Network Working Group
Request for Comments: 3285
Category: Informational

M. Gahrns
Microsoft
T. Hain
Cisco

May 2002

Using Microsoft Word to create Internet Drafts and RFCs

Status of this Memo

Copyright Notice

Abstract

This document describes the steps to configure the Microsoft Word application to produce documents in Internet Draft and RFC format.

Table of Contents

1. Overview

This document describes the steps to create a Microsoft Word 97 or later template to assist those producing Internet drafts. The resulting configuration allows for simple WYSIWYG editing of drafts and RFCs while producing output that is in accordance with IETF draft and RFC submission specifications. (72 Characters per line, 58 lines per page, each line terminated by a CRLF, and each page followed by a LF, etc.) Using Word's text justification and table capabilities may facilitate creating ASCII stick drawings.

While the authors happen to have been employed by Microsoft during much of this document's evolution, it is not a product of Microsoft and is unsupported.

Included is a detailed description of how the RFC Text and RFC Heading styles are defined. This should prove useful to those wishing to do further customization work or to create a similar template for other versions of Microsoft Word.

It also includes a description and the source of the CRLF.EXE program that is used to create the final text file output. Feedback about this program is consistent with the fact that each version of Windows has a slightly different Generic Printer driver. Since this document will not be kept current with every Windows revision, the code sample is provided as a basis for personal customizations.

Copies of the template in Microsoft Word format and the CRLF.EXE program can be found at:

ftp://ftp.ietf.org/internet-drafts/2-Word.template.rtf
ftp://ftp.ietf.org/internet-drafts/crlf.exe
ftp://ftp.rfc-editor.org/in-notes/rfc-editor/2-Word.template.rtf
ftp://ftp.rfc-editor.org/in-notes/rfc-editor/crlf.exe

While the process described in this document can be used to create Word format documents, using the editions of Microsoft Word for Windows or the Apple Macintosh, the actual text format file for submission to the I-D or RFC editors is only available from the Windows edition. This limitation is due to the lack of a Generic Printer driver for the Macintosh.

2. Conventions used in this document

In this document the steps for walking a pull-down tree are indented on subsequent lines. This allows abbreviation rather than a barrage of 'then click' or 'select' strings in a paragraph form. Example:

Help
About Microsoft Word

3. Instructions for producing Internet drafts and RFCs

1) Microsoft Word's "auto-formatting" can result in some undesired characters when creating the IETF standardized format. (I.e., it will insert special characters for quotation marks, add special formatting when creating lists, etc, which will appear as unintelligible character sequences when displayed by plain-text readers.) To avoid this, turn off "auto formatting."

   Tools
   Autocorrect

   On the property pages, 'AutoFormat' and 'AutoFormat As You Type', turn off all of the auto formatting options. If you forget, or frequently switch between IETF format and not, typing a ^Z after each auto-format event will undo the formatting change. This of course requires awareness of the event.

2) Two special styles need to be defined: RFC Heading and RFC Text. If you choose automatic reference numbering or table of contents (defined below), the style for Endnote Reference, Endnote Text, and TOC need to be modified. The entire draft must be written using these styles for the spacing to come out correctly.

   This RFC has been produced using the styles & procedures defined within. You may follow the instructions below for creating the RFC Heading and RFC Text styles or simply acquire a copy of the MS Word (.rtf) file from one of the locations above, delete the body text, insert your rfc text and apply the styles to the body and headers as appropriate.

   *** Do not use bold, underlining, italics, etc., or you will lose the WYSIWYG editing feature since these settings affect the number of characters that can occur on a line. When the resulting Internet draft is saved as plain text, all that formatting will be lost anyway. ***

3) Print the document to the Generic Text Printer, and save the output to file. If you do not have the Generic Text Printer driver installed, install it from the Control Panel. (Printers, Add Printer, local/My Computer, any LPT port (you will be printing to a file), select Generic, Generic/Text Only from the combo box). When you print to a file, a pop-up will ask for the file name.

