Table of Contents

Acknowledgements
Alias Section
Authorable Button
Browse Sequences, Creating
Build Tag Manager
Building Your Help File
Calling WinHelp()
Calling WinHelp Version 4.0
Compiler Errors 3.1
Compiler Errors 95
Compiler Options
Creating Reports
Defining Help Windows
Embedded Video (AVI)
Environment Options
File Functions
Getting Started
Glossary Wizard
Help Compiler Notes
The Help Magician Window
Help Topics Browser
Help Wizard
Importing RTF Files
Importing Text Files
Inserting
Inserting Bitmaps/Pictures
Introduction
Keyword Management
Macros, Working with
Macro Reference
Managing Topic Text
Multimedia with Winhelp
Navigator
On Line Help
Options, Building
Paragarph Styles
Project Files and Winhelp()
Project Management
SHED.EXE
Spell Checker
Testing and Navigating
Tools
Web Authoring
Working with Keywords

Introduction

The Help Magician has revolutionized the way Windows help systems are created. A highly specialized WYSIWYG editor includes embedded functions for every aspect of developing a Windows help file:

- A true WYSIWYG help system development environment that emulates WinHelp and allows you to test your help system as you develop.
- Convert help files to manuals.
- Convert manuals to help files.
- Creating Jumps and Popups is as simple as selecting the link word or phrase and clicking on the appropriate menu item, toolbar icon, or pressing the hot key.
- Define paragraph tags for your help system including spacing before, after and in between lines; select left, right or centered justification; force word wrapping; set indentation and tabs typeface, size, boldness, underline, strikethru and color.
- Load and save pre-defined paragraph styles.
- Perform spell checking on your help system text.
- Include the most popular graphics file formats in your help system with character, left, or right justification.
- Create numbered or bulleted lists with numbers, letters, or other characters.
- Create non-scrolling regions.
- Browse sequences can be created, with multiple groups, by selecting the desired page titles and, with the use of a special table, the sequence can be arranged or re-arranged to your liking. Topics will be browsed, in the compiled help system, in the order in which they appear in the table. Topics can always be added or deleted.
- The editor performs dynamic syntax checking to ensure an accurate compiled help system.
- Cut, copy, paste, find, find next, find previous, replace with verify, and replace all.
- Import and export ASCII and RTF files, including optional special formatting for the Help Magician.
- Jump to or pop up topics in secondary windows.
- Adjust the size, positioning, colors and initial attributes of all windows.
- Select the character set used in the final help system.
- Create macro definitions that contain one or more help macros and assign them to hotspots, individual topics or to be run at startup.
- Create user-defined macros and use them in your help file.
- Create your own special effects, such as sound, video, animation, custom help buttons, glossaries, and more!
- Easily add buttons to WinHelp's standard buttons with Help Magician's Create Button function .
- Scan Visual Basic source and write a complete help system shell for your application.
- Single button automatic Glossary creation.

The Help Magician provides a new level of organization and functionality to the creation of Windows help systems:

- No knowledge of Windows help files is necessary - the Help Magician handles the entire project for you, transparently. The Help Magician writes the RTF file and the project file (.HPJ), complete with map. It will also write the #define file for C programmers.
- Printouts/reports for the entire help file, browse sequences, context relationships, a link list, and more.
- The RTF file, written by the Help Magician, can be read into Word for Windows or other word processors that support RTF files.
- Dynamic browse sequence ordering.
- Edit, test, write RTF files, compile, and call WINHELP.EXE all from the Help Magician's environment.
- Call SHED.EXE, Bitmap Magician, Word for Windows and Paintbrush from the Help Magician's

menus.

- Automatic help build directory and date and time stamp maintenance.
- Maintain separate configurations for different help systems.
- Select help compiler options and set the help system index page from a simple form.
- All functions are available with the mouse or the keyboard using the assigned hot keys.
- Keywords are entered in drop down boxes on the main form.
- Automatic prevention of duplicate topic titles and context numbers.
- Quick backup menu option.
- Full project management allows multiple help developers on a network to be working on the same help project without interfering with each other.

Another help system application required 18 minutes to generate a help file from a 4000 word document. It takes the Help Magician only seconds to write the RTF file from the same document. Combine this with the WYSIWYG development environment and you have the fastest, most efficient help development system available.

Speed, organization, and ease of use are the trademarks of the Help Magician. All combined into a highly professional application, the Help Magician will increase the productivity of the professional and provide professional help systems for first time users. The Help Magician works with all Windows development languages and environments.

The development environment emulates WinHelp so that you can test your help file as you develop. No need to compile!

The Search mode duplicates the Winhelp functions that will be available to the end user when viewing the help system from your application.

HLPMAGIC

HLPMAGIC.HLP, the Help Magician's own help file was created in the Help Magician. Some of the capabilities of the Help Magician are demonstrated in the help system. More of the features are demonstrated in the DEMO400.HLP file. The source for this help file is installed to the \SOURCE sub directory of the Help Magician directory.

See Also

Function Keys
Keyboard
System Requirements
Technical Info
Version 4.0 - New Features

Acknowledgements

Thanks to the following people for their contribution to the development of the **Help Magician**:

Harald Zoschke (Zoschke Data GmbH - Germany) for his assistance in the Beta testing of the Help Magician and for the publication of the German version of the program.

Don Lambert (Software Interphase, Inc.) - for the development of the Project Management portion of the program.

Jeff Bennet (Bennet-Tech Information Systems) - for the development of and assistance in the implementation of the Alltext WYSIWYG control used as the editor in the Help Magician.

Statech Software for the development of the RTF translator, HTML translator, Macro Editor and the Manual to Help Conversion feature.

Anne Selder for her invaluable assistance Beta testing the Project Management portion of the Help Magician.

Debbie Heberger for the assistance in the development of this help file and the manual.

Thanks,
Robert B. Heberger

Version 4.0 - New Features

(1) Help Wizard
The Help Wizard is a utility that works in conjunction with the Help Magician to guide you through the creation of a help file and assist you with any specific functionality of the Help Magician.

(2) Win95 Contents Editor
WinHelp version 4.0 offers a new approach to navigating Help files. The contents and search functions have been combined into a single "Help Topics" dialog box. The Help Magician Contents Editor allows the help author to create a "contents file" that will be used to create the Contents, Index and Full Text Search tabs featured in the Help Topics dialog box.

(3) Help Topics Browser
The Help Topics Browser in Help Magician Pro 95 emulates the WinHelp 95 Help Topics Browser Window that contains the Contents Tab, Index Tab, and the Find Tab.

(4) Authorable Buttons
The Authorable Button feature allows help authors to create and insert user-defined buttons into the help text that can execute WinHelp macros.

(5) Embedded AVI
The Embedded AVI feature allows help authors to embed AVI video segments into the body of a Help topic.

(6) New Multiple Help Support
Three types of Help have their own instance of WinHelp:

- Program-invoked Help.
- Context-sensitive Help.
- Training card Help.

In addition to these three specific types, any number of instances of help may be run either by running WinHelp directly or through its association with the .HLP extension.

(7) Support for Transparent Bitmaps
Bitmaps can now be transparent againsta background. WinHelp replaces any white pixels in the bitmaps with the background color of the topic window.

(8) Default Topic Window Attributes
The help author can now specify default attributes such as position, size and color for any topic in the help file.

(9) Enhanced Secondary Window Support
Secondary windows now have the following features:

- Configurable button bar.
- Up to 255 window definitions allowed per project.
- Up to nine secondary windows may be dispayed at the same time, in addition to the main window.
- The Back button now works for secondary windows.
- Auto-size feature lets author allow window to size to text for "best-fit" of window on screen.

(10) New Macro Features

- Twenty-six new macros have been added to the list of WinHelp macros. Some older macros have

been obsoleted and removed from the list.
- The Keyword Macro feature allows the help author to assign a list of macros to a keyword so that when the end user selects the keyword from the Index Tab, the macros will execute.
- Macros may be assigned to a window definition so that when a topic is displayed in that window, the macros will execute.

(11) New Topic Linking Features

The "Jump to Keyword" feature allows the help author to provide jumps to multiple topics based on keywords rather than specific context strings. These jumps are resolved when the user clicks on them instead of when the file is compiled. The are several distinct advantages to this approach:

- Inter-file jumps can be created for Help files that may change after the original jump was created.
- Inter-file jumps can be planned for Help files that might not exist until after the program is released.
- The help author can setup inter-file jumps for Help files that might not be supported amongst various versions of the program.
- Keyword jumps can span multiple help files.

(12) New Font Handling Features

The following features have been added:

- The help author can now set the default fonts for text used in the Contents tab, Index tab, and the Topics Found dialog box.
- The language to base keyword sorting order upon can be specified to ensure that a help file created in one language will display the keywords correctly in any other language version of Windows.
- Greater character support has been added for features such as smart quotes.
- Automatic localization of quotes has been added to properly display quotation marks for several European countries.

(13) New Compiling and Testing Features

Several new features have been added to aid in debugging help file flow and interaction. These features include:

- A new Help Author mode to allow live debugging of compiled help files.
- Several new ways to launch WinHelp from the Help Magician to simulate various end-user conditions.
- New error reporting features to get a more or less detailed view of the compiler processes.
- New macros to test and compare different help files in a variety of ways.
- Missing bitmaps are reported by the compiler with greater detail.
- Help file compression may be optimized using various compression methods such as Hall and Zeck.
- Full Text Search options may be set for best performance versus space considerations.

(14) New Help to HTML Page Creation Support

Now Internet authors can create HTML pages directly from their Help files. No need to buy another editor and go through unnecessary conversions to create HTML files from your Help files. The Help Magician will generate them for you. Help Magician Pro 95 automatically converts help topic pages to individual HTML files, bitmaps to GIFs, and SHG files to GIFs with Maps. It also supports links from text to URLs.

(15) New Options for Building Help Files

(16) Assign macros to execute once a keyword is selected from the Index Tab

(17) New Visual Help Window designer with support for Windows 95 help

(18) Full-featured Keyword Management using a Keyword Database

(19) Long Filenames support for Windows 95

(20) Enhanced Project Management to include keywords across a project

(21) Support for Visual Basic 4 in the VB Source Code Scanner

(22) Supports all versions of Windows and Windows Help in one complete package

(23) Supports 256-color and higher bitmaps for Windows 95

System Requirements

Windows Version

Help Magician Pro 95 supports Windows 3.1, WFWG 3.11, Windows 95, and Windows NT. The Microsoft Help Compiler 3.1 (HC31.EXE) will generate Windows 3.1 help files from RTF files generated by Help Magician. Windows 3.1 help files are displayable by Windows 95 and Windows NT. To take advantage of the newer Windows 95 help features, you will need to use Microsoft's Help Compiler for Windows (HCW.EXE). Both compilers come with Help Magician.

Memory

The Help Magician requires a system with at least a 486 processor and at least two free megabytes of memory which means that the system should be equipped with at least eight megabytes of memory. Performance and help system size will be dependent on the capabilities of the host system.

Extended version of the Windows Help Compiler (the latest versions for Windows 3.1) require at least 500K of conventional memory to compile an RTF file.

Help Compiler

A copy of the Windows Help Compiler, installed with the Help Magician, is required to compile the RTF file written by the Help Magician into a WinHelp readable file.

Microsoft Help Compiler version 4.0 (HCW.EXE and HCRTF.EXE)
Microsoft Windows Help version 4.0 (WINHELP.EXE)

The Microsoft WordPad program that comes with Windows95 is **not** suitable for creating topic (.RTF) files.

SHED

SHED.EXE, installed with the Help Magician, is required to create segmented hypergraphics or multiple hot spots on a bitmap.

Keyboard

Alt 0-9	Place font styles 10-19	**Ctrl N**	New File
Alt C	Edit Context Number	**Ctrl O**	Open File
Alt I	Edit Context String	**Ctrl P**	Create a Popup
Alt P	Page Number	**Ctrl PgDn**	Goto next page
Alt T	Topic Title	**Ctrl PgUp**	Goto previous page
Alt W	Keywords	**Ctrl Q**	Goto Bookmark
Ctrl 0-9	Place font styles 0-9	**Ctrl S**	Save current help file
Ctrl A	Save As	**Ctrl T**	Paragraph Tag
Ctrl Alt End	Goto last page	**Ctrl U**	Multimedia
Ctrl Alt Home	Goto first page	**Ctrl V**	View/Modify Links
Ctrl Alt N	Goto nonscrolling region	**Ctrl W**	Help Wizard
Ctrl Alt S	Goto Scrolling Region	**Ctrl X**	Exit the Help Magician
Ctrl B	Inserts Bitmap/Picture	**Ctrl Y**	Load Styles
Ctrl D	Delete Links	**Ctrl Z**	Insert a new page
Ctrl E	Saves Styles	**Del**	Cut selected text
Ctrl F	Character Formatting	**End**	Goto end of line
Ctrl G	Deletes current page	**Home**	Goto beginning of line
Ctrl I	Paragraph Formatting	**Ins**	Insert/Overwrite Mode
Ctrl Ins	Copy selected text	**PgDn**	Moves down one screen
Ctrl J	Create a Jump	**PgUp**	Moves up one screen
Ctrl K	Set Bookmark	**Shift Ins**	Paste text
Ctrl L	Define Styles		

See Also
Function Keys

Function Keys

Alt F2	Goto Next Image	**F6**	Write RTF
Ctrl F3	Find	**F7**	Run Compiler
Ctrl F4	Goto Context String	**F8**	Rebuild All
Ctrl F5	Goto Paragraph Style	**F9**	Call WINHELP.EXE
Ctrl F6	Goto next paragraph style	**Shift Ctrl F2**	Goto Next Mid Topic
Ctrl F7	Call SHED.EXE	**Shift Ctrl F3**	Replace
Ctrl F8	Call Paint Brush	**Shift Ctrl F4**	Goto Context Number
Ctrl F9	Call Bitmap Magician	**Shift Ctrl F9**	Spell Check
Ctrl F11	Annotations	**Shift F2**	Goto Next Popup
F1	Help	**Shift F3**	Find Previous
F12	Navigator	**Shift F4**	Assign macro to project
F2	Goto Next Jump	**Shift F5**	Assign macro to a topic
F3	Find Next	**Shift F6**	Save Macro definitions
F4	Goto Title	**Shift F7**	Load Macro definitions
F5	Create macro definitions	**Shift F8**	View link list

See Also

Keyboard

Technical Info

Paths

All files, pertinent to the current help system, should be in the 'ROOT' directory with the exception of bitmaps and multimedia elements, which may reside in separate directories (the paths must be specified in the Bitmap Paths field in the Paths form). 'ROOT' does not mean the root directory of the default drive, but the 'ROOT' directory for the build, as described in the help compiler documentation. Essentially, the 'ROOT' directory for a help file can be any directory on your drive that contains the necessary files (such as .HLX and .RTF) to build the help system. The 'ROOT' directory is set to the directory you choose, when you open or save a help file from the File Menu. Also, the bitmap directories can be called out in the Options Menu, under Paths. If you are using the project management feature of Help Magician, all project member files (hlx, rtf, bitmaps, multimedia files) MUST be located in the same directory, otherwise access to these files wouldn't be possible across a network.

Files

The Help Magician writes one main file to maintain each help system, with an .HLX extension. This file contains all of the text you have entered into your help file and a database of the Jumps, Popups, Bitmaps, Fonts, Browse Sequences, etc. for your help system. Several other database support files are written to store keywords and help project information. These files have the following extensions: .ISD, .ISF and .ISM. The file written for the help compiler is an RTF (Rich Text Format) file and has a .RTF extension. It can be read into a word processor that is capable of reading RTF files.

Note that not all word processors that provide RTF capabilities write RTF files completely compatible with the Windows help compiler. Microsoft Word for Windows (versions 1.0, 2.0, 6.0, and 7.0) was the only word processor we tested that wrote RTF files that were completely compatible with the Windows help compiler.

In version 2.5 of the Help Magician, support for Lotus Ami' Pro was added.

A project file is also written for the Windows Help compiler, with a .HPJ extension. The compiled help file has the standard Windows help file extension, .HLP. The compiler will generate a phrase-table file with the extension, .PH if the compression option is selected.

Defined paragraph tags can be saved for use with other help files. The default extension for these files is .STY.

The file extension for a help file saved with the File/Backup option is .HLK.

Archive files created under project management will have an extension of .HZP. They are industry compatible "ZIP" files.

Up to ten previous versions of an HLX file can be automatically saved (Options/Environment Options/File Save). The extensions for these files are H01, H02, etc.

Defined macro definitions can be saved for use with other help files. The default extension for these files is .HLM.

WinHelp 4.0 creates an index file with an extension of .GID the first time a Help file is viewed. This index file is used by WinHelp to quickly display the Contents tab in the Help Topics dialog box. This file should be deleted when a new Contents file is created so that WinHelp can create a new .GID file. This can be accomplished in several ways:

- Inform the user to delete the GID files before installing an update to the Help system.
- Instruct the installation program to delete the appropriate GID files during an upgrade.
- Instruct the installation program to immediately run WinHelp with a special switch to regenerate the GID file(s).

Standard macro prototypes are stored in a file called "MACPROTO.HLD". **This file should not be modified by the novice!**

Pages/Topics

The actual number of topics is limited by available memory.

Getting Started

This information provided here is a excerpt from the "Getting Started" manual. With the assistance of the Help Wizard, you may not need the "Getting Started" manual or the "Users Manual".

When you first run the Help Magician Pro 95 application, a window will pop up over the main window. This window is the **Help Wizard**.

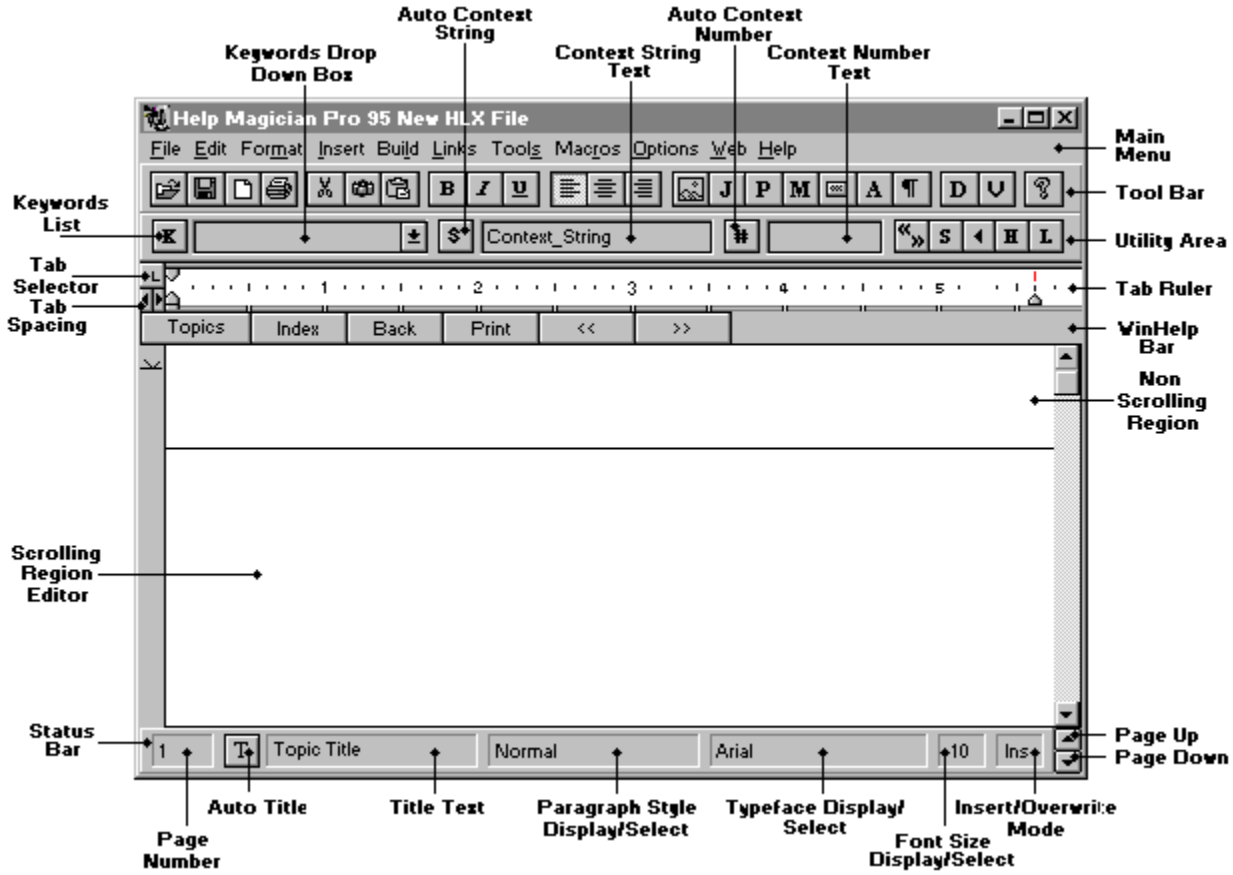
The **Help Wizard** will begin at the **Introduction** category. Please read the Introduction and, at the end of the Introduction, elect to view the **Tutorial** category. In the Tutorial, the Help Wizard will build a simple help file, performing all the steps for you. You will learn something about help files, the Help Magician and see how simple it is to create a help file in the Help Magician.

The **Help Wizard** will also give you a "tour" of most of the forms used in the program and "assist" you in the use of these forms as you create your own help file simply by selecting the appropriate sub menu from the **Category** menu.

When you're ready to create your own help file, the **Help Wizard** will guide you through all the steps necessary to setup a new help file.

Let the **Help Wizard** assist and guide you through all of the functions in the Help Magician so that you can create a professional help file in a minimal amount of time.

The Help Magician Window



Click on the desired area of the Help Magician Form for specific Help

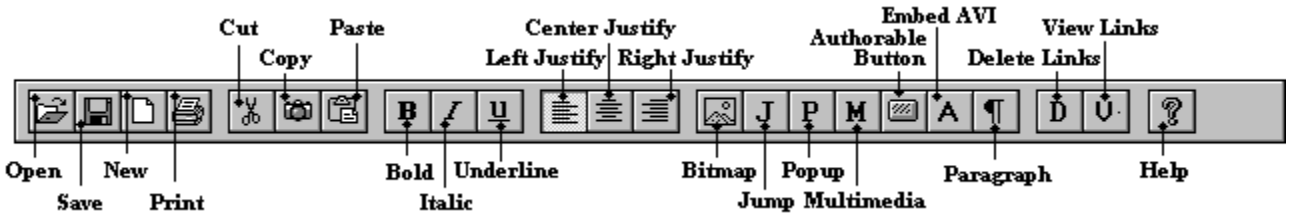
Scrolling & Nonscrolling Regions

Scrolling and nonscrolling regions are displayed in the Help Magician's editor just as they will be seen when the compiled help file is viewed with WinHelp. The nonscrolling region does not move as the user scrolls through the text in the lower, scrolling region.

Nonscrolling regions are converted from previous versions of the Help Magician. To create a nonscrolling region in a topic, select the **Format** menu and click on **Nonscrolling Region**. Click on **Nonscrolling Region** to delete an existing nonscrolling region.

To move between the two regions, click on the region or press **Ctrl+S** to move to the scrolling region or **Ctrl+N** to move to the nonscrolling region.

Tool Bar



Overview

The Tool Bar provides quick access to functions from the ***File*** Menu, the ***Format*** menu, the ***Links*** menu, and access to complete on line help. Each of these functions is described in detail in the appropriate topic.

File Commands

The ***Open*** button brings up a file dialog to open a Help Magician source file on disk. In Project mode, it brings up the Project Management form, from which you can open a source file that is a member of a project.

The ***Save*** button saves the current source file to disk.

The ***New*** button clears the current source and initializes for a new source file.

The ***Print*** button prints the current help text and graphics. Other report and print functions are available from the Filemenu.

Edit Commands

The ***Edit*** tool buttons provide quick access to standard Cut, Copy, and Paste functions.

Format Commands

Character formatting commands include Bold, Italic, and Underline.

Paragraph formatting commands include left justify, center justify, and right justify.

Link Commands

The ***Bitmap*** button is used to insert any one of the supported graphics file formats into the editor.

The ***Jump*** and ***Popup*** buttons are used to establish a link from the selected word or phrase to another topic or macro.

The ***Multimedia*** button is used to include multimedia elements in your help file.

Start The ***Add Button***

 button is used to insert an authorable button into the text portion of a help topic. This button is displayed only if the Help compiler version option is set to Win95 in Compiler Options.

The ***Embedded Avi***


button is used to insert embedded video (AVI) files into the text portion of a help topic. This button is displayed only if the Help compiler version option is set to Win95 in Compiler Options.

The ***Delete*** 

button deletes the link associated with the word or phrase at the cursor.

The ***View*** 

button brings up the appropriate form to modify the link associated with the word or phrase at the cursor.

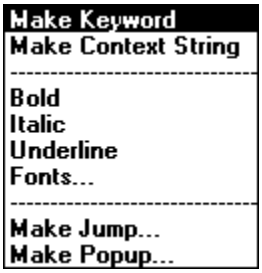
Paragraph Formatting

The Paragraph button brings up the paragraph formatting form and operates on the current or selected paragraphs.

Help

The ***Help***  button calls the Help Magician's own on line help system.

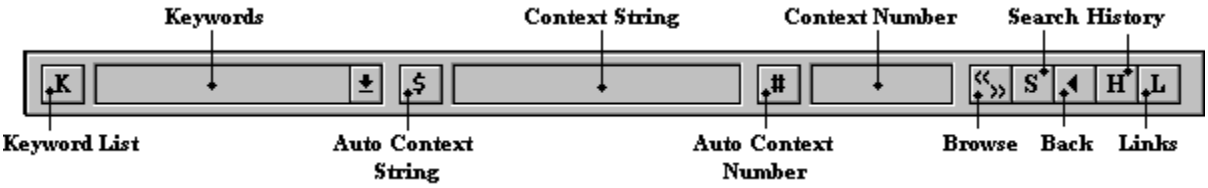
Pressing the right mouse button in the editor, when text is highlighted, pops up the **Editor Menu.**



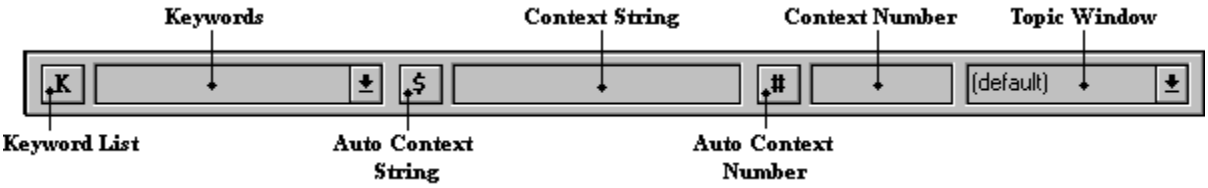
The functions available from this menu are duplicates of menu and toolbar functions and are provided as a convenience at the cursor location.

Utility Area

The Windows 3.1 Utility Area is shown here.



Start The Windows 95 Utility Area is shown here.



Windows 3.1 vs Windows 95

The difference in the **Utility Area** between the versions of Windows is found at the right end of the **Utility Area**.

In Windows 3.10 mode, there are five buttons: Browse, Search, Back, History, and Links. These buttons are replaced with a drop down box in Windows 95. This drop down box is used to select the **Topic Window** for the current topic. This is the (default) window in which the topic will be displayed unless otherwise specified by the secondary window in a jump.

The functions available to the five buttons that have been replaced are still available from the **WinHelp** bar and the **Links** menu.

Keywords

The Keywords drop down box displays the first keyword of each help topic as you page up or down. Click on the arrow to the right of the drop down box, or press Alt Up Arrow, to display all of the keywords for the current page. A new keyword can be entered, at any time. Make sure you press Enter after typing in a new keyword into the text box. Delete keywords by highlighting the desired keyword in the drop down listbox and pressing the delete key. Click anywhere in the editor or press the Esc key to return to the editor. To take advantage of full keyword management in Help Magician Pro 95, see Keyword Management under the Links menu.

Start Keywords are used by the search function of the Windows 3.1 Help system. Assign keywords to topics that are a reference to the topic by association. Use synonyms, whenever possible, and attempt to lead the user to the topic with associated words or phrases.

Start In the Windows 95 Help system, keywords are used slightly differently than in the Windows 3.1 Help System. They are now displayed in the Index tab of the Help Topics dialog box, which combined the contents and search functions under one dialog window. They are also used in linking topics by using the jump to keyword functionality. This alleviates the use of context strings and allows the end user to select from a list of possible topics to jump to. The

basic principle has not changed, however, which was to associate topics with common words and phrases to make them easier to find.

For more information see Keyword Management.

Keyword List

Clicking on the Keyword List button  to the left of the Keywords drop down box will pop up a table of Keywords already used in the help file. This is useful if a keyword is used on a number of pages and you aren't sure how it was phrased or spelled.

Double click on the Keyword you want to add or click on the Accept button and the Keyword will be added to the Keywords drop down box.

This functionality is provided as a convenience on the main form. A more comprehensive utility, **Keyword Management**, is available from the **Links** menu.

Context Strings

The Context String text box, to the right of the  button in the Utility Area, is used to enter a Context String for the current topic. When creating hot spots out of bitmaps with the SHED editor, context strings are required for the various parts of the bitmap that are sectioned off. If you are using project management while developing your help file, the context string you enter is stored into a project database at the time of saving your file. The project manager prevents other users from using your context string if they are also working on the same help file.

Context strings are referred to as Topic ID's in the Windows 95 environment. For sake of clarity, we will refer to them as context strings throughout this manual.

Temporary Context String

When you start a new file or add a new page, a temporary Context String is assigned to each topic consisting of the ***Unique File ID*** for the file (created from the system time and date, MMDDHHMM) and a four character representation of the page number when possible.

