Using Microsoft Word for Windows to Create Help Files

Despite what you may have heard, it is possible to create decent help files by using only Word for Windows and the SDK help compiler (hc31.exe or hc30.exe). There are at least two Microsoft SDK books describing Help Files and their construction -- but, neither gives the information about how all these cryptic commands can be entered using just Word for Windows.

This help file was constructed using: **1**) **Word for Windows 2.0a** **2**) **hc31.exe** and **3**) Windows' **Paintbrush** (for icons). If you want absolutely first rate Help Files, that are relatively easy to construct -- and have a spare $300 laying around -- then the WexTech package is likely what you want. You've heard the saying "You get what you pay for" -- and it's true. So just consider what you paid for this.

This hlp file and info is only meant to be a starting point in creating Help files. There is much more that can be done with Help files that is beyond the scope of this hlp file. Read the Microsoft documentation for advanced topics like macros and multiple hotspot graphics.

Requirements for Creating Help Files

For Basic Help Files -- You'll need the following items:

Microsoft Windows 3.1
Microsoft Help Compiler (hc31.exe)
Microsoft Word for Windows 2.0a

It's Sure Nice to Have --

Microsoft SDK "Programming Tools" booklet
Volume 4 of the SDK Programming Reference
Microsoft SDK Hotspot Editor (SHED.EXE)
Microsoft SDK Image Editor (IMAGEDIT.EXE)
Patience and time to experiment

Overview of Help File construction

When you create a C program, many people use a **makefile** and run **nmake makefile** when creating the executable file. The **makefile** is just a text file that gives the names of various files and instructions to the compiler/linker to create the executable program. In the same fashion, the Help Compiler (**hc31.exe**) requires a text file with instructions on how to build the help file. For a C program the makefile will usually have .c, .h, .def, etc. files referenced. The Help Compiler usually only needs to know **.rtf** (rich text format) file(s). The Help project file that contains the instructions and file names is a **.hpj** file instead of the C **makefile**. A basic Help file project can consist of just two files: **somename.rtf** and **somename.hpj**. To make the help file you would enter the command **hc31 somename.hpj** from the dos command line. When creating the files using Word for Windows -- be sure to choose **Save As** from the **File** menu and then select **text only** for the **.hpj** file and **rtf** for your **.rtf** file.

Just like the C **makefile** has syntax rules , the **.hpj** file also has syntax rules. View the topic for .HPJ syntax for more information.

A popular thing to do is to add Bitmaps (pictures) to the Help file. Bitmaps can be used as just pictures, as jumps to other topics, and pop-ups. You can use Windows' Paintbrush to create the Bitmaps (.bmp files).

Microsoft Word for Windows is used to create the **.rtf** and the **.hpj** files. Creating the **.rtf** file is easy because you should already be familiar with the basics of Word for Windows. In addition, it's easy because the text you see in Word is what you will see in the Help File. View the topic Constructing Help Files for more information about using Word to create the **.rtf** files.

Word for Windows <--> Help Files

Here are some suggestions to try when you are just starting to make help files. As you gain more experience and feel more comfortable with it -- you'll develop your own "routine".

Set Word for Windows

Before you start anything -- you need to know that much of the Help Files process makes use of "hidden" text and "footnotes". You should already have an general idea about what footnotes are. You will need to set Word for Windows to show hidden text. Start Word for Windows, but before you go too far, choose **Tools** then **Options** from the menu. In the **View** Category -- make sure the **hidden text** box is checked in the Non-Printing Characters section.

It's a pretty good idea to use Helv 10pt font for much of your text to start. It doesn't look bad, and most systems have this font on them.

First Time suggestions

1. Try making a first "test" Help file. This is just to see how things work. Make up a short file just to experiment with.
2. Taking a little time and maybe a few notes before you start your text file usually pays off in less confusion when it comes time to do jumps, etc. In other words, try as best you can to "rough out" how the Help File looks and operates before doing a lot of typing.
3. After reading through these notes.....Open a .rtf file (like wfwhlp.rtf) and take a look at some of the things you have been reading about.

Constructing .RTF files

Be aware of what Windows takes care of and what you must take care of when it comes to Help Files. Windows will handle the help window size and wrapping text in the window. You have to take care of fonts, sizes, colors, and bitmaps/graphics in your .rtf and .hpj files. Windows takes care of the Contents, Back, and History buttons. You have to tell windows to use the Search and/or Browse (<< >>) buttons in your .rtf file.

IN A NUTSHELL --

Help files are very similar to a card file. Each "screen" in the Help file is like a single card. You get to different screens by selecting one of the "jump" words. Jump words are green with a solid underline. When using Word for Windows to create the .rtf text file -- each screen is a separate page. Each of the pages is referred to as a "topic". Each topic has a unique tag(character string) associated with that page. Each topic **must** have a string. Each topic **can** also have a title, keyword, and identifier. If you use a keyword, you must also use a title. Since each topic(i.e. page) is identified with a string, jumps are possible by using the string.

The string, title, keyword, and identifier are stored with the .rtf text file in the form of footnotes. For each topic -- you enter the string(and the optional title, keyword, and identifier) as a footnote using "special" characters for the footnote mark. Instead of the footnote belonging to a particular word -- as you might in a report -- these footnotes belong to the topic(i.e. page) and are usually found at the start of the page.

