DoctorHelp

(or DOC to HLP)

A Macro for translating Word files into Windows Help files

Introduction

Tired of trying to build Windows Help files?

Trying to maintain printed documentation as well as on-line help files for your users?

This macro is for you!

Encapsulated in a Template file, HELPFILE.DOT for Word for Windows, the macro uses the document's outline structure (defined by the **heading 1, heading 2,** etc styles to build a hierarchically structured Windows HLP file. Cross references can also be defined (but are only processed by the registered version - US$20 for registration - a mere pittance). Keywords are automatically defined in both versions. Features are described later. The shareware version should include a Help version of this file, so you can have a look at the product of the registered version.

Disclaimer

You use this macro at your own risk! **You** are responsible for making suitable **backups** in case it screws up. As far as I know, it does not have any side-effects except modifying your DOC file and creating a HLP file, but you never know with computers!

This macro is not intended to be an all singing, all dancing, help development system. It is good at what it does, namely, **transforming existing written documentation into on-line Windows Help files**. The registered version will also allow you to define cross references in a very natural way, and does various other things. The intention is that you can maintain **one source file** which can be printed or transformed into on-line Help. If you want to create a hypertext that has no hierarchical structure and is more akin to an adventure game (not a clever idea in my humble opinion), then the registered version of this macro will still be useful to you, but you will need to do more work in defining all the cross references.

Getting Started

Copy the HELPFILE.DOT file into your **WinWord** directory. Start Word for Windows.

Create a new file with **File, New** and choose **HELPFILE** as the **template**. If you already have a document that you wish to translate, use the **File, Template** command to assign this new template to it, so that the macro will be available to you.

For the unregistered version, you must manually create a **.HPJ** (Help ProJect) file. A sample is included, which you can view to see how it works. The registered version will create a project file automatically if one is not present.

Run the macro by choosing **Doctor Help** from the **Help** menu. If you don't like it on the Help menu, you can move it elsewhere, or assign a keystroke to it using the **Tools, Options** command.

There are actually two macros to do the complete job. The second stage, called **Run Help**

Compiler, requires that the Help Compiler be accessible. It looks for HC.PIF, so create a PIF file for the Help Compiler and put it in your PATH. Make sure that the PIF file points to the location of the Help Compiler. You can also run the second macro independently of the first. (I do this a fair bit during development if something goes wrong in the second stage and I want to try it again). You may not need to do this unless you are making your own modifications to the macro.

What it Does

The macro reads your document one paragraph at a time. It checks to see if the paragraph **style** is a **heading** type, and if so, what **level** it is (1, 2, 3, etc). It records the level. It defines each **heading** paragraph as a **topic**, including the first line of that paragraph as a **title**, and also as the **keywords** section. A **topic** is one page in the help viewer, and is designated by a hard page break before it. It generates the **context** name automatically. This is used by the Help Compiler in constructing the links between topics. A future version might take more care with the keywords, but this is a fairly easy way to handle things, and moderately effective. The first topic is called "**contents**".

The title for the first section should **NOT** have a **heading** style. Create the top level header with a style of **Normal** or something like that. Having made a complete pass through the document, it starts again, and inserts all the menus required for a user to select sub-topics within any particular topic.

Since the macro inserts hard page breaks before topics, it's a good idea to make sure that you haven't used "Page Break Before" for any of your **header** styles. I will include in the registered version the ability to remove "Page Break Before" if it has already been inserted.

Try it out!

Assuming that you have an HC.PIF file already (if not, go and do that now!), and assuming that you've copied HELPFILE.DOT to your WinWord directory, then you should be able to convert **this** file into a Help file right now.

Although this document uses the Template **HELPFILE**, **your** version of WinWord may not yet know that it exists. Select **File, Template** to assign it again.

If you now pull down the **Help** menu, you should see that the last option is **Doctor Help**. Select it, sit back and enjoy.

Features

1. It saves your document to disk before it tries to change it (but beware of automatic savers - a problem fixed in the registered version by saving the document to its original name, and also under a name of filename.NEW).

