Manual for Tex2RTF 1.40: A LaTeX to RTF and HTML converter

Julian Smart
Artificial Intelligence Applications Institute
University of Edinburgh
EH1 1HN

April 1994

Artificial Intelligence Applications Institute
University of Edinburgh
80 South Bridge
EH1 1HN
Tel. 031-650-2746

Contents

Copyright notice

1. Introduction

This document describes a utility for converting LaTeX files into several other formats.

Only a subset of LaTeX can be processed by this utility, especially since the target document language will never perfectly match LaTeX. Whether the quality of the results is good enough will depend upon the application and your own expectations.

Tex2RTF is heavily biased towards making on-line, hypertext versions of LaTeX documents, but the *RTF* converter can be used for normal, linear documents too.

The latest version of Tex2RTF can be accessed by anonymous ftp from:

`skye.aiai.ed.ac.uk (192.41.104.6)`

in the directory /pub/tex2rtf. It is available in SPARC Open Look and Windows 3.1 versions.

Tex2RTF was developed using the free Open Look, Motif and Windows 3.1 C++ class library *wxWindows*, also available from the above FTP site in the /pub/wxwin/beta directory.

1.1 Status of Tex2RTF

Tex2RTF is under continual development, often following users' suggestions. From version 1.33, Tex2RTF is effectively in a second phase of development. In addition to the bare minimum of syntax and facilities for producing useable help systems or linear RTF, commands are being added to allow visually effective, even aesthetically pleasing, documentation to be produced.

Examples are the *indented*, *twocollist* and *marginpar* commands; over time I hope to be able to reproduce most of the popular styles of formatting and presentation in Windows Help files, whilst allowing a reasonable equivalent to be generated in the other formats.

1.2 Why use LaTeX?

LaTeX happens to be a very convenient format if you need to produce documents (such as manuals, help facilities, up-to-date information) in both printed and on-line media. Being a language rather than a WYSIWYG system, it allows explicit specification of layout and document structure, lending itself well to hypertext applications and automatic document generation. Many people also prefer to use LaTeX for ordinary use since it encourages a logical document structure and the user is not distracted by having to perfect the appearance; many layout decisions are taken by LaTeX automatically.

Although LaTeX is not as fancy as modern word processors and desk-top publishing packages, it is for many purposes quite adequate, and sometimes more flexible than its modern counterparts.

The conversion utility gives LaTeX a new lease of life by allowing virtually all other wordprocessor formats to be generated from documents containing a reasonable subset of LaTeX syntax. From the same LaTeX sources, we can now generate printed manuals, Windows Help files, *wxHelp* files, RTF-compatible word processor formats such as MS Word, and *HTML* files for use in the World Wide Web. Since the conversion tool is free, as are LaTeX, HTML viewers, wxHelp and (effectively) Windows Help, there are no financial or time penalties for providing documentation in a wide range of printed and hypertext formats.

1.3 Help versus the printed page

The purist may argue, quite rightly, that on-line help systems and printed manuals have different characteristics; help windows tend to be much smaller than pages, help topics should be more stand-alone than pages in a manual, navigation methods are very different, etc. Therefore, help systems should be *based* on printed documentation but separately hand-crafted into hypertext help, preferably by an independent person or team.

This might be the ideal, but many organisations or individuals simply do not have the time: on-line help wouldn't get done if the documentation effort had to be doubled. However, Tex2RTF does provide some commands to allow tailoring the docmentation to printed or on-line form, such as *helponly* and *helpignore*. An awareness of the design issues should go a long way to making the compromise a good one, so a book such as [1] is highly recommended.

1.4 Output Formats

At present the following output formats are supported:

- RTF (Rich Text Format). This is the most well developed converter. RTF is commonly used as a document exchange format amongst Windows-based applications, and is the input for the Windows Help Compiler. Tex2RTF supports both linear documents and Windows Help hypertext format.
- wxHelp. This is the platform-independent help system for the class library wxWindows [5]. It can display ASCII files with embedded codes for changing font styles, but no formatting is done by wxHelp.
- HTML (Hypertext Markup Language). This an SGML-like format commonly used by documents in the World Wide Web distributed hypertext system, and formats text dynamically rather like Windows Help.

1.5 What compromises must I make?

As a LaTeX user, you need to be aware that some macros or facilities don't transfer to other formats, either because they are not supported by the target format or because the converter does not support them. Maths formatting is a good example of an unsupported feature.

Sometimes LaTeX facilities must be accessed in a slightly different way to support the variety of formats, particularly hypertext formats where LaTeX references are often replaced by hypertext jumps (but must still look right in printed documentation). Tables don't transfer well to RTF (and not at all to the other formats) but an attempt is made to approximate tables so long as special row macros are used, instead of the usual end of row delimiter.

Bibliographies are handled quite well since the utilities can read in `.bib` files and resolve citations. Numbers are used in citations; the references are not yet sorted alphabetically.

Pictures are handled in a limited way: if the PSBOX macro package is used, an *image* macro can be used to place Encapsulated PostScript files in LaTeX, and Windows RGB-encoded bitmap files or placeable metafiles when converting to RTF.

Nested file inclusion (input, include, verbatiminput), is handled, and the comment environment is supported. However, using *input* to include macro packages is not advisable. If you do this, make sure you add a line in the Tex2RTF initialisation file to ignore this file, unless it's a simple LaTeX file that conforms to Tex2RTF restrictions. The file `psbox.tex` is the only file ignored by Tex2RTF by default.

Because of the way LaTeX is parsed, some syntax has to conform to a few simple rules. Macros such as *bf* and *it* need to occur immediately after a left brace, and have a block of their own, since the text within their scope is regarded as its argument. This syntax means the same thing as using *begin ... end*, which is usually a one argument macro (the argument is the text between the *begin* and *end*). See **Space** (section 1.6.1) .

As a Windows hypertext help writer, you don't have access to all RTF commands but you'll be able to get most of what you want. In particular, any LaTeX document you write will automatically be a hypertext document, because the converter takes advantage of the hierarchy of sections. Further jumps can be placed using the commands **label**, **helpref**, **helprefn**, and **popref**. Tex2RTF outputs help files that may be read linearly using the << and >> buttons, and an additonal Up button for ease of navigation.

When writing HTML, multiple files are generated from one LaTeX file since browsing HTML works best with many small files rather than a few large ones.

wxHelp files are least well supported since there is no formatting support, only font style, sizes and colours. Still, some hypertext help support on UNIX/X platforms is better than none. The class library wxWindows may be extended in future to allow using a better help viewer, such as *xmosaic*. Of course there is nothing to stop xmosaic being used as a help system, but it won't be integrated with wxWindows programs as wxHelp is.

Sometimes you will use a local macro package that is unrecognised by the converters. In this case, you may define a custom macro file where macros are defined in terms of supported LaTeX commands and text. Even if the result is not the same as in LaTeX, you can probably end up with something adequate, and at least avoid undefined macro errors. See **Initialisation file syntax** (section 4.2) for further information.

1.6 Changes to LaTeX syntax

Here are the conventions you need to observe to satisfy the Tex2RTF parser.

Some of the syntax that is OK for true LaTeX but which trips up Tex2RTF, may be detected by the TCHECK program included in the tools directory of the Tex2RTF distribution.