4) Run the CRLF program in a DOS window to automatically add carriage returns.

```
Usage is CRLF <source> <destination>
```

Where <source> is the name of the file produced by printing to the generic text printer, and <destination> is the name of the text draft you are producing. An example (where the files CRLF.EXE and draft-00.prn are in the C:/TEMP directory) would be:

```
cd c:/temp
crlf draft-00.prn draft-00.txt
```

5) Check to see if any non-ASCII characters have slipped in by viewing the document with a simple text viewer. The Unix program 'less'[1] will highlight non-ASCII characters. If a non-Microsoft operating systems is not available, the Notepad program will display and not-try to re-interpret any special characters.

3.1 Defining Microsoft Word Page Layout and Styles

These are settings used to define the RFC Text and RFC Heading styles. Note: the menu options to set these are enclosed in parenthesis and are listed for Microsoft Word 97. They may differ slightly for other versions of Microsoft Word.

1) Set measurement units to points.

```
Tools
   Options
      General
         Measurement units = points
```

2) Set margins as follows: (File, Page Setup, Margins)

```
Top:         24 pts
Bottom:      0 pts
Left:        0 pts
Right:       93.6 pts
Gutter:      0 pts
Header:      0 pts
Footer:      0 pts
```

The right margin is what determines 72 characters per line. Using 12 pt font, 10 chars/inch, 72 chars = 7.2". Using paper that is 8.5" wide. 8.5" - 7.2" = 1.3" = 93.6 pts If you get "one or more margins are outside the printable area" message, select Ignore. This seems to depend on the printer you currently have selected.

3) Set paper size as follows:

```
File
   Page Setup
      Paper Size
         Width:  612 pt (8.5")
         Height: 660 pt (12pt * 55 lines per page)
```

The height of the paper is what determines 55 lines per page.

4) Set headers/footers to be different for the first page.

```
File
   Page Setup
      Layout
```

5) Define a RFC Heading Style.

```
Format
   Style
      New
```

RFC Heading: Heading1 + Font: Courier New, 12pt, Not Bold, Line spacing exactly 12pt., Space before 0 pt after 0 pt, Level 1

NOTE: Line Spacing Exactly 12pt is very important. Set this through Format: Paragraph

Additional Heading levels can be defined by repeating this step and incrementing the Level #. If Numbered Headings are desired:

```
Format
   Bullets and Numbering
      Outline Numbered
         Select preferred style
         Customize
            More
               Link level to style RFC Heading
```

6) Define a RFC Text Style.

```
Format
   Style
      New
```

RFC Text: Normal+Font: Courier New, 12pt, Indent: Left 21.6pt, Line Spacing Exactly 12 pt.

Line Spacing and indent are set through Format, Paragraph. This leaves a 3 character left indent for the RFC text

7) Fix the Header Style.

```
Format
   Style
      Header
```

Header: Normal+Font: Courier New, 12pt, Line Spacing Exactly 12pt, Clear the tabs previously defined, and add Tabs 252 pt Centered, 504 pt Right Flush

8) Fix the Footer Style.

```
Format
   Style
      Footer
```

Footer: Normal+Font: Courier New, 12pt, Line Spacing Exactly 12pt, Tabs 252 pt Centered, 504 pt Right Flush

9) Define your headers and footers for the first page.

```
View
   Headers
   (  on first page)

Header: No Header
Footer:  Blank line
         Blank line
AuthorName <tab> <tab> [Page <page number field>]
```

10) Define subsequent headers and footers.

```
View
   Headers
      (on second page)

Header: <tab> Title <tab> Month, Year
         Blank line
         Blank line
Footer:  Blank line
         Blank line
AuthorName <tab> Expiration <tab> [Page <page number field>]
```

11) Set Tabs to be every three spaces.

```
Format
   Style
      RFC Text
         Tabs: Left 21.6, 43.2, 64.8, 86.4, 108, 129.6,
         151.2,172.8, 194.4, 216, 237.6, 259.2, 280.8,
         302.4, 324, 345.6,367.2, 388.8, 410.4, 432,
         453.6, 475.2, 496.8
```