2. It creates a **hierarchical structure** for your Help file automatically. Most help files have a basic hierarchical structure, so this makes this part of the job really easy. Your challenge is to then define other cross references, which are also very easy, because you can use ambiguous cross references.

3. It defines **keywords** using the **Title** (first line) of each **heading** section. I will eventually allow the title to be parsed, defining each component word as a keyword.

4. It creates a **context name** for each topic. These are used for building the links.

5. It creates a **Browse** sequence from start to finish so that the document can be read in a linear fashion, just like the printed version. This gives the document the same feel as a book, with the added benefit of the partial tables of contents at each branch point.

6. It calls the Help compiler (assuming you have a **HC.PIF** file available).

7. The registered version does many other things. These are documented later, together with a list of planned enhancements.

Project File

An example Project File is shown below. It can be modified to suit yourself. Note that you should be using the Help Compiler which comes with the **SDK for Windows 3.1** in order for some of these options to work. This Help Compiler also comes with Borland products such as Turbo Pascal and C++ for Windows.

You can change the **icon**, remove **compress**ion, alter the **title** and the **copyright** notice. The [FILES] section contains the name of the file (must end with RTF and be the same name as your DOC file). The **BrowseButtons** option makes sure that the browse buttons are included. The macro creates a browse sequence from the start to the finish, so the document can be read in linear fashion if desired.

```
[OPTIONS]
compress=on
icon=drhelp.ico
title=any old title
copyright=© Roger Hadgraft, 1992

[FILES]
drhelp.rtf

[CONFIG]
BrowseButtons()
```

Limitations

The shareware version has limited capability. The full version can be obtained by registering. Details later.

The shareware version does not include cross referencing, nor does it automatically create a .HPJ file. These are available in the registered version. It is also explicitly limited to 4 levels of outlining, although there's no reason why that can't be changed.

There is a limitation of 200 topics at the moment. You can change that by modifying the first DIM line. I'm not sure how large you can make it without running out of memory. I guess it depends on how long your title names are.

The macro modifies your DOC file. Be careful that you don't save the modified version on

top of the original document. The registered version saves the modified document as filename.NEW before any changes are made, just in case you use a utility for automatic saves (as I do).

The Registered Version

Registration

Registration costs US$20 or AUD$25. Send cheques (preferably in Aussie dollars, but US dollars will do) to:

Roger Hadgraft
Department of Civil Engineering
Monash University
Clayton, Victoria, 3168.
Australia

Please provide an e-mail address if possible to which I can send the registered version.

If credit card facilities would be easier, please contact me by e-mail or fax (as below). Such things can be arranged through the University.

Report bugs/problems/suggestions to:

e-mail: roger.hadgraft@eng.monash.edu.au (preferred)
fax to: +61 3 565 4944.

Updates

Registration includes updates free for 12 months if an e-mail address is supplied.

Using the Registered Version

Extra features in the registered version are:

Cross references

Cross references are detected, and predefined jumps (eg. to a topic in another file) and other calls to macros (eg. to launch a program) are bypassed. Cross references are defined by using all or part of a card title. If this is not unique, a table of possible jumps is built, and appears as a popup for the user. If the cross reference cannot be resolved, a warning is issued.

> **"Ambiguous cross reference: xxx yy zzzz"**

Cross references are defined by using **double underlining** (Ctrl-D) on the word or words to be used. If there are multiple targets, the double underlining is automatically changed to single underlining so that a popup, and not a jump appears.

When defining a cross reference, it is only necessary to use enough of the title as is unique, eg. "Office" might be enough to match "Office Procedures" as long as nothing else starts with "Office". It is entirely up to you how much care you take with these titles.

Ambiguity can be useful to find several sections containing the same word, eg. "Windows" will match Windows 1, Windows 2, Windows 3.0, Windows 3.1, Windows NT. In this case, a popup will appear from which the user may choose one of these or none at all.

Cross references are **not** case sensitive, and the word may match anywhere within a title. However, although the cross reference cannot be longer than the target, although it may be a subset of it. For example, if the target is "cross reference", you cannot use "cross references" as a phrase on which the reader may click. (You could, of couse, just omit the last "s".