1.6.1 Space

Tex2RTF attempts to insert spaces where LaTeX assumes whitespace. However, for the benefit of RTF conversion, you need to use the *rtfsp* macro where a command or brace within a paragraph begins or ends with a macro. For example:

```
Within a paragraph, you need to be careful about commands that begin
\rtfsp {\it at the start} of a line.
```

As normal with LaTeX, two newlines represents a paragraph break, although **par** can also be used at the end of a paragraph.

You need to have a blank line between section and some environment commands and the first paragraph or your document will look rather weird, e.g. headings running into paragraphs.

wxHelp is more fussy than LaTeX or RTF: you need to use percent characters at line ends liberally to eliminate newlines after commands on single lines.

1.6.2 Command arguments

Commands that have one or more arguments can be used in the following three ways:

```
\bf{Some text.}

\begin{bf}
Some text.
\end{bf}

{\bf Some text.}
```

The first method is a normal LaTeX command.

The second method is called an *environment*; LaTeX has specific environments that do not always correspond to normal commands, but Tex2RTF recognizes environments and normal commands interchangeably, so long as the command has no more than two arguments.

With the third method, it is important that the command has its own pair of braces, and that the command immediately follows the first brace. Otherwise, the parser cannot parse the argument(s) properly. With multiple arguments, each should be enclosed in braces.

Optional arguments are specified using square brackets or parentheses.

The braces that start command arguments must not be seperated from the other arguments by whitespace. For example, the following produces an error:

```
\image{5cm;0cm}
{picture.eps}
```

and should be replaced by

```
\image{5cm;0cm}{picture.eps}
```

1.6.3 Avoid the setlength macro

Using the \setlength command doesn't work, since its first argument looks like a macro with the wrong number of arguments. Use an alternative form instead, e.g.

```
\parindent 0pt
```

instead of

```
\setlength{\parindent}{0pt}
```

1.6.4 Units

Only a subset of LaTeX units may be used for specifying dimensions. Valid units are *pt, mm, cm* and *in*. Units should usually be specified for dimensions or the results may be unexpected.

1.6.5 Labels

The *label* command may be used for sections and figure captions, but must come immediately after the section or caption commands with no intervening whitespace.

1.6.6 Tables

For best layout, table rows should be enclosed in a *row* or *ruledrow* command, since Tex2RTF can't cope with parsing the LaTeX tabular syntax unaided. However, if you really don't want to go through LaTeX files inserting new syntax, set the **compatibility** flag to TRUE in your tex2rtf.ini file. In this mode, Tex2RTF tries to make the best of a bad job, but the results won't be optimal (e.g., no table borders). Without this flag set, normal LaTeX tables can crash RTF readers such as Word for Windows.

1.7 Tex2RTF change log

Changes in version 1.40:

- Added **generateHPJ** option for generating the .HPJ WinHelp project file
- Added support for DDE via a small command set

Changes in version 1.39:

- Option for using Word's INCLUDEPICTURE or IMPORT field, since the method that works for Works, doesn't work for Word! See **bitmapMethod** in the settings section.

Changes in version 1.37-1.38:

- Improved bibliography reading and cured some minor bugs
- Added `\ss` German sharp s
- Added rudimentary `\special` command (simply copies the argument to the output)
- Added missing '.' in subsubsection reference
- Added primitive internationalisation support with contentsName, tablesName etc.

Changes in version 1.36:

- All HTML special characters now correctly delimited by a semicolon.
- Cured HTML section-duplicating bug I introduced in 1.35.
- Cured too much spacing after sections in RTF, introduced in 1.35.

Changes in version 1.35:

- Added TCHECK tool, to help track down common Tex2RTF syntax problems.
- Included Kresten Thorup's LACHECK LaTeX checking tool with DOS executable.
- Now ignores `\@` command.
- Table of contents now includes numbered subsubsections.

Changes in version 1.34:

- Added *multicolumn* 'support' to stop RTF readers crashing.
- Added *useWord, defaultColumnWidth, compatibility* options to .ini file.
- *comment* environment now doesn't complain about unknown syntax.
- Added *toocomplex* environment that treats its contents as verbatim in output, treated as normal output in true LaTeX.
- End-of-line comments allowed in in .ini files, using semicolon, percent or hash characters to denote a comment.

- For linear RTF, Word for Windows support for *printindex*, *index*, *pageref*, *listoftables*, *listoffigures*, contents page.
- Added RTF support for various symbols.
- Added colour support, with *definecolour, fcol* and *bcol* commands.
- Fixed some bugs: page numbering problems, macros deleted after first pass.

Changes in version 1.33:

- Added -charset command-line switch.
- Added *itemsep, twocolumn, onecolumn, setfooter, setheader, pagestyle, pagenumbering, thechapter, thesection, thepage, thebibliography, bibitem* commands.
- New environment called *twocollist* for making two-column lists, with formatting optimized for target file format.
- New *indented* environment for controlling indentation.
- List indentation and bulleting improved.
- Added commands *normalbox, normalboxd* for putting borders around text.
- Many options can now be specified in the .ini file along with custom macros.
- Cured bug that put too much vertical space after some commands.
- Improved table formatting.
- Optional ‘Up’ button in WinHelp files for easier navigation.
- Verbatim lines followed by *par* in RTF, to improve WinHelp wrapping.
- Conversion may now be aborted under Windows by attempting to close the application.
- Added conditional output for all formats: *latexignore, latexonly, rtfignore, rtfonly, winhelpignore, winhelponly, htmlignore, htmlonly, xlpignore, xlponly.*
- HTML generator can now add Contents, Up, ¡¡ and ¿¿ buttons (text or bitmap) to each page except titlepage.

Changes in version 1.32:

- *footnote* command now supported in WinHelp RTF, and *footnotepopup* added.

Changes in version 1.31:

- *footnote* command now supported, in linear RTF only.
- Added -bufsize option, for converting large documents.

Changes in version 1.30:

- *image* command now scales metafiles (but not bitmaps).

- Fixed macro loading bug, now informs the user of the found macro filename.
- Now supports paragraph and subparagraph commands.
- Support for some accents added.
- *verb* command now supported.
- Bug in subsubsection handling fixed.
- Can save conversion log in a text file.

Changes in version 1.22:

- More informative, warns against use of some commands.
- Added compile-time support for non-GUI environments (such as plain UNIX).
- Improved HTML support.

2. Hypertext features

LaTeX is inherently suitable for specifying hypertext documents since it encourages description of the logical structure of a document using section commands. Therefore, a LaTeX document is automatically a hypertext document, without any further editing.

For Windows Help, a single RTF file is generated with topics corresponding to sections. A top level contents page shows each chapter or top-level section, and each chapter or section ends with a list of further sections or subsections. Tex2RTF outputs help files that may be read linearly using the << and >> buttons.

Similarly, a single wxHelp XLP file is generated.

For HTML, a different file is generated for each section, since the XMOSAIC browser works best with a large number of small files. The files are named automatically based on the name of the output file, with the contents page filename being formed from the output filename with `_contents` appended to the name. This may result in the generation of several hundred files for a large LaTeX input file.

To specify explicit jumps around a hypertext file, the **helpref** macro is used. The first argument is the text to be displayed at the point of reference, which will be highlighted in a hypertext file to allow jumping to a reference. The second argument is the reference label (there should be a corresponding **label** command in the file, following a section or figure).

To use extra Tex2RTF features in proper LaTeX, such as **helpref** and the C++ and CLIPS class reference documentation features, include the style file `texhelp.sty`.

3. Special sections

The treatment of bibliography, glossary and index are worth special mention.

