RTF (rich text format) from scratch

ADUG Melbourne, 20 January 2014

Tony Bryer

Greentram Software

tony@greentram.com

www.greentram.com

Greentram's programs

- *Each program handles beam and column design for domestic-scale structures*
- **SuperBeam** – steel and timber design
- **ProSteel** – more advanced steel design
- **EuroBeam** – combines virtually everything in SuperBeam and ProSteel, with design to the new pan-European Eurocodes

Greentram's programs

- *Programs produce calculation sheets*

Steel calculation to BS449 Part 2 using S275 (Grade 43) steel
SECTION SIZE : 203 x 133 x 30 UB S275 (Grade 43)
D=206.8 mm B=133.9 mm t=6.4 mm T=9.6 mm
L_E/r_y = 5.40 x 100/3.17 = 170 D/T = 21.5
Permissible bending stress, p_{bc} = 89.4 N/mm² (Tab
Actual bending stress, f_{bc} = 19.9 x 1000/280.0 = 70.9 N/mm² OK
Maximum shear in web, f_s = 14.7 x 1000/(6.4 x 206.8) = 11.1 N/mm² OK
Check unstiffened web capacity with load of 14.72 kN
Bearing: p_b = 210N/mm² (Table 9); C1 = 40.0 kN; C2 = 1.34 kN/mm
Buckling: p_c = 146N/mm² (Table 17a); C1 = 96.6 kN; C2 = 0.934 kN/mm
Unstiffened web bearing capacity, F_w = 40.0kN: no minimum stiff bearing length r
Total deflection = 60.3 x 1e8/(205,000 x 2,900) = 10.1 mm (L/532) OK

Extract from SuperBeam steel beam calculation

Internal text tagging

Within each of our programs text to be output to screen or printer is held in a stringlist with formatting controlled by ruler lines and tags

Fundamental principle is no read ahead (I started back in the days of dot matrix printers!)

Screen MDI child window and print routines just handle this text without knowing or caring what it is

PrintLine tagging

- Formatting is set by single character ***<x>***
- **<n>,<e>,<s>,<h>,<*>,<->** : set font
- **<\>..</>**, **</>..<\>** : subscript, superscript
- **<1>,<2>** … superscript number
- **!a,b,c** ruler line sets tabs, e.g. !10,-30,-40
  (- = right aligned tab)
- **<!>** jumps to next tab

Sample

```
<I>Column calculation to EN1993-1-1 using S275 steel
<e>Location: Mu
Min depth
Length: 4.0 m.
!1005,8,-28,-38,-48,-57,-68,-78,-88,-98
Pos<!>Dur<!>Load<!>kN  <!><!>Factored load<!>Offset<!><!>Moment y-y  <!><!>Moment
z-z
<!><!><!><!>6.10a<!>6.10b<!><!>6.10a<!>6.10b<!>6.10a<!>6.10b
A<!>QA<!>Axial load<!>10.00<!>10.50<!>15.00<!><!><!><!><!><!>
A<!>G<!>Axial load<!>5.00<!>6.75<!>6.24<!><!><!><!><!><!>
A<!>QA<!>Axial load<!>10.00<!>10.50<!>15.00<!><!><!><!><!><!>
Y<!>QA<!>My moment<!>2.00<!><!><!><!>2.10<!>3.00<!><!><!>
Z<!>QA<!>My moment<!>3.00<!><!><!><!><!><!>3.15<!>4.50<!>
!-28,-38,-48,-57,-68,-78,-88,-98
<e>Total load<->27.75<o><!><e><O>36.24<o><-
><!><!><O>2.10<o><!><e><O>3.00<o><->3.15<o><!><e><O>4.50<o><->
<n><c>Load durations: G: Dead; Qx: Imposed: QA: Residential
<e>SECTION SIZE<-> : <s>152 x 152 x 23 UC  S275
Section properties: <>>B  = 152.2mm   D = 152.4mm   T = 6.8mm   t = 5.8mm
A<\>g</> = 29.2cm<2>
<!>i<\>y</> = 6.54cm   i<\>z</> = 3.70cm  Wc\>pl,y</> = 182cm<3>   Wc\>el,y</> =
164cm<3>   Wc\>pl,z</> = 80.1cm<3>   Wc\>el,z</> = 52.6cm<3>
```