If you have a title such as "3.6<TAB>Windows 3.1", it can be matched by "3.6" or "Windows 3.1", whichever you prefer.

If an ambiguous cross reference has already been defined, then another similar reference will use that in preference to an existing title. For example, if there is already an ambiguous cross reference for "Windows", and "Windows" is used again, then the same cross reference table will be used (which is probably what you intended). If instead, there was a multi-table for "Windows NT", and you used "Windows" it will match with "Windows NT". Perhaps I should make sure that any multi-table match must be exact.

Popups

Popups are treated in the same way as cross references, except that instead of using double underlining, **single underlining** is used (Ctrl-U). Again, these can be unique or ambiguous.

The only problem with popups is that the window for them doesn't have a scroll bar, so the amount of text you can present is limited. They are really designed for defintions and short explanations. As a result, the number of target references in an ambiguous cross reference is limited to the resolution of your screen. This may be 20 to 40 depending on the resolution. The macro allows up to 50 to be recorded, but I would suggest that that would be too many for someone to scan.

Defining context names

In most cases, the types of cross references described above will be adequate. They are easy to define, and the macro will automatically allow you to have ambiguous ones. Sometimes, however, you will want to craft your own links, eg.

- a link to a specific topic in another help file (using JumpContext), or
- if you want a graphic as a button, or
- if you are using the hot spot editor to define multiple hot spots on a graphic.

A standard link is defined by having a first part (the button part) indicated by single or double underlining. The button component can be text or a graphic. The second part is the context name of the topic. Normally, you don't need to worry about this, because the macro searches for a match between your button text and one or more of the titles. However, if the button is a bit of text, this is not possible.

Hence, you need to provide the context name. Since context names are assigned automatically by position within your file, I had to provide a mechanism for you to define a context name of your own for topics which are to be accessed in this way. It's very easy. Simply type the name (no blank characters) at the **start** of the title, and make it **hidden** (Ctrl-H). It will be removed during the scanning process, and that context name will replace the automatically generated one.

You can use a graphic as a button by inserting it where you want it, selecting it, then giving it a single or double underline. Insert after the graphic an **underlined** version of the context name as you inserted it before the title of the destination topic. Make sure you turn off the underlining after the button graphic, because there must be no space between the button and the context name.

You can also use these context names for unambiguous jumps by inserting it in the text and using single or double underlining. If you do that, you will want to use names which are sensible, because the reader will see the context name as the button text. I suspect that unambiguous jumps like this are best handled by making your own link as described above. Handcrafted links are ignored by the macro, and looks like this:

Project file

A **project file** is automatically created if one is not present. This means that given a Word file, it can be translated to a Help file **without** any other activities, assuming that it has a structure defined using **heading**s. The default project file is called DEFAULT.HPJ. You can edit this to suit your own needs.

New Buttons

I have defined an **Up** button that allows a user to go up one level in the hierarchy. This is a really useful way of moving back through a file, and is a feature not available in most Windows Help files. It is implemented by a special macro in the DEFAULT.HPJ file, and by the insertion of a short macro for every topic. The **Up** button will sometimes get confused if you branch off to another Help file. In this situation, you may get a "Macro was not found" message.

If you already have a .HPJ file, you may wish to copy the relevant line from the DEFAULT.HPJ file into your own.

I have also added an **Annotate** button to the Help toolbar to encourage users to annotate the Help file. This just triggers the standard Annotate feature, and is similar to scribbling in the margins. Annotations are stored in a .ANN file in the reader's Windows directory. For example, for this Help file, the annotation file would be DRHELP.ANN. An annotation is indicated by a paper clip symbol on the top left corner of the viewing window.

Explanatory Help file

The registered version now comes with a small help file called DRHLPUSR.HLP which describes some of the extra features in your Help file which are not standard in other Help files. The source is included so that you can customise it.

Minor changes

The modified document is saved with a .**NEW** extension to avoid over-writing the original.

The **number of topics** has been increased to 500.

Other improvements have been made to its **robustness**.

The macro automatically uses the **old phrase table** (used during compression). If you have made substantial changes to your DOC file, delete the old phrase table (filename.PH).