3.1 Bibliography

Tex2RTF recognises standard LaTeX bibliography files (usually with `.bib` extension) and resolves citations. The **bibliography** command reads the given `.bib` file and includes a list of references at that point in the input. Only numbered, unsorted references are catered for at the moment, with no variation in bibliography style. A **References** heading is placed in the contents section. Note that Tex2RTF must be run twice to ensure the citations are resolved properly.

Tex2RTF can also cope with the *thebibliography* environment, with *bibitem* commands, so long as the text following the first *bibitem* argument is enclosed in braces as if it were a second argument.

3.2 Glossary

Glossaries are formatted according to the following scheme. The **helpglossary** environment is used together with the **gloss** command for glossary entries. In LaTeX this is interpreted as a description list, and each glossary entry is an item. In on-line help, each glossary entry is a section.

A labelled glossary entry command may be referenced by **popref** to provide a quick popup explanation of a term.

3.3 Index

The explicit index is assumed to be redundant in on-line help, since search facilities are provided. Therefore the *printindex* command does nothing in on-line versions. In linear RTF an index field is added, and **index** marks words for inserting in the index.

In Windows Help, all section headings and C++ function names are treated as keywords. A keyword may be ambiguous, that is, refer to more than one section in the help file. This automatic indexing may not always be adequate, so the LaTeX **index** command may be used to add keywords.

In wxHelp, all section headings are indexed.

4. Running Tex2RTF

Tex2RTF may be run in a number of ways: with or without command line arguments, interactively or in batch mode, and with an optional initialisation file for specifying LaTeX macros and detailed options.

Tex2RTF accepts two arguments (input and output filenames) and trailing (optional) switches. If both filenames are given, the utility will work in batch mode. Otherwise, if Tex2RTF has been compiled for GUI operation, a main window will be shown, with appropriate menu items for selecting input and output filenames, starting off the conversion process, and so on.

Note that if the file `bullet.bmp` is found by Tex2RTF, this bitmap will be used as the bullet for items in *itemize* lists, for WinHelp output. Otherwise, a symbol will be inserted (linear RTF) or bold 'o' will be used instead (all other formats).

Syntax error reporting is fairly minimal. Unrecognised macro errors may actually be produced by an unbalanced brace or passing the wrong number of arguments to a macro, so look in the vicinity of the error for the real cause.

> Some of the syntax that is OK for true LaTeX but which trips up Tex2RTF, may be detected by the TCHECK program included in the tools directory of the Tex2RTF distribution. Some LaTeX errors may be picked up by the LACHECK program, also found in the tools directory.

It is recommended that you run Tex2RTF twice in order to be sure of of resolving all references and including an up-to-date contents page.

If importing RTF files into Word for Windows, you may need to reformat the document. The easiest way to do this is to select all text with CTRL-A, then reformat with F9. Reformat again to ensure all references are resolved. For the second format, respond with *Update Entire Table* to prompts.

4.1 Command line arguments

These are the optional arguments you may give Tex2RTF on the command line.

-bufsize Specifies buffer size in K (default 60 under Windows, 500 under UNIX). Large files (particularly large verbatim environments) may require a large buffer size, equal to the largest argument of a LaTeX command. Note that this value may not be larger than 64 under Windows.

-html Specifies HTML (World Wide Web) output.

-interactive Forces interactive mode even if both filenames are given.

-charset charset Specifies a character set for RTF production. This can be one of ansi, mac, pc, and pca. The default is ansi.

-macros filename Specifies a file for the custom macro file – see **Macro not found error** (section 7.2.1) .

-rtf Specifies linear RTF output.

-sync Forces synchronous mode (no yielding to other processes) – usually use this in non-interactive mode.

-twice Tells Tex2RTF to run the conversion twice to ensure all references and citations are resolved and the contents page included.

-winhelp Specifies Windows Help RTF output.

4.2 Initialisation file syntax

The initialisation file contains further detailed options for customising Tex2RTF's behaviour. A file may be specified with the **-macros** command line switch, otherwise Tex2RTF looks for the file `tex2rtf.ini` in the working directory or input file directory.

The file may comprise macro definitions or option settings.

The syntax for a macro definition is:

```
\name [number of args] {...LaTeX code...}
```

For example:

```
\crazy      [2]{{\bf #2} is crazy but #1 is not}
\something  [0]{}
\julian     [0]{Julian Smart}
```

The syntax for an option setting is:

```
name = value
```

or

```
name = "value"
```

For example:

```
conversionMode = RTF
runTwice = true
titleFontSize = 12
authorFontSize = 10
headerRule = yes
footerRule = yes
```

Options expecting boolean values accept *1, 0, true, false, yes, no* in any combination of upper or lower case.

End-of-line comments are allowed in an initialisation file, using the hash, semicolon or percent signs to denote the start of a comment, which runs until the end of the line.

4.2.1 Tex2RTF options

These are the allowable options in an initialisation file.

General options

Option	Description
compatibility	Set to true for maximum LaTeX compatibility, e.g. if tables crash RTF readers. Should be false (default) if the Tex2RTF guidelines are followed, e.g. use of *row* command in tabular environment.
conversionMode	One of RTF, WinHelp, XLP (or wxHelp), and HTML.
ignoreInput	Adds the filename to the list of files ignored by the *input* command. The only default filename in the list is `psbox.tex`.
isInteractive	If true, runs in interactive mode (the default).
runTwice	If true, runs the converter twice.

PRESENTATION OPTIONS

Option	Description
authorFontSize	Specifies the point size for the author and date (RTF only).
chapterFontSize	Specifies the point size for chapter headings (RTF only).
documentFontSize	One of 10, 11 and 12, to specify the main font size independently of the LaTeX document style command.
sectionFontSize	Specifies the point size for section headings (RTF only).
subsectionFontSize	Specifies the point size for subsection headings (RTF only).
titleFontSize	Specifies the point size for the title (RTF only).
chapterName	The string used when referencing chapters. The default is “chapter”.
sectionName	The string used when referencing sections. The default is “section”.
subsectionName	The string used when referencing subsections. The default is “subsection”.
subsubsectionName	The string used when referencing subsubsections. The default is “subsubsection”.
indexName	The string used for printing the index heading. The default is “Index”.
contentsName	The string used for printing the contents heading. The default is “Contents”.
tablesName	The string used for printing the list of tables heading. The default is “List of Tables”.
tableName	The string used when referencing a table. The default is “table”.
figuresName	The string used for printing the list of figures heading. The default is “List of Figures”.
figureName	The string used when referencing a figure. The default is “figure”.
glossaryName	The string used for printing the glossary heading. The default is “Glossary”.
referencesName	The string used for printing the references heading. The default is “References”.

RTF/WinHelp options

Option	Description
defaultColumnWidth	The width in points for columns in tables where the width of the column is not set by using *p* in the tabular argument. The default is 100.
footerRule	If true, draws a rule above footers (linear RTF only).
generateHPJ	If true, generates a .HPJ project file (WinHelp mode only).
headerRule	If true, draws a rule below headers (linear RTF only).
listLabelIndent	Specifies the size of list item label indentation, in points. The default is 18.
listItemIndent	Specifies the size of list item indentation, in points. The default is 40.
indexSubsections	If true (the default), subsection and subsubsection titles are indexed in RTF mode.
mirrorMargins	If true, margins are mirrored in twosided documents (linear RTF only).
useWord	If true (the default), Word for Windows RTF formatting is used where possibly, e.g. for the table of contents, list of tables, and list of figures.
useHeadingStyles	If true (the default), sections are marked with appropriate heading styles for generating the table of contents in RTF.
useUpButton	If true (the default), WinHelp files will be generated with an **Up** button to make browsing easier. Note that you need to put an extra line in the CONFIG section of your .HPJ file: `CreateButton("Up", "&Up", "JumpId('name.hlp', 'Contents')")` where `name.hlp` is the name of your help file.
winHelpTitle	Windows Help file title, inserted into the project file if **generateHPJ** is true.