Mapping

Context Strings are normally constructed from the topic title or are imported along with an RTF file. Spaces are replaced with underscores and other illegal characters are re-mapped (Context Strings).

Uses

Normally, it isn't necessary to alter Context Strings. However, when editing a Bitmap with SHED.EXE, you will need the name of the Context String associated with the topic, or page, to be linked. A complete printout of the Context Strings is available by selecting Context Relations from the File/Print Menu. A description of how the Context Strings are constructed from the topic titles is provided on the Context String page. Help on SHED.EXE is available on SHED.EXE page.

You can create a link to several topics by using the jump to keyword feature.

Include Files

Another use for Context Strings would be for compatibility with symbolic references to help topics in an Include file, specified in the [MAP] section of the project (.HPJ) file. (The Help Magician writes the .HPJ file every time it writes the RTF file.) In this case, you may want to edit the Context Strings so that the references match those in your Include file. If this method is used, the name of the Include file must be entered in the Context String Map text box in the Compiler form available from the Options Menu. Select

"Use Existing File" from the three map file options. If a Context String Map file name is provided and the "Use Existing File" option is selected, the Help Magician will enter the file name in the [MAP] section of the project file each time the help file is compiled.

Auto Context String

Clicking on the  button will cause the Help Magician to automatically construct a Context String from the topic title and enter it into the Context String text box. You will have to confirm the replacement of the current Context String.

Context Numbers

Context numbers can be assigned to help pages, or topics, to provide a context sensitivity link between the help system and your application. For instance, if you wanted to provide help for a certain menu item, there would be a help topic page created for that menu item with a unique context number assigned to it. Your application would call WINHELP and pass the context number for help on the menu item.

The editor will not accept duplicate context numbers for different topics. The title page (or index page), as selected in Compiler Options, must have a Context Number. If a Context Number is not assigned at design time, the Help Magician will assign the next available Context Number in order to build the help system. If you are using project management while developing your help file, the context number you enter is stored into a project database at the time of saving your file. The project manager prevents other users from using your context number if they are also working on the same help file.

Auto Context Number

If the Context Number text box is empty, clicking on the  button will cause the Help Magician to automatically enter a Context Number. The number entered will be one greater than the highest Context Number in the help file.

Pressing the Esc key will restore the original contents of the Context Number text box and return focus to the editor.

Renumber All Context Numbers

The Context Numbers for the entire file can be entered or renumbered by selecting ***Renumber Context #'s...*** from the ***Tools*** menu. The purpose of this function is to eliminate the need to manually enter context numbers for all topics and/or to reorder the context numbers sequentially. Use this option only if you haven't already established context sensitivity between your application and the help file.

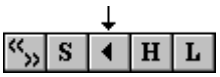
Browse

Clicking on the ***Browse***  button will bring up the Browse Group Definitions window. See Creating Browse Sequences for information on **Browse Sequences**.

Search

Clicking on the ***Search*** button  brings up a replica of the Search dialog used by WinHelp.

Back

Clicking on the ***Back*** button  returns the editor to the topic from which a link was executed.

History

Clicking on the ***History*** button 

pops up a replica of the History window used by WinHelp. The last 40 topics viewed are listed. Double click on a topic to go to that page in the editor.

Links From

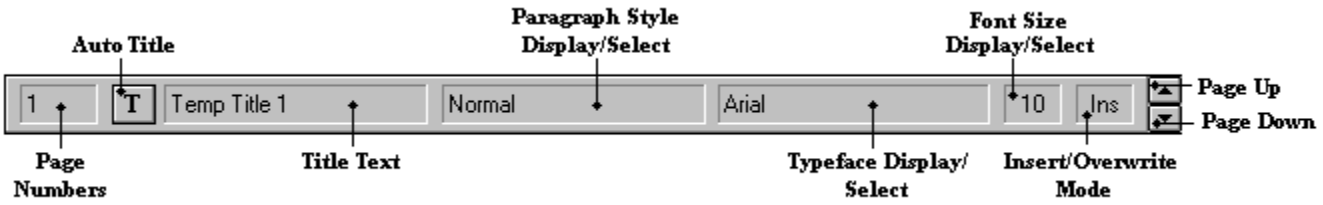
Clicking on the ***Links From*** button 

pops up a form displaying the links from other topics to the current topic.

See Also

Context Strings
Unreferenced Topics

Status Bar



Page Number

The text box in the lower left hand corner of the main window is continuously updated to reflect the current page number. A page number may also be entered into this text box to 'go to' any page. If a page number greater than the number of pages in the help file is entered, the editor will display the last page in the file. Be sure to press the Enter key after entering a page number to go to**.**

Title

The text box after the T button is continuously updated to reflect the title of the current page, topic, or opitonal hotspot link data (provided option for this is turned on in Environment Options). You can edit the title by tabbing to or clicking on the text box. Titles can contain any ASCII characters except colons.

Auto Title

Clicking on the ***Auto Title*** button T will cause the Help Magician to copy the contents of the Title text box to the top of the current page, automatically creating the topic heading from the title. If a paragraph style (Define Styles) named "Topic Heading" has been defined, the paragraph style will be assigned to the text.

Pressing the Esc key will restore the original contents of the Title text box and return focus to the editor.

Paragraph Style Display/Select

The ***Paragraph Style*** label displays the paragraph style assigned to the paragraph at the current text cursor. Paragraph styles can be assigned by clicking on this label or by selecting ***Paragraph Tag*** from the ***Format*** menu.

Typeface Display/Select

The ***Typeface*** label displays the typeface of the character at the current text cursor position. The typeface can be changed by clicking on this label or by selecting ***Character*** from the ***Format*** menu.

Font Size Display/Select

The ***Font Size*** label displays the font size of the character at the current text cursor position. The font size can be changed by clicking on this label or by selecting ***Character*** from the ***Format*** menu.

Insert/Overwrite Mode

The Insert/Overwrite Mode label displays the current typing mode. Clicking on this label toggles between insert and overwrite. In insert mode, characters to the right of the cursor are moved to the right as you type. In overwrite mode, typing replaces characters to the right of the cursor. The typing mode can also be changed by pressing the Insert key.

Page Up

The ***Page Up*** button  causes the editor to display the previous page, if there is one. You can also display the previous page by pressing the Ctrl PgUp key combination.

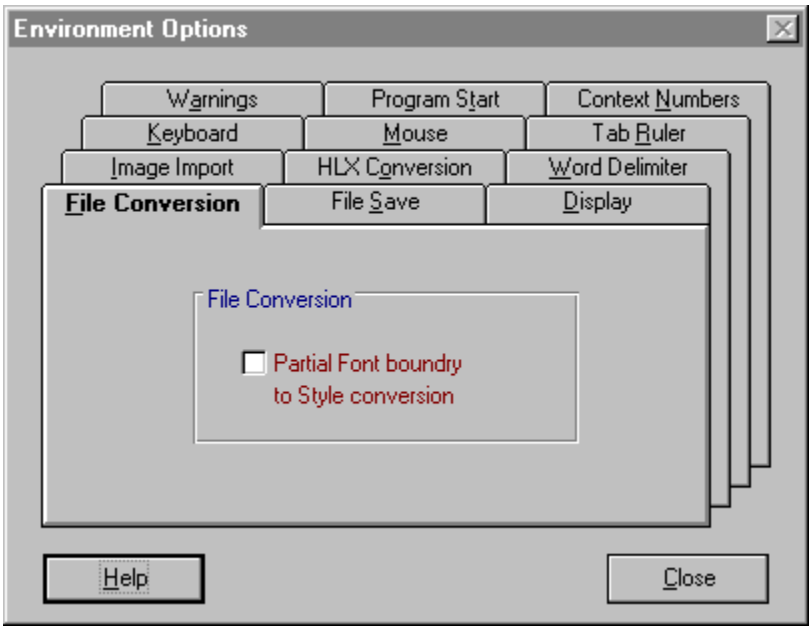
Page Down

The ***Page Down*** button  causes the editor to display the next page in the help file, if there is one. You can also display the next page by pressing the Ctrl PgDn key combination. The *Page Down* button is also used to create a new page. If you press ***Page Down*** on the last page of the help file, the editor creates a new page with a temporary title consisting of the ***Unique File ID*** for the file (created from the system time and date, MMDDHHMM) and a four character representation of the page number when possible. The page is not considered edited and will not be saved with the help file until the page has been edited. Another new page cannot be created until the most recent new page has been edited.

See Also
Define Styles

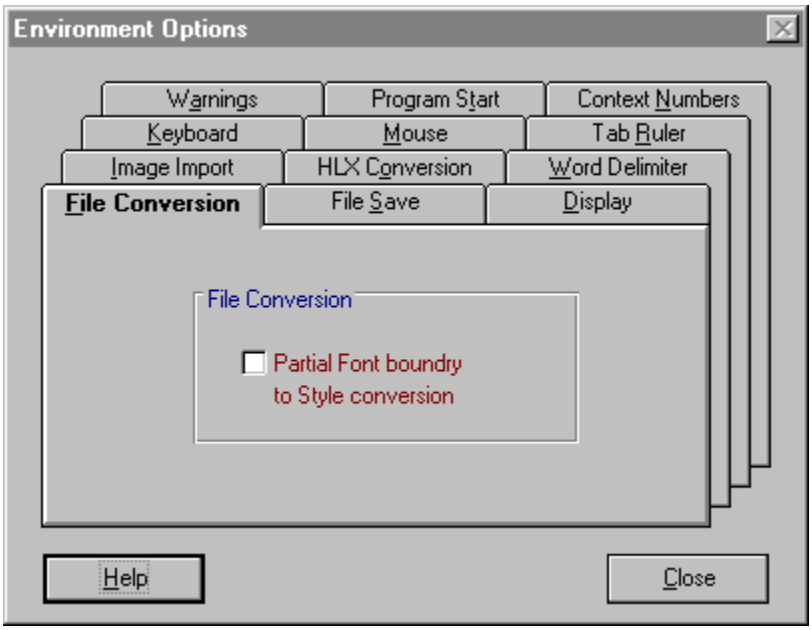
Environment Options

The Environment Options dialog is available from the Options Menu, Environment Options sub menu. The Environment Options dialog contains nine categories: Partial Font Boundary, File Save, Display, Image Import, HLX Conversion, Word Delimiter, Keyboard, Mouse, and Tab Ruler.



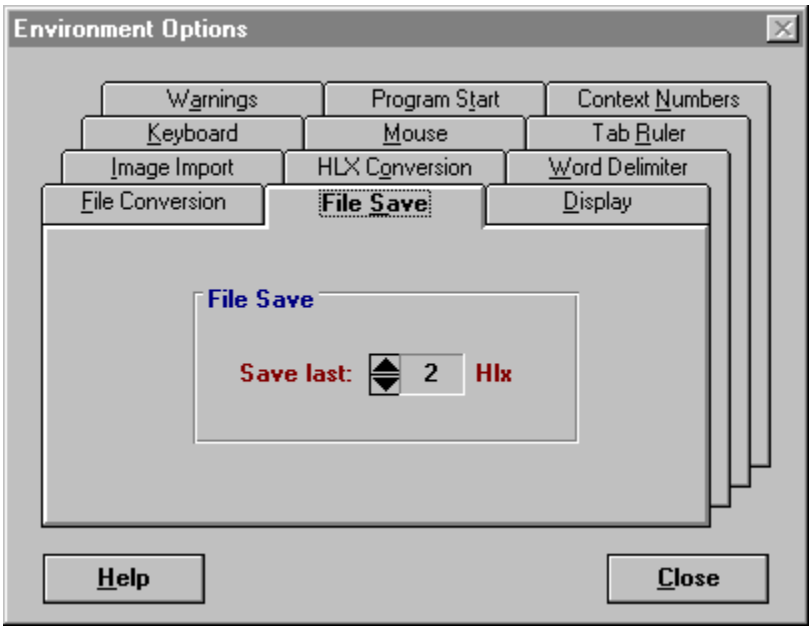
Click on the desired tab of the Environment Options bitmap for specific Help

Partial Font Boundary



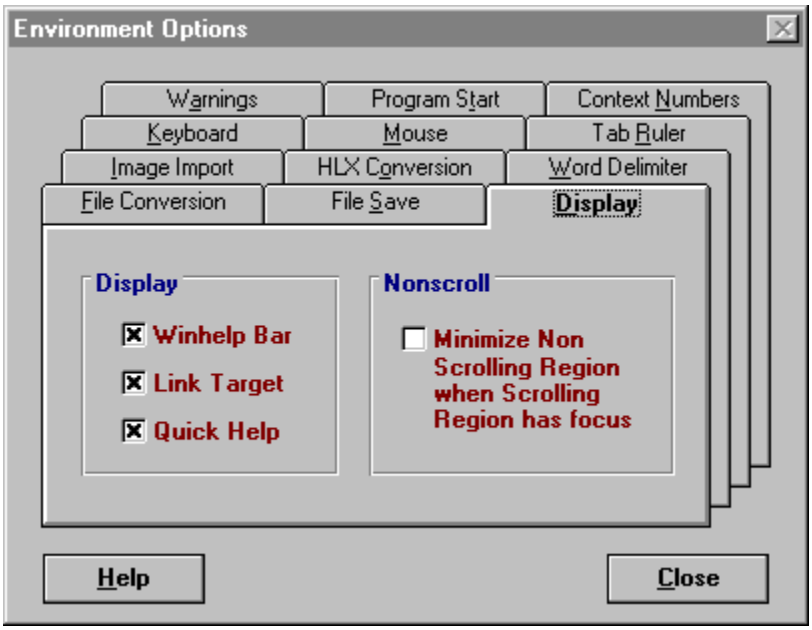
The Partial Font Boundary to style conversion option determines how font markers in older HLX files are interpreted. When possible individual font and paragraph formatting in Help Magician versions prior to 3.0 are converted to paragraph tags for use in 3.0/3.1. A paragraph is considered to have font formatting only if the font markers completely encompass the paragraph.

File Save



The File Save option specifies the number of HLX files to automatically back up. The range is one through ten. Click on the up arrow to increase the value and the down arrow to decrease it.

Display



Display

Winhelp Bar

This option determines whether the WinHelp Bar is displayed. If selected, the standard WinHelp Bar is displayed just above the editor window.

Link Target

If this option is selected, The "target" topic for the link under the cursor is displayed in the Status Bar at the bottom of the main window.

The page number is displayed in the page number text box preceded by a "**>**" symbol to distinguish it from the current page number.

The title of the target topic is displayed in the title text box preceded by one or more letters and a "**>**" symbol.

J> Jump
P> Popup
JF> Jump to another help file
PF> Popup to another help file
M> Jump to a macro
JS> Jump to another source file
PS> Popup to another source file
C> Jump to a mid topic Context String
JK> Jump to a keyword
JU> Jump to URL Address

Quick Help

This option determines whether the Quick Help on the main form is displayed as the mouse cursor is passed over the controls on the form. If Quick Help is enabled, a small yellow window appears near the

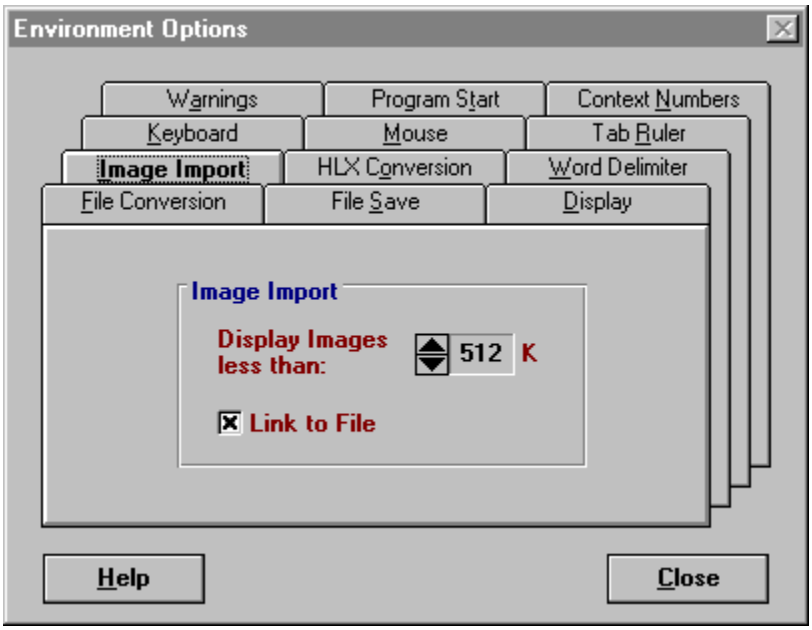
control with a word or phrase that identifies the control.

Nonscroll

Minimize Non Scrolling Region...

When the **Minimize Non Scrolling Region when Scrolling Region has focus** option is checked, the nonscrolling region of the editor will be reduced to one line to provide the maximum amout of editor space for the scrolling region.

Image Import



Display Images Less Than

This setting determines the maximum size of the images that will be actually displayed in the Editor and those that will be referenced by file name. The settings are from 0 to 1024k, and 16k increments. The setting is used when reading older HLX files, importing RTF files, and when entering bitmaps into the Editor, from the Edit menu or the Bitmap button on the toolbar. The larger this value, the longer it will take the editor to display the bitmap.

Display Image

Stores the image in memory and in the HLX file. This method uses the most memory and disk space but provides the fastest display of the image in the editor.

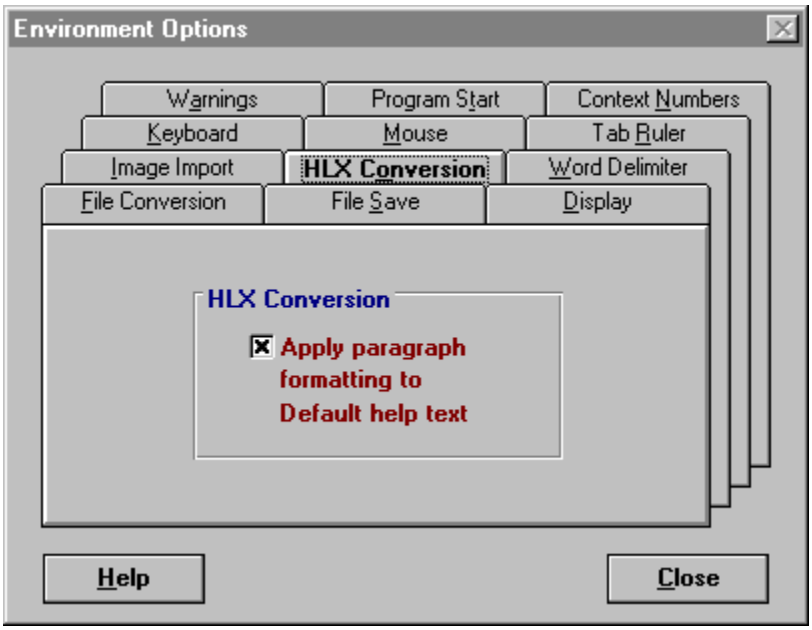
Display Filename

Stores the filename only in memory and in the HLX file. This method uses the least memory and disk space of all the display options.

Link to File

Stores only the the image filename in memory and in the HLX but displays the image in the editor. This method is a compromise between the two methods described above.

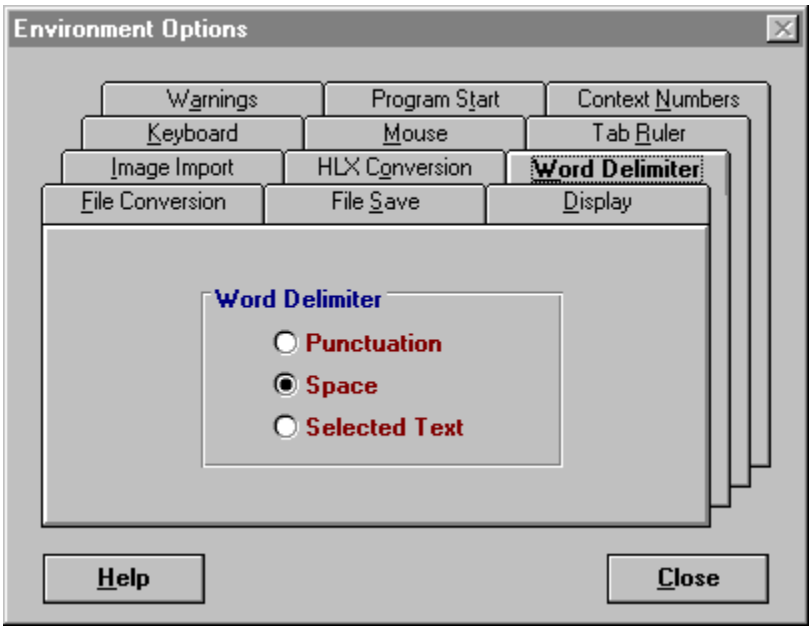
HLX Conversion



Apply Paragraph Formatting

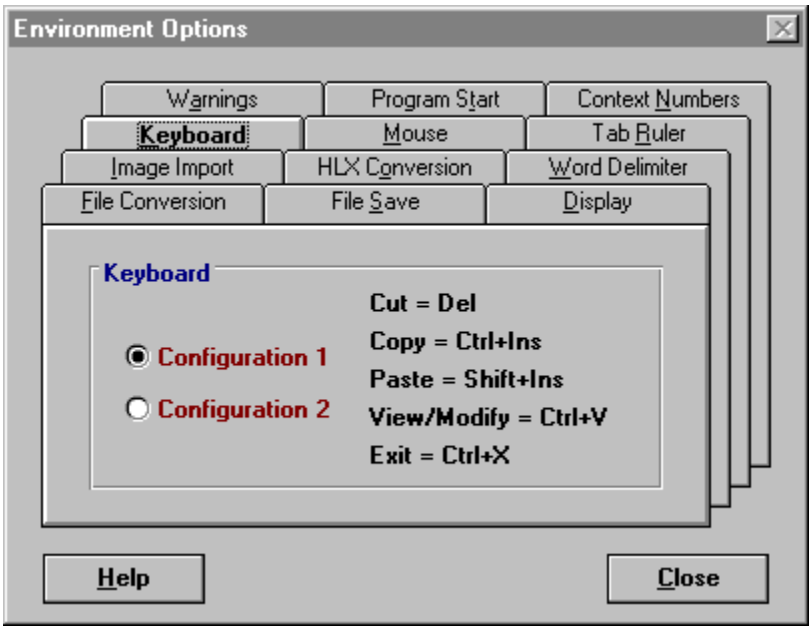
Since most of the text in the Help file will be default or normal style, you may or may not want the paragraph formatting to be applied to the paragraphs as they are assigned. If the applied paragraph formatting to default help text is unchecked only the font attributes will be assigned to the paragraphs as the normal or default style is assigned. If this box is checked, both font and paragraph formatting will be assigned to the paragraphs.

Word Delimiter



The Word Delimiter option determines how the Help Magician finds the beginning and end of a word of phrase when establishing a jump or a Popup. If the punctuation option is selected, the Help Magician will search from the current cursor position backward and forward until it finds a space or punctuation character and highlight that portion of the word or phrase. If the space option is selected the Help Magician will search backward and forward until a space is found and highlight that portion of the word or phrase. If the selected text option is selected the user must highlight the portion of the word or phrase that will be used as the jump or Popup text. A warning will be issued if this option is selected and no text is highlighted at the time a jump or a Popup is established.

Keyboard Configuration



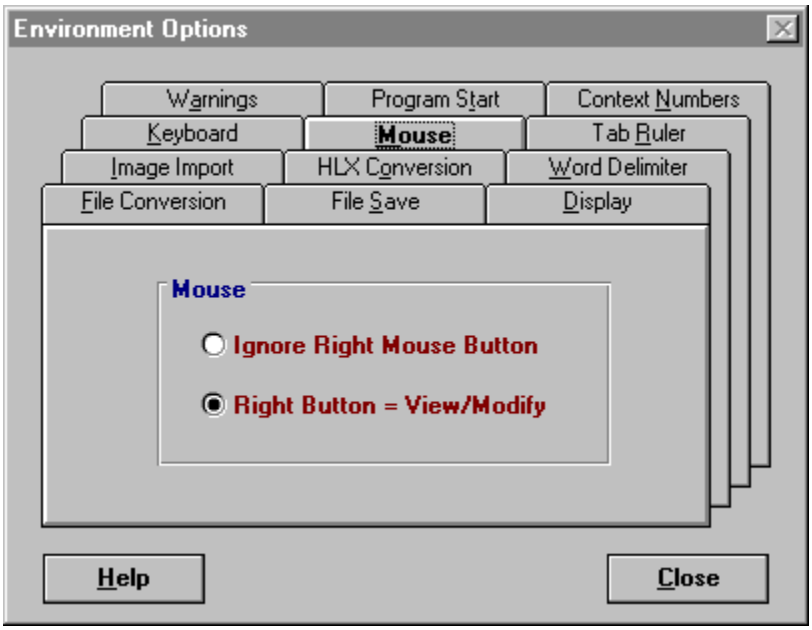
Keyboard Configuration

There are two possible keyboard configurations provided to maintain compatibility with earlier versions of the Help Magician and to offer the option to use the new standard keystrokes for cut, copy and paste.

Configuration 1 maintains compatibility with earlier versions of the Help Magician:
Cut = Del
Copy = Ctrl+Ins
Paste = Shift+Ins
View/Modify = Ctrl+V
Exit = Ctrl+X

Configuration 2 provides compatibility with the new cut, copy, and paste keystrokes:
Cut = Ctrl+X
Copy = Ctrl+C
Paste = Ctrl+V
View/Modify = Ctrl+M
Exit = Ctrl+R

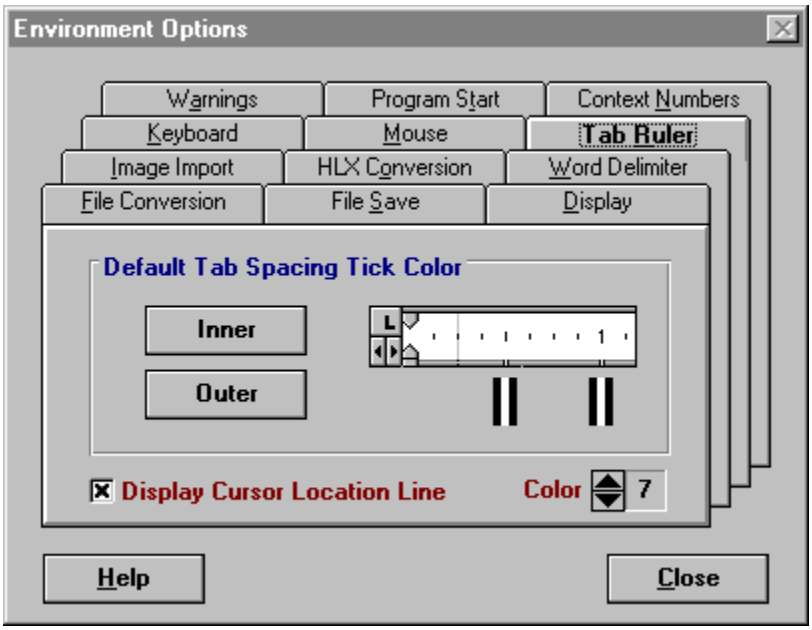
Mouse



Right Mouse Button

By default, the right mouse button is used to quickly bring up the appropriate form to modify a hot link or the attributes of an included image. The ***Ignore Right Mouse Button*** option is provided to avoid conflicts with certain mouse drivers that provide right mouse button functionality such as left button double click.

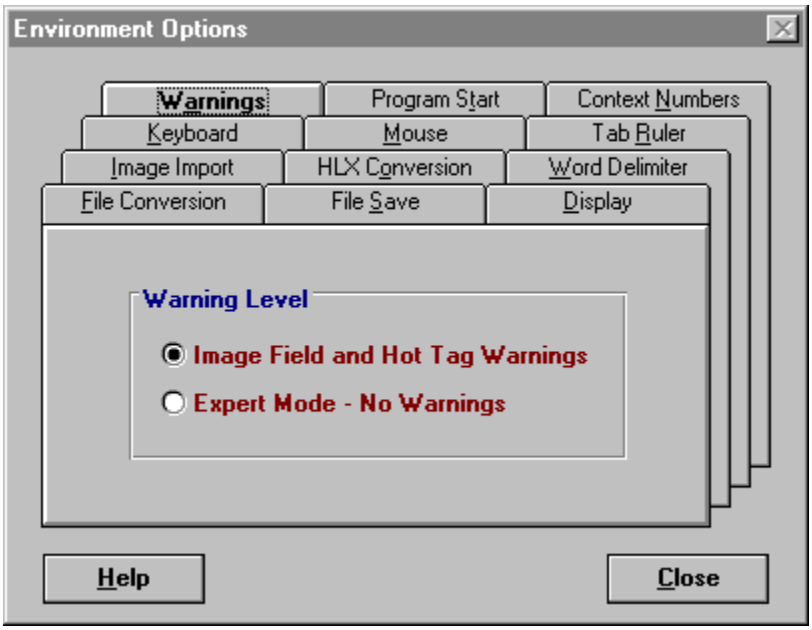
Tab Ruler



Default Tab Spacing Tick Color

The default tab spacing for a paragraph is displayed at the lower edge of the tab ruler. Three lines make up the ticks. By default, the outer lines are black and the inner line is white. This option allows setting the colors of the lines for better visibility on some monitors.

Warnings



Warnings

The **Warnings** tab is provided to select the level of warnings regarding hot spots and image fields.