Does just what **we** need

But is not compatible with anything else

Which didn't matter since we didn't want users to be able to copy content because of

- Anti-piracy concerns – each printout carries the program build and serial number traceable back to the licensee
- Output integrity concerns – fear of unscrupulous users editing program output

But …

- Anyone who has a PDF generator (e.g. Acrobat) can set this as their printer and then can generate open PDF files which can edited or copied from (though copies lose a lot of formatting)

In the meantime …

- I'm working on an Intraweb web-hosted version of EuroBeam.
- Initial plan was to generate HTML in parallel with custom tagging
- Instead I settled on writing a converter which converts a tagged text stringlist to HTML

So when a user asked …

- For a way of copying program output to the clipboard in RTF or another portable format so he could paste it into a CAD drawing
- (a) on reflection the former security misgivings no longer held; and
- (b) having tackled the conversion issues already, duplicating this for RTF looked much less challenging

So what is RTF?

- Text file format first introduced by Microsoft in 1987, readable by many programs.
- The bad old days: *"RTF is defined as whatever Word saves when you ask it to save as RTF"*

Anon

RTF history

- **1.0:** 1987: Word 3.0 for Macintosh
- **1.0:** June 1992: Word for Windows v2
- **1.1, 1.2:** Unknown, unavailable
- **1.3:** January 1994: Word v6
- **1.4:** September 1995: Word v7 (Word 95)
- **1.5**: April 1997: Word v8 (Word 97)
- **1.6**: May 1999: Word v9 (Word 2000)
- **1.7**: August 2001: Word v10 (Word 2002)
- **1.8**: April 2004: Word v11 (Word 2003)
- **1.9**: January 2007: Word v12 (Word 2007)

So what is RTF?

- Now documented (1.9: 278 pages!)

Rich Text Format (RTF) Specification
Version 1.9.1

Why RTF?

- Didn't they use it in the 1990s?
- I just remember It from WinHelp files
- Why use it now?

Why RTF?

- Didn't they use it in the 1990s?
- I just remember It from WinHelp files
- Why use it now?
- Because it's like English – virtually everyone understands it to a greater or lesser extent

Why not use the real thing?

A normal programmer would conclude that Office's binary file formats:

- *are deliberately obfuscated*
- *are the product of a demented Borg mind*
- *were created by insanely bad programmers*
- *and are impossible to read or create correctly*.

Joel Spolsky (who then explains why this isn't totally true)

Why RTF?

If you really want to generate fancy formatted Word documents, your best bet is to create an RTF document. Everything that Word can do can be expressed in RTF, but it's a text format, not binary, so you can change things in the RTF document and it'll still work.

You can create a nicely formatted document with placeholders in Word, save as RTF, and then using simple text substitution, replace the placeholders on the fly. Now you have an RTF document that every version of Word will open happily.

Joel Spolsky

So what is RTF?

- RTF only uses 7-bit ASCII characters – all others are encoded – you can edit RTF in a simple editor like Notepad
- Text – like HTML, line breaks don't matter
- Very simple format in principle … though RTF generated by Word is not human-friendly

Key RTF options

- Fonts: name, size, bold, italic, underline, superscript, subscript, colour etc
- Lines: left/centre/right align, margins, paragraph and line spacing
- Tabs

Old but good

A simple RTF document

- **{\rtf1\ansi{\fonttbl\f0\fswiss Arial;}\f0\fs22\pard**
- **This is some {\b bold} text.\par**
- **}**

A simple RTF document

- **{\rtf1\ansi**{\fonttbl\f0\fswiss Arial;}\f0\fs22\pard
  This is some {\b bold} text.\par
- }
- **{** - open document
- **\rtf1\ansi** – prologue: all RTF is v.1
- **{..}** is a group. Generally formatting specified within a group affects only the text within that group
- **\xxx** is a control word, max 32 chars, case sensitive, terminated by character other than a letter or digit (usually a space or another control word)
- Most measurements are in twips, 1/20 point = 1/1440”