HTML options

Option	Description
htmlBrowseButtons	Allows generation of Contents, Up, browse back and browse forward buttons on each HTML page except title page. Specify none, text or bitmap. If you specify bitmap, make sure that the files contents.gif, up.gif, back.gif and forward.gif are in the directory where the HTML files will reside: samples are given in the docs directory.

4.3 DDE commands

A Windows program can hold a conversation with Tex2RTF using DDE. The Tex2RTF server name is "TEX2RTF", and the topic name to use is also "TEX2RTF".

Tex2RTF functionality is accessed using the DDE **Execute** message. The **Execute** data should consist of a command name and possibly one argument, e.g.
`INPUT c:\docs\mine.tex`

If the command is not recognised, a standard TEX2RTF.INI option is assumed.

The **Request** DDE message can be used to query the return status of an **Execute** command, and will be one of **OK** (no error), **CONVERSION ERROR**, or a more specific error string.

The following DDE commands may be used:

Command	Description
EXIT	Takes no argument, and exits Tex2RTF.
GO	Takes no argument, and initiates the conversion.
INPUT	Takes a file name as the argument, and sets the input file to be this name.
MINIMIZE	Takes no argument, and minimizes Tex2RTF.
OUTPUT	Takes a file name as the argument, and sets the input file to be this name.
RESTORE	The same as SHOW.
SHOW	Takes no argument, and unminimizes Tex2RTF.

5. Tex2RTF for non-LaTeX users

You don't need to have LaTeX installed to use Tex2RTF. You can still output RTF files to be imported into your favourite word processor, and hypertext files for on-line help.

This chapter gives a very brief introduction to LaTeX. For further information, Kopka and Daly's *A Guide to* LaTeX [2] is recommended.

5.1 What is LaTeX?

LaTeX is a macro package built on top of the typesetting package, TeX. TeX was written by Donald Knuth in the 1970s, and Leslie Lamport wrote LaTeX as a higher-level, easier way to write TeX.

TeX was quite advanced for its day, and is still used (particularly by academics) because of its free availability and its flexibility in typesetting maths and other symbols. It's more like a programming language than a word processor, with embedded commands prefixed by a backslash and block structure. Like programs, TeX documents are processed by a 'compiler', outputting a .dvi file, which is a device independent file which can be read by many converters for output onto physical devices, such as screens and printers.

A reason for its longevity is the ability to add facilities to TeX, using macro packages that define new commands.

LaTeX is the most popular way to write TeX. Although WYSIWYG word processors and DTP packages are outstripping LaTeX, the increasing interest in hypertext and mark-up languages makes LaTeX relevant as a similar language to SGML documents (such as World Wide Web HTML files).

Also, languages such as LaTeX (and Rich Text Format, which it resembles in many ways) are *complementary* to WYSIWYG packages. These languages allow automatic production and translation of documents, where manual mark-up is impractical or undesirable.

Since the source code of TeX and LaTeX is in the public domain, there are many free and commercial implementations of LaTeX for almost every computer in existance. Of PC implementations, EmTeX is arguably the best and most complete. You can download it from various FTP sites.

If you don't want to use LaTeX itself, you may wish to use a program called lacheck to check your documents before using Tex2RTF, since it catches some mistakes that Tex2RTF doesn't.

5.2 Document structure

Here is a sample of a typical LaTeX document:

```
\documentstyle[a4,texhelp]{report}
\title{A title}
\author{Julian Smart}
\date{October 1993}
\begin{document}
\maketitle

\chapter{Introduction}

...

\section{A section}

...

\end{document}
```

The first line is always a *documentstyle* command. The square brackets enclose optional *style* files (suffix .sty) that alter the appearance of the document or provide new commands, and the curly brackets enclose the mandatory style, in this case 'report'.

Before the document begins properly with *\begin{document}*, you can write various commands that have an effect on the appearance of the document or define title page information. The *maketitle* command writes the title page using information defined previously (title, author, date).

A report has chapters, which are divided into sections, and can be further divided into subsections and subsubsections. To start a new section, you write the appropriate section command with the section heading; there is no specific end section command, since a new section heading or the end of the document will indicate the end of the previous section.

An article is divided into sections, subsections and subsubsections, but has no chapters. This is so an article can be included in a report as a chapter.

Tex2RTF is written to deal with reports best, so stick with the report style if you can.

5.3 Command syntax

There are several kinds of commands in LaTeX. Most involve a keyword prefixed with a backslash. Here are some examples:

```
\titlepage

\centerline{This is a centred line}

\begin{center}
This is a centred
paragraph
```

```
\end{center}

{\bf This is bold font}
```

The first example has no arguments. The second has one argument. The third example is an *environment* which uses the begin and end keywords instead of a pair of braces to enclose an argument (usually one). The fourth is an example of using a command within a pair of braces: the command applies to the scope within the braces. Tex2RTF treats this form as if it were a command with one argument, with the right brace delimiting the argument. In this case, the command must immediately follow a left brace as shown.

Commands may be nested, but not overlapped.

5.4 Space

In LaTeX, white space is mostly ignored, line breaks make no difference. However, LaTeX interprets two successive newlines (a blank line) as denoting a paragraph break. You may also use the *par* command to end a paragraph.

To be continued!

6. Macro reference

The following lists macros which are recognised by the converters. Other macros not not mentioned can be assumed to be unrecognised or ignored.

Each command is listed with its name, the number of arguments it takes (excluding optional arguments), and a description. Note that if the command is used as an environment (using *begin* and *end*) then the number of arguments must be either one or two. For example, the *tabular* environment takes two arguments: a first argument for specifying the formatting, and the second argument for the body of the environment.

```
\begin{tabular}{|l|l|}
\row{One}{Two}
\row{Three}{Four}
\end{tabular}
```

6.1 Commands

abstract:1

This standard LaTeX environment prepares an abstract page, and is treated as an ordinary chapter or section in on-line help.

addcontentsline:3

Adds a chapter title to the contents page. Linear RTF. Rarely required.

author:1

Defines the author, for output when *maketitle* is used.

backslash:0

Outputs a backslash in math mode (should be enclosed by two dollar symbols).

bcol:2

Sets the background colour for a block of text (RTF only). Has no known effect in the RTF readers currently tried (Word for Window and Windows Help).