Just so you won't get confused when reading Microsoft documentation; the string I mentioned above is called a "context string". The items and their associated footnote marks are as follows:

Item	Footnote mark
context string	#
title	$
keyword	K
identifier	+
macro	!

Simple Help File listing --

```
.....................................................................
  #   First page title          ( "#" footnote mark for pg 1)

  press jumppage_two to jump to the next page.

.....................................................................
  #                                          ("#" footnote mark for pg 2)
  came here from page one.

  press jumpmain_index to return to page 1.
.....................................................................
      (the footnote list window in Word for Windows)
  # main_index                        (footnote from page 1)
  # page_two                          (footnote from page 2)
.....................................................................
```

First Time Tips -- Take a look at the wfwhlp.rtf file provided to see the footnotes, etc.

Footnote Syntax

footnote mark	meaning
#	context string
$	title
K	keyword
+	browse sequence
!	macro

Footnotes are associated with a topic (page) as opposed to an individual word or phrase. Generally, footnote marks appear at the beginning of the page. The exception is the macro (!) command (see Microsoft docs).

Jumps Syntax

Jumps are created by identifying a word or phrase to be clicked on by the user to jump to another context string (page). In Word for Windows the jump word is double underlined by selecting Format then Character then double underline from the menu. Imediately following the double underline jump word is the context string name in hidden text. In Word for Windows hidden text is done by selecting Format then Character then hidden from the menu. Hidden text is displayed with an underline of dots. Windows takes care of making the word green colored in the help file. Sample of a jump with hidden context topic:

jumppage_two

Pop-up Syntax

Pop-ups are created by identifying a word or phrase to be clicked on by the user to pop-up a dialog box containing context string info. In Word for Windows the pop-up word is single underlined by selecting Format then Character then single underline from the menu. Imediately following the single underline pop-up word is the context string name in hidden text. In Word for Windows hidden text is done by selecting Format then Character then hidden from the menu. Hidden text is displayed with an underline of dots. Windows takes care of making the word green colored in the help file. Sample of a pop-up with hidden context topic:

popuppopup_string

Bitmap Displays

Only 16 color bitmap displays are supported.

Grapics can be incorporated by using the normal procedures in Word for Windows.

Bitmaps can also be used for jumps or pop-ups. Bitmaps can also be displayed in either the right or left margin with text wrapping around them as shown by the bitmaps at left and right. Bitmaps can also be displayed as a "character" where the bitmap will be displayed with the bottom and lower left starting at a character location as shown below:

some text  some more text.

Bitmap Jumps

Bitmap jumps are created by identifying the bitmap to be clicked on by the user to jump to another context string (page). In Word for Windows the bitmap style and filename is double underlined by selecting Format then Character then double underline from the menu. Imediately following the double underline bitmap info is the context string name in hidden text. In Word for Windows hidden text is done by selecting Format then Character then hidden from the menu. Hidden text is displayed with an underline of dots. Sample of a bitmap jump with hidden context topic:

{bmc filename.bmp}jumpto_contextstring

Bitmap Pop-ups

Bitmap pop-ups are created by identifying the bitmap to be clicked on by the user to pop-up another context string (page). In Word for Windows the bitmap style and filename is single underlined by selecting Format then Character then single underline from the menu. Imediately following the single underline bitmap info is the context string name in hidden text. In Word for Windows hidden text is done by selecting Format then Character then hidden from the menu. Hidden text is displayed with an underline of dots. Sample of a bitmap pop-up with hidden context topic:

{bmc filename.bmp}popup contextstring

.HPJ Syntax

There are lots of styles/options to the .hpj file. Best to read the documentation if you have it. The following is a sample .hpj text file with comments (comments begin with a ";"):

```
; sample .hpj file
[OPTIONS]
CONTENTS=main_index     ;usually context of 1st pg
TITLE=Help Window Title string
COMPRESS=TRUE           ;compress hlp file
WARNING=3               ;higher more strict
   << optional >>
BMROOT=path             ;path to .bmp files
ROOT=path               ;path to project
ICON=c:\name.ico        ;minimize icon to use
OLDKEYPHRASE=OFF
REPORT=ON               ;report hc31 progress

[FILES]
filename.rtf            ;must include all .rtf's
   << and/or >>
#include <filenamelist.h>

[CONFIG]
BrowseButtons()         ;to use browser

[BITMAPS]
first.bmp               ;section can be skipped
second.bmp              ;if BMROOT is used
```

NOTES

The usual copyright info is in effect. Mostly everything here is by Microsoft including:
hc31.exe -- Word for Windows -- Windows -- Paintbrush -- etc.

After going to all the trouble to add browser identifiers in your .rtf file -- don't forget to enable the browse buttons by adding **BrowseButtons() macro in the **[CONFIG]** section ot the .hpj file.

**Multiple keywords ("K" footnote mark) can be assigned by separating with semicolons (i.e. K First Keyword;Second Keyword;etc.)

You **must provide a title ("$" footnote mark) if you assign keywords.