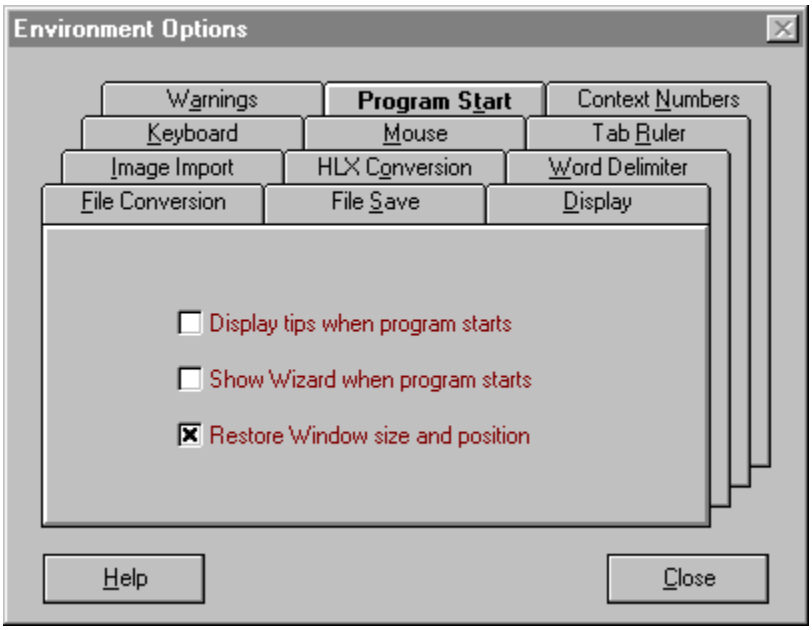
Hot spots and image fields must be separated by at least one space. Image fields cannot be modified in the editor. Their attributes must be modified from the Links menu, View/Modify (right mouse button) sub menu.

The **Esc** key will detach the cursor from a hot spot or image field.

By default, warnings are issued when you attempt to type (including the Return key) inside one of these fields. On hot spots, you have the option to edit the text, detach the cursor from the hot spot, or to cancel. On image fields, you can only select OK and the cursor will be detached from the image field.

If you become proficient at maintaining a space between these fields, select the **Expert** option to eliminate the warnings associated with them.

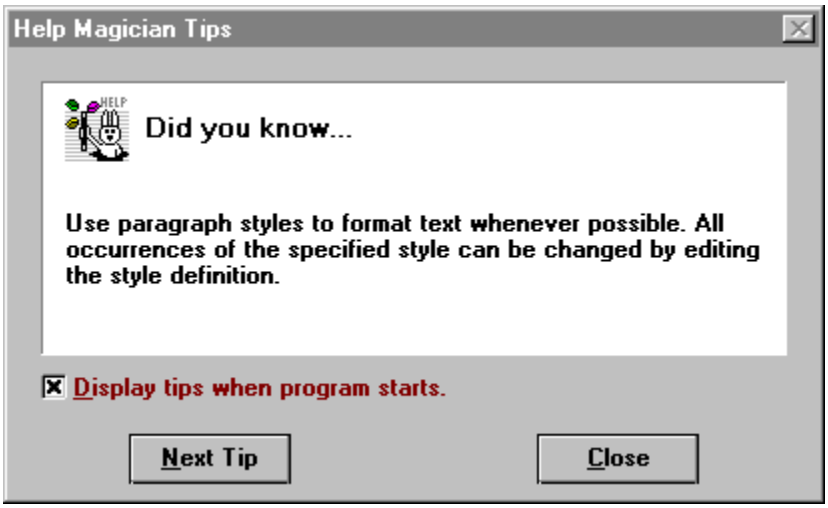
Program Start



Display tips when program starts

If this option is checked, the **Help Magician Tips** form pops up every time the program is started.

The Window shown below displays a useful hint about the **Help Magician** or help systems in general.



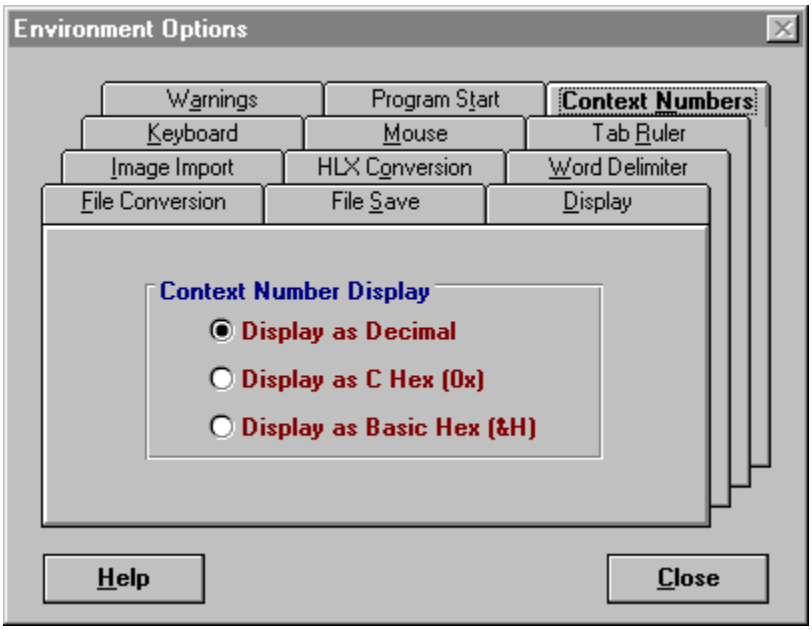
Show Wizard when program starts

If this option is checked, the **Help Wizard** pops up every time the program is started.

Restore Window size and position

If this option is checked, the Help Magician's main window will be displayed at its location when the program was last terminated. The size last of the window will also be restored.

Context Numbers

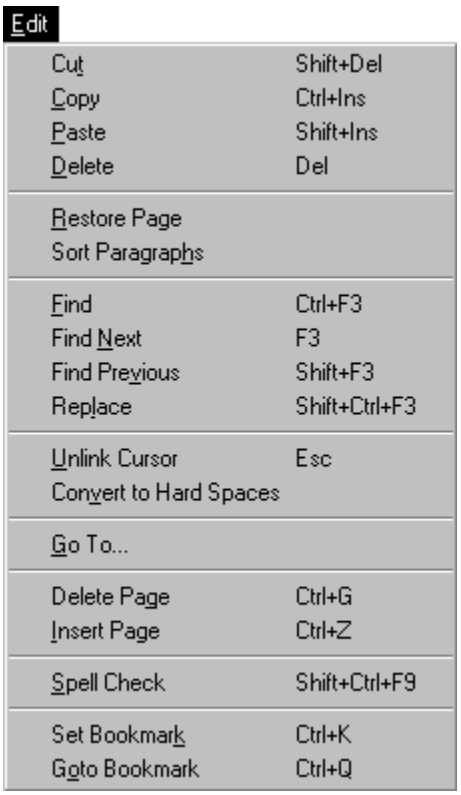


This option will show Context Numbers in the Context Numbers box just above the editor as decimal, C-style hexadecimal, or Basic-Style hexadecimal.

Managing Topic Text

Cut, Copy, Paste, Delete

The Help Magician Pro 95 Edit menu is shown here.



Click on the desired area of the Edit Menu for specific Help

Cut Text

Select Cut to delete selected text from the editor. Text is selected in the standard Windows manner, by pressing the left mouse button and dragging the mouse cursor over the desired text or, using the keyboard, holding the Shift key while moving the cursor over the desired text.

The hot keys, Shift+Del & Del or Ctrl+X (depending on the Environment Options settings), can be used to Cut selected text. Any text cut is copied to the clipboard.

Adding Text Using Other Windows Applications

You can use the paste command to add text from other sources using the DDE (Dynamic Data Exchange) capabilities of Windows. Copy the text from another application, switch to the Help Magician, place the cursor at the desired insertion point and paste the text.

Copy Text

Copying text is accomplished exactly the same as Cutting text except that the text is not removed from the editor.

The hot keys, Ctrl+Ins or Ctrl+C (depending on the Environment Options settings), can be used to copy selected text. Selected text is copied to the clipboard.

Paste Text

Paste will insert the text in the clipboard, if any, at the current cursor position. If text is highlighted, it will be replaced by the pasted text. A message box will notify you if the clipboard does not contain any text.

The hot keys, Shift+Ins or Ctrl+V (depending on the Environment Options settings), can be used copy selected text.

Delete Text

The Delete key removes text from the editor without copying it to the clipboard.

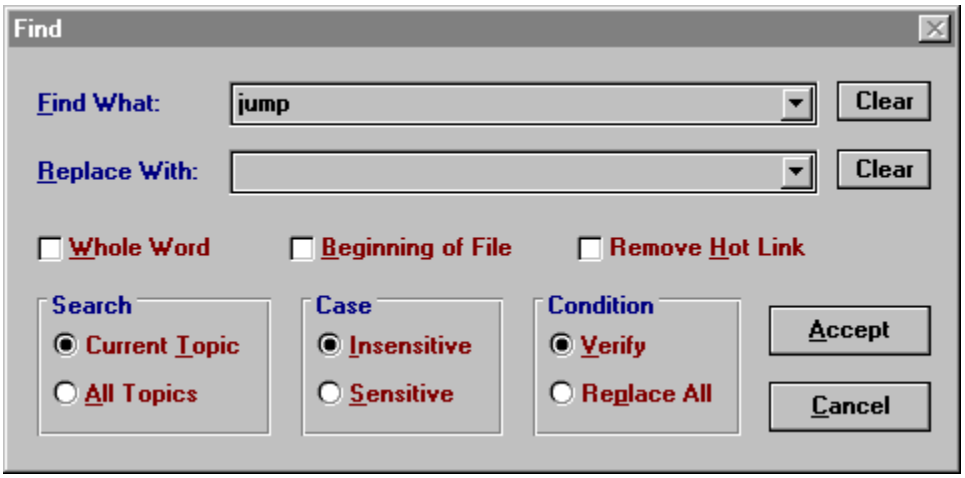
Sort Paragraphs

This function will sort paragraphs alphabetically. Select the paragraphs to be sorted by highlighting them (highlighting need not be precise) and select the **Sort Paragraphs** function.

Find

Find

Selecting Find from the Edit Menu will bring up the Find dialog.



Find What

The search text can be part or all of a word and it can be case sensitive or case insensitive, depending on the options selected below.

Whole Word

Click on the Whole Word check box if you want to limit the text found to a complete word, with a space or punctuation mark on either side. If the Whole Word box is not checked, the search text can be included in, or part of, a word.

Beginning of File

Click on the Beginning check box if you want to start the search from the beginning of the file instead of the current cursor position.

Current Topic

Selecting Current Topic will limit the search to the current help page, or topic.

All Topics

Selecting All Topics will cause the search to be performed on all help pages from the current page forward.

Insensitive

During a case insensitive search, the search text will be compared with the text being searched without regard for case. Both are converted to upper case before comparison.

Sensitive

In a case sensitive search, the text must match exactly, with regard to the case of each letter in the search text, entered in the Find What text box.

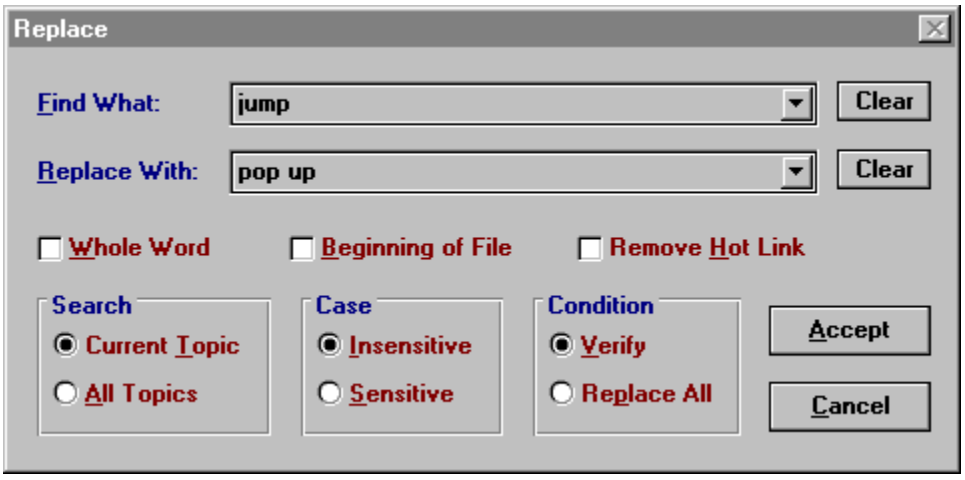
Find Next

Find Next will find the next occurrence of the search text, based on the arguments previously entered in the Find form. The search begins at the cursor and ends at the bottom of the last page. The search will

not wrap to the beginning of the help file. If Find Next is selected and there has not been an initial search, the Find form will pop up.

Find Previous

Find Previous will find the previous occurrence of the search text. If there has not been an initial search, no action will be taken. The search begins at the cursor and ends at the top of the first page in the help file. The search will not wrap.



Replace

Replace

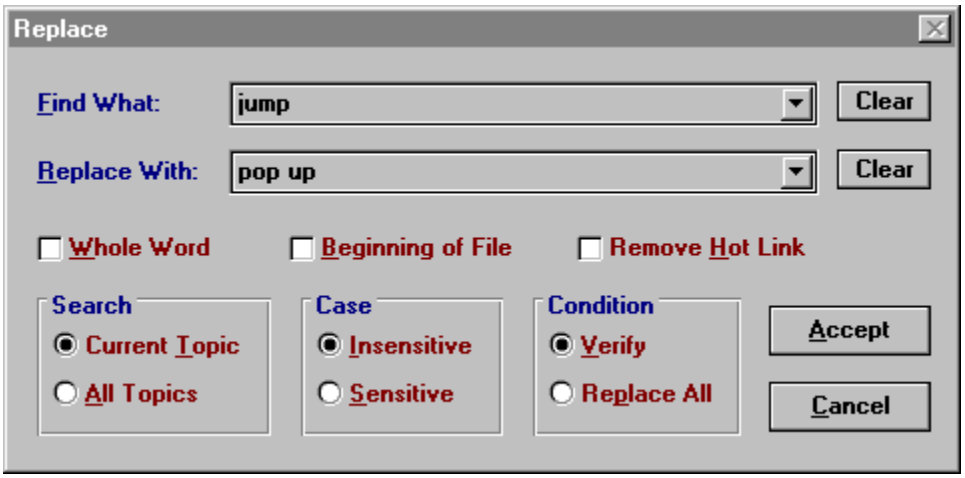
Because there are more options, Replace uses a different form than Find. In addition to the Find form, you must enter a replacement text string. Also, there are options to verify each replacement or to replace all occurrences. If you select verify, you will have the option, at each occurrence, to replace the text, leave the text as is, or to cancel the search and replace. In verify mode, Replace begins at the current cursor position on the current page. It does not wrap to the beginning of the file.

Verify

This option will cause the Help Magician to stop on every occurrence of the search text and request confirmation before replacing.

Replace All

This option will cause the Help Magician to scan the either the current topic or the entire help file, depending upon whether the Current Topic or All Topics option is chosen, and replace all occurrences of the search text, based on the Case Sensitivity and Whole Word options. The editor will return to the page you were editing when Replace is complete.



Unlink Cursor

As with any word processor, typing in the Help Magician's editor continues with the same font and paragraph attributes as the previous text. When a hot link is established, the attributes for the text are automatically set in the editor.

For example, if the text is green and underlined, in the case of a Jump, you don't want to continue to type with these attributes for normal text. If the cursor is just to the right of the hot link, you will continue to type with these attributes and the text will become part of the hot link. The same is true for an image field.

The Insert/Overwrite indicator, in the bottom right corner of the main window, displays the status of the cursor relative to hot spots and image fields. If the indicator is red, the cursor is inside of an image field. If the indicator is green, the cursor is inside of a hotspot. If the indicator is yellow, the cursor is inside both fields.

Selecting the ***Unlink Cursor*** sub menu or pressing the Esc key will detach the cursor from the hot link or image field and its attributes.

Convert to Hard Spaces

When **Center** or **Right** aligned tabs are used, the spaces in the text must be hard spaces. A hard space can be entered from the keyboard with the **Shift+Space** key combination.

If text has already been entered, the spaces can be converted to hard spaces by highlighting the text and selecting ***Convert to Hard Spaces*** from the ***Edit*** menu.

Goto

Goto...

The Goto... sub menu offers ten options. You can goto a Topic by page, title, Context String, or Context Number. You can goto the next Jump, Popup, Mid Topic Context String, or unresolved hot link. You can also goto the next occurrence of a paragraph style. You can also goto a page by entering the page number in the text box to the right of the Page: label, in the Status Bar.

Goto Page	
Topic Title	F4
Context String	Ctrl+F4
Context Number	Shift+Ctrl+F4
Next Jump	F2
Next Popup	Shift+F2
Next Mid Topic	Shift+Ctrl+F2
Next Unresolved	Ctrl+F2
Paragraph Style	Ctrl+F5
Next Paragraph Style	Ctrl+F6
Next Image	

Goto Page

When you select the ***Goto Page*** sub menu, the text cursor moves to the text box in the lower right hand corner of the screen and the current page number is highlighted. Enter the number of the page that you want to go to, press Return and the editor will display the selected page. If the page number entered is greater than the number of pages in the file, the editor will display the last page.

 A shortcut to the Goto Page sub menu would be to click in the page text box with the left mouse button.

Goto Title

When you select Goto Title from the Goto... sub menu, a form will pop up with a table of the topic titles in your help system. Double click on the title to goto that topic or select the title and click on Accept. You can preview any topic by highlighting the title and clicking on the Preview button.

Goto Context String

When you select Goto Context String from the Goto... sub menu, a form will pop up with a table of the Context Strings in your help system. Double click on the desired Context String or select the Context String and click on the Accept button. You can preview any topic by highlighting the Context String and clicking on the Preview button.

Reminder: Context Strings are refered to as Topics ID's in Win95 environment.

Goto Context Number

When you select Goto Context Number from the Goto... sub menu, the Page: text box will receive the focus and the cursor will be preceded by the letter 'C'. Enter the context number and press return and, if the context number exists on any page, the editor will goto that page.

 The letter 'C' can be entered in the Page text box at any time to goto a page by context number.

Goto Jump, Popup, Mid Topic, Unresolved

You can goto the next hot link by selecting the Jump, Popup, Mid Topic, or Unresolved sub menu. A beep will sound if there are no more links of the selected type.

The purpose of the Unresolved sub menu is to locate unresolved links caused by importing an RTF where the destination for a hot link was not available.

Goto Paragraph Style/Next Paragraph Style

You can locate occurrences of paragraph styles throughout the help file by selecting these options from the Edit/Goto menu. After selecting the paragraph style, the goto Next Paragraph Style with locate successive occurrences of the style.

Goto Next Image

Goto Next Image will go to the next occurrence of an embedded image, an image filename, or a link to file image.

Insert Page

Insert Page will insert a page, at the current page number, moving all pages, from this page number, forward one page. The new, inserted page, will have the same attributes as a new page that is created by clicking on ^PgDn while on the last page. The title will be Temp Title plus the page number.

Delete Page

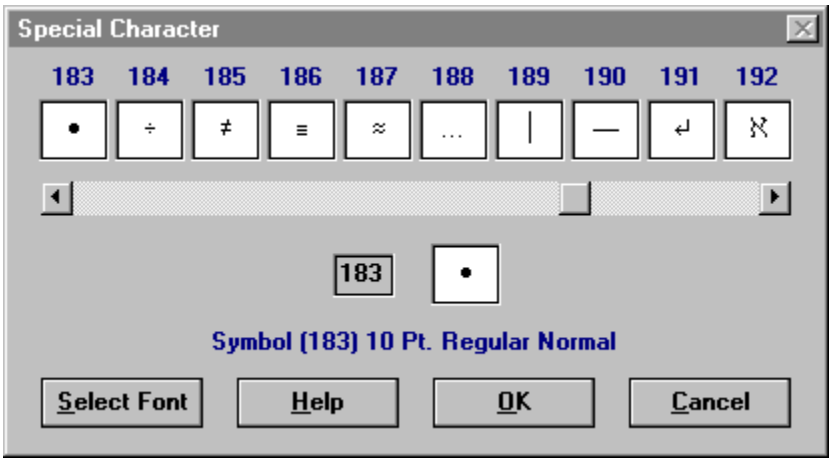
This will delete an entire page from the current help file. If there are links in the text on this page, they also will be deleted. If other pages are linked to the page to be deleted, the pointers to this page will be deleted and the links will be converted to "unresolved" in the database. If the last page is deleted and was the Help Index Page, then the first page will default to the new Index Page, otherwise the new Help Index Page will be the next consecutive page.

Bookmarks

You can set and goto any one of ten bookmarks anywhere in the help file. Press and hold the Ctrl key and press the "K" key to set a bookmark. Press and hold the Ctrl key and press the "Q" key to goto a bookmark.

Inserting Special Characters

You can insert special characters from symbol or other decorative fonts by selecting **Special Character** from the **Insert** menu to bring up the Special Character form. This is the same form used to define characters for bulleted paragraph formatting.



Special Character Form

Ten characters are displayed at the top of the form with the ASCII value of each character above the sample. You can scroll through the characters by moving the horizontal scroll bar with the mouse.

The currently selected character is displayed in a label in the center of the form, to the right of the ASCII value text box. Clicking on one of the ten characters at the top of the form selects that character.

You can also enter an ASCII value (0 - 255) in the text box to the left of the selected character label to select a character.

Select Font

Click on the Select Font button to set the typeface, point size, style, effects, and color for the character.

Done

When you have set all the attributes for the character, click on OK. The character will be inserted into the editor at the current cursor location.

Inserting a Horizontal Rule

The horizontal rule is included for use with WEB page authoring but it can be used in a help system just as well. Click on the **Insert** menu, **Horizontal Rule** sub menu and the rule will be inserted into the editor at the current cursor location.

Paragraph Styles

Defining Styles

Define Styles

Selecting Define Styles brings up the Font Style Definition window. The figure shows the default list of defined styles. Up to 600 paragraph styles can be defined for the Help system.

Default Font

The first style on the list is the default paragraph style for the entire help system. All text will be viewed in this style unless it is specifically marked as another style. The Default Help Text style cannot be deleted and the name cannot be changed. As you cursor through the list, or click on a style name, a sample of the actual font is displayed at the bottom of the window.

Other Defaults

Other Paragraph Styles are provided. They may be re-defined or deleted as desired.

Delete

Clicking on the Delete button will remove the currently highlighted font style from the list of defined font styles. The Default Help Text style cannot be deleted.

Style Name

To change the style name, tab to or click on the Defined Style Name text box and edit the name.

Explicit Re-assignment

This check box is not enabled until a style has been changed. Checking this box causes the re-assignment of fonts to ignore exceptions to the existing style. See the note below.

When a style is assigned, an effort is made to preserve embedded exceptions to the existing style such as a bold or italic word or color assigned to a word or phrase. If you would rather that the entire paragraph be re-formatted according to the selected style, click on the style with the right mouse button or press Shift+Return.

When assigning styles with the hot keys (Ctrl+0-9 and Alt+0-9) add the Shift key to the hot key for explicit re-formatting.

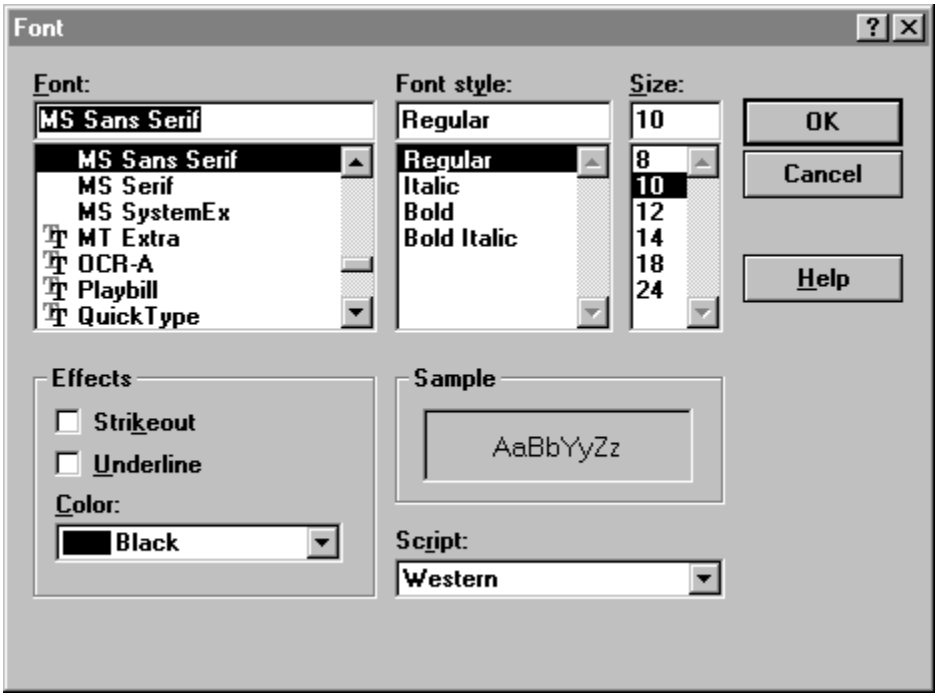
See Also

- Alignment
- Applying Styles
- Borders
- Bullets
- Character Formatting
- Defaults
- Indents
- Load/Save Styles
- Merge Styles
- Modify Character Attributes
- Paragraph Formatting
- Spacing
- Tabs

Modify Character Attributes

Modify Character Attributes

To modify the character attributes of a defined style, click on the ***Character*** button or double click on the style name in the list box. This will bring up the Select Font Properties window.



Font Face

The list of typefaces provided in the list box include all of the screen fonts on your system.

Font Style

Select whether you want the font to be Regular, Italic, Bold, or Bold Italic.

Font Size

The point sizes available for the selected Font are listed in the Size list box.

Effects

Check the appropriate box in the Effects frame for Strikeout or Underline.

Font Color

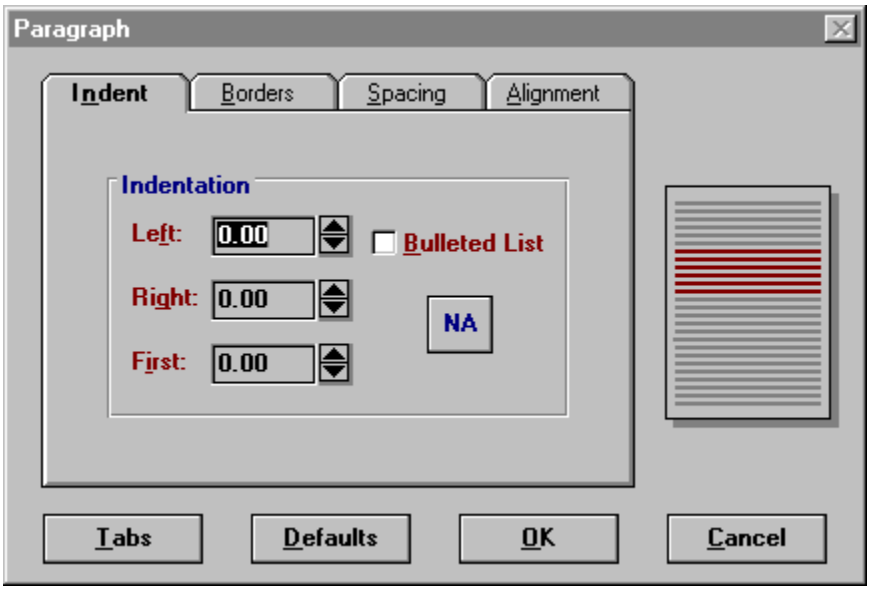
Select the font color from the 16 colors in the drop down box.

Modify Paragraph Attributes

To modify the paragraph attributes of a paragraph style, click on the

Paragraph ¶ button. This will bring up the Paragraph Formatting form.

Indents



Left Indent

To indent a paragraph from the left margin, enter the desired indentation in the Indent text box labeled "Left" and click on OK.

Right Indent

To indent a paragraph from the right margin, enter the desired indentation in the Indent text box labeled "Right" and click on OK.

Hanging Indent

To produce a hanging indent, set both the Left and First indents. The Left indent applies to the entire paragraph and the First indent applies only to the first line.

The First indent will be a negative number to move the first line of the paragraph back towards the left margin relative to the indentation of the entire paragraph. Use the Sample window on the right side of the Paragraph formatting form to visualize the effect of the settings.

Delimiter

There must be a delimiter between the Left justified text and the hanging indent text. If a single word numbered listing or hanging indent is created, a space will serve as the delimiter. If multiple words are used, the delimiter must be a Tab character.

During the writing of the RTF file, you will receive a warning message for every line that is missing a proper delimiter. This is not fatal, however, just select OK and the RTF writer will continue.

Numbered Lists

Numbered Lists can be created with hanging indent paragraphs. Enter a Tab character after the number.

1 Item one.
2. Item two.
3) Item three.
A.) Item A

Words

Numbered lists can also be created using words or phrases in place of the number character. If words are used, the rules are the same as for lists with numbers. If phrases are used, there can be spaces between words but the indent must be separated from the phrase by a Tab character.

In order to produce indented non-numbered paragraphs after a numbered paragraph, you may intentionally leave out the number before the delimiter to achieve this effect:

1. TAB This is an example of the first
paragraph that is word wrapped.

TAB This is the next paragraph with
no numbers or text before
the space or tab delimiter.

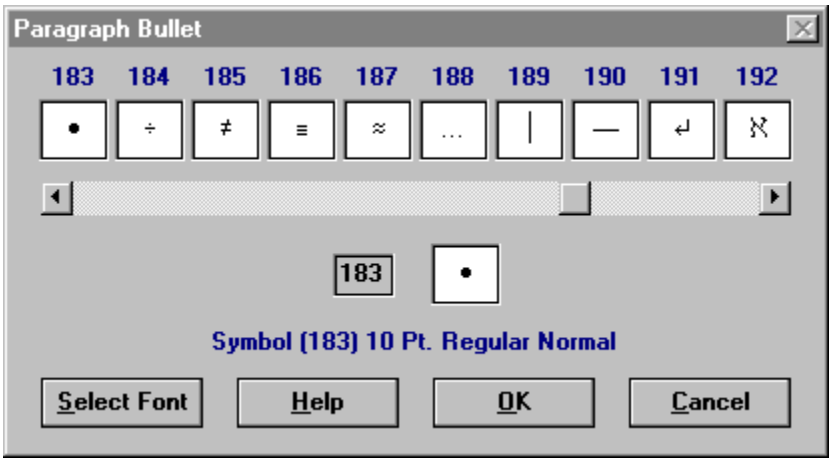
2. TAB This is the next paragraph with
a delimiter.