See also **definecolour**, **fcol**.

bf:1

Specifies bold font.

bibitem:2

For parsing convenience, *bibitem* requires two arguments: a cite key and item. LaTeX syntax permits writing this as if it were two arguments, even though it is in fact only one. This command is used within a **thebibliography** environment. The preferred method is to store references in .bib files and use the **bibliography** command to generate a bibliography section automatically.

bibliographystyle:1

Currently doesn't affect the style of bibliography, but probably will in the future.

bibliography:0

Includes the bibliography at this point in the document. See the section on **bibliographies** (section 3.1) .

caption:1

Specifies a caption (within a **figure** environment). This may be followed immediately by a **label** command.

cdots:0

Outputs three dots.

centerline:1

Centres (or centers!) a line of text.

center:1

Centres a block of text.

cextract:0

Prints a C++ extraction operator (>>).

chapter*:1

Outputs a chapter heading with no contents entry.

chapter:1

Outputs a chapter heading.

chapterheading:1

Like **chapter**, but does not increment the chapter number and does not print a chapter number in the printed documentation contents page, or in the chapter heading. Used to implement **glossaries** (section 3.2) and other sections that are not real chapters.

cinsert:0

Prints a C++ insertion operator (<<).

cite:1

Cite a reference. The argument is a reference key as defined in a LaTeX .bib file.

class:1

Outputs the argument, an index entry (LaTeX only) and a keyword entry (WinHelp only). Used in class reference documentation.

clipsfunc:3

Formats a CLIPS function, given the return value, function name, and arguments.

comment:1

An environment that allows large comments in LaTeX files: the argument is ignored in all formats. Useful for commenting out parts of files that cannot be handled by LaTeX, such as the picture environment. See also **toocomplex**.

copyright:0

Outputs the copyright symbol.

cparam:2

Formats a CLIPS type and argument. Used within the third argument of a **clipsfunc** command.

date:1

Specifies the date of a document; only output by **maketitle**.

definecolour:4

Defines a new colour that can be used in the document (RTF only). This command can also be spelt *definecolor*.

The first argument is the lower-case name of the colour, and the following three arguments specify the red, green and blue intensities, in the range 0 to 255.

The default colours are equivalent to the following definitions:

```
\definecolour{black}{0}{0}{0}
\definecolour{cyan}{0}{255}{255}
\definecolour{green}{0}{255}{0}
\definecolour{magenta}{255}{0}{255}
\definecolour{red}{255}{0}{0}
\definecolour{yellow}{255}{255}{0}
\definecolour{white}{255}{255}{255}
```

To use colours in a document, use the **fcol** and **bcol** commands.

Note that a document that defines its own colours should be converted twice within the same Tex2RTF session.

description:1

A list environment, where each **item** command must be followed by optional square-bracketed text which will be highlighted.

documentstyle:1

Specifies the main style (report, article etc.) and, optionally, style files such as `texhelp.sty`. A report has **chapters**, while an article's top-level sections are specified using **section**.

document:1

This environment should enclose the body of a document.

em:1

Emphasizes text (italic in RTF).

enumerate:1

Enumerate list environment: numbers the **items**.

fcol:2

Sets the foreground colour for a block of text (RTF only).

For example:

```
This sentence is brightened up by some \fcol{red}{red text}.
```

gives:

This sentence is brightened up by some .

See also **definecolour**, **bcol**.

figure:1

A figure enviroment: does nothin in RTF.

flushleft:1

Flushes the given text to the left margin.

flushright:1

Flushes the given text to the right margin.

footnote:1

In linear RTF, a footnote is created. Whether this appears at the end of the section or the bottom of the page appears to depend on the current document style, at least for MS Word 6.0 for Windows. The default seems to be to put the footnotes at the end of the section, which is probably not the best assumption.

In WinHelp RTF, a bracketed number is generated for the footnote and the footnote becomes a popup topic. It is probably preferable to change footnote commands to **footnotepopup** (section 6.1) , or **popref** (section 6.1) references to glossary entries.

This command is not supported for formats other than LaTeX, linear RTF and WinHelp RTF.

footnotepopup:2

In linear RTF, a footnote is created following the first argument, as with **footnote** (section 6.1) .

In WinHelp RTF, a the first argument is highlighted and becomes a popup reference to the second argument. See also **footnote** (section 6.1) and **popref** (section 6.1) .

This command is not supported for formats other than LaTeX, linear RTF and WinHelp RTF.

functionsection:1

Defines a subsection, adding the C++ function name to the LaTeX index or the WinHelp keyword list.

Should be followed by a **func** command to specify function details.

func:3

Defines a C++ function, given the return type, function name, and parameter list.

Should occur after a **functionsection** command.

gloss:1

Marks a glossary entry. In LaTeX, this is a synonym for an **item** with an optional argument, within a **description** environment, and the argument is added to the index.

In Windows Help, this is identical to a **section*** in a report.

If labels are associated with the glossary entries, they can be referenced by **helpref** (section 6.1) or **popref** (section 6.1) jumps. A glossary entry is currently the only type of destination that popref may refer to.

This is an example of making a glossary in a report:

```
\begin{helpglossary}

\gloss{API}\label{api}

Application Programmer's Interface - a set of calls and classes defining
how a library (in this case, wxWindows) can be used.

\gloss{Canvas}\label{canvas}

A canvas in XView and wxWindows is a subwindow...

\gloss{DDE}\label{dde}

Dynamic Data Exchange - Microsoft's interprocess communication protocol.
wxWindows provides an abstraction of DDE under both Windows and UNIX.

\end{helpglossary}
```

helpglossary:1

An environment for making a glossary (not standard LaTeX). See **gloss** for usage.

helpignore:1

Ignores the argument in Tex2RTF generated files, but not LaTeX.

helponly:1

Only outputs the argument in Tex2RTF generated files.

helpinput:1

Only includes the given file in Tex2RTF generated files.

helpfontfamily:1

Specifies the font family for Tex2RTF generated files. The argument may be Swiss or Times.

helpfontsize:1

Specifies the font size for Tex2RTF generated files.

helpref:2

Specifies a jump to a labelled chapter, section, subsection subsubsection or figure.

The first argument is text to be highlighted (mouseable in help systems) and the second is the reference label. In linear documents, the section number is given following the text, unless the **helprefn** command is used instead, where the section number is suppressed.

Note that when generating HTML, the label *contents* is automatically defined, and may be referenced using *helpref*.

helprefn:2

Specifies a jump to a labelled chapter, section, subsection subsubsection or figure.

The first argument is text to be highlighted (mouseable in help systems) and the second is the reference label. See **helpref** for the form where the section number is printed in linear documents.

hline:0

Within a **tabular** environment, draws a horizontal rule below the current row. Note that this does not work in RTF for the last row of a table, in which case the command **ruledrow** should be used instead.

hrule:0

Draws a horizontal line below the current paragraph. For example:

```
This paragraph should have a horizontal rule following it.\hrule
```

gives:

This paragraph should have a horizontal rule following it.

---

htmlignore:1

Ignores the argument in HTML.

htmlonly:1

Only outputs the argument in HTML.

huge:1

Outputs the argument in huge text.

Huge:1

Outputs the argument in huger text that **huge**.

HUGE:1

Outputs the argument in huger text that **Huge**.

include:1

Include the given file. The command must not be preceded by any whitespace, and spurious whitespace between elements of the command will also trip up Tex2RTF.

indented:2

Environment supplied by Tex2RTF to allow (possibly nested) indentation of LaTeX and RTF text. The first argument is the amount to be indented.

For example:

```
\begin{indented}{2cm}
This text should be indented by a couple of centimetres. This can be
useful to highlight paragraphs.
\end{indented}
```

produces:

> This text should be indented by a couple of centimetres. This can be useful to highlight paragraphs.

input:1

Include the given file. The command must not be preceded by any whitespace, and spurious whitespace between elements of the command will also trip up Tex2RTF.

itemize:1

Indents each **item** of a list and precedes with a bullet. If the file `bullet.bmp` is found by Tex2RTF, this bitmap will be used as the bullet (WinHelp RTF); otherwise, a symbol or bold 'o' will be used instead, depending on output format.