A simple RTF document

- {\rtf1\ansi**{\fonttbl\f0\fswiss Arial;}**f0\fs22\pard
- This is some {\b bold} text.\par
- }
- {
- \rtf1\ansi
- **{\fonttbl** – font table
- **\f0\fswiss Arial;** – font 0 definition
- **}** – end table
- Font table defines fonts used in document
- Font definitions do not have to be in numeric order. The font family can be fnil, froman, fswiss, fmodern, fscript, fdecor – get using EnumFonts or EnumFontFamiliesEx

A simple RTF document

- {\rtf1\ansi{\fonttbl\f0\fswiss Arial;}**\f0\fs22**\pard
- This is some {\b bold} text.\par
- }
- {
- \rtf1\ansi
- {\fonttbl\f0\fswiss Arial;}
- Document starts here
- **\f0** – use font 0 (Arial)
- **\fs22 –** font size 11pt – RTF sets font size in half points

A simple RTF document

- {\rtf1\ansi{\fonttbl\f0\fswiss Arial;}\f0\fs22**\pard**
- This is some {\b bold} text.\par
- }
- {
- \rtf1\ansi
- {\fonttbl\f0\fswiss Arial;}

- Document starts here
- \f0\fs22

- **\pard** – start new para with default properties

A simple RTF document

- {\rtf1\ansi{\fonttbl\f0\fswiss Arial;}\f0\fa11\pard
- **This is some {\b bold} text.**\par
- }
- {
- \rtf1\ansi
- {\fonttbl\f0\fswiss Helvetica;}
- Document starts here
- \f0
- \pard

- **This is some**
- **{\b bold}** - \b bold affects following text in {} only
- **text.**

A simple RTF document

- {\rtf1\ansi{\fonttbl\f0\fswiss Arial;}\f0\fs22\pard
- This is some {\b bold} text.**\par**
- **}**

- {
- \rtf1\ansi
- {\fonttbl\f0\fswiss Arial;}
- *Document starts here*
- \f0
- \pard
- This is some {\b bold} text.

- **\par** – end paragraph
- **}** – finish doc (pair of opening '{')

A little more RTF

- Another basic construct is a colour table

- ***{\colortbl***
- ***;*** ***color 0 – use default***
- ***\red0\green77\blue187;*** ***color 1***
- ***\red0\green176\blue80;*** ***color 2***
- ***}***

Unlike fonts, colours are not numbered in the table. First undefined colour (0) means use system default. Others are defined by RGB values, 0-255. Use as **\cfN** and **cbN** – latter not always supported. You don't have to use all defined fonts and colours.

A little more RTF - characters

- **\'hh** – hex value (e.g. \'a3 is a pound sign, '£')
- **\uc1\u*number**** – unicode character n 255-32768, e.g. \uc1\u21487*
- **\uc1\u-*number**** – unicode character n >= 32768, number=n-65536
- **\b, \i, \ul, \scaps, \strike** – bold, italic … cancel with following '0'
- **\sub, \super, \nosubsuper** – subscript, superscript, cancel
- **\\, \{, \}** – escapes
- **\~** - non-break space
- **\lquote, \rquote, \ldblquote, \rdblquote** – single/double quotes

A little more RTF – para and page

- **\slN\smult1** – space lines (N=mult of single x 240: 240=typewriter 1/6")
- **\liN** – Left indent (twips: 1 twip = 1/20pt = 1/1440")
- **\fiN** – First line indent relative to above
- **\riN** – Right indent
- **\ql, \qr, \qc, \qj** – left, right, centre, justify
- **\margtN, \margbN, marglN, margrN** – margins in twips
- **\tql, \tqr, \tqc, \tqdec** – tab l/r/c/dec – follow with **\txN**
- **\pvpg\phpg\posxN\posyN\abswN** – start para x,y + width
- **\page** – page break
- **\sbN, \saN** – space before/after (above/below) paragraph