Bullets

To create a bulleted paragraph style, check the Bulleted Listing box by clicking on it with the mouse. This enables the Bullet

selection button 

. Click on the Bullet selection button to bring up the Paragraph Bullet form.



Paragraph Bullet Form

Ten characters are displayed at the top of the form with the ASCII value of each character above the sample. You can scroll through the characters by moving the horizontal scroll bar with the mouse.

The currently selected character is displayed in a label in the center of the form, to the right of the ASCII value text box. Clicking on one of the ten characters at the top of the form selects that character.

You can also enter an ASCII value (0 - 255) in the text box to the left of the selected character label to select a character.

Select Font

Click on the Select Font button to set the typeface, point size, style, effects, and color for the bullet.

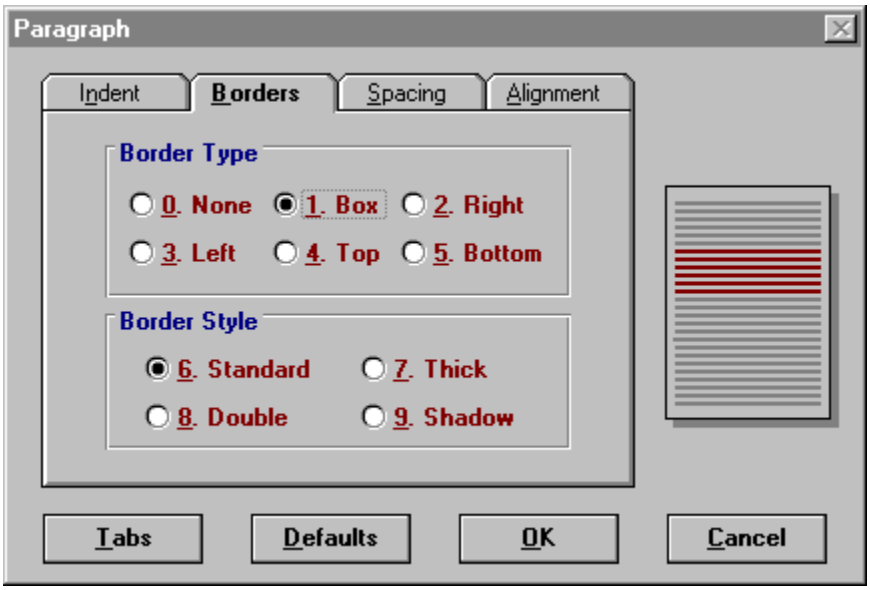
Done

When you have set all the attributes for the bullet, click on OK. When you return to the Paragraph form, the selected bullet will be displayed on the Bullet selection button.

Borders

Borders Tab

Click on the Borders tab to access the paragraph border attributes.



Border Type

Click on the appropriate radio button to select the border type. The Border Style options will not be available until a Border Type has been selected.

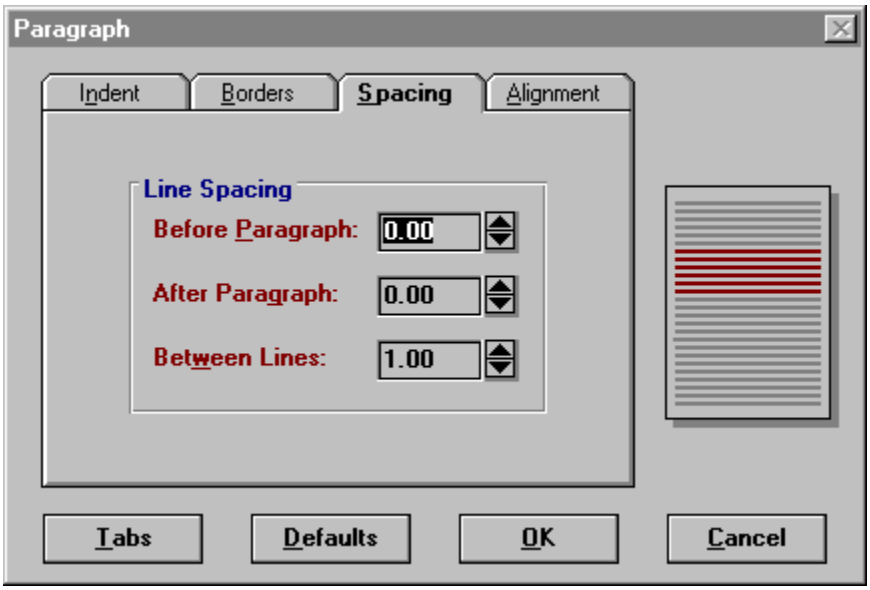
Border Style

Click on the radio button for the type of border. Click on OK to accept your choices.

Spacing

Spacing Tab

Click on the Spacing tab to access the paragraph spacing attributes.



Before Paragraph

Click on the up or down arrows to the right of the Before Paragraph text box to increment or decrement the line spacing before the paragraph or enter the value into the text box (up to 2 decimals). The sample window on the right side of the form reflects the value entered.

After Paragraph

Click on the up or down arrows to the right of the After Paragraph text box to increment or decrement the line spacing after the paragraph or enter the value into the text box (up to 2 decimals). The sample window on the right side of the form reflects the value entered.

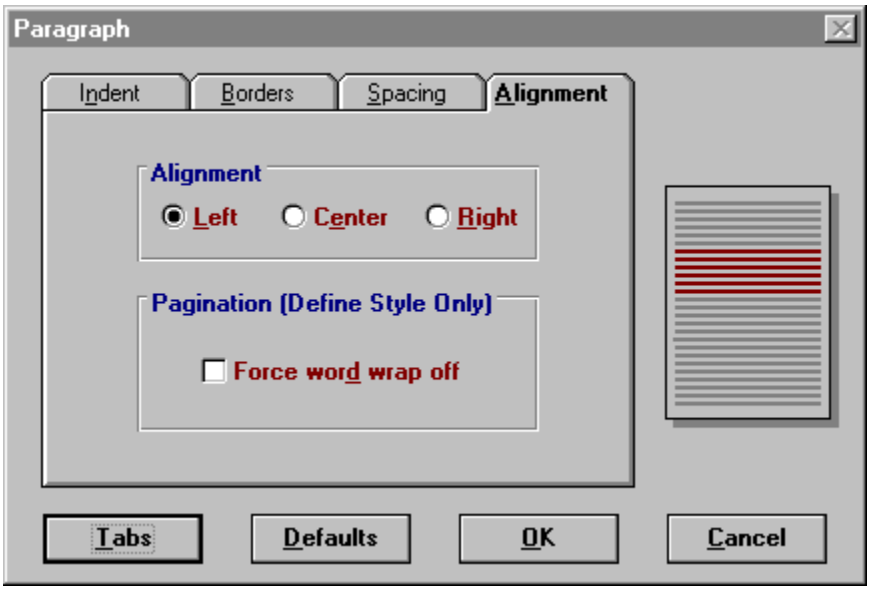
Between Lines

Click on the up or down arrows to the right of the Between Lines text box to increment or decrement the spacing between lines or enter the value into the text box (up to 2 decimals). The sample window on the right side of the form reflects the value entered.

Alignment

Alignment Tab

Click on the Spacing tab to access the alignment attributes.



Alignment

Click on the appropriate radio button to set the alignment for the paragraph text.

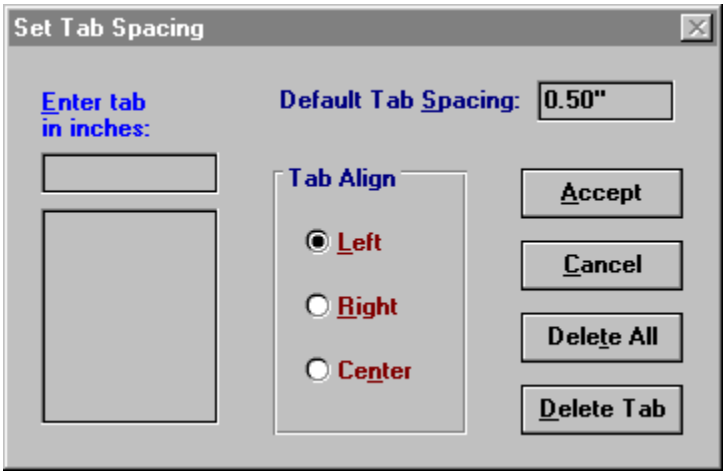
Pagination

Check the ***Force word wrap off*** check box to eliminate word wrapping of the text in the paragraph when viewed with WinHelp. If the width of the WinHelp window is less than the width of the paragraph, the text will have to be scrolled to be viewed.

Check the ***Non-scrolling region*** check box to define the paragraph as a non-scrolling region, or banner. Styles defined with this attribute must be at the top of the topic and there cannot be any type of paragraphs defined above a non-scrolling region paragraph. When viewed with WinHelp, this area will remain in place and will not scroll as the rest of the text in the window is scrolled.

Tabs

Setting Tabs
To set tabs for a paragraph, select the "Tabs" button. This brings up the ***Set Tab Spacing*** form.



Set tabs by setting a tab alignment of Left, Right or Center; typing in a tab position, in inches, in the text box and selecting the Enter key on the keyboard. Duplicate tabs will be ignored as well as tabs greater than eight inches. To delete a tab or all tabs, select the tab and select "Delete tab" or 'Delete All' button, respectively. To set the default tab positions to something other than half an inch, change the setting in the "Default tab spacing:" text box. Select Accept to return to the Paragraph form. If tabs have been set the color of the Tabs button will change from black to red.

Defaults

Pressing the ***Defaults*** button will restore all of the paragraph attributes to their default settings.

Load/Save Styles

Save Styles

You can save your Font Style definitions to a file, to be used in other help files. Select Save Styles from the Format Menu and assign the file a name in the file form. The default extension for styles is .STY.

Load Styles

When you start a new help file, you can load styles that were defined for another help system. The file form will prompt for files with the default extension for styles .STY. You can only load defined styles in a new help file.

See Also

Merge Styles

Merge Styles

Font Styles, defined in another help file can be merged with the Font Styles in the current help file.

As each Style is read in from the Paragraph Style (.STY) file, its attributes are compared with the attributes of the Font Styles in the current help file. Font Styles with duplicate attributes will not be merged.

Font Styles with duplicate Style names but different attributes will be merged and the Style name will be appended with a -A, -B, etc. as necessary.

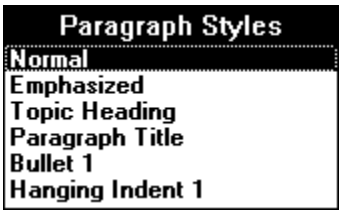
Applying Styles

Selecting Paragraphs

To apply a paragraph style to a paragraph, the cursor can be any where in the text and highlighting is not necessary for single paragraphs. If more than one paragraph is to be formatted, highlight from anywhere in the first paragraph to anywhere in the last paragraph. The Help Magician automatically finds the beginning and end of paragraphs. A carriage return is considered to be the end of a paragraph.

Defined Style List

The list of Defined Paragraph Styles can be accessed from the Format menu, Paragraph Tag sub menu or by clicking on the style label in the Status Bar at the bottom of the main window.



Select Style

Select the style for the paragraph or paragraphs by clicking on the desired style or by moving to the desired style with the cursor keys and pressing Return.

When a style is assigned, an effort is made to preserve embedded exceptions to the existing style such as a bold or italic word or color assigned to a word or phrase. If you would rather that the entire paragraph be re-formatted according to the selected style, click on the style with the right mouse button or press Shift+Return.

When assigning styles with the hot keys (Ctrl+0-9 and Alt+0-9) add the Shift key to the hot key for explicit re-formatting.

Paragraph Formatting

Paragraph formatting can be applied independent of the paragraph styles. Select the paragraph or paragraphs just as you would to apply a style and select Paragraph from the Format menu.

Select the attributes for the paragraph in the Paragraph form and click on OK to apply the formatting.

Character Formatting

Character formatting can be applied independent of the paragraph styles. Select the text to be formatted by highlighting it with the mouse and select Character from the Format menu.

Select the attributes for the text in the Font dialog and click on OK to apply the formatting.

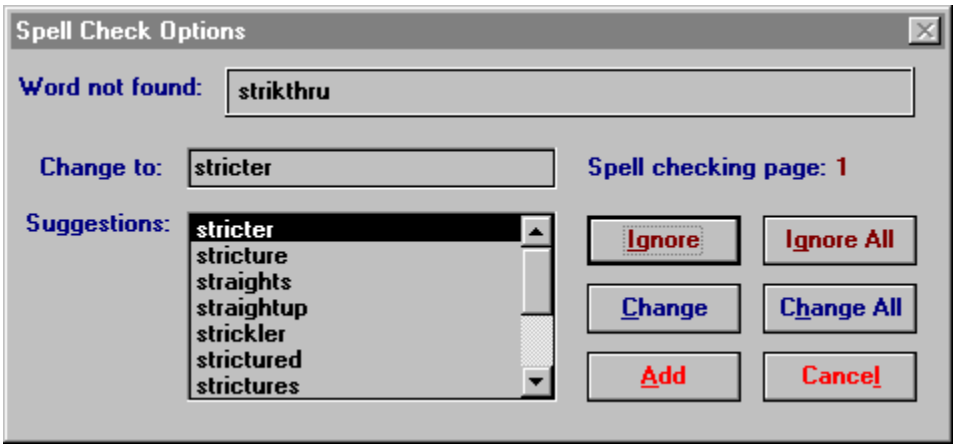
Spell Checking

Spell Check

The Spell Check option will check the spelling in your help text from the current page, at the current cursor position, to the end of the file.

Replace Options

The Replace Options window pops up when a Spell Check session is initiated. The page number being scanned is displayed after the Spell checking page: label.



Word not found:

If the spell checked finds a misspelled word, it is displayed after the Word not found: label on the Replace Options form. Note that in the tutorial file shipped with the Help Magician, the word "occurrence" is misspelled.

Ignore

Press the Ignore button to ignore this single occurrence of the word.

Ignore All

The Ignore All option will cause the spell checker to ignore this and all successive occurrences of the word.

Change

Pressing the Change button replaces the misspelled word in the editor with the word in the text box above the suggested words list box. When you click on a word in the suggested words list box, it is copied into the text box. You may also type your own replacement for the misspelled word in the text box.

Change All

The Change All option will replace the misspelled word and all successive occurrences of the word.

Add

This will add word after the Word not found: label to the "User Dictionary". This and all successive occurrences of the word will be ignored.

User Dictionary

The User Dictionary is named "DICT.U" and can be found in your Windows directory. If you need to remove words from the dictionary, edit the file with an ASCII editor. The words in the User Dictionary must

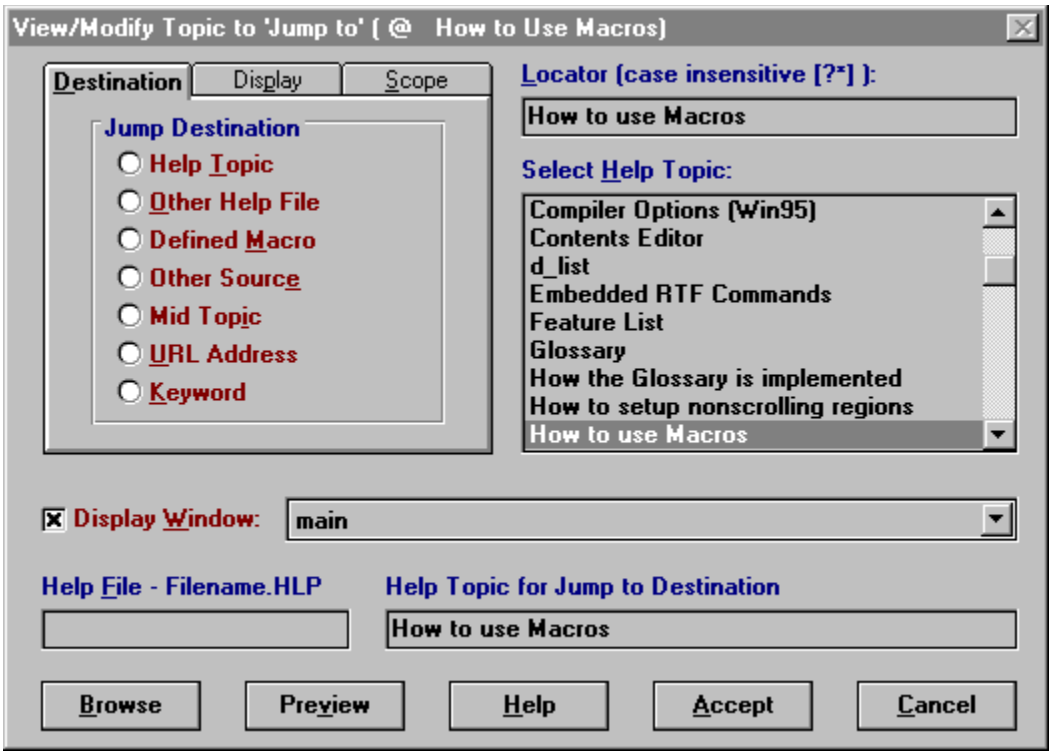
be upper case first letter and separated by commas. The last word must be followed by a comma.

Jumps and Popups

Jump

Select Jump from the Marker Menu to set up a Jump, or link, between the selected word or phrase and another help topic in the current help file, a topic in another help file, or a macro. When you select Jump, the Jump Destination form will pop up. There are options for the display of the Jump text, secondary windows, and the Scope of the Jump, all explained in detail in the following sections. Some of the options on the form are enabled and disabled depending on the Jump Destination option selected.

Destination



Locator

There is a text box above the list box used to display help topics and macros. It is used to quickly locate items in the list by typing some of the characters in the help topic or macro. The list will automatically scroll to the item most closely matching the entry in the text box.

Another Topic

To establish a Jump to another topic in the same help file, select the topic from the list box labeled, Select Help Topic and click on the Accept button. In the compiled help system, the user will Jump to the selected page when they click on the word or phrase. The current page is not included in the list because it would not be a valid Jump link.

Another Help File

To Jump to a topic in another help file, first select the Other Help File radio button. This will enable the Help File and Context String text boxes and the Browse button. Enter the name of the help file in the Help File text box and the enter the Context String of the topic to Jump to in the Context String text box. If necessary, refer to a copy of the Context Relations printout for the other help file.

Macro

To Jump to a macro, first select the Defined Macro radio button. Note that this button will read "No Macros available" and will not be enabled if macros have not yet been defined. (Defining Macros)

If there are macros to select from, they will be displayed in the same list box used to display topic titles. The caption above the list box will be replaced with Select Macro to Run.

Other Source

If you are using multiple files to build your help system, you will want to use Jumps and Popups to Other Source to combine the functionality of the individual files. Click on the Other Source radio button and enter the Context String for the topic in any of the other source files. If necessary, refer to a copy of the Context Relations printout for the other source file. When compiled, the Jump will be linked to that topic in the other source file (Multiple Files).

Keyword

To jump to a keyword, first select the Keyword radio button. If keywords have been defined, they will be displayed in the list box used to display topic title. Select a keyword from the list.

When the end user selects a jump that is linked to a keyword, the topic that had the keyword assigned to it will be displayed. If more than one topic had the keyword assigned, a list of the topics will be displayed for the user to select from.

Besides offering a list of topics to jump to, the Jump to Keyword feature also allows help authors to easily include future updates, like new topics, without having to redistribute a potentially large help system. To accomplish this, the help author can add new topics to a separate help file and assign them keywords that were used in the original help system. When the update is completed, the new topics will be automatically linked into the old system via the keyword jumps. If the original help file is planned out carefully, future expansion can be less painful than before.

The help author should distribute an updated contents file (.CNT) with the new help file. To complete the upgrade the user's configuration file (.GID) needs to be updated as well. To accomplish this, the user can be instructed to delete their old configuration file or the help author's installation program can call WinHelp with the -g option. This will instruct WinHelp to force an update on the user's configuration file.

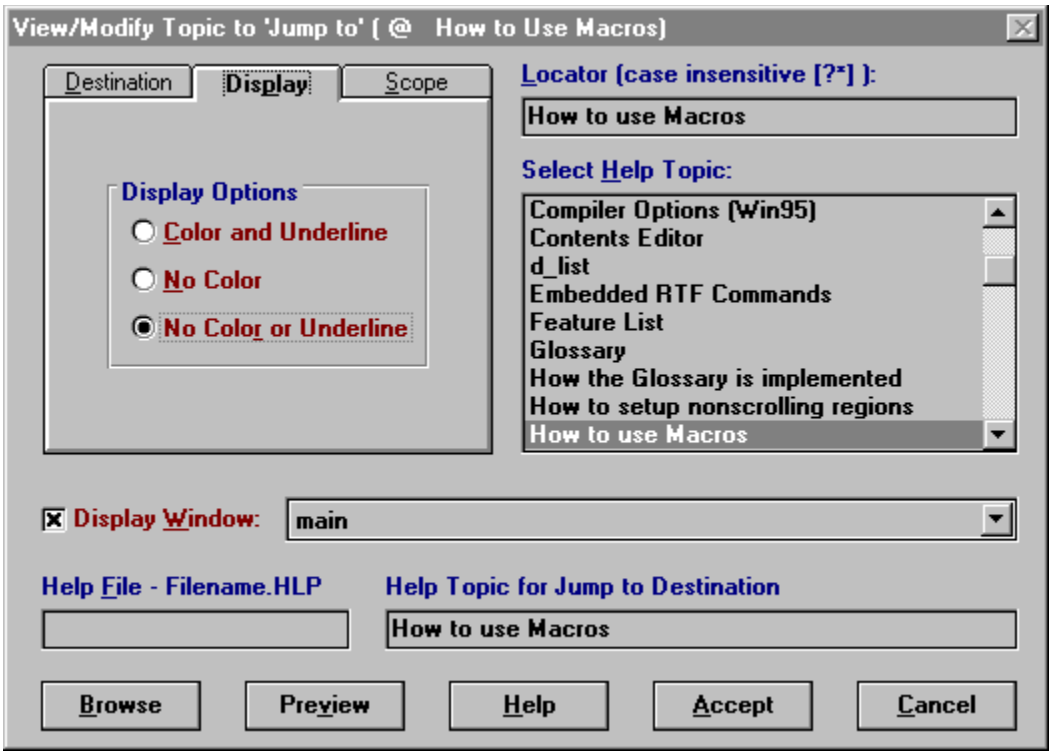
Refer to the section on *Working with Keywords* for more information on keywords in general.

Web Site URL

When this destination is selected, you will be required to enter a URL Address as the target for the jump. Enter the address exactly as you would in your net browser (http://www.whatever.com/web_page/).

If you have previously entered web addresses, they will be displayed in the listbox.

Display

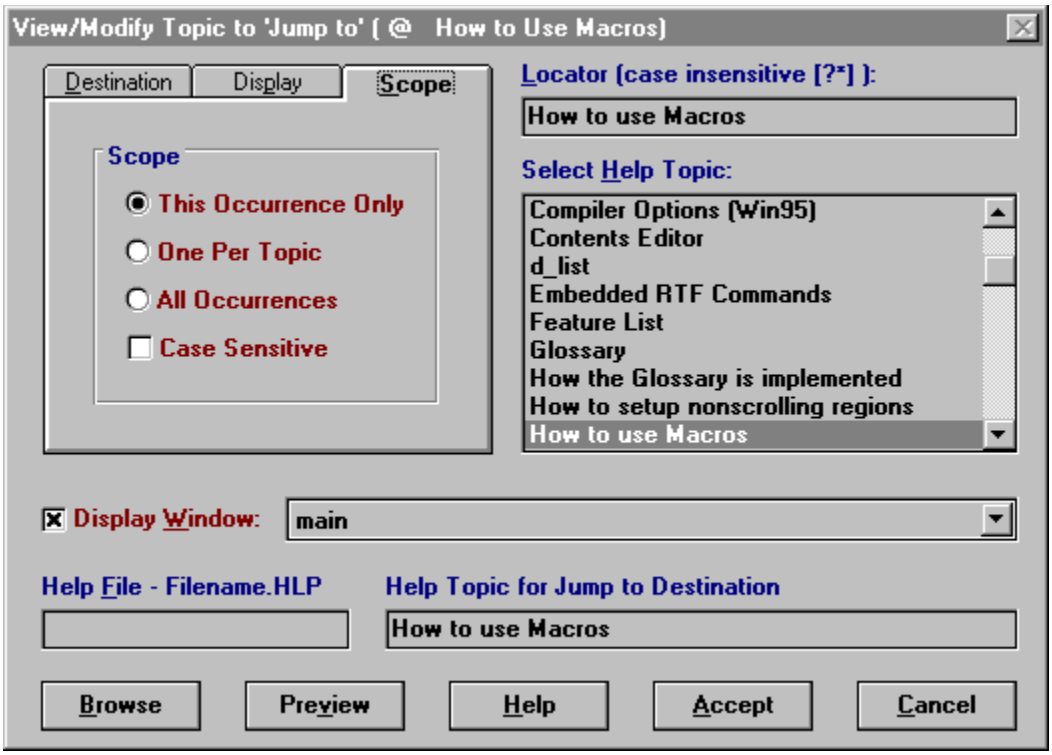


Display Options

The manner in which the Jump text is displayed in the compiled help file can be controlled with the three radio button in the Display Options frame. Normally, a Jump is green underlined text. This option is selected with the Color and Underline radio button.

To display the Jump text without color, select the No Color radio button. To display the Jump text without the color or the underline, select the No Color or Underline radio button.

Scope



Scope

When you establish a Jump of any type, you can set the link for the currently selected word or phrase, for one occurrence of the selected text on every page, or for every occurrence of the text in the help file. This eliminates the need to find and mark text throughout the help file.

Display Window

If you want the Jump to be displayed in a window other than the Main window, check the Display Window check box. Select the window for the Jump from a list of defined windows in the drop down box to the right of the Display Window check box.

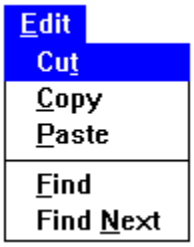
To force a Jump from a secondary window back to the Main window, check the Display Window check box and select the Main Window.

If no other windows have been defined, the only window in the list will be Main (Defining Help Windows).

If you've selected Jump to a URL, this option will be disabled.

Segmented Hypergraphics

Not only can pictures be used as jumps or Popups but portions of the them can be defined as Jumps and Popups to various topics. One obvious use for this feature would be to display an entire menu from the application and, clicking on a menu item would display information about that particular menu item. This type of image hypertext is known as segmented hypergraphics, or hot spots.

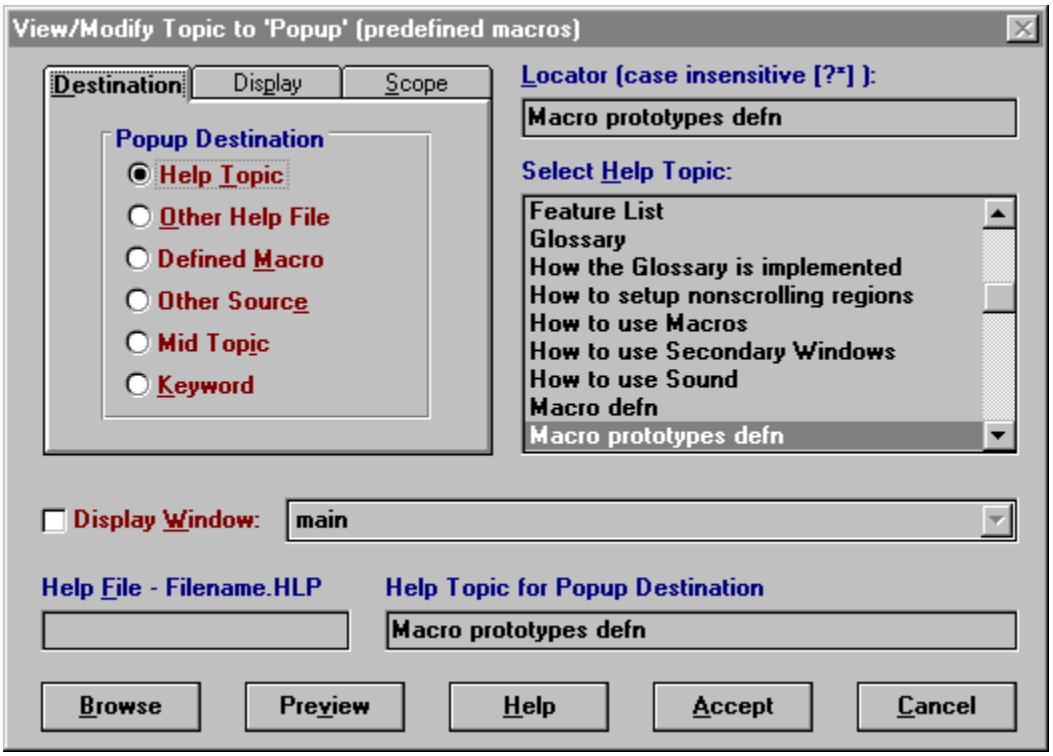


Each of the items in the sample menu could be defined as a Jump or a Popup to another topic. This requires the use of SHED.EXE, Microsoft's Segmented Hypergraphics Editor, supplied with Help Magician Pro, Windows SDK, and the Visual Basic 2.0/3.0 professional version.

Popup

Selecting Popup will establish a link between the selected word or phrase and another help topic in the current help file, a topic in another help file or a macro. A Popup is used primarily for definitions of words or as an explanation of a phrase. When run under WINHELP.EXE, the contents of the linked topic will be displayed in a window which pops up over the current help topic. The user will not Jump to the selected topic page.