When the Help compiler is finished it asks if you want to delete the NEW and RTF files to save space. If so, it automatically reloads the DOC file. Normally you should answer **yes**, since both of these files will be regenerated when you next run the macro.

I have included the ability to ignore a preset "Page Break Before" if it has already been inserted for any heading. This removes one more impediment to having a single source document for printed documents (for which a page break before certain headings may be desirable), and help documents, for which it is essential for **every** heading.

Planned Enhancements

There are a whole lot of things I have in mind. I guess the rate at which these things are done depends on how many people register and request such things!

Keywords

I plan to split the Title into separate words and add each as a keyword. To save having trivial words in the list, I will reject all 1, 2 and 3 letter words. (I may also check words against a list of other common words - from, than, etc).

Calling Macros

Other macros will be written for defining macro calls. These will be useful if you are building a system of help files launched from a top level menu. They would be used for defining common links such as:

- calls to execute programs
- jumps to topics in other help files

Keep with Next

When I implement the dialog box for setting parameters, I will add the ability to turn on the "Keep with Next" option for all headings so that the title is not lost when the user scrolls the page.

Title in the Project File

The macro should prompt for a suitable title to go in the Project File.

Options Dialog Box

I plan to add a dialog box at the start which allows various parameters to be set.

Cross References to other files

It would be a neat idea to be able to resolve cross references into other Help files. This could

probably be done by writing an output file (filename.XRF) which contains titles and context names which could be scanned when cross references from other files are being resolved.

It is already possible to define a jump into another file if you know the context name of the topic to be accessed. Since my macro creates context names, this may not always be easy. If you need to know, the name created for each card is "cardn" where n is the topic number in the file. There is no "card1". Instead it is called "contents".

Error File

I could provide the option to write warnings to an error file rather than have a continuous barrage of popup message boxes. With the introduction of ambiguous cross references, this seems to be less of a problem, so it currently has a low priority.

Deluxe Version

I am considering having two different registered versions at different prices. It seems to me that there are a whole lot of things I could incorporate. You may wish to start with the basic registered version for US$20, and upgrade to the deluxe version for probably US$40. The latter does not yet exist, but will probably include things like the dialog box for customising its behaviour and the ability to handle a suite of interacting help files by creating a cross reference table for each file.

Helpful Hints

Keep with Next

One item not explained in the Help Compiler manual is the use of the "Keep with Next" and "Keep Together" options for headings. I usually use these for written documents because they prevent stray titles on the last line of a page. In a Help context, they produce a page for which the title will not scroll off the screen, which is usually a good idea.

Just beware because if you use a "Keep with Next" on a topic that is to become a popup, then you'll only see the Title. I hope that the next version of the Help Compiler fixes this problem. I assume that this is a bug (in the Help Compiler) and not a feature.

Useful Macros

It is possible to include a range of macros in your Help files. These access in-built capabilities within the Help engine. The full range of them is described in Microsoft's documentation. Here are a few that you might find useful.

In each case, there must be some text that is underlined (single or double) followed by a macro string which is hidden (Ctrl-H). The macro always starts with an exclamation mark: ! This is how it might appear on screen in Word:

Execute a Program

It is often useful to be able to launch a program from within a Help file. With this capability you could develop a specialised environment for running a small number of programs, or you might want to start a program within a tutorial document. The macro is:

!ExecProgram("progname filename1 ...", n)

The n=0 for a Normal window, n=1 is minimised, and n=2 is maximised.

For instance, click below to run Notepad using: !ExecProgram("notepad.exe"):

Run Notepad

Jump to another Help file

Sometimes you may have a collection of Help files which you want to link together into a Help system, or you might just have one other Help file (eg. DRHLPUSR.HLP) which you want to let the user read, without including it in your own file. You can allow the reader to jump to the contents page of that file with the JumpContents macro like this:

Jump to Outline Help

It is then possible to backtrack to the original file using the **Back** button. Also, if you click on the **History** button, you will see that the topics in the original file are available, eg.

Contents
WINHELP:Contents for How to Use Help
WINHELP:Copying a Help Topic onto the Clipboard
Contents

where the prefix WINHELP indicates that a topic in that file.