12) Fix the Table-of-contents Styles. Repeat for each level.

```
Format
   Style
         TOC1: RFC text +, Automatically update, Clear all tabs,
            Add tab Rt. Flush, 504pt, ... leader
         TOC2: RFC text + Indent: Left 43.2pt,
            Automatically update, Clear all tabs,
            Add tab Rt. Flush, 504pt, ... leader
         TOC3: RFC text + Indent: Left 64.8pt,
            Automatically update, Clear all tabs,
            Add tab Rt. Flush, 504pt, ... leader
```

3.2 Positioning the document identifiers on the first page

The 'Table' tool can be used to assist with justification of the document identifiers on the first page. Each cell in the table maintains its own justification characteristics, so getting left and right justification on the same line is simplified. On the Toolbar select the icon that looks like a grid with a dark bar across the top. This will pop-up a table array. Drag the mouse across to select the number of rows and columns (for the opening header 4 rows x 2 columns, unless there are several authors). Select the table that was just inserted by click-and-hold in the left margin, and then clear the boarders.

```
Format
   Borders and Shading
      None
```

Select the cells on the right (position the cursor just above the top cell, when the cursor becomes an arrow pointing down, click) and set justification right. (The default is to take justification from the line it is being positioned on, so the left column shouldn't need changing.)

```
Format
   Paragraph
      Right
```

If necessary, move the center divider to the right for the document title. Select the left column of cells, then position the cursor over the dividing line. When it changes to parallel bars with right/left arrows, click-and-hold, then drag the line as necessary.

3.3 Automatic date

For those who frequently update drafts, and find they occasionally forget to update the current save and expire dates, there is a way to automate those fields. While it is rather complex to set up the expire-month field, it only needs to be done once in a template file, and all future drafts benefit.

To automatically set the current date on save, select the lower right cell in the table created above, and insert the save date.

```
Insert
   Field
      Date and Time
         SaveDate
         In the box below the sample "field codes",
            modify as necessary to make it look like:
            - SAVEDATE \@ "MMMM YYYY" - (between the -'s).
         OK
```

The field will have a gray background on the screen, but will not affect the printed version. Double click on the field, copy, and then replace the Month, Year in the header (10 in Layout Styles above) with a paste.

Setting up the expire-date is similar, but requires inserting nested fields. Select the location for the month then insert an IF field.

```
Insert
   Field
      MailMerge
         IF
         OK
```

This will result in an error. Right click on the error message, and select Toggle Field Codes. This will allow further editing. Select the space after the initial IF, then insert another field: SaveDate (as above but this time only the month digit is used "M"). Right click on the number it inserts and Toggle Field Codes again. Follow

the right brace } with =, then the month to test, followed by the month name 6 months later. At this point loop and insert another IF, until all 12 are done. Follow the last one with a "" to complete the syntax. The resulting expanded field code will look like:

```
{ IF { SAVEDATE  \@ "M" \* MERGEFORMAT } = 1 July { IF { SAVEDATE  \@
"M" \* MERGEFORMAT } = 2 August { IF { SAVEDATE  \@ "M" \*
MERGEFORMAT } = 3 September { IF { SAVEDATE  \@ "M" \* MERGEFORMAT} =
4 October { IF { SAVEDATE  \@ "M" \* MERGEFORMAT } = 5 November { IF
{ SAVEDATE  \@ "M" \* MERGEFORMAT } = 6 December { IF { SAVEDATE  \@
"M" \* MERGEFORMAT } = 7 January { IF { SAVEDATE  \@ "M" \*
MERGEFORMAT } = 8 February { IF { SAVEDATE  \@ "M" \* MERGEFORMAT } =
9 March { IF { SAVEDATE  \@ "M" \* MERGEFORMAT } = 10 April { IF {
SAVEDATE  \@ "M" \* MERGEFORMAT } = 11 May { IF { SAVEDATE  \@ "M" \*
MERGEFORMAT } = 12 June "" \* MERGEFORMAT } \* MERGEFORMAT } \*
MERGEFORMAT } \* MERGEFORMAT } \* MERGEFORMAT } \* MERGEFORMAT } \*
MERGEFORMAT } \* MERGEFORMAT } \* MERGEFORMAT } \* MERGEFORMAT } \*
MERGEFORMAT } \* MERGEFORMAT }
```