When you select Popup, the Popup Destination form will pop up. There are options for the display of the Popup text, secondary windows, and the Scope of the Popup. Some of the options on the form are enabled and disabled depending on the Popup Destination option selected. The options are the same as the options for Jumps except that you cannot Popup to a secondary window. See the Jump section earlier in this topic for more information.



SHED Files

When editing a Bitmap with SHED.EXE, you will need the name of context strings associated with the

topic, or page, to be linked. A complete printout of the context strings is available by selecting Context Relations from the File/View Menu. A description of how the context strings are constructed from the topic titles is provided in Context Strings is further explained in SHED.EXE.

See Also

- Add Button
- Context Relations
- Creating Browse Sequences
- Define Macros
- Defining Help Windows
- Delete
- Embedded Video (AVI)
- Links From
- Mid Topic Jump
- Multimedia With WinHelp
- Multiple Files
- Unreferenced Topics
- View/Modify
- Working with Keywords

Unreferenced Topics

Selecting ***Unreferenced Topics*** from the ***Links*** menu will initiate a scan of the relationships of all the topics in the current source file.

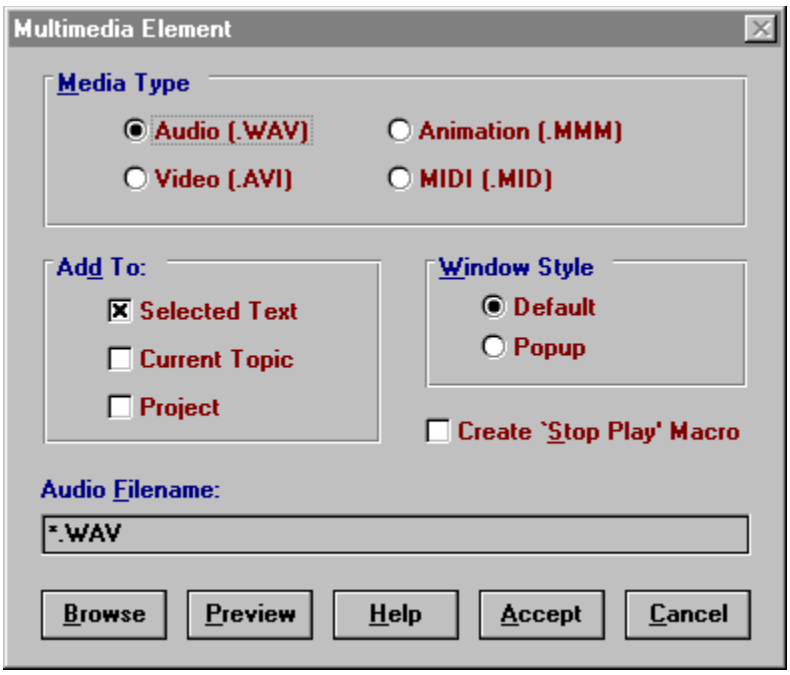
Every topic will be cross referenced for calls from other topics in the form of Jumps and Popups or from Shed files. Links to Mid Topic Context Strings will be included in the scan.

If unreferenced topics are found, they will be displayed in a text box at the end of the scan. Press the ***Print*** button for a hard copy of the list.

Multimedia with WinHelp

Multimedia

The Help Magician automates the use of audio (.WAV), midi (.MID), video (.AVI), and movie (.MMM) files in your help system. Selecting Multimedia from the Marker Menu or pressing the Multi button in the tool bar pops up the Multimedia dialog.



Media Type

Select the media type by pressing the corresponding radio button. The caption above the text box and the default extension in the filename text box will change to reflect the type of file.

Add To

Select the type of link to be established for this Multimedia element.

Selected Text: has the same effect as a the Multimedia element will be played when the user clicks on the text in the compiled help system.

Current Topic: has the same effect as a topic macro in that the Multimedia element will be played when the user selects the current topic in the compiled help system.

Project: has the same effect as a project macro in that the Multimedia element will be played when the help system is first called.

Window Style

Select the type of window to be used when displaying a video multimedia element. A Popup window will not have a border or system menu.

Stop Play

Put an X in the 'Stop Play' check box if you want a "Close" macro to be created along with the other related macros for the current multimedia element. This macro can later be linked to a close text/button of

your choice using the Jump to Macro capabilities of the Jump dialog.

Macros

When a multimedia element is linked, the Help Magician writes several macros and adds them to the existing macros, if any.

Project Macro

If a Project Macro does not exist, one is created to register the function(s) necessary to support the selected multimedia element.

Macro Names

The play and close macro definitions are created by adding the prefix "Play" to the filename of the media element to be played, without the file extension. If duplicate filenames exist, the filename will be appended with a three character representation of the next available numeric suffix for the filename.

Wave

The following line is added to the project macro if wave files will be played in the help system.

RegisterRoutine("MMSYSTEM", "sndPlaySound", "Si")

A play macro definition is created for each multimedia element used. It uses a call to the **sndPlaySound** function in MMSYSTEM.DLL.

sndPlaySound(`DRIVE:\PATH\WAVEFILE.WAV',0)

MCI

Video (.AVI), Movie (.MMM), and Midi (.MID) files are all played with the use of MCI calls. The following line is added to the project macro if MCI files will be played in the help system.

RegisterRoutine("MMSYSTEM", "mciExecute", "S")

Play w/ Stop

Typically, the following four macros would be included in a macro definition written to play a Video (.AVI) file if the "Create 'Stop Play Macro' check box is checked:

IfThen(IsMark("AVI000"),'mciExecute ("Close AVI000")')
mciExecute("Open DRIVE:\PATH:\AVIFILE.AVI alias AVI000 type AVIVideo style Popup")
mciExecute("Play AVI000")
SaveMark("AVI000")

Close

The following macro would be written to the Close macro.

IfThen(IsMark("AVI000"),'mciExecute("Close AVI000");DeleteMark("AVI000")')

This macro can later be linked to a close text/button of your choice using the Jump to Macro capabilities of the Jump dialog.

Microsoft Video for Windows or the Video for Windows runtime module is required to play AVI files.

Alias's

Alias's are created for use in the MCI open statement. The alias is created by using a three character

prefix for the media type (AVI, MMM, WAV, MID) and a three character suffix representing the next available numeric value for the media type.

SaveMark

The string used in the "SaveMark" and "IfThen(IsMark..." macros is exactly the same string as the alias. These macros are used to avoid errors caused by opening an MCI event that is already open or closing an MCI event that is already closed.

Play w/o Stop

Typically, the following three macros would be included in a macro definition written to play a Video (.AVI) file if the "Create 'Stop Play Macro' check box is unchecked:

mciExecute("Open DRIVE:\PATH\AVIFILE.AVI alias AVI000 type AVIVideo style Popup")
mciExecute("Play AVI000 wait")
mciExecute("Close AVI000")

The "**wait**" statement causes the multimedia element to be played in its entirety before control is returned to the help system.

Filename

Enter the filename for the Multimedia element or press the Browse button as described below.

Browse

Pressing the Browse button will pop up a file dialog which can be used to select the filename for any of the Multimedia elements. The default extension is already set for the type of Multimedia element selected in the Media Type frame.

Preview

When a filename has been entered, the Multimedia element can be "previewed" by pressing the Preview button.

Audio: requires a sound board and drivers or a PC Speaker driver.

Midi Sequence: requires a sound board and drivers.

Video: requires Microsoft Video for Windows drivers supplied with Microsoft Video for Windows or the Microsoft Video for Windows runtime module.

Animation: requires drivers supplied with Microsoft Multimedia Development Kit or Microsoft Viewer 2 Development Kit.

Accept

Pressing the Accept button will place Jump Markers around the selected text, if the Selected Text option is chosen. The Status label in the Status Bar will reflect the type of link.

Multimedia elements can also be previewed in the editor by double clicking on the hot spot that references the multimedia element. All multimedia elements included in the macro definition will be played. Click in the Paragraph Style frame ("Click to Stop"), on the Main Menu, or anywhere in the editor to stop the process.

Mid Topic Jump

Overview

Mid Topic Jumps are used to display a topic at a particular line in the topic. When jumped to in WinHelp, the topic text is scrolled to the line that contains the Mid Topic Context String. WinHelp will not, however, scroll the topic past the viewable area of the window and Mid Topic Jumps near the end of the text will not be scrolled to the top of the window.

Implementing

Select ***Mid Topic Jump*** from the ***Links*** menu. Enter the Context String in the form that pops up. Remember that Context Strings cannot contain spaces and many other characters. See Context Strings for a description of the mapping of illegal characters.

To create a Jump to this Mid Topic position, select ***Jump*** from the ***Links*** menu and click on the ***Mid Topic*** radio button on the ***Destination*** tab on the ***Jump*** form. Select the Context String from the list and select ***Accept***.

Embedded Video (AVI)

This is a Windows 95 feature only.

The term "Embedded Video" means that the window that displays the video file is positioned in the topic during design and will be displayed at that location when viewed with WinHelp rather than displaying the video in a separate window.

To embed a video (AVI) file into your help system, select Embedded Video (AVI) from the Links menu. This will pop up the Embedded Video form.

Embedded Video Options

First, select the **options** for the multimedia control window that is created.

EXTERNAL Keeps the file outside of the Help file.
NOPLAYBAR No playbar is shown (useful for auto-play and repeat).
NOMENU No menu button is shown if there is a playbar.
REPEAT The file automatically repeats when play is done.
PLAY The file automatically plays when shown.

Next, select the **orientation**.

Character The window acts like a character of text.
Left Margin The window will be at the left margin and text on the same line will wrap around on the right side of the window.
Right Margin The window will be at the right margin and text on the same line will wrap around on the left side of the window.

Next, select the **editor display option** for the Help Magician editor. These options are the same as the options for an image.

Embed Frame

Select the **Embed Frame** option to include the image of the first frame of the AVI file in memory and in the HLX (Help Magician source) file. This method produces the fastest load and display but uses the most memory and disk space.

Show Filename

When the Show Filename option is selected, the image will not be displayed in the editor, it will not be stored in memory, and it will not be stored in the HLX file. The image filename will be displayed in the editor in red text. The compiled help file will show the image, however.

Link to File

The **Link to File** option provides the best combination of performance, memory conservation, and disk space usage. The data for the image is not stored in the HLX file and the data for images is only stored in memory when the topic containing the image is displayed. Images for other topics are not kept in memory.

Video (AVI) Filename

Enter the filename of the AVI file to embed. Use the **Browse** button to use a file dialog to locate the file on your system.

Preview

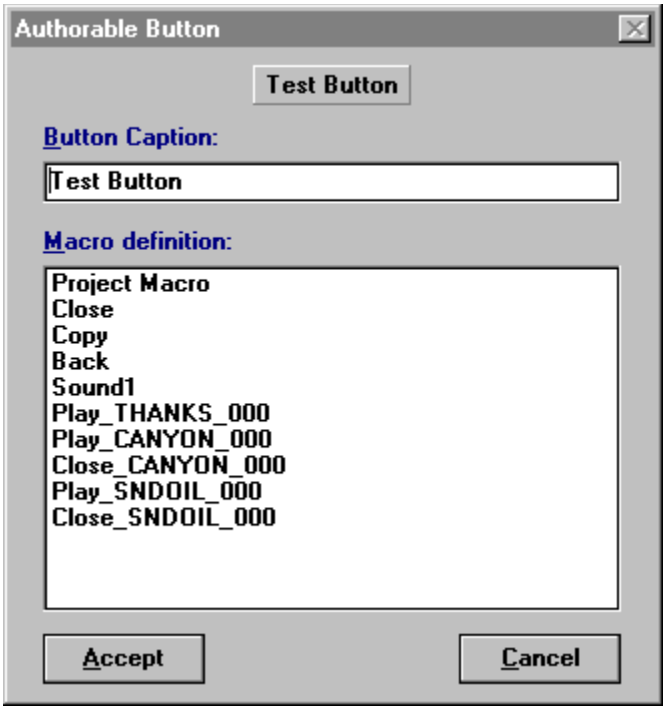
Pressing the **Preview** button will play the AVI file. It will be displayed in a separate window but it will 'play' just as it will in the compiled help file.

Authorable Button

Windows 95 Only

An **Authorable Button** gives the help author the ability to define a button in the help text by supplying the caption for the button and the functionality when the button is pressed. An **Authorable Button** could be used to do something as simple as calling the "About" dialog or something more interesting such as playing a multimedia element. All the capabilities of macros are at your dispsal for use with these buttons.

To add an authorable button to your help text, click on the **Links** menu, **Authorable Button** or click on the image of teh button in the toolbar. This will invoke the **Authorable Button** dialog.



Enter the Caption

Type the caption you want on the button in the textbox below the "Button Caption" label. The button changes as you type to reflect the actual appearance of the button when viewed with WinHelp.

Select a Macro

Select the macro definition to be executed when the user presses the button. The list is taken from the currently defined macros in the help system. See Macros, Working with for detailed information on macros.

Accept

When you have defined the button, click on the **Accept** button and an image of the button will be inserted into the editor.

View/Modify

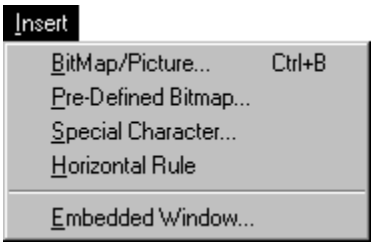
The ***View/Modify*** function, available from the ***Links*** menu, allows modification of a Jump, Popup, Mid Topic Jump, or an embedded image.

Shortcuts to the ***View/Modify*** function include the ***Ctrl+V*** key combination and clicking on the link with the right mouse button.

Delete

The ***Delete*** function, available from the ***Links*** menu, deletes any of the links used by the Help Magician. In the case of an embedded image, the image is also removed.

Inserting



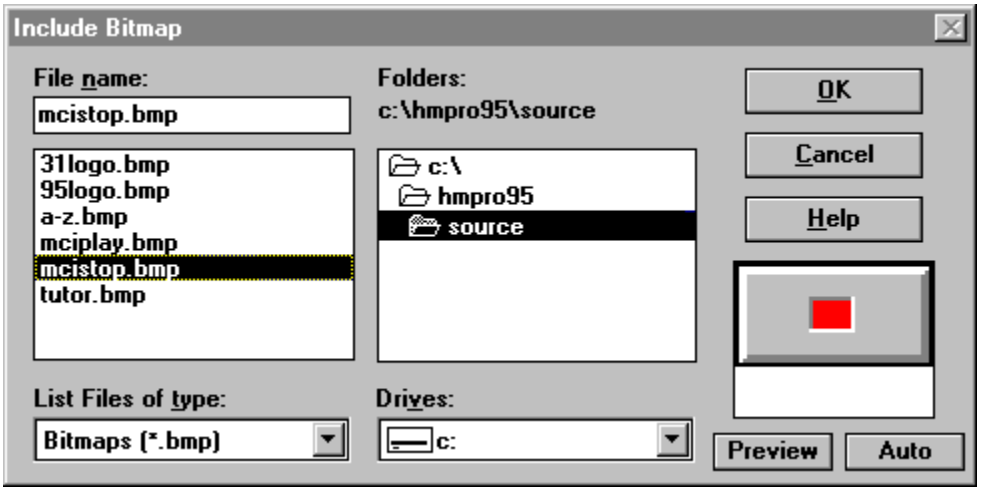
Click on the desired area of the Insert Menu for specific Help

Inserting Bitmaps/Pictures

The Help Magician supports BMP, GIF, ICO, PCX, SHG, TGA, TIF, and WMF graphics file formats.

To insert a picture into you help text, select Insert Bitmap/Picture from the Insert menu or click on the Bitmap button  in the Toolbar (Windows 3.1 mode).

This brings up a special file import dialog with image preview capabilities..



List Files of Type:

Select the type of image files to list by clicking on the down arrow under the ***List Files of Type:*** label and clicking on the desired file type.

Auto / Manual Display

Clicking on the Auto/Manual button toggles the state of the button. In **Auto** mode, the images will be displayed in the sample window as they are selected in the file list box. In **Manual** mode, they will not be displayed as they are selected in the file list box.

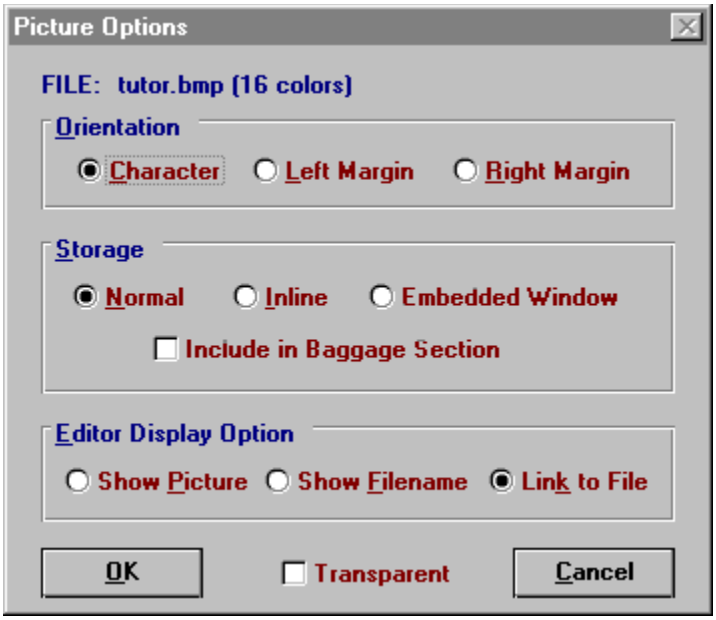
Note that the images in the sample window are stretched or reduced to fit the window and do not represent the actual size of the image.

Preview

When an image is selected in the file list box, it may be previewed in its actual size by pressing the ***Preview*** button, which brings up the image viewer.

Picture Options

When you have selected the image, the Picture Options form will appear.



Color Depth

If the image to be inserted is detected as having a color depth of 256 colors, the Help Magician will automatically insert the image as an embedded window (Windows 3.1 only). This is done through the use of the HMEW2.DLL, supplied with the Help Magician.

If embedded windows are used in your help system, HMEW2.DLL must be shipped with the compiled help file and installed to the WINDOWS directory (Windows 3.1 only). In addition, all your 256-color image files must be shipped and installed in the same directory as your compiled help file. Another option to get around using the HMEW2.DLL is to use Microsoft's Help Compiler version 3.10.505 (Extended). This version supports the inclusion of 256-color images directly in the HLP file without needing a DLL. However, we found this version of the compiler to have quirks, and thus have not shipped it with Help Magician. You can get this version off our BBS. You can try it and if it works for you then use it.

All color depths are supported by the Windows 95 help compiler and the use of a DLL to display the image is not necessary.

Orientation

Select the orientation of the image by clicking on the appropriate radio button. When character orientation is selected, the image will act as any other character in the text. It will flow with the text and wrap when necessary. Left and right orientation will cause the image to stay at the left or right margin regardless of the flow of the text.

The editor will not display left and right aligned bitmaps as such but they will be properly aligned when the help file is compiled.

Storage

When *Normal* storage is selected, the data for the image is stored only once in the compiled help file even if the image is used many times. This can result in a slightly slower display time but it reduces the size of the compiled help file.

When *Inline* storage is selected, the data for the image is stored for each occurrence of the image in the compiled help file. This method produces a faster display of the image but results in a larger compiled help file.

When *Embedded* is selected, the image will be displayed in an embedded window in the topic. This is used primarily for 256 color bitmaps.

The Include in Baggage section is used to force the compiler to store the bitmaps in the baggage section of the Help file.

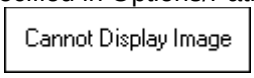
Editor Display Option

Select the **Show Picture** option to include the image data in memory and in the HLX (Help Magician source) file. This method produces the fastest load and display but uses the most memory and disk space.

When the **Show Filename** option is selected, the image will not be displayed in the editor, it will not be stored in memory, and it will not be stored in the HLX file. The image filename will be displayed in the editor in red text. The compiled help file will show the image, however.

The **Link to File** option provides the best combination of performance, memory conservation, and disk space usage. The data for the image is not stored in the HLX file and the data for images is only stored in memory when the topic containing the image is displayed. Images for other topics are not kept in memory.

Cannot Display Image

If a link to file image is not available (in one of the bitmap directories as specified in Options/Paths/Bitmap Directories) when the topic that contains that image is displayed, a bitmap 

will be temporarily displayed.

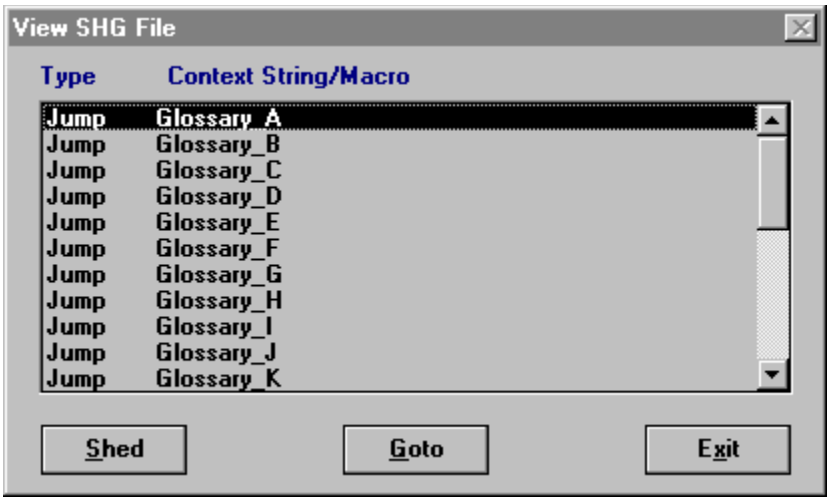
If you compile the source without the images available, you will get warnungs from the help compiler.

The images will be properly displayed and compiled when they become available.

Transparent Bitmap Option

Select the Transparent Bitmap option if you want the bitmap to be transparent in your help file. A transparent bitmap will have any color WHITE changed to the background color of the WinHelp window the bitmap is displayed in.

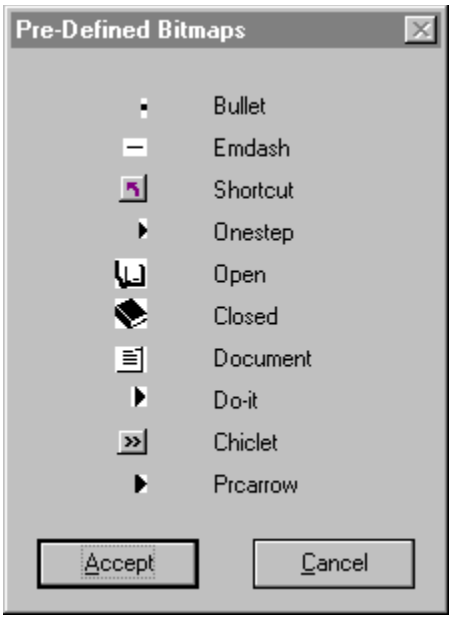
A "local" bitmap is created in memory from Shed (SHG) files for display in the editor. Note that double clicking on a SHG image file will bring up the View SHG form listing all the context strings referenced in the SHG image and an option to launch the SHED editor.



Bitmap files (.bmp) are created, in the 'ROOT' or working directory, from GIF, ICO, PCX, TGA, and TIF files to be sent to the Help Compiler.

Pre-Defined Bitmaps

When using the Windows 95 help compiler, ten bitmaps are provided by the compiler for use in your help system. Selecting the **Insert Menu**, **Pre-Defined Bitmap**, will pop up a form from which you can select one of the pre-defined bitmaps.



To select a bitmap, either click on the bitmap or its name and click on **Accept** or double click on the bitmap or its description.

The bitmaps do not need to exist in any directory in order to use them. However, if a bitmap with the same name does exist in the build ROOT directory or in any of the designated bitmap directories, it will be used instead of the compiler's internal bitmap.

Embed Window

The Help Magician supports the embedded window feature of WinHelp. The embedded window feature allows users to place a window inside a topic in their help file that can be controlled by programs written for this purpose. These programs, called "DLL's", can be purchased from software vendors or created by the more experienced user.

If embedded windows are used in your help system, **HMEW.DLL** must be shipped with the compiled help file and installed to the **WINDOWS** directory. Any bitmaps displayed with **HMEW.DLL** must also be shipped with the help system.

Some of the types of things that can be done with an embedded window are animation (sequential display of several bitmaps), display of lists of information read from external data files like phonebooks or file names, and display of 256 color bitmaps.

To use this feature you will need the dollowing information about the particular DLL you are using:

- The name of the DLL, i.e. "HMEW.DLL"
- The name of the window class, i.e. "ewBitmap"
- A list of arguments that the DLL needs to run.

This information should be found in the documentation supplied by the manufacturer of the DLL that you are using.

To insert an embedded window into your help text, select Embedded Window from the Edit menu. This brings up the embedded window dialog.

Placement

Select the orientation of the embedded window by clicking on the appropriate radio button. When character orientation is selected, the embedded window will act as any other character in the text. It will flow with the text and wrap when necessary. Left and right orientation will cause the embedded window to stay at the left or right margin regardless of the flow of the text.

The editor will not display left and right aligned embedded windows as such but they will be properly aligned when the help file is compiled.

DLL Name

Enter the name of the DLL that you intend to control the embedded window. This information should be found in the documentation supplied by the manufacturer of the DLL that you are using. You may use the **BROWSE** button to search for a DLL on your disk.

Class Name

Enter the class name from the documentation provided by the DLL manufacturer.

DLL Data

Enter the data or list of arguments required by the particular DLL that you are using. This information can be found in the documentation supplied by the manufacturer. Please separate the arguments by spaces: do not use commas to separate arguments.

When you have finished entering all of the above information, select Accept to create the embedded window or Cancel to abandon the changes.

After creating an embedded window, a picture of a box will be put into the Editor to remind you that you have placed an embedded window there. The words "Embedded" will be displayed followed by a single letter indicating which type of placement you chose (Character, Left or Right). After this the name of the DLL will be displayed in the box.

You may modify the embedded window information at any time by clicking on the box with the right mouse button.

Restore Page

This function restores current topic text to the state of the page when last paged to, until paging to another topic, or using any of the following functions:

Find/Replace, Spell Check, Build, One Page Preview, VB Help Wizard, Save File/Backup, or Setup Browse.

Working with Keywords

Keyword Management and WinHelp Keyword Simulation

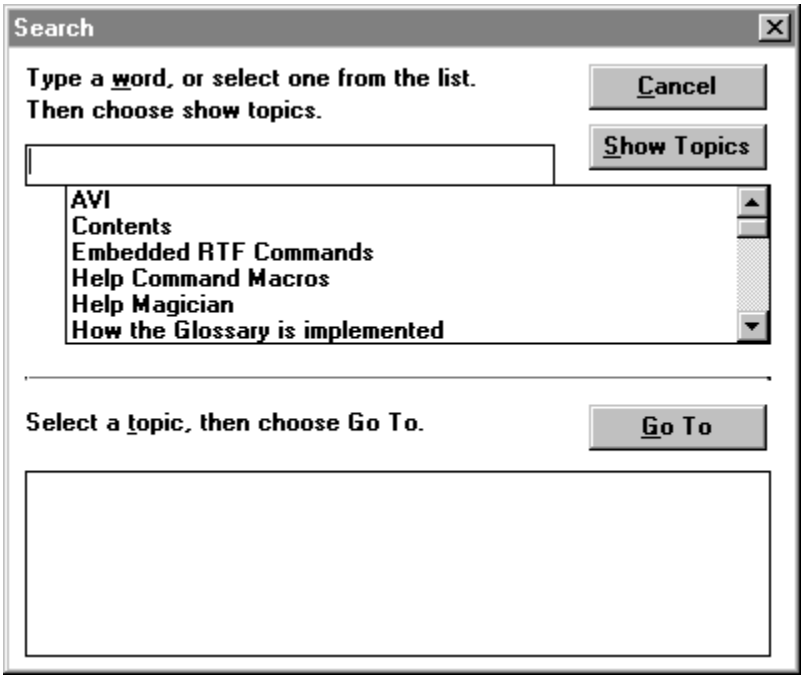
Overview

Keywords play a very important role in the development of any help file. Keyword entries make up the *Index* of your help file. The Index is often overlooked by many help authors. A quality help file should contain a well constructed index; thus you should put a lot of thought into creating your Index.

Help Magician Pro 95 offers a database approach to managing keywords in your help file. You can view, work, and print the keyword database in either of two ways: by a list of keywords with their associated topics or by a list of topics and their associated keywords. In addition, Help Magician Pro 95 simulates WinHelp keyword/topic access during the testing of your help file.

The WinHelp 3.1 Index

The WinHelp 3.1 Index screen is shown below. The user selects one of the keywords in the keyword list or types a word or a phrase in the text box above the list of keywords. When the user selects a keyword, a list of topics that are associated with that keyword appear in the Topics Found box. The user then clicks on one of these topics to display the topic. Microsoft found that users were confused by this interface and redesigned it for Windows 95.