Play with some RTF

- Using a little program based on this video by Alister Christie
- http://www.youtube.com/watch?v=A5TSI9NKyAg

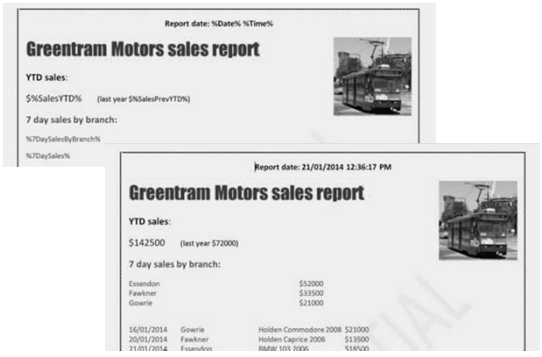
Using a RTF template

- RTF doesn't contain any 'nasties' such as offsets or jump tables, so we can - assuming we follow the rules - change the content without affecting the validity of the file
- `{\rtf1\ansi{\fonttbl\f0\fswiss Arial;}`
  `\f0\fs22\pard` **`%content%`**`\par`
- Because RTF is text, the placeholder replacement can be done using any text-handling program – Delphi, C++, PHP etc.

Using a RTF template

- I showed off a little program that produced various reports from dummy car sales data. The program reads in an RTF template and replaces placeholders such as %SalesByBranch% with the corresponding report.
- The WordPad-produced report is very basic
- The Word-produced template contained an image, page background, 'Confidential' watermark and other embellishments
- In a real life situation the end user can be given a list of recognised placeholders and make their own templates, saved as RTF

Template and output



Using a RTF template

- Reports are an obvious application
- Invoices etc
- Custom letter/agreement/quote generator – select required paras then generate
- etc

Generating RTF from scratch

Things to note:

- Syntax (obviously)
- Need to escape \, { and } as \\, \{ and \}
- Need to encode characters > 127
- Need to ensure that conversions to twips return an integer value
- You probably want to define page size and margins (My Windows defaults to letter size paper)

MyRTF conversion code

- Not 100% but works OK

My RTF conversion

- Basically straightforward.
- Allow users to pick fonts used for RTF output
- For load data we expand the data to fill printer page width – tab positions are proportionate to page width
- RTF tabs are absolute distances – we assume an default average character width of 100 twips (trial and error) – so if rightmost tab is 92 and print width 6.75”
- TabTwips := round(6.75 * 1440 / 92) = 106 twips

MyRTF conversion code

- Key issues:
- RTF doesn’t allow two tabs in same place
- In our code !-30,30 is a valid construct so
- **<!>20<!>kN** - 20 is right aligned, kN left aligned, so this returns
- `.........!.............`
- `          20kN`
- Fix is for code picks this up and nudge second instance 10 twips

MyRTF conversion code

- **Key issues:**
- In our code <!><!><!> jumps to third tab (as defined by ruler) regardless of current position. RTF (as most WPs) jumps to third tab from where cursor is
- `...>..........<.....<`
- **1.<!>Load<!>10.0<!>10.0**
- **Total load<!><!>10.0<!>10.0**
- Returns
- **1 Load 10.0 10.0**
- **Total load 10.0 10.0**
- Literal translation to RTF
- **1 Load 10.0 10.0**
- **Total load 10.010.0**
- Fix is to insert an extra ruler line omitting redundant tab

MyRTF conversion code

- **Key issues:**
- For right-aligned tabs we need to generate any tagged output in reverse order thus to get right-aligned πr^2
- `<2>r<*>p<->`
- Is rendered from tab pos working RTL
- `<2>` - superscript 2
- `r` - r
- `<*>` - switch to symbol font
- `p` – p = pi in symbol font
- `<->` - revert to normal