Use **itemsep** to specify the separation between list items. Currently this only works for linear or WinHelp RTF output. If the value is more than zero, an extra paragraph is inserted.

item:0

Marks an item of a **itemize**, **description** or **enumerate** list. Items within a description environment should have an 'optional' argument in square brackets which will be highlighted.

itemsep:0

Use this command to specify the separation between list items. Currently this only works for linear or WinHelp RTF output. If the value is zero, no extra paragraph is inserted; if the value is more than zero, an extra paragraph is inserted.

image:2

This is translated to a PSBOX macro package *psboxto* command in LaTeX, the first argument being a sizing command and the second a filename.

In HTML mode, the second argument is used to generate a PostScript file reference.

In RTF mode, the second argument is tried with first a BMP extension and then a WMF extension to find a suitable Windows bitmap file or placeable metafile. If a suitable file is found, in Windows Help mode a *bmc* command is inserted into the RTF file with a reference to the file. In linear RTF mode, the bitmap or metafile is converted into hex and inserted into the RTF document.

Note that only RGB-encoded Windows bitmaps, or placeable metafiles, are valid for input to Tex2RTF. You can convert a RLE (run length encoded) bitmap file into a (bigger) RGB file using a program such as Paintshop Pro. A placeable metafile has a special header with dimension information. One may be constructed by a wxWindows program by calling the function wxMakeMetafilePlaceable. The Microsoft Windows SDK has a sample program that loads and steps through placeable and ordinary metafiles.

Another wrinkle is that programs differ in the methods they use to recognise pictures in RTF files. You may need to use the *bitmapMethod* setting, which can be "hex" (embed the hex data in the file with a `\dibitmap` keyword), "includepicture" (use the MS Word 6.0 INCLUDEPICTURE field) or "import" (an earlier name for INCLUDEPICTURE).

Here is an example of using the *image* command.

```
\begin{figure}
$$\image{5cm;0cm}{heart.ps}$$

\caption{My picture}\label{piccy}
\end{figure}
```

The dollars centre the image in the horizontal plane. The syntax of the first argument to *image* is taken from syntax used by the *psbox* package: it allows specification of the horizontal and vertical dimensions of the image. Scaling will take place for PostScript and metafile images. A value of zero indicates that the image should be scaled in proportion to the non-zero dimension. Zeros for both dimensions will leave the image unscaled in the case of metafiles, or scaled to fit the page in the case of PostScript.

index:1

In WinHelp mode, adds a keyword to the keyword list for the current topic. This keyword must currently be straight text, with no embedded commands. The conversion process must be run twice (without quitting Tex2RTF inbetween) to resolve the keyword references.

it:1

Marks the argument in italic.

label:1

Labels the chapter, section, subsection, subsubsection or figure caption with the given label. This must be an ASCII string, and duplicate items with different case letters are not allowed.

The command must follow immediately after the section or caption command, with no intervening whitespace.

large:1

Marks the argument in large text.

Large:1

Makes the argument display in larger text than **large**.

LARGE:1

Makes the argument display in larger text than **Large**.

LaTeX:0

Outputs the annoying $\LaTeX$ upper and lower case name.

latexignore:1

Ignores the argument in $\LaTeX$.

latexonly:1

Only prints the argument in $\LaTeX$.

ldots

Outputs three dots.

maketitle

Makes the article or report title by outputting the **title**, **author** and optionally **date**.

marginparwidth:1

Specifies the width of a margin paragraph.

marginpar:1

Inserts a marginal note. It is best to use the Tex2RTF extensions **marginparodd** and **marginpareven** for best results.

marginpareven:1

Inserts a marginal note on even pages. This is required for RTF generation since it is impossible for Tex2RTF to know in advance which side of paper the marginal note will fall upon, and the text has to be positioned using absolute dimensions. If only one sided output is required, use **marginparodd** instead.

marginparodd:1

Inserts a marginal note on odd pages. This is required for RTF generation since it is impossible for Tex2RTF to know in advance which side of paper the marginal note will fall upon, and the text has to be positioned using absolute dimensions.

Also, even if one-sided output is required, this command should be used instead of *marginpar* because the LaTeX macro allows it to be used just before a paragraph. Normally, if this were done, the marginal note would not be aligned with the paragraph succeeding it. For example:

```
\marginparodd{{\it Note:} if nothing happens, perhaps you have not plugged
your computer in at the mains.}%
To start using your WhizzyGig Computer 4001, push the Power button and
wait for some kind of response.
```

Note the percent sign after the *marginparodd* command: without it, LaTeX refuses to believe that the following text is part of the same paragraph, and will print the note at the wrong place.

You should use **textwidth** to allow space for marginal notes, and also **marginparwidth** to specify the size of the marginal note.

In WinHelp, HTML and wxHelp, marginal notes are treated as normal text delineated with horizontal rules above and below.

membersection:1

Used when formatting C++ classes to print a subsection for the member name.

member:1

Used to format a C++ member variable name.

multicolumn:3

Used in **tabular** environment to denote a cell that spans more than one column. Only supplied for compatibility with existing LaTeX files, since all it does in RTF is output the correct number of cell commands, with the multicolumn text squashed into one cell.

newcommand:3

Define a new command; arguments are the command, the number of arguments, and the command body. For example:

```
\newcommand{\crazy}[2]{{\bf #1} is crazy but {\bf #2} is not.}
```

The command must have no whitespace at the start of the line or between the three arguments.

New commands may also be defined in the `tex2rtf.ini` file using slightly different syntax (see **Macro not found error** (section 7.2.1)).

newpage:0

Inserts a page break.

nocite:1

Specifies that this reference should appear in the bibliography, but the citation should not appear in the text.

See also **cite**.

noindent:0

Sets paragraph indentation to zero. See also **parindent**.

normalbox:1

Draws a box around the given paragraph in LaTeX and RTF. In HTML and XLP formats, horizontal rules are drawn before and after the text.

For example:

```
\normalbox{This should be a boxed paragraph for highlighting important
information, such as information for registering a shareware program.}
```

gives:

This should be a boxed paragraph for highlighting important information, such as information for registering a shareware program.

See also **normalboxd** for double-bordered text.

normalboxd:1

Draws a double border around the given paragraph in LaTeX and RTF. In HTML and XLP formats, horizontal rules are drawn before and after the text.

For example:

```
\normalboxd{This should be a boxed paragraph for highlighting important
information, such as information for registering a shareware program.}
```

gives:

This should be a boxed paragraph for highlighting important information,such as information for registering a shareware program.

See also **normalbox** for single-bordered text.

normalsize:1

Sets the font size back to normal.

onecolumn:0

Sets the number of columns to one. LaTeX and linear RTF only.

pageref:1

In linear RTF, generates a page reference to the given label.

pagestyle:1

If argument is *fancyplain* or *fancy*, Tex2RTF separates the header from the rest of the page with a rule. This command must be defined for headers and footers to work properly. See also **setheader**, **setfooter**.

LaTeX and linear RTF only.

pagenumbering:1

The argument may be one of:

alph a, b, ...

Alph A, B, ...

arabic 1, 2, ...

roman i, ii, ...

Roman I, II, ...

LaTeX and linear RTF only.

paragraph*

Behaves as for a subsubsection.

paragraph

Behaves as for a subsubsection.

param:1

Formats a C++ type and argument pair. Should be used within the third argument of a a **func** command.

parindent:1

Indents the first line of succeeding paragraphs by the given amount.

parskip:1

Changes the spacing between paragraphs. In fact, in RTF this will cause two **par** commands to be output if parindent is greater than zero.

par:0

Causes the paragraph to end at this point. LaTeX and Tex2RTF also treat two consecutive newlines as a paragraph break.

printindex

In linear RTF, inserts an index.

popref:2

Similar to **helprefn**, except that in Windows Help, the destination text is popped up in a small window to be dismissed with a mouse click, instead of going to a separate section.