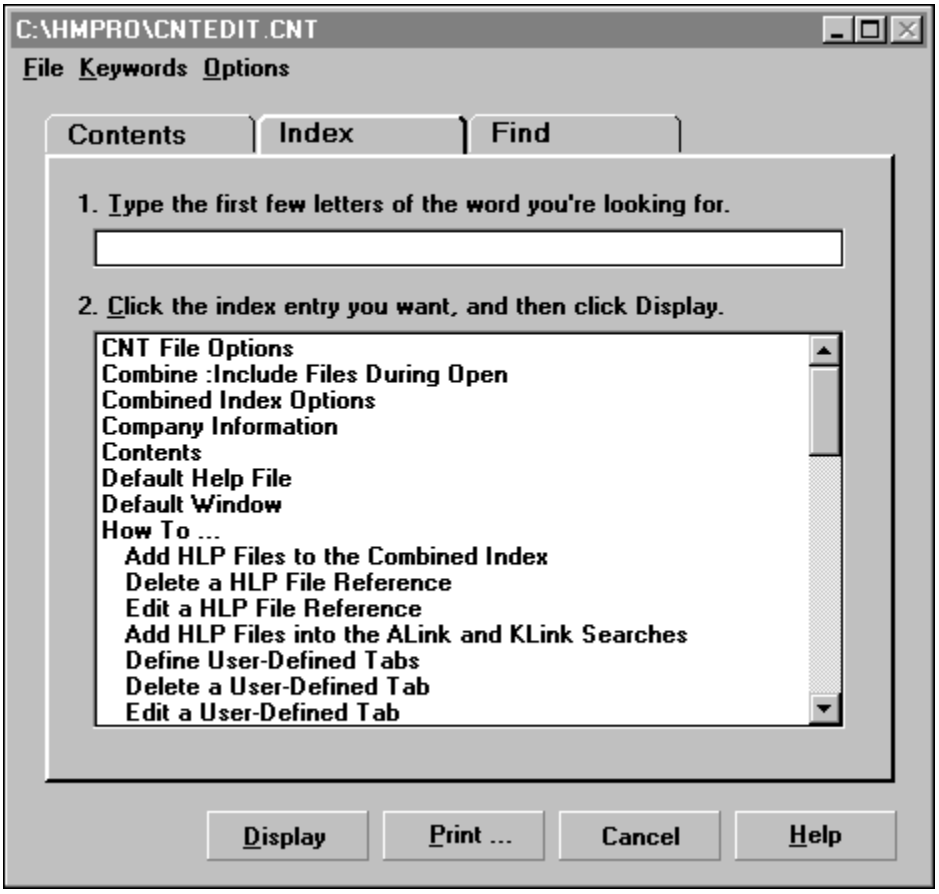


Help Magician Pro 95 simulates the WinHelp 3.1 Search dialog provided that you've selected Windows 3.1 compiler mode (see Options...Compiler...Compiler Tab). To simulate the WinHelp 3.1 Search dialog in Help Magician Pro 95, click on the Search button in the WinHelp button bar just above the editor (if it's not visible, turn it on by selecting Options...Environment Options...Display Tab). You must have created some keywords in Help Magician prior to using this feature.

The WinHelp 95 Index

The WinHelp 95 Index screen is shown below. Its user interface resembles a print index. This index can contain subentries (called "Related Keywords" in Help Magician) under main entries (called "Keywords" in Help Magician). Related Keywords are shown indented under Keywords, both in the keyword database

and in the WinHelp 95 simulated Index Tab in Help Magician Pro 95.



Help Magician Pro 95 simulates the Help Topics browser provided that you've selected Windows 95 compiler mode (see Options...Compiler...Compiler Tab). It functions just like the WinHelp 95 Index Tab. To simulate the WinHelp 95 Index in Help Magician Pro 95, click on the Index button in the WinHelp button bar just above the editor (if it's not visible, turn it on by selecting Options...Environment Options...Display Tab). Once the Help Topics browser shows, click on the Index Tab to view the Keywords. You must have created some keywords in Help Magician prior to using this feature.

See Also

Keyword Management

The Keyword Management Screen

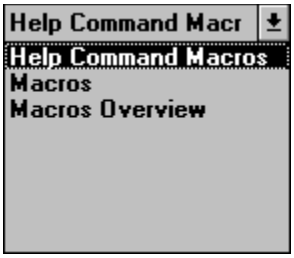
Keyword Mangement

Help Magician Pro 95 provides a dual database approach to managing keywords within a single help source file or multiple source files (project). Either way is completely transparent to the help author. When you create a new source file (hlx) or project file (hmp), Help Magician Pro 95 creates two databases: topics and keywords. The keyword database contains all the keyword information and cross-references to the topic database. The topic database contains topic information and any cross-references to the keyword database. The keyword database uses three files with the filenames *.ksd, *.ksf, and *.ksm. The * represents the first 8 characters of the hlx or hmp filename. By contrast, the topic database also uses three files with the filenames *.isd, *.isf, and *.ism. The * represents the first 8 characters of the hlx or hmp filename. Thus, both the keyword and topic databases are related to one another. All database files are stored in the same directory as the HLX or HMP files AND SHOULD NOT BE MOVED SEPARATELY.

Whenever you save your work, Help Magician Pro 95 saves a copy of the keyword information specific to your source file within the HLX file. Thus if something should happen to the keyword database, it is possible to reconstruct it from the data contained in the HLX file.

Help Magician Pro 95 offers two ways of working with keywords. The first way allows you to add and delete keywords related to a specific topic you are currently working on. They can be added and deleted through a drop down box that appears on the left just above the editing window. The second way is to use the keyword database manager, available from the Links Menu. The first way is very simple and doesn't offer the ability to create related keywords, whereas the second way is very powerful.

Using the Keyword Drop Down Box on the Editor



Keywords

The Keywords drop down box, located just above the editor window on the left) displays the first keyword of each help topic as you page up or down. Click on the arrow to the right of the drop down box, or press Alt Up Arrow, to display all of the keywords for the current page. A new keyword can be entered, at any time. Make sure you press the enter key after each keyword entered. Delete keywords by highlighting the desired keyword in the list and pressing the delete key. Click anywhere in the editor or press the Esc key to return to the editor.

Keyword List

Clicking on the Keyword List button 

to the left of the Keywords drop down box will pop up a table of Keywords already used in the help file. This is useful if a keyword is used on a number of pages and you aren't sure how it was phrased or spelled.

Double click on the Keyword you want to add or click on the Accept button and the Keyword will be added to the Keywords drop down box.

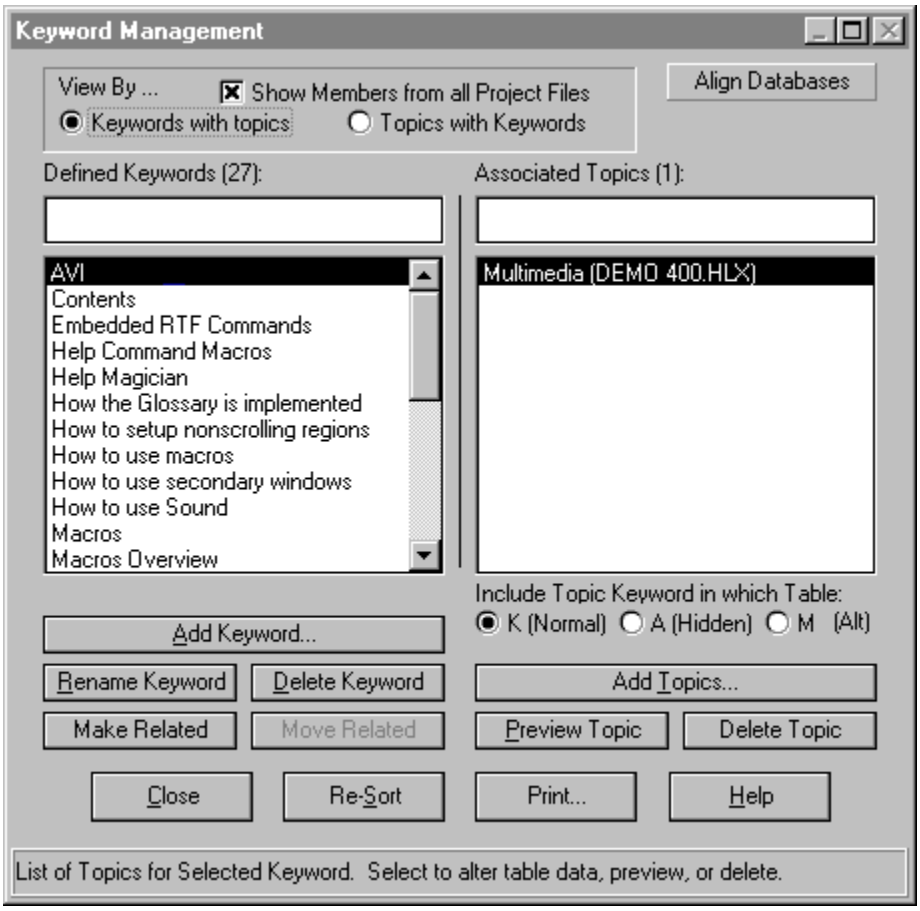
See Also

The Keyword Management Screen

Macro Keyword Associations
View by Keywords with Topics Mode
View by Topics with Keywords Mode
Working with Keywords

The Keyword Management Screen

Selecting the menu Link...Keyword Management will bring up the following screen



There are two modes that the Keyword Management screen will function: Keywords with associated topics and Topics with associated keywords. Selecting View By...Keywords with Topics provides all the functionality within the Keyword Management screen. You can add and delete keywords, make related keywords (i.e. make a keyword a subentry in your index), adding and deleting topic associations from keywords, and modify which table (Normal, Hidden, or Alternate) that the keyword gets put in. Selecting View By...Topics with Keywords provides limited functionality including Adding and Deleting Keywords. From this mode you cannot Add or Delete Topics, because the topics that are shown are all the topics for the entire project (including multiple file spans).

In the View By Keywords with Topics mode, the left listbox shows all the keywords currently defined throughout your help project and the right listbox will show the list of topics that are associated with a SELECTED keyword in the keyword list. The filename in parentheses (next to the topic in the listbox) is the file where the topic resides. Any defined Related Keywords will appear in the keyword listbox properly indented under its parent.

In the View By Topics with Keywords mode, the right listbox shows a list of all the available topics within your help project and the left listbox will show a list of keywords that are associated with a SELECTED topic in the topic list. If there are no keywords associated with a selected topic, then the listbox will be empty. In this mode, you cannot tell what are main keywords and what are related keywords.

Navigating the Lists

For each list, there is a text box located above the list. Typing characters in the text box will cause the listbox to scroll to the next nearest match within the text box.

Keyword Tables

Help Magician Pro 95 supports three keyword tables: Normal, Hidden, and Alternate. Normal keywords appear in the Index Tab and can be referenced with the Klink macros ("Jump to Keyword" functionality within Help Magician). *Normal* keywords are the most often used. Hidden keywords do not appear in the Index Tab but are available with the Alink macros (again, "Jump to Keyword" in Help Magician). *Hidden* keywords are useful for those topics that you do not want to appear in the Index Tab. Note that Alink and Klink macros ("Jump to Keywords") functionality are available only for Windows 95. *Alternate* keywords are rarely used by help authors, but can be specified in the call to WinHelp. Help Magician defaults to the letter M for the alternate keyword table. You do have the option of using other letters other than M for each Alternate keyword. To change the alternate keyword table "M" to another table, simply click on the "M" (or any other letter that may be there) and enter the letter in the textbox.

Tip: the Jump to Keyword function in Help Magician Pro 95 spares you from having to deal with Alink and Klink macro programming! (See Jumps and Popups).

See Also

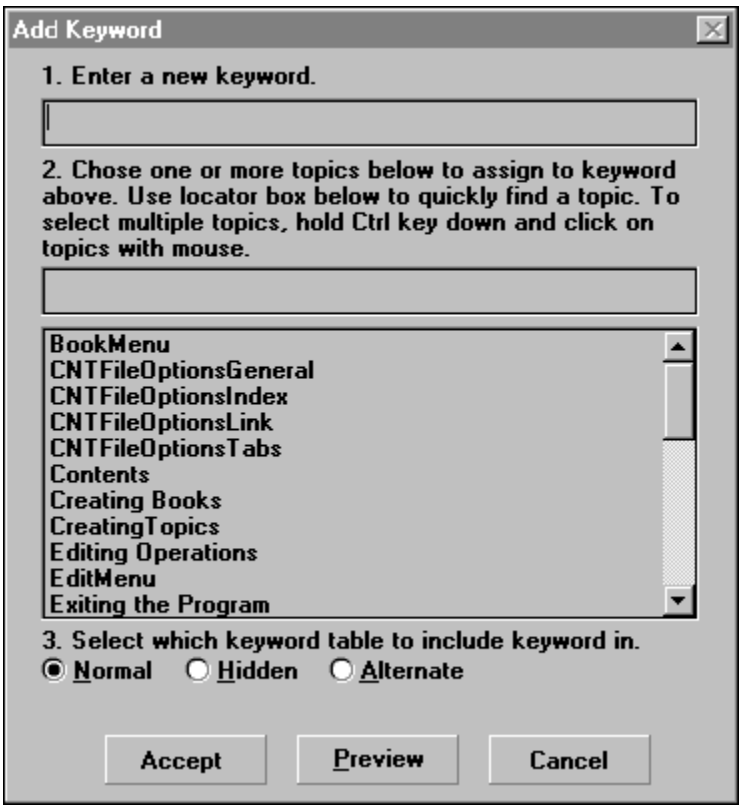
- Keyword Management
- View by Keywords with Topics Mode
- View by Topics with Keywords Mode
- Working with Keywords

View by Keywords with Topics Mode

Adding Keywords

1. To Add a keyword to the keyword database, click on Add Keyword...
2. The Add Keyword screen will appear. Type in a new keyword, then select an existing topic to assign the keyword to. Then select which keyword table to include the keyword in.

Notes:
You cannot add keywords to the database if you do not choose a topic to assign it to.
You can preview a topic if it exists in your current source file.



Deleting Keywords

1. To delete a keyword, select it from the keyword list and click on Delete Keyword. Doing this will delete the keyword and all topic associations from the database.

Renaming Keywords

1. To rename a keyword, select it from the keyword list and click on Rename Keyword. Any changes you make to the keyword name becomes permanent to the keyword database, even if you do not save your help source file.

Adding Topics to Keyword List

1. Select (or highlight) a keyword from the keyword list.
2. Click on Add Topics...
3. The "Add Topics" screen will appear. Simply select the topic you want to be associated with your keyword. Click on Accept.
4. The added topic will be displayed in the topics list box.

Deleting Topics from Keyword List

1. To delete a topic associated with a selected keyword, click on Delete Topic. Once all the topics associated with a keyword have been deleted, the keyword is removed from the database.

Previewing Topics in Keyword List

1. You can preview any topic that exists in your currently opened HLX source file by selecting a topic from the topic list and clicking on Preview Topic.

Making a Related Keyword (or subentry)

1. To make a keyword a Related keyword (or subentry in the WinHelp Index), select it in the keyword list, then click on Make Related.
2. Next choose which keyword you want to be its parent.
3. The keyword will appear indented under the "parent" keyword.

Note: This function results in instant permanent changes to the keyword database.

Moving a Related Keyword to another Keyword

1. If you want to move a related keyword to another keyword parent, select the Related Keyword and click on Move Related.
2. Next, from the popup keyword list, select the new parent.

Note: This function results in instant permanent changes to the keyword database.

Printing a List of Keywords and Associated Topics to Printer

1. To print an alphabetical list of keywords and their associated topics to the printer, make sure you have View By...Keywords with Topics mode selected.
2. Then click on Print...

See Also
Keyword Management
The Keyword Management Screen
View by Topics with Keywords Mode
Working with Keywords

View by Topics with Keywords Mode

Adding Keywords to a Topic

1. To add a keyword to a topic, select the topic you want to add the keyword to and click on Add Keyword...
2. Then when the Add Keyword screen appears, type in the keyword and click on Accept.

Deleting Keywords from a Topic

1. To delete a keyword from a topic, select the topic in the topic list.
2. When a list of keywords appear in the left listbox, select the keyword in that list.
3. Then click on Delete Keyword.

Note: If the keyword is tied to any other topics, it is not removed from the database until there are no more topics assigned to that keyword.

Renaming Keywords

1. To rename a keyword, select the topic in the topic list.
2. When a list of keywords appear in the left listbox, select the keyword in that list.
3. Then click on Rename Keyword.
4. Type in the new keyword name.

Previewing Topics

1. You can preview any topic that exists in your currently opened HLX source file by selecting a topic from the topic list and clicking on Preview Topic.

Printing a List of Topics and Associated Keywords to Printer

1. To print an alphabetical list of topics and their associated keywords to the printer, make sure you have View By...Topics with Keywords mode selected.
2. Then click on Print...

What Happens if the Keyword or Topic Database Becomes Problematic?

There may be an occasion whereby the Topic Database becomes out of sync with the Keyword database (could happen during a power failure). If this occurs, you will notice a keyword reference in the drop down box may not match the keyword in the database. Simply click on Align Databases and Help Magician will scan the Keyword database for topic references and rebuild the topic database with the referenced keywords.

Exiting the Keyword Database Screen

To exit Keyword Management, simply click on the Close button. Any adding or deleting changes made to the database will not become permanent until you save your file.

Technical Notes About the Databases

Help Magician Pro's Keyword Database has been designed to work for both single file and multiple file project management. As such, a few facts should be noted.

(1) When a keyword is added to the topic database, a commit flag is set and the keyword is temporarily added to the database. When you quit Help Magician without saving, the keywords are removed next time you come back into the file. Likewise, if you save your work, the keywords are "committed" to the database permanently.

(2) When a keyword is deleted, the a delete flag is set within the database. If you quit Help Magician without saving, the delete flag is removed from the database next time you open the file and the keyword is not deleted. Likewise, if you save your work, the keyword is permanently removed from the database.

(3) If you rename a keyword, Make Related, or Move Related, the changes are permanent.

(4) A copy of the database (related to the specific file you're working on) is written to the HLX file when you save. This allows the database to be reconstructed, should you ever have to. When Help Magician opens the file you want to work on, it checks to see if a keyword database exists. If it doesn't, it is automatically constructed. Thus, if you wanted Help Magician to reconstruct the databases, just delete the three related keyword database files (*.ksf, *.ksm, and *.ksd). A word of caution, if the topic database ever becomes corrupted before you save, improper information may be written to the HLX file regarding keywords, thus a successful reconstruction of the keyword database isn't possible.

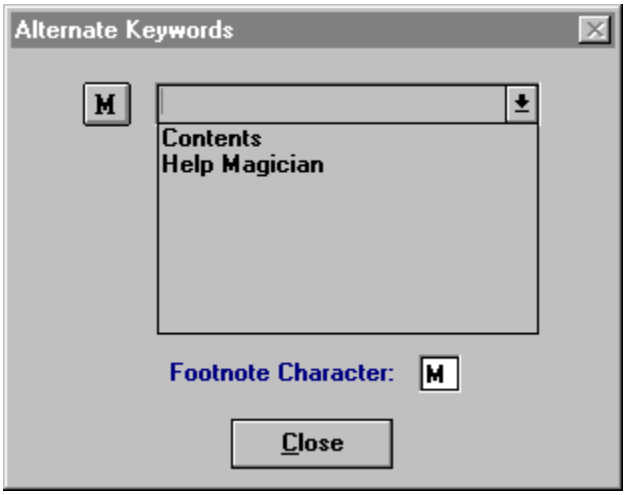
(5) In Project Management mode, the Topic Database is the same database as the Project Database mentioned in the topic on Project Management.

See Also

- Keyword Management
- The Keyword Management Screen
- View by Keywords with Topics Mode
- Working with Keywords

Alternate (Multiple) Keywords

Alternate keyword tables enable a program to look up topics that are defined in alternate keyword tables. This feature can be used to provide context sensitive help on words in an editor, for instance. The Alternate Keyword form is available from the ***Build menu***. Add and delete Alternate keywords in the same manner as described for Keywords.



Multiple List

Clicking on the Multiple Keyword List button 

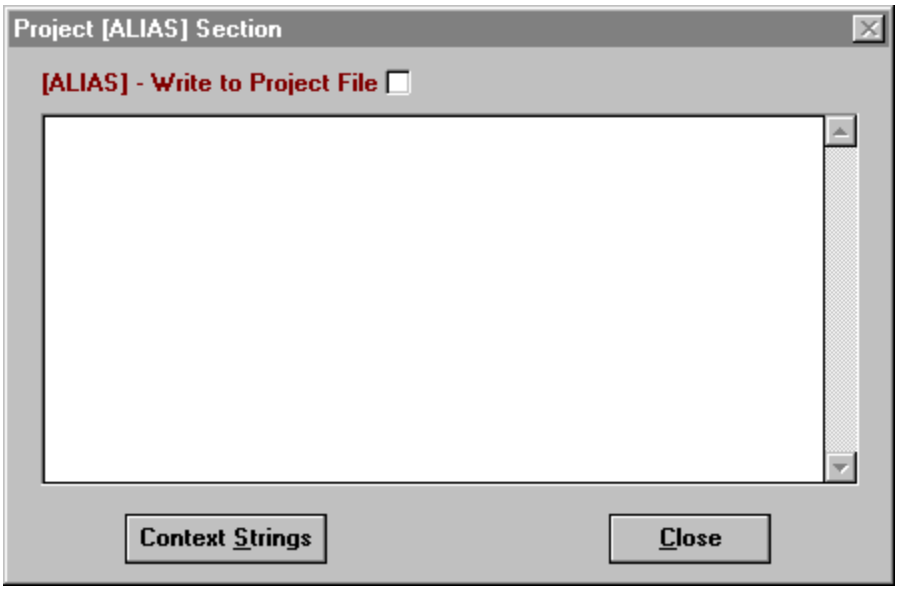
to the left of the Multiple Keywords drop down box will pop up a table of Multiple Keywords already used in the help file. This is useful if a Multiple Keyword is used on a number of pages and you aren't sure how it was phrased or spelled.

Double click on the Multiple Keyword you want to add or click on the Accept button and the Multiple Keyword will be added to the Multiple Keywords drop down box.

Alias Section

Project [ALIAS] Section Dialog

The Project [ALIAS] Section Dialog is available from the **Build Menu**, **Alias Section** sub menu.



Write to Project File
A check box is provided to determine whether the Alias information is written to the Project file. This provides a means to store the information without writing it to the Project. When the box is checked, the information will be written to the Project file.

Enter the Alias information as described below.

Do not include the [ALIAS] header in the text.

Syntax

[ALIAS]
context_string=alias context string

The **[ALIAS]** section associates one set of context strings with an alternate set of context strings. The alias strings correspond to context strings assigned to topics in the # footnotes of the Help file. This section is optional; however, if it is included, it must precede the **[MAP]** section in the Help project file.

Parameters

context_string
Specifies the application ID or other context ID that you want to reassign.

alias context string
Specifies the context string that appears in the # footnote of the topic you want Help to recognize. An alias context string has the same form and follows the same conventions as standard context string. That is, it is not case-sensitive and may contain the alphabetic characters A through Z, the numeric characters

0 through 9, and the period (.) and underscore (_) characters.

Comments

Because context strings must be unique for each topic and cannot be used for any other topic in the Help project, the **[ALIAS]** section provides a way to remap context strings that are no longer used or invalid. For example, suppose the application defines a context ID for each field in a dialog box, but your Help file only provides one topic for all the fields. You can use the **[ALIAS]** section to map all the application context IDs to your one Help topic. In this way, no matter which field the user has selected in the dialog box, Help will display your Help topic when the user requests context-sensitive Help.
You can also use the **[ALIAS]** section to combine Help topics without recoding your files. For example, if you create a topic that replaces the information in three other topics, you could manually search through your files for invalid cross-references to the deleted topics. The easier approach, however, would be to use the **[ALIAS]** section to assign the name of the new topic to the deleted topics.

You can use alias names in the **[MAP]** section of the Help project file. If you do, however, the **[ALIAS]** section must precede the **[MAP]** section.

Example

The following example creates several aliases within an **[ALIAS]** section:

Do not include the [ALIAS] header in the text.

```
[ALIAS]
sm_key=key_shrtcuts
cc_key=key_shrtcuts
st_key=key_shrtcuts          ;combined into keyboard shortcuts topic
 clskey=us_dlog_bxs
maakey=us_dlog_bxs           ;covered in using dialog boxes topic
chk_key=dlogprts
drp_key=dlogprts
lst_key=dlogprts
opt_key=dlogprts
tbx_key=dlogprts             ;combined into parts of dialog box topic
frmtxt=edittxt
wrptxt=edittxt
seltxt=edittxt               ;covered in editing text topic
```

Creating Browse Sequences

Browse

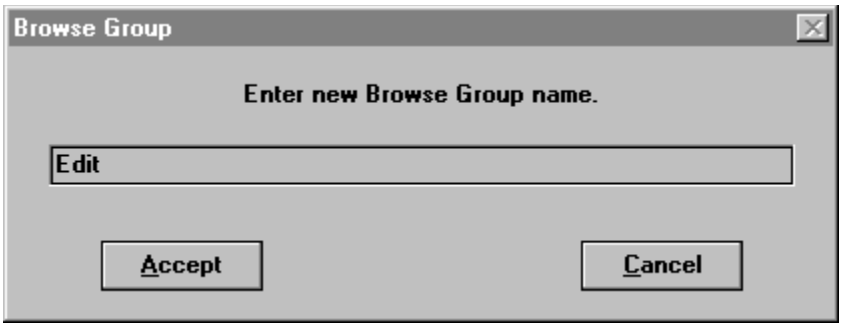
Clicking on the Browse button will bring up the Browse Group Definitions window.

Add Group

When first entering Browse groups, click on the red Add button below the Groups list box. Type in the desired name of the group and select ***Accept***. Keep group names simple but descriptive. Browse groups define a section of topics, or help pages, that will be displayed as the Browse buttons (<< and >>) are pressed while viewing a Windows Help file. These should be topics that are directly related to each other and maintain a degree of continuity while browsing. This allows the user to view related topics without the need to initiate a new search or to return to the Contents page.

Input Form

The Input form is used here to add a Browse group.



Add Topics

Once a Browse group name has been entered, you can add the titles of help topics to the Titles list box. When you press this Add button, a list of titles, taken from the current help file, appears. Titles already belonging to a Browse group do not appear in the list because a topic can belong to only one Browse group. Highlight the desired title and select ***Accept*** or double click on the title. The title will be added to the end of list box.

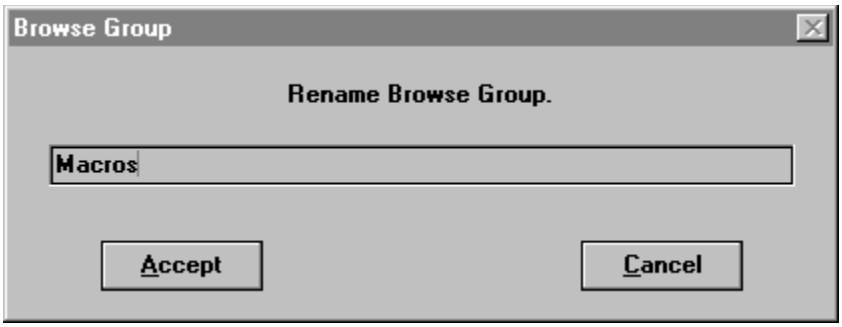
To find an existing topic, enter the topic name or at least the first few characters into the locator fields above the topic list box. The list is alphabetically sorted so as you type each character the next closest match will be displayed in the Browse Group list box.

Rename

To rename a Browse Group, highlight the desired Browse Group name and click on the Rename button or double click on the Browse Group name. A dialog will Popup with the current name to be edited or replaced.

Input Form

This Input form is used here to rename a Browse group.



Browse Order

Titles will be Browsed in the order in which they appear in the Titles list box. There are two ways to re-arrange the order of the titles. One is accomplished with the keyboard and the other is done with the mouse. To move a title with the keyboard, highlight the title to be moved, activate the Move button with the hot key if a mouse is not available, cursor to the new position, and activate the Move button again, which has been renamed to Done, and the title is moved to the new position.

To move a title with the mouse, click on the Move button, grab the title with the mouse cursor (press the left mouse button), drag the title to the desired new position, and release the mouse button.

Delete

Browse groups or titles contained in Browse groups can be deleted at any time by highlighting the desired entry and pressing the Delete button under the corresponding list box.

Locators

There are text boxes above the list boxes used to display the groups and titles. They used to quickly locate items in the list by typing some of the characters in the help group or title. The list will automatically scroll to the item most closely matching the entry in the text box.

Defining Help Windows

The method by which Help Windows is designed is greatly improved over previous versions of Help Magician. The newest version offers Visual Help Window design as well as support for all the options for Windows 95 help.

Overview

WinHelp provides three kinds of help windows to display topics in: main windows, secondary windows, and popup windows. Windows 95 allows more flexibility in controlling the use of windows than does Windows 3.1.

Popup Windows

Popup windows are very specific windows that popup next to the hotspot you are clicking on. You cannot alter the size or position of a popup window, nor can you define buttons, menus, or non-scrolling regions. Popup windows are generally used for short descriptions or definitions. To make a popup window within Help Magician, simply define a hotspot as a "popup". Popup hotspot text will generally appear as green dotted-underlined text. When you *test* a hotspot that is a popup, Help Magician will show that topic on its own page, rather than show it in a popup window. When viewing the compiled help file, the popup will show up normally.