Jump to a Topic in Another File

If you know a topic name in another Help file, you can jump to it with the JumpContext macro, eg:

Browse Buttons

This is handled by defining context names in the other file.

For some comments about backtracking from other files, see the topic: Jump to another Help file

Opening a second window

Sometimes you may want to open another Help file in a second window, rather than replacing the one that is currently open (eg. Jump to another Help file or Jump to a Topic). You can do this using the ExecProgram macro by running another copy of winhelp.exe:

!ExecProgram("winhelp.exe drhlpusr.hlp")

Don't forget to insert the winhelp.exe part. The reader of your document will need to close that window, or switch back to the original window by clicking on it. Which option you provide depends on the sophistication of your readers. Too many open windows on the screen can be confusing.

Bitmaps

If you intend to include large bitmaps in a Help file, I suggest you store them in seaparte files, rather than paste them into your DOC file. I have had some problems with the Help Compiler reporting errors on embedded Help files.

Section Breaks

I ran the macro on one of my papers which had a section break in it. For some reason, the function that returns the filename of the current document returns a blank instead. This means that the "Run Help Compiler" macro will crash. I can't think of any solution to this problem except for removing section breaks. Perhaps I can do this automatically.

Bullets

Bullets don't show up in a Help file.

Subscripts & Superscripts

Subscripts and superscripts don't show in a Help file. Perhaps Microsoft will fix this?

Distribution

If you wish to pass the **shareware macro** to others for evaluation, please include **all** the files:

HELPFILE.DOT
DRHELP.DOC
DRHELP.HLP
DRHLPUSR.HLP
DEFAULT.HPJ
DEFAULT.ICO
HC.PIF

Note that the .HLP file included in the shareware version was generated with the registered version, so that you can see the advantages of registering.

Please do not remove the copyright notice from the text of the macro, and please do not pass on the registered version!

Customisation

I am happy to customise the macro to suit registered users. If the work involved is likely to be of general use, there will probably be no charge, but if it is specific, then we can negotiate a charge. This will be a fixed price contract.

Glossary

Ambiguous cross references

An unambiguous cross reference is one which has several targets. This is possible where the phrase being is used is found in the Title of several Topics. This macro builds a table of destinations, and presents this table to the reader as a popup. The reader can choose a destination. Once the list of destinations gets over 10, this will be a bit overwhelming, so make sure that this doesn't happen.

Browse

Browse is the ability to read topics in sequential order. The macro creates just one browse sequence from the first topic to the last, so that one of these help files can be read as if it was a book.

Context names

The context name is the code by which the Help system refers to each topic. This macro generates context names automatically.

Cross References

Most hypertext systems have numerous cross references to other parts of the same document, or even to other documents, so that the user is pointed towards potentially useful information.

Help Compiler

You need the Help compiler to generate Help files. This macro writes out the required input file for the compiler. The compiler is included as part of the Microsoft Windows SDK, and other developers, such as Borland, have licensed it from MS, and have included it with their Windows products.

Keywords

The Windows Help system allows keywords to be defined by the author. This macro adds each title as a "keyword". A later version will probably add the component word separately.

Styles

Styles are used in Word to format paragraphs of text. There is a style called **Normal** for normal text, and there are a set of **heading** styles for defining headings. **Heading 1** is the top level section heading, then comes **heading 2** and so on. This macro uses the **heading**s to define the document's hierarchical structure. Note that the title of the document should have **no heading** style. The major sections should be defined by **heading 1**.

Template

Word for Windows uses Template files (with a .DOT extension) for storing the look of

particular documents. They are editable in the normal way, and you can set styles and all sorts of other things to suit the different types of documents that you produce. Templates can also contain **macros** which is how this macro is distributed. Simply copy the HELPFILE.DOT file into your Word directory.

Title

The title of each topic is assumed to be the line which has the **heading** style. The title is used by the Help system for backtracking, for bookmarks, and for references in the **Search** capability.

Topic

A **topic** is the basic unit in a Help file. It corresponds to a page. You can only read one topic at any one time, but you can branch to see other topics in a variety of ways. Normal topics will have a scroll bar, while popup topics do not.