Currently this command can only refer to a labelled glossary entry; see **gloss**.

psboxto:2

Identical to **image**.

quote:1

Indents a short quotation.

quotation:1

Indents a long quotation.

ref:1

In LaTeX and linear RTF, refers to a **label** and causes the number of that section or figure to be printed.

rm:1

Causes the argument to be formatted in a plain, roman font.

row:1

A Tex2RTF command signifying the row of a table within the **tabular** environment. See also **ruledrow**.

ruledrow:1

A Tex2RTF command signifying a ruled row of a table within the **tabular** environment. See also **row**.

rtfignore:1

Ignores the argument in linear RTF.

rtfonly:1

Only outputs the argument in linear RTF.

rtfsp:0

Outputs a space in RTF. Tex2RTF tries to insert a space where one is implied by a newline, but cannot cope where a line starts or ends with a command, in the middle of a paragraph. Use this command to insert a space explicitly.

sc:1

Prints the output in small capitals.

section*:1

Section header, with no entry in the contents page.

section:1

Section header, with an entry in the contents page.

sectionheading:1

Like **section**, but does not increment the section number and does not print a section number in the printed documentation contents page, or in the section heading.

setfooter:6

Tex2RTF has a non-standard way of setting headers and footers, but the default macro definitions in texhelp.sty may be altered to your current method.

The arguments are as follows:

1. Left footer, even pages
2. Centre footer, even pages
3. Right footer, even pages
4. Left footer, odd pages
5. Centre footer, odd pages
6. Right footer, odd pages

For many documents, the first three arguments will be left empty.

The behaviour for first pages of a chapter, section or document is to have a blank header, but print the footer.

For best results, define headers and footers for *each chapter or section*.

Note that this command works only for LaTeX and linear RTF. See also **setheader**.

setheader:6

Tex2RTF has a non-standard way of setting headers and footers, but the default macro definitions in texhelp.sty may be altered to your current method.

The arguments are as follows:

1. Left header, even pages
2. Centre header, even pages
3. Right header, even pages
4. Left header, odd pages
5. Centre header, odd pages
6. Right header, odd pages

For many documents, the first three arguments will be left empty. If **pagestyle** is not plain or empty, the header will separated from the rest of the page by a rule.

The behaviour for first pages of a chapter, section or document is to have a blank header, but print the footer.

For best results, define headers and footers for *each chapter or section*.

Note that this command works only for LaTeX and linear RTF. See also **setfooter**.

shortcite:1

The same as **cite**.

small:1

Prints the argument in a small font.

special:1

Simply copies the argument to the output file without processing (except `\}` is translated to `}`, and `\{` is translated to `{`, to allow for insertion of braces).

ss:0

Outputs the German sharp S character ß.

subparagraph*:1

Behaves as for a subsubsection.

subparagraph:1

Behaves as for a subsubsection.

subsection*:1

Subsection header, with no entry in the contents page.

subsection:1

Subsection header, with an entry in the contents page.

subsubsection*:1

Subsubsection header, with no entry in the contents page.

subsubsection:1

Subsubsection header, with an entry in the contents page.

tabbing:1

Tabbing environment: doesn't work properly in RTF.

tableofcontents:0

Inserts the table of contents at this point. In linear RTF mode, a proper Word for Windows table of contents will be inserted unless the variable *insertTOC* is set to *false*.

tabular:2

Tabular environment: an attempt is made to output something reasonable in RTF format, although currently only simple tables will work. The first argument specifies the column formatting. a pipe symbol (|) denotes a vertical border, one of *l, r, c* signifies a normal column of default width, and *p* followed by a dimension specifies a column of given width. It is recommended that the *p* is used since Tex2RTF cannot deduce a column width in the same way that LaTeX can.

Horizontal rules are achieved with **hline**; two together signify a double rule.

Use the Tex2RTF **row** and **ruledrow** commands for best effect.

For two-column tables that work in WinHelp files, use **twocollist** instead.

Example:

```
\begin{tabular}{|l|p{8.5cm}|}\hline
\row{{\bf A.I.}&{\bf Simulation}}\hline\hline
\row{rules&constraints/methods}
\row{planning&design of experiments}
\row{diagnosis&analysis of results}
\ruledrow{learning&detection of connections}
\end{tabular}
```

This produces:

A.I.	**Simulation**
rules	constraints/methods
planning	design of experiments
diagnosis	analysis of results
learning	detection of connections

TeX:0

Outputs the annoying TeX upper and lower case name.

textwidth:1

Sets the text width (valid for RTF only). This might be used in conjunction with **marginpar**, for example, to leave space for marginal notes.

thebibliography:1

An environment for specifying the a bibliography as a series of **bibitem** commands; the preferred method is to use .bib files and **bibliography** instead.

title:1

Sets the title, to be output when the command **maketitle** is used.

tiny:1

Prints the argument in a very small font.

today:0

Outputs today's date.

toocomplex:1

An environment for dealing with complex LaTeX commands that Tex2RTF cannot handle. In normal LaTeX, the argument will be output as normal. In Tex2RTF output, the argument will be output as verbatim text, for the user to hand-translate into the desired output format.

See also **comment**.

tt:1

Outputs the argument in teletype font.

typeout:1

Outputs the text on the Tex2RTF text window.

twocolitem:2

Used to specify a row for a two column list, a Tex2RTF extension to optimize two-column lists for different file formats. See **twocollist**, **twocolitemruled**.

twocolitemruled:2

Used to specify a ruled row for a two column list, a Tex2RTF extension to optimize two-column lists for different file formats. See **twocollist**, **twocolitem**.

twocollist:1

A Tex2RTF environment for specifying a table of two columns, often used in manuals and help files (for example, for listing commands and their meanings). The first column should be one line only, and the second can be an arbitrary number of paragraphs.

The reason that a normal tabular environment cannot be used is that WinHelp does not allow borders in table cells, so a different method must be employed if any of the rows are to be ruled. In LaTeX, a table is used to implement this environment. In RTF, indentation is used instead.

Use this environment in conjunction with **twocolitem** and **twocolitemruled**. To set the widths of the first and second column, use **twocolwidtha** and **twocolwidthb**.

Example:

```
\htmlignore{\begin{twocollist}}
\twocolitemruled{{\bf Command}}{{\bf Description}}
\twocolitem{File}{The file menu is used to select various file-related
operations, such as saving, loading, exporting, importing, saving as
and various other bits and pieces.}
\twocolitem{Edit}{The Edit menu is used for selection, copying, pasting
and various other bits and pieces.}
\end{twocollist}
```

This produces:

Command	**Description**
File	The file menu is used to select various file-related operations, such as saving, loading, exporting, importing, saving as and various other bits and pieces.
Edit	The Edit menu is used for selection, copying, pasting and various other bits and pieces.

twocolwidtha:1

Sets the width of the first column in a two column list to the given dimension. See also **twocollist** and **twocolwidthb**.

twocolwidthb:1

Sets the width of the second column in a two column list to the given dimension. See also **twocollist** and **twocolwidtha**.

twocolumn:0

Sets the number of columns to two. LaTeX and linear RTF only.

underline:1

Underlines the argument.

verbatiminput:1

Include the given file as if it were within a **verbatim** environment. The command must not be preceded by any whitespace, and spurious whitespace between elements of the command will also trip up Tex2RTF.

verbatim:1

Uses a fixed-width font to format the argument without interpreting any LaTeX commands.

verb

The *verb* command is like the **verbatim** (section 6.1) environment, but for small amounts of text. The syntax is:

```
\verb<char><text><char>
```

The character *char* is used as a delimiter; it may be any character not ocurring in the following text, except asterisk.