MyRTF conversion code

- **Solution:**
- Pull out this bit of text and reverse it
- `<2>r<*>p<->`
- Become
- `<->p<*>r<2>`
- On the right lines but the <*> and <-> symbol andnormal font tags are now the wrong way round so we have to fix these up. Likewise for the <\></> subscript, </><\> superscript constructs

MyRTF conversion code

- We use <O><o> tag pair for overlines. First saves the x and y position, second draws a line from here to new x position.
- RTF has no simple way of doing this (neither does Word – you have to use the equation editor).
- The answer is to use the Unicode character 0773 – overline previous character. For this to work, overlines must be inmost tags. So lambda bar has to tagged as <*>**<O>l<o>**<-> not **<O>**<*>l<->**<o>**
- \f6 <O>l<o>\f0 -> \f6 l\uc1\u773*\f0
- `<O>\f6 l\f0 <o> \\uc1\u773* 6 \uc1\u773* !!!!`

Copying RTF stringlist to clipboard

```
procedure CopyRTFToClipboard(RTF: TStringList);
var
  SS: TStringStream;
  hMem: THandle;
  pMem: pointer;
  fmt: Cardinal;
begin
  if length(RTF.Text) = 0 then
  begin
    WMessBox('RTF copy error', 'Nothing to copy', '', [cbOK]);
    exit;
  end;

  fmt := registerClipboardFormat(CF_RTF);
  SS := TStringStream.Create;
  try
    try
      RTF.SaveToStream(SS);
      { Rewind stream position to start }
      SS.Position := 0;
      { Allocate a global memory  block the size of the stream data }
      hMem := GlobalAlloc(GMND or GMEM_DDESHARE, SS.Size);
      if hMem <> 0 then
      begin
        { Succeeded, lock the memory  handle to get a pointer to the memory }
        pMem := GlobalLock(hMem);
        if pMem <> nil then
        begin
          { Succeeded, now read the stream contents into the memory  the pointer points at }
          try
            SS.ReadpMem^, SS.Size);
            { Rewind stream again, caller may be confused if the stream position is
              left at the end }
            SS.Position := 0;
          finally
            { Unlock the memory  block }
            GlobalUnlock(hMem);
          end;
          { Open clipboard and put the block into it. The way the Delphi clipboard
            object is written this will  clear the clipboard first. Make sure the
            clipboard is closed even in case  of an exception. If left open it would
            become  unusable for other apps. }
          Clipboard.Open;
          try
            Clipboard.SetAsHandle(fmt, hMem);
          finally
            Clipboard.Close;
          end;
        end
        else
        begin
          { Could not lock the memory  block, so free it again and raise an out of
            memory exception }
          GlobalFree(hMem);
          OutOfMemoryError;
        end;
      end
      else
      { Failed to allocate the memory  block, raise exception }
      OutOfMemoryError;
    except
      on E: Exception do
      WMessBox('RTF copy error', 'Error copying RTF to clipboard: ' + E.ToString, '', [cbOK]);
    end;
  finally
    SS.Free;
  end;
end;
```

Delphi TRichEdit alternatives

- Delphi uses RichEdit 2.0 as shipped with Windows 98 so is limited in some respects. QC40408: Update TRichEdit: "*Please bring the Delphi TRichEdit control up to date with the one shipped with XP*", filed 2007, still open!
- Native controls for editing and viewing RTF (not tried by me):
- WPTools (wpcubed.com) – E398+
- TRichView (trichview.com) US$330

Resources

- RTF Pocket Guide book – part 1 free download
  oreilly.com/catalog/rtfpg/chapter/part1.pdf
- Official RTF specification – Microsoft
  www.microsoft.com/en-au/download/details.aspx?id=10725
- More friendly RTF 1.5 specification
  www.biblioscape.com/rtf15_spec.htm
- The RTF cookbook
  search.cpan.org/~sburke/RTF-Writer-1.11/lib/RTF/Cookbook.pod
- Language numeric codes
  msdn.microsoft.com/en-us/goglobal/bb964664.aspx
- Alister Christie video – extracting text from a TRichEdit
  www.youtube.com/watch?v=A5TSI9NKyAg