Space over and set the expire-year with a field in a similar manner. This time there are only 2 IF fields, comparing halves of the year. The printed value on true will be the SaveDate year value and the expanded result will look like:

```
{ IF { SAVEDATE \@ "M" \* MERGEFORMAT } < 7 { SAVEDATE \@ "YYYY" \*
MERGEFORMAT } { IF { SAVEDATE \@ "M" \* MERGEFORMAT } > 6 { = {
SAVEDATE \@ "YYYY" \* MERGEFORMAT } + 1 \*MERGEFORMAT } "" \*
MERGEFORMAT }
```

Revert the field codes to normal text by right click, Toggle Field Codes or Update Field. Select both of these fields by clicking on one, then shift click on the other. Copy, then paste in the footer (9 & 10 in Layout Styles above), replacing the Month, Year.

3.4 Automatic reference numbering

To support automatic updates of reference numbers, make the following changes. (Requires the document to be a single section prior to the Reference heading.)

1) Insert a section break on the line after Reference heading.

```
Insert
   Break
      Section Break
         Continuous
```

2) Format the style of the Endnote References and Text.

```
Format
   Style
      Endnote reference
      Modify
         Based on 'underlying paragraph'
         Format Font
         clear the check box for 'superscript'
      Endnote text
      Modify
         Based on 'RFC text'
         Format Paragraph
         Indentation
            Left    21.6
         Special
            Hanging 21.6
```

3) Set up the location of the references, and number style.

```
Insert
   Footnote
      Endnote
      Autonumber
      Options
         Place at 'End of section'
         Numeric style '1,2,3'
```

4) Select the location for the first reference. Between the user typed [ ] characters, insert an endnote.

```
Insert
   Footnote (endnote will already be selected,
       as will auto 1,2,3)
   OK
```

When the endnote is inserted, the lower pane will appear. Type in the text describing the reference. The first time a reference is inserted, the Endnote Separator should be cleared (the continuation separator may need it as well). Find the pull down, just above the reference text, and change it to each of the options to make sure all but the 'All Endnotes' are cleared.

```
Endnote Separator
   Select and delete any text
```

The reference number in the text and the endnote table will automatically track as changes are made. If the endnote window is closed and changes need to be made, select:

```
View
   Footnotes
```

To automatically add updated cross-references for previous footnotes, select the location of the cross-reference. Between the user typed [ ] characters insert a cross-reference.

```
Insert
   Cross-reference
      Select reference type 'endnote'
      Clear the checkbox for 'Insert as hyperlink'
      Select the reference from the endnote list
      Insert
```

4. Final fixup: the CRLF program

Each line needs to be terminated by a CRLF, but when printing your document to the Generic Text Printer driver, some blank lines will be terminated only with a line feed. Consider a traditional text line printer, printing a line of text, followed by 3 blank lines. The output would look as follows:

Line of Text<CR><LF><LF><LF>.

This was done because there was no need to move the print carriage head for the blank lines, only line feeds were necessary.