Main Windows

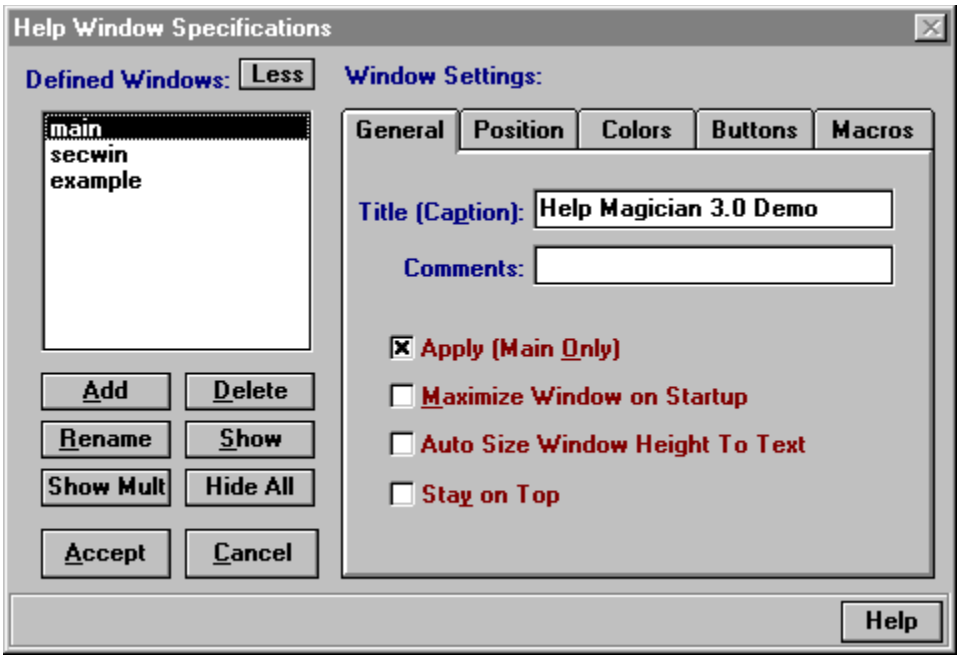
Main windows normally feature a menu bar and below the menu bar, a toolbar. In Windows 3.1, the toolbar contains the buttons Contents, Search, Back, History, and optional Browse buttons. In Windows 95, the toolbar typically contains Help Topics, Back, Print, and optional Browse buttons. There can be only one main window in WinHelp. By default all help topics are displayed in the main window. In Windows 95 and Help Magician Pro 95, you can define a "default" main or secondary window for each topic.

Secondary Windows

Secondary Windows in Windows 3.1 were limiting and could have only 5 defined secondary windows and only one could be displayed. In Windows 95, you can define up to 255 secondary windows and up to 9 can be displayed at one time. You can define scrolling and non-scrolling windows as well as background colors. In Windows 95, secondary windows can have button bars just like the main window. They can also be made to autosize based on the length of the topic being displayed in them. You typically would use secondary windows to augment the information contained in the main help window.

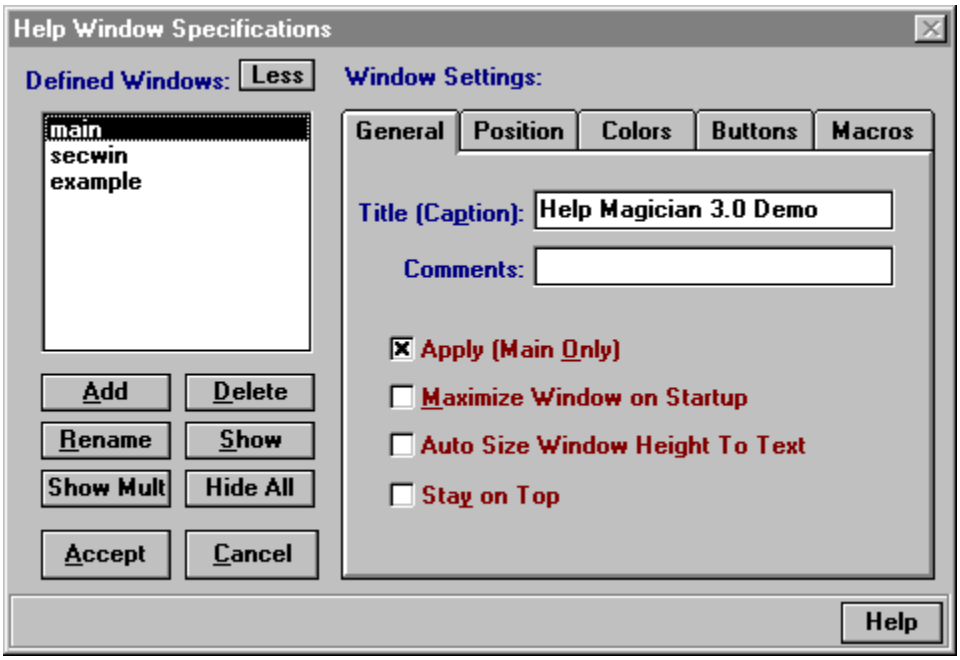
Visual Layout

Help Magician Pro allows you to visually layout your main and secondary windows relative to each other on the screen. You can modify window attributes such as colors, window title (which appears at the top of the help window), position, and buttons (Windows 95 only). To define help windows in Help Magician, click on the **Options** menu and then click on **Windows**. The following Help Window Specifications screen will appear.



Click on the desired area of the Window Definitions Bitmap for specific Help

General Tab



A list of defined windows will appear in the Defined Windows list. There will always be a "main" window. The "main" window cannot be deleted or renamed.

If you are working with multiple files within Project Management in Help Magician, all windows defined in the project will appear in this list. Be careful when deleting or altering windows because that action may affect other members of the project. If another user on the network opens a project member file, the specifications you define here will be available to the other users. In Project Management mode, all window specifications are stored in the HMP file in the [WINDOWS] section (you should not need to modify this section manually).

More or Less

The Help Window Specifications window can be collapsed or expanded to full size. You would collapse the window to make room on your screen while visually resizing and positioning your defined help windows. For convenience, the Help Window Specifications window will always appear on top of other windows.

To collapse the Help Window Specification window, click on the **Less** button. To expand the Help Window Specification window, click on the **More** button. The More/Less button is a toggle button and will change according to the state of the Help Specifications Window.

Adding a New Secondary Window Definition

1. Click on the Add button.
2. You will be prompted for a new window name. Enter it here and click on Accept.
3. The new defined window will appear in the Defined Windows list.
4. Now you can setup window attributes such as title, position, colors, buttons, etc.

Deleting a Secondary Window Definition

1. To delete a window definition, click on a defined window name in the Defined Windows list.
2. Click on the Delete button.

3. Confirm that you do want to delete the window definition.

Notes about deleting windows
A. You cannot delete the main window.
B. Be careful when deleting windows in multiple file Project Management mode. Your action could affect other members of the project.

Renaming a Secondary Window Definition

If you make a mistake in defining a secondary window name, use the rename feature to change its name.
1. To rename an already defined secondary window, click on its definition name in the Defined Windows list.
2. Click on the Rename button.
3. You will be prompted to type in a new name. Click on Accept.

Showing the Defined Windows on the Screen

When designing windows for your help file, you can view what the windows look like in relation to each other and the entire screen. When you show a window, you can also position and resize it (see section below). Like WinHelp, Help Magician Pro 95 will only show the windows that you're allowed to see (1 main and 1 secondary for WinHelp 3.1, or 1 main and up to 9 secondary for WinHelp 95).

1. To show a window, click on a defined window name in the Defined Windows list.
2. Click on Show.
3. To Show Multiple windows, hold down the Ctrl key then click on the windows you wish to show in the Defined Windows list. Then click on the Show Mult button. Help Magician will display all your selected windows.

The Hide and Show commands have no effect upon the final compiled help file. These commands are for your convenience in designing your windows.

Hiding Defined Windows

1. To hide a window, click on a shown window listed in the Defined Windows list.
2. Click on the Hide button (if no window is shown, there will not be a Hide button, instead there will be a Show button).

OR

1. Double-click on the shown window's icon in the upper left corner of the window.

To hide all the shown windows, click on Hide All.
The Hide and Show commands have no effect upon the final compiled help file. These commands are for your convenience in designing your windows.

Setting the Window Title (Caption)

You can specify the text that appears at the top of each window that is shown in WinHelp. This area is known as the Window Title.
1. To set the Window Title, select the window you want to put a title in from the Defined Windows list.
2. Click on the General Tab.
3. Type in the Window Title in the "Title (Caption):" textbox and press Enter.

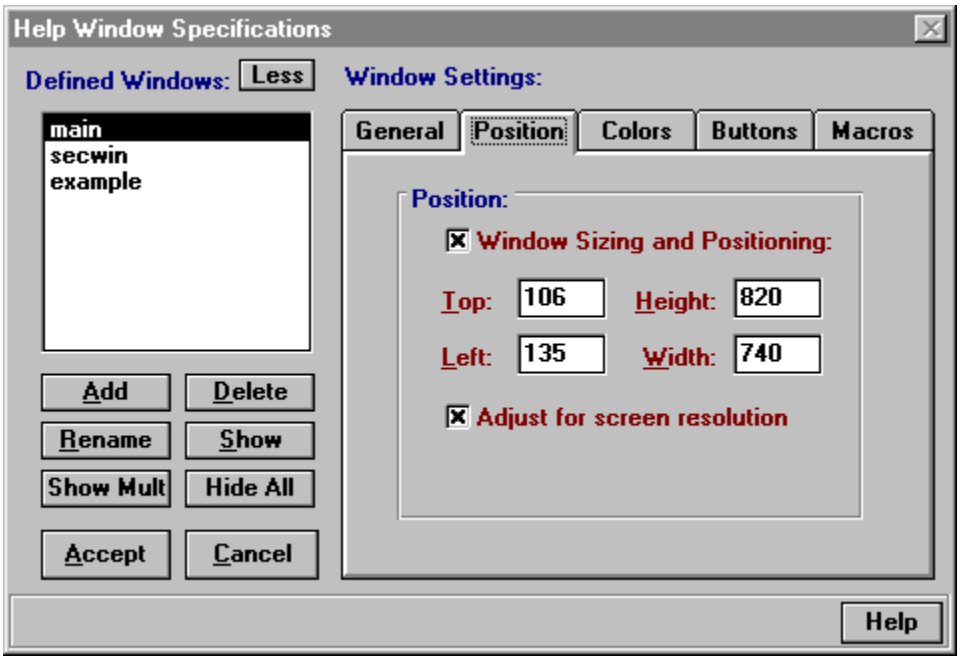
Another menu option in Help Magician allows you to set the title of the main window. It is called the Help System Title and it's in the Options...Appearance menu. The Help System Title there is the same as the Main Window Caption here.

See Also
- Buttons Tab
- Colors Tab
- Macros Tab
- Position Tab

Position Tab

Setting the Window Size and Position

The size and position parameters tell WinHelp where to display the window when it is initially displayed. There are two ways you can set a window's size and position. The first is to manually enter screen coordinates and the other is to visually lay out the window. There may be times where you want a secondary window to appear right next to the main window. The trick here is to show both windows, then move them close together.



Manually entering coordinates

1. Click on the window you wish to modify size and position in the Defined Windows list.
2. Click on the Position tab.
3. Enter screen coordinates in the appropriate textboxes, followed by the Enter key for each textbox.

The maximum value is 1023 and the minimum value is 0. The values could be in "help units" or pixels, depending on the state of the "Adjust for screen resolution" checkbox (see below). With this checkbox checked, a window that takes up the entire screen would have Top and Left coordinates 0 and Height and Width coordinates 1023, regardless of the resolution of your screen.

4. If the window is shown, the window will move to the new coordinates or be resized.

The **Adjust for screen resolution option** should be checked for normal use. This tells WinHelp to use the coordinates as a percentage of screen size. Thus, if you create a window that is 512x512 help units, then it will always occupy one-half of the screen, regardless of the user's screen resolution. If you need precise pixel control over your help window, make sure the "Adjust for screen resolution" checkbox is unchecked, then the values in the textboxes will be in pixels. Precise pixel control is useful in situations where you've created a bitmap for a lower resolution display and you want to display it in a help window on a higher resolution display. When the Adjust for screen resolution is checked, there would be white space around the right side/lower part of the bitmap in the help window on a higher resolution display. With precise pixel control, the window size is kept at an absolute value and is not resized on a higher resolution display.

Note that in WinHelp, text and bitmaps are not sized within the help window, just the window itself.

Visually laying out the Windows

1. Click on the desired window in the Defined Windows list.
2. Click on the Show button.
3. Optionally show another window if you want to move them in relation to each other.
4. Optionally click on the Position tab to view the coordinates of the window you're moving/resizing.
5. Now, with the mouse, resize and/or move the window just like you would with any other Windows window.
6. If the window you're moving is selected in the Defined Windows list and you clicked on the Position tab, it will show the updated coordinates after you're done moving/resizing the window.

Note that you can also change the background color of the window by double-clicking the mouse inside the non-scrolling or scrolling area of the window.

See Also

Buttons Tab
Colors Tab
General Tab
Macros Tab

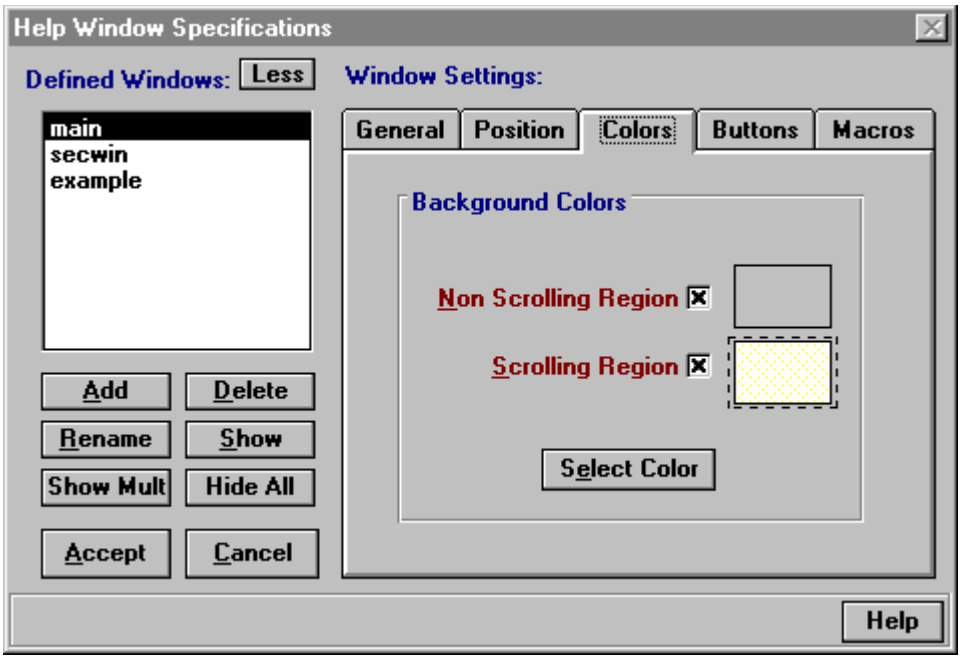
Colors Tab

Setting the Window Background Colors

Each window in WinHelp contains two areas: the non-scrolling region and the scrolling region. The scrolling region is where you display your topic and this area can be scrolled if your topic is larger than the window height. The non-scrolling area is above the scrolling area and is used for displaying banners, logos, headlines, or whatever. This area is fixed and does not scroll.

You can set the background colors for each of these two areas. When choosing colors there is an important issue to consider when you're developing a help file that will run on both 16-color standard VGA systems and 256-color Super VGA systems. It is recommended that you stick with the basic 16 colors, otherwise a color from the 256-color palette will appear dithered on systems that can display only 16 colors. Dithered backgrounds make the text very hard to read. If you know that your help file will only be displayed on 256 color systems, indulge yourself!

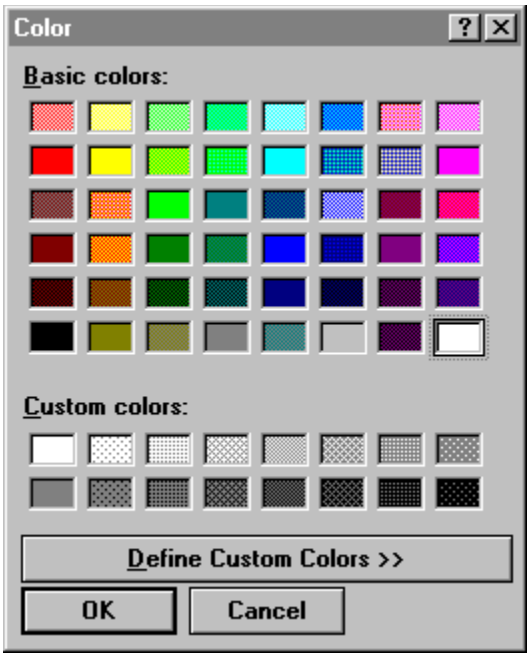
Start Tip: You can set a non-scrolling region within a topic by selecting Non-Scrolling Region from the Format menu on the main Help Magician screen.



1. To set the background colors, click on the window in the Defined Windows list.
2. Click on the Colors tab.
3. Double-click on either the Non-scrolling region or scrolling region color box. You will be shown the color palette dialog below.

OR

1. If you have a help window shown, you can change the color of either the non-scrolling or scrolling region by double-clicking inside the desired region in the window. You will be shown the color palette dialog below.



Select a color and click on OK.

Other Window Settings

There are some other settings that are available for help windows. They are available within the General tab.

Comments

This field is for the help author only and will not be used in the final compiled help file.

Apply (Main Only)

Clicking this option will tell the help system to use the settings you make here for the main window. If you do not check this option, WinHelp will use the last known help window size and position (stored in [Windows Help] section in win.ini). The Apply option works with the main window only.

Maximize Window on Startup

This feature will cause the window to fill the entire screen when it first appears. If the user "normalizes" the window, the sizing specified in the Position tab will be used.

Auto Size Window Height to Text

A very useful feature that causes WinHelp to automatically resize the secondary window height to fit all the text in it. Autosizing is not available for the main window. You cannot have midtopic jumps to secondary windows that are set for autosizing. If you do, the jumps become ineffective.

Stay On Top

This option causes the window to stay on top, regardless if you switch to another application. You should avoid using large windows with the feature because it takes over the screen.

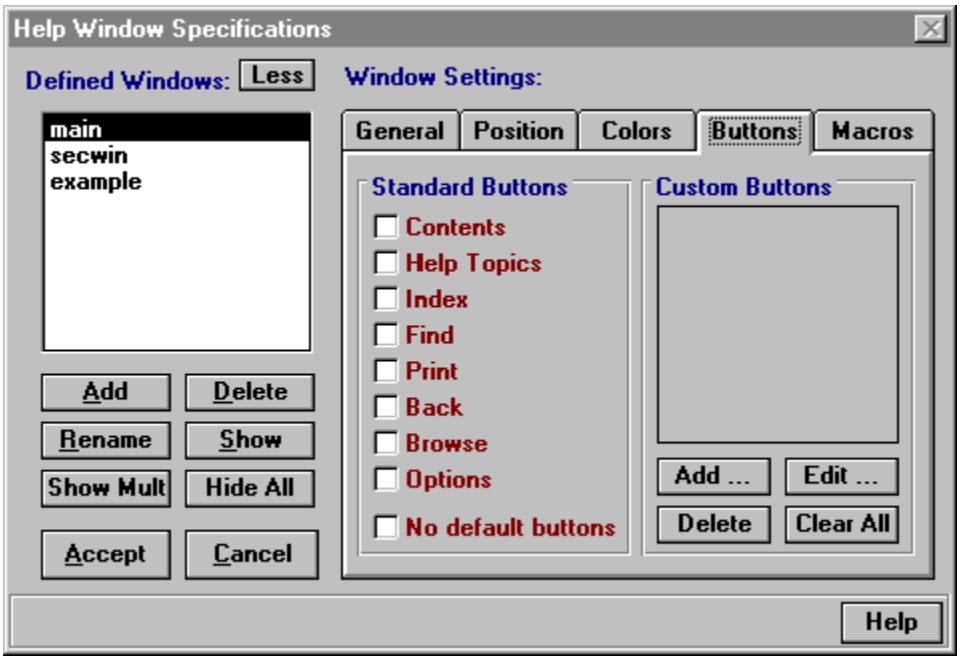
See Also
Buttons Tab
General Tab
Macros Tab
Position Tab

Buttons Tab

Defining Buttons for the Help Window

(Windows 95 feature only)

The Buttons tab allows you to set up the default button bar that will appear on the selected window when it is initially displayed. In WinHelp 95, secondary windows may have button bars. There are two button types that can be defined for each window-standard and custom. Standard button types are pre-defined buttons that perform pre-defined functions. Custom buttons are buttons that you can define and can be made to execute any of the WinHelp macros. At least one standard button must be defined before you can define any custom buttons. Up to 22 buttons can be defined per window. Click on No Default Buttons to disable displaying all default buttons for the currently selected window.



Defining Standard Buttons for a Selected Window

1. Select a defined window in the Defined Windows list.
2. Click on the Buttons tab.
3. Click on the button options you want in your help window.

Defining Custom Buttons for a Selected Window

1. Select a defined window in the Defined Windows list.
2. Click on the Buttons tab.
3. Make sure you have at least one of the eight standard buttons checked.
4. The listbox in the Custom Buttons frame contains all the custom buttons defined for the selected window. If you click on one of the buttons in the list, the macro assigned to that button will be shown in the status bar at the bottom of the form. To delete a defined button, highlight the button in the list and click on Delete. To Edit an existing button, click on the button to edit and then click on Edit... To add a custom button, click on Add Button. An Add or Edit operation will bring up the Create Button dialog box:

5. Enter a button id for the button. It can be anything you want to name it. If you were to use any of the WinHelp macros that would affect this button at runtime (DestroyButton, DisableButton, or EnableButton), the parameter for these macros would require the button id. If you did nothing with the macros, then the button would be enabled all the time.
6. Enter the caption for the button, i.e. the text that would appear inside the button. You should keep your text as small as possible, like a single word.
7. Select a macro to execute when the button is pressed at runtime. You must have previously defined a macro set using the Macros...Define Macros option from the main menu. The "Macro to Execute" option allows you to choose from the macro definition list in Help Magician. Each macro definition can contain one or more WinHelp Macros. You cannot select a WinHelp macro directly for the "Macro to Execute" option.
8. Click on Accept and you'll return to the Window Definition dialog. To add more buttons, repeat steps 4-8 as necessary.

See Also
Colors Tab
General Tab
Macros Tab
Position Tab

Macros Tab

Defining a Macro to be Executed when the Window is Displayed

(Windows 95 feature only)
The Macros Tab allows you to assign Macro Definitions to a window. These macros will be executed before the window is initially displayed. Clicking on the Macros tab will show the following information.

Defining a Window Specific Macro
1. You must have previously defined a macro to be executed by using the Macros...Define Macros command on the main menu. See Macros, Working with on how to do this.
2. Select a defined window from the Defined Windows List.
3. Click on the Macros tab.
4. Choose one of your previously defined macros from the macro drop-down list.

Note: You can type any comments about this macro in the Comments field. It will not be displayed in the compiled help file.

Make sure you select a window from the list box before setting any attributes. Failure to do so will result in the currently select window being modified, rather than the intended window.

Using the Defined Windows Within Your Help File

There are many ways you can reference and use these windows within your help file. They can be used as part of a Jump Destination to force a topic to appear in it; they can be used as part of the Link Destination in a SHED file (or segmented hotspot graphic); they can be used as a default window for each help topic; or they can be used as part of the call to WinHelp.

Referencing a Window in a Jump Destination
When you define a hotspot jump, there is an option on the Jump Options screen that allows you to define which window will be used to display the topic. You can select from any of the windows you've defined by the methods in this section. If you don't select any windows for the jump, the default window ("main") is used. If you've selected a window for that hotspot jump, the topic will appear in the selected window when the user clicks on the hotspot. (See Jumps and Popups).

Referencing a Window in a SHED file
When you define multiple hotspots on a bitmap graphic using Microsoft's SHED editor, you can specify which window the topic will appear in. In the hotspot attributes dialog, select Type "Jump", Attribute "Invisible", and enter the Context String of the topic you want to display. After the Context String text, add ">windowname", where windowname is the name of one of the defined windows in your help file (do not type the quotation marks after the context string text!). For example, if the Context String of the topic you want to display is "Company Directory" and the Window you want to display that topic in is called "Directory", you would enter the following text in the Context String box: Company Directory>Directory

Referencing a Window as a Default Window for each Help Topic
When Help Magician is placed into Windows 95 support mode, the Utility bar supports the selection of a default help window for each topic that is shown in the editor. All the windows defined by the methods in this section are available in the combo box at the right, just above the editor. Simply select the Window you want this topic to be displayed in when it is shown.

Referencing a Window in a Call to WinHelp
For Advanced Programmers Only. The WinHelp API call has a feature that allows you to specify a window for a help topic to be displayed. Its reference syntax is basically the same as the reference in the SHED

discussion above. When you pass the name of the help file to WinHelp, you can add the ">windowname" suffix to that string.

For example: CALL WINHELP(hWnd, "myhelp.hlp>secwin", 0, 0) displays the contents page of myhelp.hlp in a secondary window called "secwin".

Specifying a Default Window Type for All Topics

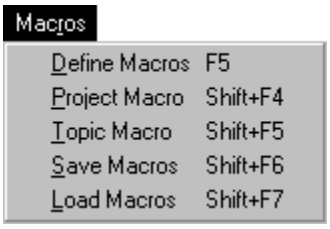
(Windows 95 feature only)

You can specify which window to use as the default window when it is opened from the Context, Index, and Find tabs. You can set the ":Base filename.hlp>window" option in the .CNT file. For an easier way to set this, see the Default Help File and Default Help Window options in Software Interphase's WinHelp '95 Contents Editor package.

See Also

- Buttons Tab
- Colors Tab
- General Tab
- Position Tab

Macros, Working with



Click on the desired area of the Macros Menu for specific Help

Macros

The ***Macros*** menu allows you to perform maintenance on macros that can be used in Jumps, Popups, Topics and the Project.

Note: If you are defining macros in Project Management mode, any macros you create, modify, or delete will affect other members of the project- so be careful what you do!

The user has the ability to use pre-defined macros (prototypes) to manipulate and control the Windows Help System. The macros allow control of the Help System menu items and buttons and even creation of new buttons.

For example, a glossary button may be added to the Help System main menu which displays a glossary in a secondary window. For instance, this could be accomplished by using the CreateButton macro that would create the glossary button and place it on the Windows Help main menu at run-time. The button itself could be defined to call a JumpID macro that would display a glossary topic in a secondary window. The programming possibilities are vast.

See Also

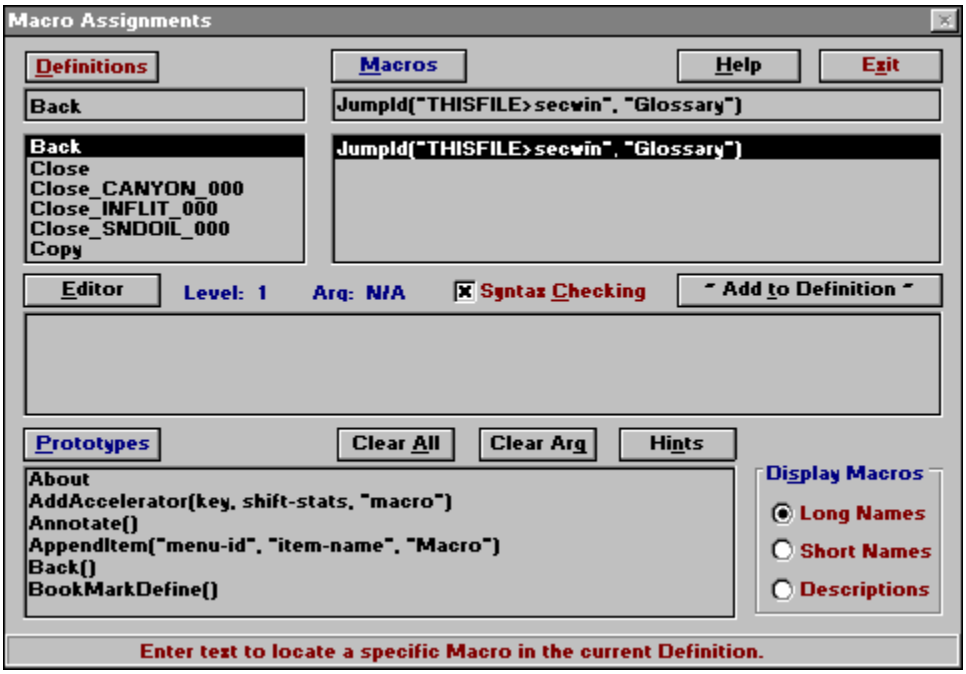
- Macro Form Definitions
- Step by Step
- Macro/Keyword Associations

Define Macros

Define Macros

Selecting Define Macros from the menu displays the Macro Assignments form. This form provides all the functions necessary to create, modify, and maintain macros for your help system.

26 new macros have been added for WinHelp 4.0. Definitions for these macros can be found in the Help Magician's on line help system under **Macro Reference**.



Step by Step

A. Create a Macro Definition.

1. Select the Definitions button.
2. Select Add from the menu.
3. Enter a name for the Macro Definition (ex. "Play_Music")
4. Select the Accept button.

B. Create Macros and add them to the Definition.

1. Create a macro in the Editor using one of the following ways:

- Copy a macro Prototype to the Editor by double clicking on it or using the Prototypes menu.
- Copy an existing Macro from the Macros list to the Editor using the Macros menu.
- Turn Syntax Checking off and enter a macro from scratch (not advised for novices).

2. Edit the macro in the Editor.

- Click on or move over each argument, if they exist, and either edit the argument or use the Hints button to choose a value from a list of options.

