=$gender.;
  define result1   / display
                     style(column)={just=d};
  compute before ;
    line ' ';
  endcomp;
  compute after sex ;
    line ' ';
  endcomp;
  compute result1;
    if _name_='n' then
      call define(_col_,"format","8.0");
    else if _name_ in('Mean','Median') then
      call define(_col_,"format","8.1");
    else if _name_='SD' then
      call define(_col_,"format","8.2");
  endcomp;
run;
ods rtf close;
```

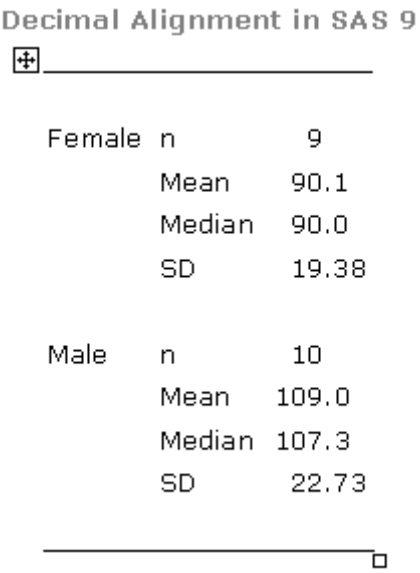


Figure 8: Decimal Alignment in SAS 9

Users of SAS 8.2 can achieve the same effect by specifying the RTF code “tqdec” and the distance to align the decimal from the left border in a unit called twips (there are 1440 twips in an inch or, one twip is 1/20th of a point, i.e. the standard unit of measurement for fonts in Microsoft Windows).

For examples within a PROC REPORT define statement, decimal alignment can be specified with 400 twips from the left cell margin by adding the following style statement:

```
style(column)={protectspecialchars=off pretext="\tqdec\tx400 "};
```

AND BEYOND

WITHIN TABLE PAGE NUMBERING

During the four years since the original version of this paper several requests have been received to display within table page numbering in the document, rather than page numbering the entire document. This was not possible in SAS 8.2, because an RTF control code needs to be embedded at the start of a new section to force the automatic page numbering to start again.

SAS 9 has a new ODS RTF statement option that enables users to enter their own RTF controls via the SECTIONDATA keyword. The following example forces the page numbering to restart for whatever output follows. Note that it is assumed that the ODS RTF document is opened previously in the program.

```
* Create in table page numbering within the document;
ods rtf sectiondata="\pgnrestart\pgnstarts1";
proc print data=sashelp.class ;
  title '{Page \field{\*\fldinst { PAGE   \\* MERGEFORMAT }}} of {\field
{\*\fldinst SECTIONPAGES \* MERGEFORMAT } in this section}';
run;
```

The output shown in Figure 9 shows the third page in a Microsoft Word document. The previous two pages have each been generated by individual procedures and hence a section break is added between each of those outputs. The third section, shown below, has page numbering beginning at Page 1 thanks to the "\pgnrestart\pgnstarts1" code. After that the statements used put the current page number and total number of pages in the current section into the title. Notice also that the final RTF control character in the SECTIONDATA keyword is the number 1. By altering this value page numbering can begin from any number.

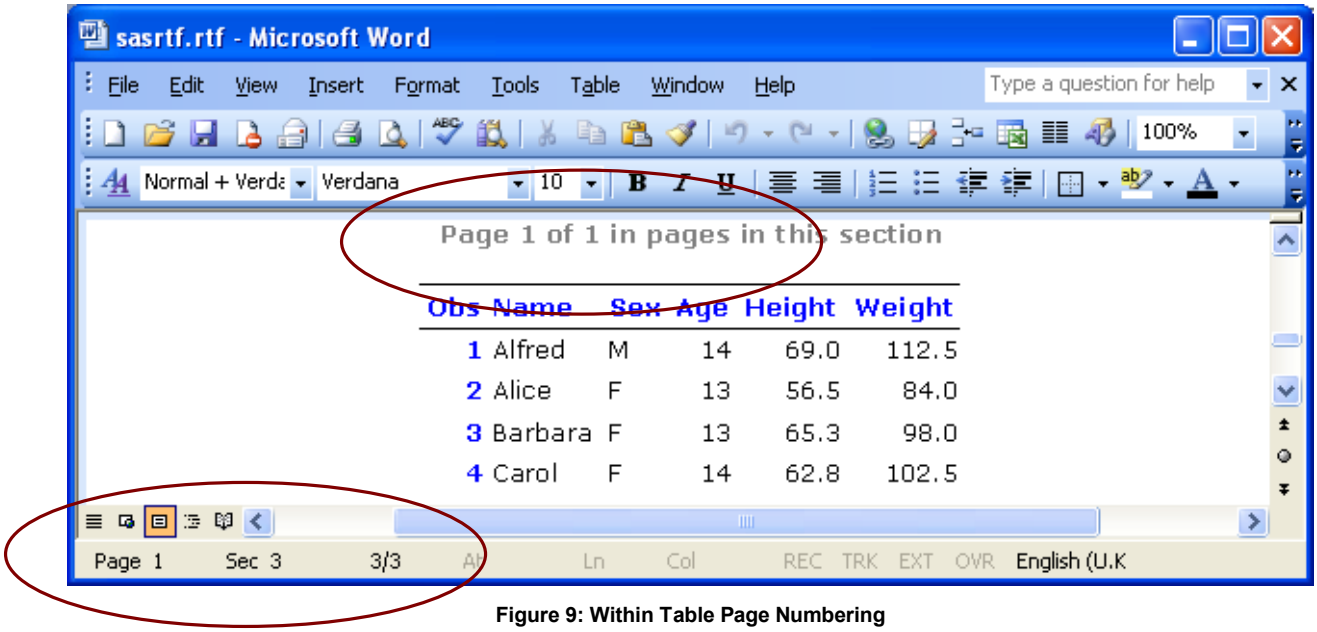


Figure 9: Within Table Page Numbering

DOCUMENT INFORMATION

It has been requested to me on more than one occasion that the filename and path of the document containing a report output to printed in a footnote within the report. There are several disadvantages of hard coding a path into output as the storage location may change with distribution of our reports.

With ODS RTF we can embed the field codes into our documents which always show the current path and location of our document. This can be achieved with the following syntax:

```
ODS TEXT= "{Document: \field{\*\fldinst { FILENAME \\* Lower\\p \\* MERGEFORMAT
}}} ";
```

This code was submitted after a procedure call; producing the following output, shown partially in Figure 10 :

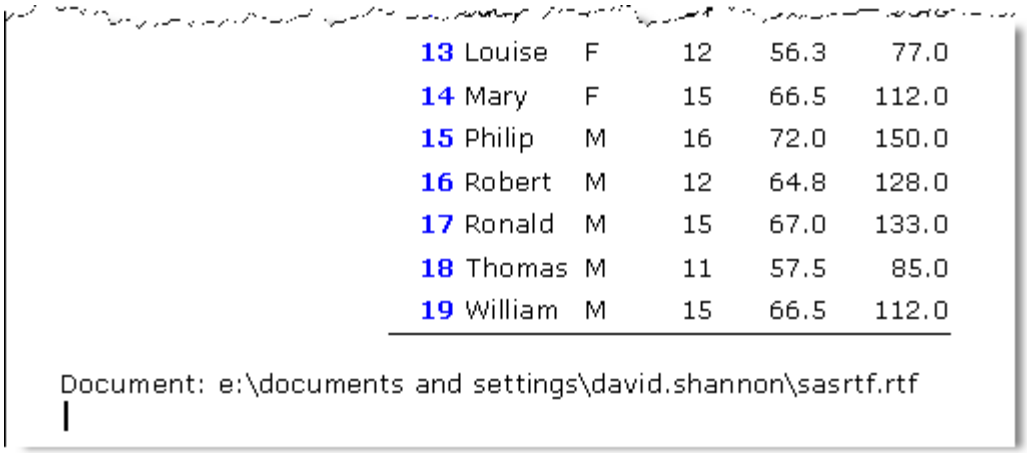


Figure 10: Adding Document Information

OH THOSE BLINKING TEXT EFFECTS!

A very common approach to reporting today is to make use of traffic lighting techniques. This simply means that attributes of individual datum are presented according to their value, for example negative values are shown in red as opposed to black etc.

Microsoft Word enables some special text formatting effects, the first of which is the ability to make text blink. This feature is also accessible via RTF control words.

The following example demonstrates how, with the use of a picture format, such special effects can be applied to data. Three p-values are presented with text effects applied presented according to the significance of the p-value:

```
proc format ;
  picture rtfpvalue
    0-<0.001        = "9.9999}" (prefix="{\b\animtext2 ")
    .001 - <  0.05  = "9.9999}" (prefix="{\animtext2 ")
    .05 - high      = "9.9999";
run;

ods rtf style=plain;
title "Oh Those Blinking Text Effects";
proc report data=blinking noheader nofs style={frame=box};
  column label result1;
  define label / display style={cellwidth=4cm just=l};
  define result1 / display style={just=c} f=rtfpvalue.;
run;
ods rtf close;
```

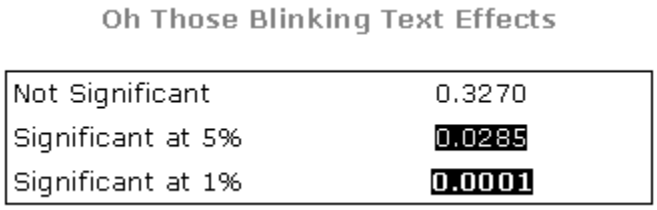


Figure 11: Applying Effects According to Data Value

Here the non-significant p-value is presented without any style effect. The second p-value 0.0285 is statistically significant at the 5% level, hence is made to blink. The third p-value is considered to be highly statistically significant (<0.001) hence is presented as both blinking and in bold font.

I couldn't really advocate the use of presenting statistical findings in such a manner… but you can see how the concept could be applied to any data!

INSERTING IMAGES

Dynamically inserting images into the body of a report can be approached in several ways. The simplest method I have found takes the following steps:

- ❶ Create the table of data that is to be reported;
- ❷ Add a column to the table which contains the path and filename of the image to be reported;
- ❸ Using proc report and compute block, apply a call define statement to dynamically change the value of the PREIMAGE style attribute;
- ❹ Take the variable holding the image filename and set its value to blank.

Notice also that an image can be inserted above the header of the report (❺). The following snippet shows how the 2005 Champions League winning team, their career statistics and photos were loaded into a data set for reporting:

```
data lfc; ❶
  length surname $20 forename $20 image $256;
  input shirt surname $ forename $ games goals ;
  image="&imagepath.No"!!compress(put(shirt,best.))!!".jpg"; ❷
datalines;
1 Dudek Jerzy 174 0
3 Finnan Steve 86 1
.
.
<data removed>


ods rtf style=plain;
title "Inserting Images";
proc report data=lfc nofs style={ preimage="&imagepath.\lfc.png"}; ❺
  column image shirt surname forename games goals;
  define image / "" display style={cellwidth=3cm};
  compute image;
    call define(_col_,"Style","Style=[PREimage='"!!image!!"']"); ❸
    image="";  *<<< Avoid image location being printed; ❹
  endcomp;
run;
ods rtf close;
```

The output in Figure 12 below shows ❻ how the club logo for Liverpool FC is inserted above the table (but below any titles) and ❼ how the images are dynamically loaded into each row of the table.

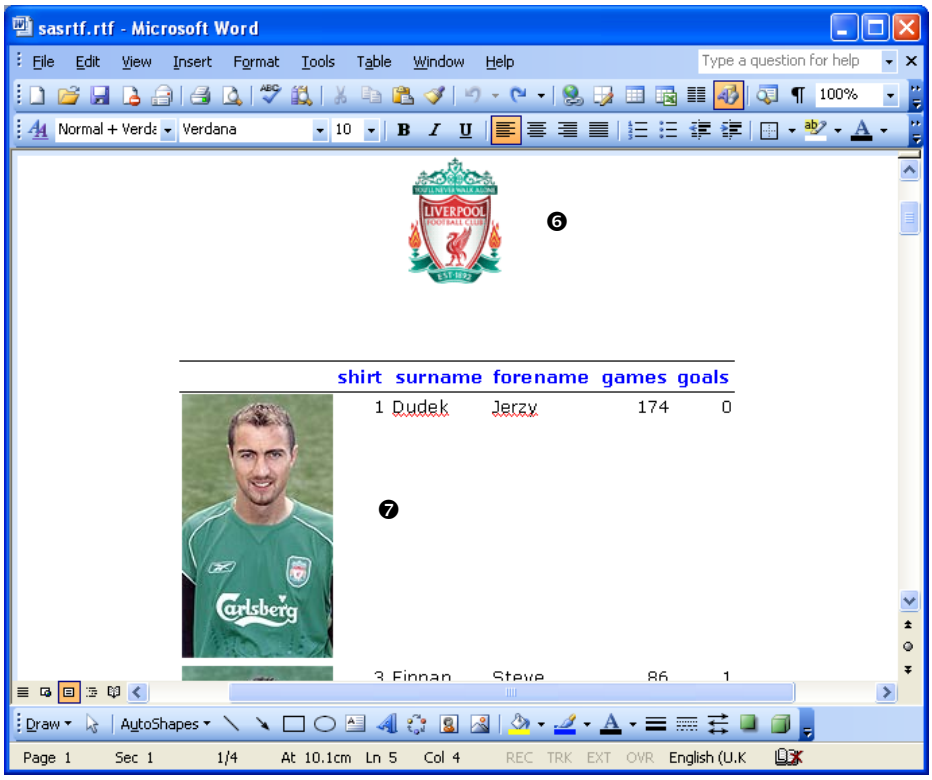


Figure 12: Inserting Images into Microsoft Word

INSERTING SCIENTIFIC EQUATIONS

To correctly present formulae or perhaps something as simple as a square root sign has previously been difficult if not impossible using traditional monospace output. As with many of the examples seen here with ODS RTF, we can take advantage of Microsoft Word's capability of displaying various symbols. By inserting a field called EQ we can construct equations in our Word document.

The following code creates three records in a data set which are then sent to a RTF file via PROC REPORT. Each record uses a different style of equation.