The following example provides the source for a CRLF fixup program.

```
/*************************************************************
 * CRLF.C - Sample source code to format documents produced by
 * the MS Word IETF template so that they comply to IETF draft
 * and RFC guidelines
 * Change CR/FF ; FF/CR/LF ; FF/LF ; CR/FF/CR/LF  into CR/LF/FF
 *************************************************************/

   #include <stdio.h>
   #include <io.h>
   #include <fcntl.h>
   #include <sys/types.h>
   #include <sys/stat.h>
   #include <memory.h>
   #include <string.h>
   #include <stdlib.h>
```

```
#define CR 13
#define LF 10
#define FF 12
#define TRUE 1
#define FALSE 0

typedef int BOOL;

int main(int argc, char *argv[])
{
     int fSrc, fDest;
     int iNumBytesRead;
     int iNumLines;
     char cr = CR;
     char lf = LF;
     char ff = FF;
     unsigned char buff[3];
     BOOL bPrecedingCR = FALSE;
     BOOL bPrecedingLF = FALSE;
     BOOL bPrecedingFF = FALSE;

     if(argc != 3)
     {
             printf("Usage:\n\n");
             printf("    crlf <srcfile> <dstfile>\n\n");
             return 0;
     }

fSrc = open(argv[1], O_RDONLY | O_BINARY);
fDest = open(argv[2], O_CREAT | O_RDWR | O_BINARY |
   O_TRUNC, S_IREAD | S_IWRITE);

if(fSrc == -1)
{
   printf("Could not open file (%s) for reading.\n",
            argv[1]);
         printf( strerror(errno));
             return 0;
     }

     if(fDest == -1)
     {
             printf("Count not open file (%s) for writing.\n",
            argv[2]);
         printf( strerror(errno));
             return 0;
     }
```

```
  // Using the MS Word with the generic text printer, an
  // extra CR LF starts the file.  Skip over these first 2
  // bytes,
  iNumBytesRead = _read(fSrc, buff, 2);

bPrecedingCR = FALSE;
bPrecedingLF = TRUE;
bPrecedingFF = FALSE;
iNumLines = 0;

  // Prepare to parse through the file
  iNumBytesRead = _read(fSrc, buff, 1);
  while(iNumBytesRead > 0)
  {
   if (buff[0] == FF)
   {
      // Found FF
      if (bPrecedingCR == TRUE)
      {
         // Some drivers write CR/FF w/o LF
         // Insert LF between
         _write(fDest, &lf, 1);
         _write(fDest, &(buff[0]), 1);
      }
      else if (bPrecedingLF == TRUE)
      {
         // If driver writes LF/FF, assume preceding CR
      }
      else if (bPrecedingFF == TRUE)
      {
         // If we just set FF from line count, ignore this
         // one
      }
      else if (bPrecedingLF == FALSE && bPrecedingCR == FALSE)
      {
         // Some drivers write FF alone ; insert CR/LF
         // for RFC rule of FF on line by itself
         _write(fDest, &cr, 1);
         _write(fDest, &lf, 1);
         _write(fDest, &(buff[0]), 1);
      }
      // reset flags
      bPrecedingFF = TRUE;
      bPrecedingCR = FALSE;
      bPrecedingLF = FALSE;
      iNumLines = 0;
   }
```

```
            else if (buff[0] == CR)
            {
               // Found CR
               if (bPrecedingFF == TRUE)
               {
                  // Some drivers write CR/FF/CR/LF
                  // ignore second CR/LF as it creates a 59th line
               }
               else
               {
                  // This CR counts
                  bPrecedingCR = TRUE;
                  bPrecedingLF = FALSE;
                  bPrecedingFF = FALSE;
                  if (++iNumLines < 59)
                  {
                     // Not end of page write it out
                     _write(fDest, &(buff[0]), 1);
                  }
                  else
                  {
                     // Some drivers write 66 lines per page as LF
                     // write end of page & skip to next CR in LF test
                     _write(fDest, &cr, 1);
                     _write(fDest, &lf, 1);
                     _write(fDest, &ff, 1);
                     bPrecedingFF = TRUE;
                     bPrecedingCR = FALSE;
                     bPrecedingLF = FALSE;
                     iNumLines = 0;
                  }
               }
            }
            else if (buff[0] == LF && bPrecedingFF == TRUE)
            {
               // Ignore up LF to next CR
            }
            else if (buff[0] == LF && bPrecedingCR == TRUE)
            {
               // Found a LF after a preceding CR
               // write it out and reset flags
               bPrecedingLF = TRUE;
               bPrecedingCR = FALSE;
               bPrecedingFF = FALSE;
               _write(fDest, &(buff[0]), 1);
            }
            else if (buff[0] == LF && bPrecedingCR == FALSE)
            {
```

```
            // Found a LF without a preceding CR
            if (bPrecedingLF == TRUE)
            {
               // Inject a CR to precede the LF only
               // if still in the first col
               if (++iNumLines < 59)
               {
                  // Not end of page write it out
                  _write(fDest, &cr, 1);
                  _write(fDest, &(buff[0]), 1);
                  bPrecedingLF = TRUE;
               }
               else
               {
                  // Driver writing longer than page
                  // write end of page & skip to next CR
                  _write(fDest, &cr, 1);
                  _write(fDest, &lf, 1);
                  _write(fDest, &ff, 1);
                  iNumLines = 0;
                  bPrecedingFF = TRUE;
                  bPrecedingCR = FALSE;
                  bPrecedingLF = FALSE;
               }
            }
            else
            {
               //ignore the random LF and clear flag
               bPrecedingLF = FALSE;
            }
         }
         else
         {
            // Other text, write it out and clear flags
            bPrecedingCR = FALSE;
            bPrecedingLF = FALSE;
            bPrecedingFF = FALSE;
            _write(fDest, &(buff[0]), 1);
         }
         // Read next byte
         iNumBytesRead = _read(fSrc, &buff[0], 1);
      }
      _close(fSrc);
      _close(fDest);

      return 0;
   }
```