For example, `\verb$\thing%^&$` produces `\thing%^&`.

winhelpignore:1

Ignores the argument in WinHelp RTF.

winhelponly:1

Only outputs the argument in WinHelp RTF.

xlpignore:1

Ignores the argument in XLP mode (wxHelp files).

xlponly:1

Only outputs the argument in XLP mode (wxHelp files).

6.2 Accents

The following LaTeX accents work for RTF production:

- `\'{a}` produces á. Valid for a, e, i, o, u, A, E, I, O, U
- ``\`{a}`` produces à. Valid for a, e, i, o, u, y, A, E, I, O, U, Y
- `\^{a}` produces â. Valid for a, e, i, o, u, A, E, I, O, U
- `\~{a}` produces ã. Valid for a, n, o, A, N, O
- `\"{a}` produces ä. Valid for a, e, i, o, u, y, A, E, I, O, U, Y
- `\.{a}` produces ȧ. Valid for a, A

7. Bugs and troubleshooting

7.1 Bugs

Command parsing. If a command is used followed by inappropriate argument syntax, Tex2RTF can crash. This an occur when a command is used in an asterisk form that is only formed in the non-asterisk variety. The non-asterisk form is assumed, which makes the following asterisk trip up the parser.

Setlength. Using the \setlength command doesn't work, since its first argument looks like a macro with the wrong number of arguments. Use an alternative form instead, e.g. `\parindent 0pt` instead of `\setlength{parindent}{0pt}`.

Newcommand bug. Environments in a command definition confuse Tex2RTF. Use the command form instead (e.g. \flushleft{...} instead of \begin{flushleft} ... \end{flushleft}.

Bibliography. There's no flexibility in the way references are output: I expect I'll get round to doing something better, but only if people tell me they need it!

Tables. Tables can't handle all LaTeX syntax, and require the Tex2RTF `\row` commands for decent formatting. Still, it's better than it was (RTF only).

Indexes and glossaries. Not completely supported.

Crashes. Crashes may be due to an input file exceeding the fixed-size buffer used for converting command arguments, especially for the *verbatim* command. Use the **-bufsize** switch to increase the buffer size.

Verbatiminput. Verbatiminput files which do not end with a blank line can trip up following commands.

7.2 Troubleshooting

Below are some common problems and possible solutions.

Some of the syntax that is OK for true LaTeX but which trips up Tex2RTF, may be detected by the TCHECK program included in the tools directory of the Tex2RTF distribution. Some LaTeX errors may be picked up by the LACHECK program, also found in the tools directory.

7.2.1 Macro not found

This error may indicate that Tex2RTF has not implemented a standard LaTeX macro, or that a local macro package is being used that Tex2RTF does not know about. It can cause spurious secondary errors, such as not recognising the end document command.

You can get round this by defining a macro file (default name `tex2rtf.ini`) containing command definitions, such as:

```
\crazy      [2]{{\bf #2} is crazy but #1 is not}
\something  [0]{}
\julian     [0]{Julian Smart}
```

New commands may be defined in LaTeX files, but custom macro files will have to be defined when local style files are being used. See **Initialisation file syntax** (section 4.2) for further details.

The 'Macro not found' error can also be caused by a syntax error such as an unbalanced brace or passing the wrong number of arguments to a macro, so look in the vicinity of the reported error for the real cause.

Here is one obscure situation that causes this error:

```
\begin{center}
{\large{\underline{A}}}
\end{center}
```

The problem is too many curly brackets. This should be rewritten as:

```
\begin{center}
{\large \underline{A}}
\end{center}
```

7.2.2 Unresolved reference

References and citations are usually resolved on a second pass of Tex2RTF. If this doesn't work, then a missing label or bibliographical entry is to blame.

7.2.3 Output crashes the RTF reader

This could be due to confusing table syntax. Set 'compatibility ' to TRUE in .ini file; also check for end of row characters backslash characters on their own on a line, and insert correct number of ampersands for the number of columns. E.g.

```
hello & world\\
\\
```

becomes

```
hello & world\\
&\\
```

7.2.4 Erratic list indentation

Try increasing the value of the variable *listItemIndent* (default 40 points) to give more space between label and following text. A global replace of `\item [` to `\item[` may also be helpful to remove unnecessary space before the item label.

7.2.5 Missing figure or section reference

Ensure all labels *directly* follow captions or sections (no intervening white space).

7.2.6 Unresolved references in Word for Windows

If question marks appear instead of numbers for figures and tables, select all (e.g. CTRL-A), then press F9 *twice* to reformat the document twice. For the second format, respond with *Update Entire Table* to any prompts.

Bibliography

[1] Scott Boggan, David Fakas, and Joe Welinske. *Developing on-line help for Windows.* Sams Publishing, 11711 North College, Carmel, Indiana 46032, USA, 1993.

[2] Helmut Kopka and Patrick W. Daly. *A Guide to LaTeX.* Addison-Wesley, 1993.

[3] Katherine Shelly Pfeiffer. *Word for Windows Design Companion.* Ventana Press, 1994.

[4] Gabriel Robins. The ISI grapher: a portable tool for displaying graphs pictorially (isi/rs-87-196). Technical report, University of South California, September 1987.

[5] Julian Smart. *wxWindows 1.50 User Manual.* University of Edinburgh, 80 South Bridge, Edinburgh, EH1 1HN, 1993.

Glossary

GUI Graphical User Interface, such as Windows 3 or X.

HTML Hypertext Markup Language; an SGML document type, used for providing hypertext information on the World Wide Web, a distributed hypertext system on the Internet.

LaTeX A typesetting language implemented as a set of TeX macros. It is distinguished for allowing specification of the document structure, whilst taking care of most layout concerns. It represents the opposite end of the spectrum from WYSIWYG word processors.

Metafile Microsoft Windows-specific object which may contain a restricted set of GDI primitives. It is device independent, since it may be scaled without losing precision, unlike a bitmap. A metafile may exist in a file or in memory. wxWindows implements enough metafile functionality to use it to pass graphics to other applications via the clipboard. A placeable metafile is a metafile with a 22-byte header which can be imported into several Windows applications, and is a format used by the Microsoft help compiler.

Open Look A specification for a GUI 'look and feel', initiated by Sun Microsystems. XView is one toolkit for writing Open Look applications under X, and wxWindows sits on top of XView.

RTF Rich Text Format: an interchange format for word processor files, used for importing and exporting formatted documents, and as the input to the Windows Help compiler.

wxHelp wxHelp is the hypertext help facility used to provide on-line documentation for UNIX-based wxWindows applications. Under Windows 3.1, Windows Help is used instead.

wxWindows wxWindows is a free C++ toolkit for writing applications that are portable across several platforms. Currently these are Motif, Open Look, Windows 3.1 and Windows NT.

XView An X toolkit supplied by Sun Microsystems, initially just for porting SunView applications to X, but which has become a popular toolkit in its own right due to its simplicity of use. XView implements Sun's Open Look 'look and feel' for X, but is not the only toolkit to do so.