```
data t;
  length x $150;
  x="{A capital sigma: \field{\*\fldinst {EQ \\I\\su(i=1,n, ) }}}";
  output;
  x="{A cube root: \field{\*\fldinst {EQ \\R(3,27) }}}";
  output;
  x="{We can display \field{\*\fldinst {EQ \\x \\to \\bo(text with borders) }}}";
  output;
run;

ods rtf style=plain;
proc report data=t noheader nofs style={protectspecialchars=off rules=none};
run;
ods rtf close;
```

The output generated is shown in Figure 13 below:

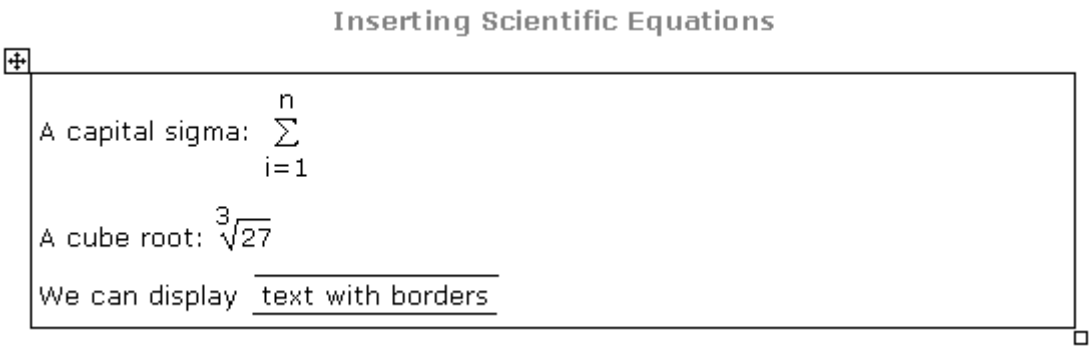


Figure 13: Inserting Scientific Equations

THE LIFE OF ODS RTF IS YOUNG

Since ODS RTF became production in SAS 8.1 there has been a steady stream of new features introduced as the versions have been released and there is no sign of this trend diminishing as we are informed about the next release (SAS 9.2) as I write.

The most significant changes to how output can be organised in the document is with the introduction of ODS LAYOUT and ODS DESIGN, which are experimental features in SAS 9.1.3. These are intended to enable ODS RTF/PDF/PRINTER users to specify either absolute or relative regions within a page that will contain output.

The direction taken of new features is generally guided because of the feedback given to SAS Institute R&D, so if you are a user of ODS RTF; take the time once in a while to provide some feedback.

CONCLUSION

Compared to the traditional listing output we can quickly create reports which are both professional in appearance and easy to distribute.

There are benefits to both the programmer in terms of reduced manipulation of output and also to the user for enhanced and professional appearance.

Although initially complex it is possible to extract some of Microsoft Word's features to further enhance RTF output, with more features added as new versions of SAS are released.

CONTACT INFORMATION

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

David Shannon
Amadeus Software Ltd.
Orchard Farm
Witney Lane
Leafield
OX29 9PG
Work Phone: +44 (0)1993 878287
Email: david.shannon@amadeus.co.uk
Web: www.amadeus.co.uk