5. Known problems

5.1 Margins

During the development of this document there were reports that some version combinations of Windows and Word cut off characters on the left. One approach to resolve this is to set the left and right margins to 36 and 57.6, thus shifting the text right. If text clipping was not a problem for the version combination, these values produce leading spaces. This doesn't affect the overall appearance, but makes the file larger than necessary, and violates the RFC line length rule. Adjustment of the margins for any specific version combination of Windows and Word will have to be locally appropriate; just make sure to move both in equal increments of 12 to the point where all characters appear.

5.2 Printing

If you try to print the draft you are working on from within Microsoft Word to an actual printer (not to a file using the Generic Text printer driver), you may receive an error message indicating the margins are outside of the printable area of the printer. If you continue printing, the first 2 characters of each heading will be truncated. It is recommended that you produce a printed copy of the draft you are working on by using the CRLF program to produce a text file, and then redirect it to a printer (so that you do not need to deal with other programs like NOTEPAD, etc. adding their own margins.) Example:

- Print to a file using the generic text printer
- CRLF draft.prn draft.txt
- NET USE lpt1 <\\printername\sharename>
- TYPE draft.txt > LPT1

As an alternative, if the final draft.txt file is opened with Word, setting all 4 margins to .65" will position it on the page.

File
   Page Setup
   Top .65
   Bottom .65
   Left .65
   Right .65

5.3 The Underscore character

If you use the underscore character "_" within the RFC Text and RFC Heading style, it will not be displayed on most screens. (It appears as a blank space.) It will print correctly and will appear as an underscore character in the final draft output.

6. Formal Syntax

The formal definition of RFC format is defined in RFC 2223 [2] and Internet Draft instructions are available at [3].

7. Security Considerations

Caution is advised when opening any document that may contain a macro virus. The template files originally provided to the Internet-drafts & RFC editors did not contain any macros, and unless tampered with, should not now. If there are concerns about using the template doc file, the instructions provided here will allow the creation of one from scratch. Further details about Microsoft Word macro virus concerns are available at: http://www.microsoft.com/. To find the current documents, search for 'macro virus'.

References

[1] http://www.greenwoodsoftware.com/less/

[2] Postel, J. and J. Reynolds, "Instructions to RFC Authors", RFC 2223, October 1997.

[3] http://www.ietf.org/ietf/1id-guidelines.txt

Acknowledgements

The authors would like to acknowledge the comments from around the community in helping refine this document. We would like to give particular recognition to DJ Son and Aaron Falk, of the RFC Editor staff, for aligning the details of this document with the current RFC Editor process.

Authors' Addresses

Mike Gahrns
Microsoft
One Microsoft Way
Redmond, Wa. USA

Phone: 1-425-936-9833
EMail: mikega@microsoft.com

Tony Hain
Cisco
500 108th Ave
Bellevue, Wa. USA

Phone: 1-425-468-1061
EMail: ahain@cisco.com

Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.