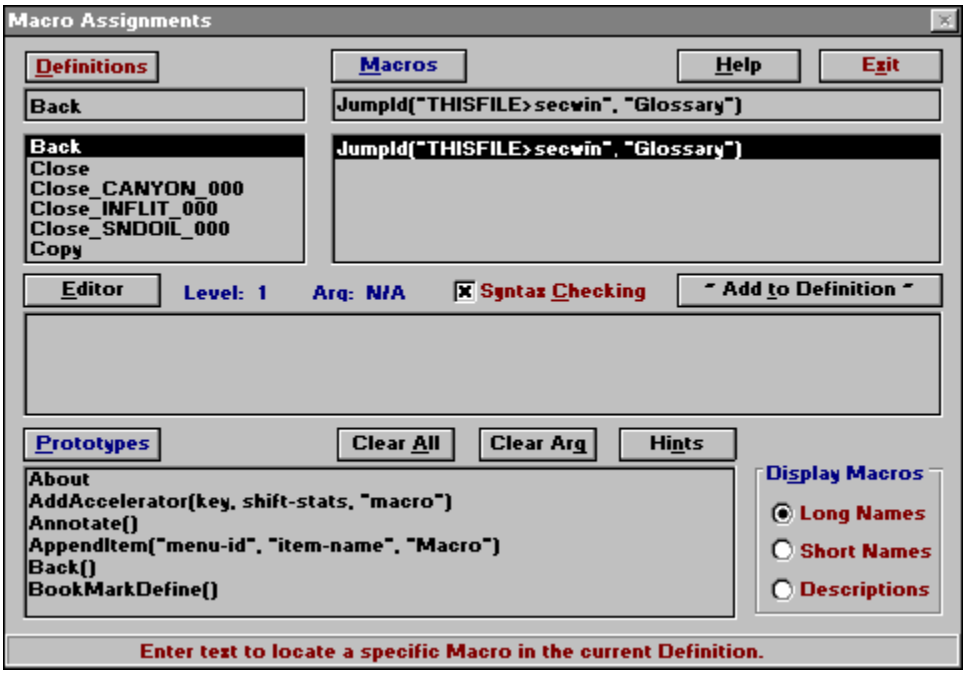
3. Add the macro to the Macros list using one of the ways below:

- Select the Add Macro to List button.
- Select the Editor button and then Add Macro to List.
- Type Ctrl-W.

See Also

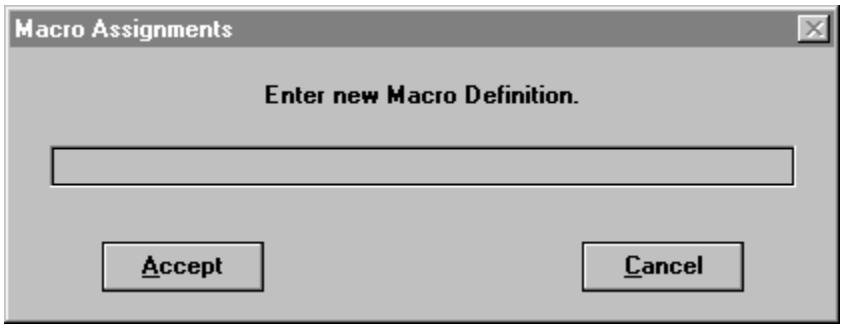
Macro Form Controls

Macro Form Controls



Click on the desired area of the Macro Definintion Form for specific Help

Definitions Button



Add

Each macro or group of macros must be assigned a name or unique identifier called a "Macro Definition". This is accomplished by selecting the Definitions button and then the Add menu item. A form will appear allowing you to enter a new Macro Definition. Enter the definition and click on the Accept button. After adding a Definition, macros may be added by editing a macro prototype or creating a new macro from scratch.

Delete

Removes a Definition from the Definitions list box. Highlight a Definition and select this menu item.

Rename

Allows editing a Definition name. Duplicate names are not allowed.

Copy

Duplicates a Definition and its associated macros. You will be prompted for a name for the new definition.

Definitions List

The definitions list box contains the names of the macro definitions used by the Help Magician. Each definition can contain one or more macros. To add a definition, click on the Definitions button and select Add from the menu.

Locator Fields

There are text boxes above the list boxes used to display Definitions or Macros, respectively. They are used to quickly locate items in the list by typing some of the characters in the Definition or Macro. The list will automatically scroll to the item most closely matching the entry in the text box.

Macros Button

Copy To New Macro / Copy To Editor

Copies the currently selected macro to the Editor. Once in the Editor, the macro may be modified and added to the list. If syntax checking is turned on, the macro will be validated before placing it into the Editor.

If there is a macro already in the Editor, a sub menu can be displayed that allows you to choose a position to place the currently selected macro into the Editor. If syntax checking is turned on, the macro will be validated before placing it into the Editor. The insertion options are listed below:

At Beginning - Chains the macro before the text in the Editor.

Append to End - Chains the macro after the text in the Editor.

Replace All - Replaces all the text in the editor.

Before Argument - Chains the macro before the currently selected argument.

After Argument - Chains the macro after the currently selected argument.

Replace Argument - Replaces the currently selected argument with the macro.

Current Position - Inserts the macro into the current cursor position in the Editor text. (only available if syntax checking is turned off)

Delete

Removes the currently selected macro from the list.

Edit

Allows you to modify the currently selected macro in the Editor. Prompts you before erasing the contents of the Editor. Select Done to update the current macro with the edits or Cancel to abort any changes.

Move

Allows you to change the execution order by repositioning the macros in the list. To move a macro, drag it with the mouse to the desired position. Alternatively, you may select the macro and then this menu item and then use the arrow keys to re-position it. When you are done select either Done or Cancel.

Macro List

Located just below the Macros button, this is a list of macros that are assigned to the currently highlighted Definition. The macros will be executed in the order that they appear in the list. There are several ways to add a macro to the list:

1. Create a macro by editing a Prototype and then add it to the list.
2. Create a macro from scratch by turning off Syntax Checking and then add it to the list.
3. Copy an existing macro to the Editor and then add it back to the list.
4. Copy an existing macro to the Editor, modify it and then add it back to the list.

Help button

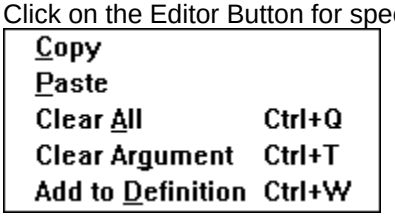
Displays help for creating and maintaining macros in the Help Magician. Context sensitive help is available for each function on the form by selecting the item and pressing F1 function key.

Exit Button

Select the Exit button to return to the main form of the Help Magician. Any changes will be accepted and can be permanently saved by typing Ctrl+S or selecting Save from the Help Magician main form File menu. If you are using macros in a project, the macros will be stored in a HMP file and be made available to all other project members. Make sure that the macros you edit will not affect other project members.

Editor Button

Click on the Editor Button for specific help



Copy

Copies the currently highlighted text to the Clipboard. Markers and special characters will not be copied, however.

Paste

Pastes the current contents of the Clipboard to the current cursor position in the Editor. Markers and special characters will not be pasted, however. This feature is only available when syntax checking is turned off.

Clear All

The Clear All button removes all text from the Editor.

Clear Argument

The Clear Arg button removes the currently selected argument within the macro in the Editor.

Add to Definition

Adds the macro in the Editor to the Macros list for the currently selected Definition. If no definitions exist, an error message will be displayed.

Editor

The macro editor is where you will edit the arguments in a macro. With Syntax Checking on, the editor will not allow an editing error. Hints are available for the standard arguments.

Macro Status Indicators

The Macro Status Indicators are located just to the right of the Macros button.

Level

Displays the current argument nesting level. There are a maximum of five levels of macro nesting.

Arg

Displays the current relative argument position within the current macro level.

Syntax Checking

The Syntax Checking check box is located just to the right of the Status Indicators.

This feature forces syntax checking on the currently edited macro or macros. If a macro from an existing Definition is edited, the macro will be syntax checked before being place into the Editor. If there are syntax errors then the macro will still be placed in the Editor and syntax checking will be turned off.

To create or edit a macro "free hand", which is not advised, simply turn off syntax checking and perform any modifications manually. When completed, please turn syntax checking on to validate the macro before adding it to a Definition. This will save development time by catching any syntax errors before compiling the file and producing potentially ambiguous and confusing errors.

Paste operations are not allowed with syntax checking enabled.

Prototypes Button

Insert at Beginning
Chains the currently selected Prototype before the text in the Editor.

Append to End
Chains the currently selected Prototype after the text in the Editor.

Edit Prototype
Places the currently selected Prototype into the Editor.

Insert Before Arg
Chains the Prototype before the currently selected macro argument in the Editor.

Insert After Arg
Chains the Prototype after the currently selected macro argument in the Editor.

Replace Arg
Replaces the currently selected macro argument in the Editor with the Prototype.

Current Position
Places the Prototype into the current cursor position in Editor.

- To quickly add a Prototype to the Editor, double click on the desired Prototype.
- To chain a Prototype before a macro in the Editor, place the cursor on the macro name, before the first left parenthesis "(", and then double click on the Prototype.
- To chain after a macro in the Editor, position the cursor after the macro's name, after the left parenthesis, and then double click on the Prototype..
- To replace a macro argument in the Editor with a Prototype, position the cursor anywhere between the markers enclosing the argument, and then double click on the Prototype. If the argument is a macro that contains other macros as arguments, place the cursor on the macro's name before the left parenthesis and double click.

See Also
Macro Reference

Prototypes List

The Prototypes List, located just below the Prototypes button, contains the standard WinHelp macros and macros necessary to play multimedia files. These macros contain arguments that will be replaced with the appropriate data when edited in the macro editor, described later.

Edit Prototype / Replace All

Places the currently selected Prototype into the Editor. If the menu displays "Replace All" then the entire contents of the Editor will be replaced with the currently selected Prototype.

Hints / Macro Ref

The Hints button provides lists of options for macro arguments in the Editor. Syntax checking must be turned on to enable this function. When selected this button can provide a list of valid options for certain arguments, such as "filename", "menu-id", "acckey" and so on. To use this feature simple select the desired option from the list and select Accept. The option will replace the previous argument contents, if any existed.

When the Prototypes list box has the focus, the Hints button label is changed to "Macro Ref". Clicking on this button calls the Help Magician's on line help system with complete help for all the macros in the Prototypes list.

Display Macros

Use Short Names

This feature controls the display of the Macro Prototypes in the following manner:

Long Names	Displays full macro name.
Short Names	Displays abbreviated macro name (useful for combined macros where the 255 character limitation is a factor).
Descriptions	Displays a description of the macro.

WinHelp 4.0 will internally change all long name macros into their short names during the compilation process.

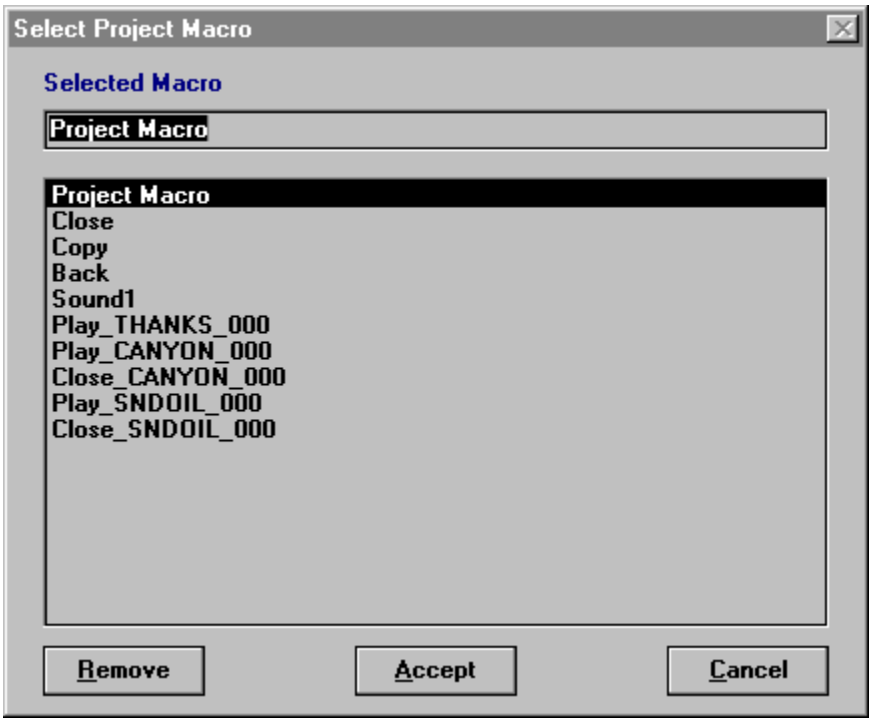
Help Label

The label at the bottom of the Macro window displays "quick help", and hints on macro arguments.

Project Macros

Project Macros

A macro definition that contains one or more macros may be assigned to run automatically when the finished help system is loaded via WinHelp. For example, you may wish to do this to create new buttons upon startup of the Help system. This is accomplished by assigning the macro definition to the project in Project Macros under the Macros main menu item. The Select Project Macro form will appear allowing you to select a macro definition that you previously created in the Macro Definitions form, and assigning it to the project by choosing the Accept button. To remove a previously assigned macro definition, simply select the Remove button and then Accept.



Topic Macros

Macro definitions may be assigned to a help topic so that the macro or macros that are assigned to the macro definition will be executed anytime the help topic is displayed in WinHelp. Note that in WinHelp, topic macros are executed before the actual text is displayed in the WinHelp window. To **assign** a macro definition to a help topic, simply display the help topic in the Help Magician main form and select Topic Macros from the Macros main menu list. The Select Topic Macro form will appear allowing you to select or remove a macro definition, previously created in the **Macro Definitions** form, from the list.

Save Macros

You can load macro definitions that were defined for another help system and save the set defined for your current system to be shared with other projects. The file form will prompt for files with the default extension for styles .HLM.

Load Macros

You can load macro definitions that were defined for another help system and save the set defined for your current system to be shared with other projects. The file form will prompt for files with the default extension for styles .HLM.

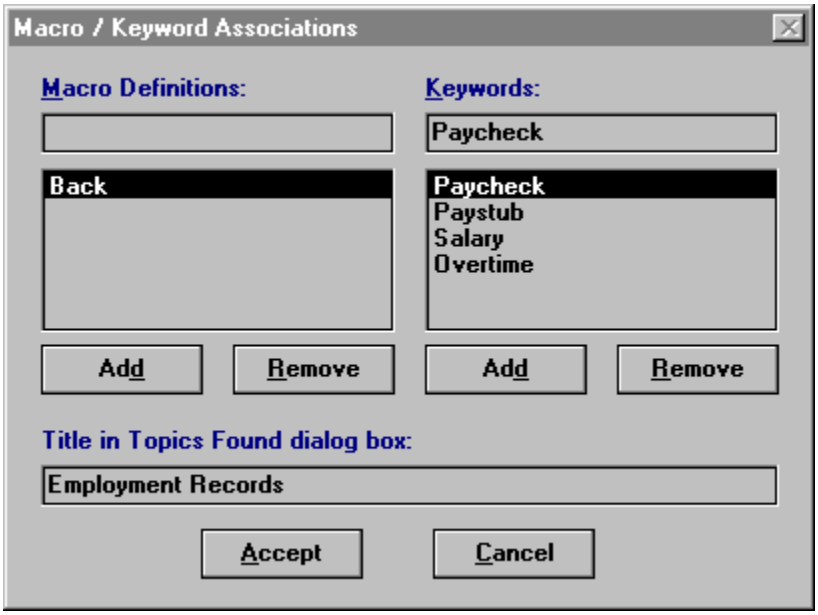
To **merge** other macro definitions from another Help Magician project into your current project, simply **load** the .HLM file using the Load macro definitions procedure described above. Any duplicate macro definitions will be ignored and not loaded.

Macro/Keyword Associations

Start WinHelp 4.0 allows help authors to associate keywords and macros to extend the programming capabilities of the Help search system. The basic idea is to allow selected macros to run after a user has selected an Index entry (i.e. a keyword) from the Index dialog box and have the macros complete execution before the destination topic is displayed. This feature allows the Help author more control over the manner in which certain topics are displayed and also provides a way of possibly introducing a topic with multimedia effects or running a utility program, a wizard or a DLL.

To associate a macro with a keyword or keywords, you must first create some macro definitions using the macro editor. Then you will need to add keywords to your topics. After you have done this, select the Macros main menu item, then Keyword Macro submenu item. The Macro / Keyword Associations window should appear as is below, but without the items in the lists.

Start Since one macro definition can have potentially many keywords associated with it, this form is set up so that selecting a definition will display the keywords mapped to it. This method greatly simplifies managing the list.



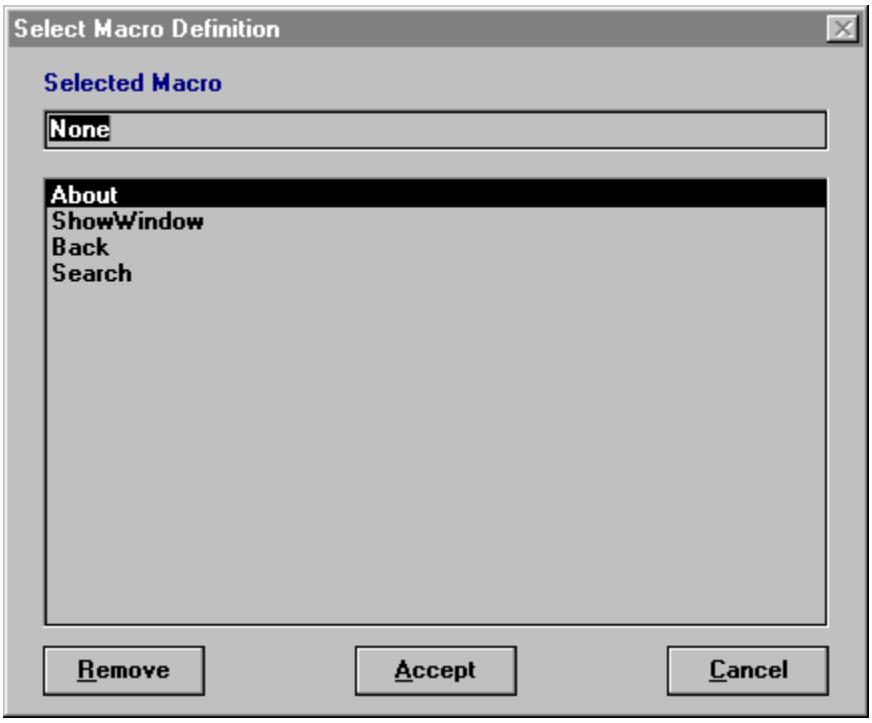
Keywords

Select the Add button to add a keyword to the Keywords list. To remove a keyword from the list, select the keyword and then the Remove button.

Macro Definitions

First select the keyword you wish to associate a macro with, then select the Add button to add a macro definition to the list. The Select Macro Definition dialog box will be displayed as seen below. Select a macro from the list and then click on the Accept key or hit Enter. The macro should be added to the list.

To automate the association process, several keywords may be associated to one macro definition. Simply select several keywords and then select a macro definition to associate them with. To remove a macro definition from the list, select the definition and then the Remove button. Any keyword associations will be removed as well.



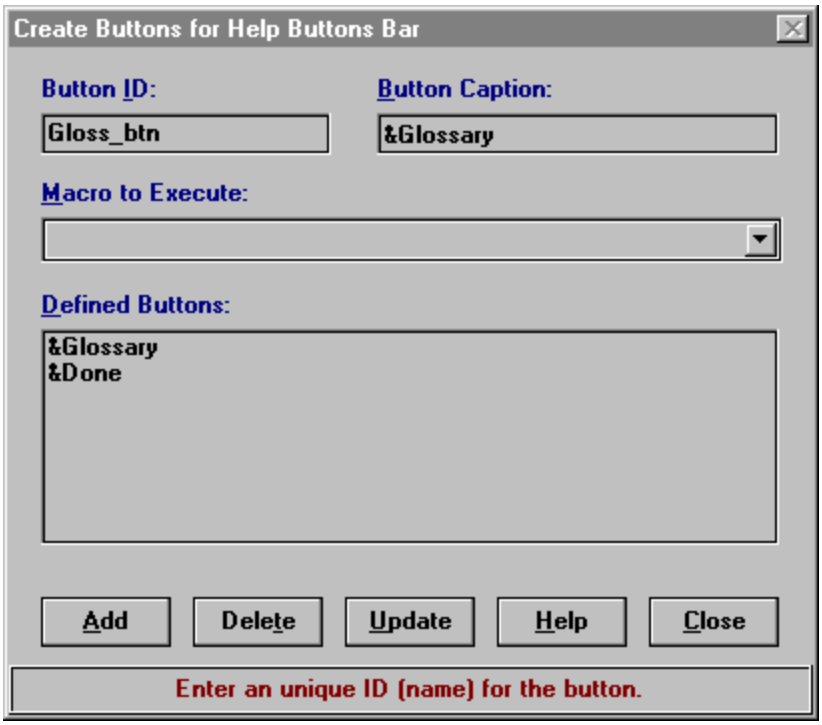
Title in Topics Found dialog box

An Index entry, or keyword, may be associated with several topics and consequently will display the Topics Found dialog box after the Index entry is selected. The title that is displayed in the Topics Found dialog box can be entered in this text box. Every association can have its own title by selecting a macro definition then entering a title for the dialog box.

Create Button

The **Create Button** feature is used in Windows 3.1 only. Buttons can only be added to the Main window in Windows 3.1.

For the creating buttons under Windows 95 mode, see Defining Help Windows, Buttons Tab.



The **Create Buttons** form provides an easier, higher level way to add buttons to the help project than the **Macro Assignments** form.

These buttons are displayed just below the main menu along with the standard buttons, Contents, Search, Back, History, and the browse buttons, when the help system is viewed with WinHelp.Exe.

Button Id

Enter an **unique** name, or ID for the button in the **Button Id** field. This name is used internally by WinHelp.Exe and in low level macro programming.

Button Caption

Enter the **caption** that will appear on the button when the help system is viewed with WinHelp.Exe.

Macro to Execute

Select the **macro to be executed** when the button is pressed from the list of macros in the drop down box. In keeping with the simplicity of the **Create Button** form, the macros selected for this list box are limited to macros that do not require arguments or additional information.

More complex button macros may be added in the Macro Assignments form.

Defined Buttons

All of the buttons you have created will be listed in the **Defined Buttons** list box. Clicking on a defined button updates the display of the individual fields for that button. You can **edit** the fields for any of the defined buttons when the button is selected.

Add

To **add** a button to the list, fill in the fields as described above and press the **Add** button.

Delete

To **delete** a button from the list, highlight the entry in the Defined Buttons list box and press the **Delete** button.

Update

To **change** the definition of a button, edit the fields and press the **Update** button.

Help

Displays the Help Magicians help file beginning at the Create Button page.

Close

When you are finished creating or editing buttons for the help system, press the **Close** button. All entries will be recorded in the Macro database.

Integrated Test Mode

Introduction

The Help Magician's Integrated Test Mode is an invaluable, time saving tool in the development of your help system. You can instantaneously test the help file without the need to write the RTF file or to compile the RTF file with the Windows help compiler. Jumps to other topics, Jumps to secondary windows, Jumps to macros, Jumps to other help files, Popups, Browse sequences, and keyword links can all be tested in Test Mode.

The Editor

Jumps and Popups are displayed in the editor exactly as they will appear when the compiled help file is viewed with WinHelp. When the mouse cursor is moved over a link, the cursor changes to a hand, just as it would in WinHelp. Double clicking on the link will cause the editor to "Jump" to that topic.

Utility Help Buttons

The Utility Help Buttons, found in the Utility Area, provide the Search, Back, and History functions. They also provide access to the Browse and Links From forms.

SHED (SHG) File

If an image in the editor is a representation of an SHED file (Segmented Hypergraphics file created with SHED.EXE), double clicking on the image will display a table of the hot links contained in the file. The hot links can be Jumps, Popups, or macros.

The Help Magician does not distinguish between the hot link types contained in an SHG file. If the help topic title exists, double clicking on the hot link name or clicking the accept button will cause the Help Magician to jump to that page. An error message will be displayed if the title does not exist in the current help source file. If the title exists, the Help Magician will jump to the page and History window and Back button will be updated.

Clicking on the SHED button will launch the SHED editor. Any changes to the image hotspots will be reflected next time you double-click on the SHED image file.

Multimedia elements can be previewed in the editor by double clicking on the hot spot that references the multimedia element. All multimedia elements included in the macro definition will be played. Click in the Paragraph Style frame ("Click to Stop"), on the Main Menu, or anywhere in the editor to stop the process.

See Also
WinHelp Bar

WinHelp Bar

The WinHelp button bar, located just above the editor, provides the same functionality found in the button bar when a help file is viewed with WinHelp. The display of the WinHelp Bar is optional and can be set in Environment Options available from the Options menu.

This is the Windows 95 WinHelp Bar.

Contents

The Contents button displays the Contents or Index page of the help file as defined in Compiler Options.

Topics

When you click on the Topics button, you have a choice. You can either click on yes or no. If you click on yes, a simple CNT file will be created for you. If you click on no, you will be brought to the Contents Editor.

Search

The Search function works exactly like the Search function in WINHELP.EXE.

The upper list box contains all of the keywords in the help system. As you enter text into the text box, above the keyword list box, the highlighted bar moves to the keyword that matches the text entered. As you delete characters, the process reverses.

Keyword

You can select a keyword by highlighting the entry in the keyword list box and clicking on the Show Topics button or by double clicking on the keyword entry itself.

Topics

When you have selected a keyword, the titles of the pages linked to the keyword appear in the Topics list box. Select the topic by double clicking on it or by clicking on the Goto button. The selected help topic will be displayed and History and Back will be updated.

Index

Provides a comprehensive index to the topics in one or more Help files. The index is the list of keywords that you specify for each topic. A user can scroll to the desired keyword either by clicking the scroll arrows or by typing the word. If more than one topic is associated with the keyword, WinHelp displays the Topics Found dialog box that lists the associated topics. WinHelp version 4.0 now supports second level index entries and can combine the keywords from multiple Help files.

The Index tab replaces the Search dialog box in WinHelp version 3.1.

Back



The Back button will only be enabled if there are topics in the Back buffer. When you press the Back button, the help system will back up or retrace the steps you took to get to the current topic until the first topic has been reached. At that point, the Back button will again be disabled. The Back buffer will store the last 100 help topics.

History


Clicking on the History button will pop up a window containing a list of the topics you have visited. A total of 40 (actually 41, including the Contents page) will be stored in reverse order. The most recent topics are listed first. Double clicking on a topic causes the help system to display that topic. Clicking anywhere else on the form, or pressing the Esc key, closes the History window.

Print


The Print button allows you to print the document that is currently open.

Browse



The Browse Left and Browse Right Browse buttons will be enabled and disabled to reflect the ability to Browse in either direction based on the Browse sequences set up in the Browse function while designing the help system.

Testing and Navigating

Button Bar, Contents Tab, Index Tab, & Find Tab

Overview

Help Magician Pro 95 allows you to test your help file and offers many ways to navigate around your help file as you are developing it. Some ways simulate the way WinHelp works and other ways are unique to Help Magician Pro. Certain navigational aids within Help Magician Pro 95 will be available depending on what platform you are developing your help file for (i.e. Windows 3.x, Windows NT, or Windows 95). These navigational aids are discussed below. Note that when you navigate to a different topic, that topic will be shown in the Help Magician's editor window, regardless if the topic is a popup or is destined to be displayed in a secondary window.

Built-in Test Mode

Help Magician's integrated test mode is an invaluable, time-saving tool in the development of your help system. You can simultaneously test the help file as you are developing it without the need to write the RTF file or to compile the RTF file with the Windows help compiler. Jumps to other topics, jumps to secondary windows, jumps to macros, jumps to other help files, popups, browse sequences, multimedia objects, and keyword links can all be tested as you are developing.

When you create a jump or popup link within the editor, the link will appear the same in the editor as in the final compiled help file. When you move the mouse over a link, the mouse cursor changes to a hand, just as it would in WinHelp. Double-clicking on the link will cause the editor to "jump" to that topic. You can return to where you started from by clicking on the "back" button in the WinHelp Bar just above the editor. If the WinHelp Bar is not shown in the editor, you can enable it by setting the WinHelp Bar checkbox in menu Options...Environment Options in the Display Tab.

Goto ...

The Goto function is unique to Help Magician and offers a quick way to go to a topic by supplying certain information. You can "Goto" a topic by page number, topic title, context string, or context number. You can also goto the next jump, popup, mid topic context string, or unresolved hot link. You can also go to the next occurrence of a paragraph style or graphic image. The Goto option is available in the Edit...Goto Menu and is discussed more in depth in Managing Topic Text. A shortcoming of the goto method is that it is not visual in nature.

Visual Navigational Aids

Help Magician Pro offers several visual navigational aids. In WinHelp 3.x emulation mode, the WinHelp button bar is available, complete with the Contents, Search, Back, History, and Browse buttons. In WinHelp 95 emulation mode, the functionality of the Contents Tab, Index Tab, and Find Tab is available. In addition, Help Magician Pro 95 has a special feature, the LinkMap Navigator, which displays all your hot links graphically in a hierarchical tree.

WinHelp 3.x Emulation - WinHelp Button Bar

The WinHelp button bar, located just above the editor, provides the same functionality found in the button bar when a help file is viewed with WinHelp in Windows 3.x. The display of the WinHelp Bar is optional and can be set in Environment Options available from the Options menu.

Click on the desired button of the Windows 3.X button bar bitmap for specific Help

See Also

- Help Topics Browser
- Utility Help Buttons

Contents

The Contents button displays the Contents or Index page of the help file as defined in Options...Compiler Options...Index. For any help file, the Contents page must be set and that topic must have a context number defined (usually 1).

Search

The Search function works exactly like the Search function in WINHELP.EXE.

The upper list box contains all of the keywords in the help system. As you enter text into the text box, above the keyword list box, the highlighted bar moves to the keyword that matches the text entered. As you delete characters, the process reverses.

Keyword

You can select a keyword by highlighting the entry in the keyword list box and clicking on the Show Topics button or by double clicking on the keyword entry itself.

Topics

When you have selected a keyword, the titles of the pages linked to the keyword appear in the Topics list box. Select the topic by double clicking on it or by clicking on the Goto button. The selected help topic will be displayed and History and Back will be updated.

Back

The Back button will only be enabled if there are topics in the Back buffer. When you press the Back button, the help system will back up or retrace the steps you took to get to the current topic until the first topic has been reached. At that point, the Back button will again be disabled. The Back buffer will store the last 100 help topics.

History

Clicking on the History button will pop up a window containing a list of the topics you have visited. A total of 40 (actually 41, including the Contents page) will be stored in reverse order. The most recent topics are listed first. Double clicking on a topic causes the help system to display that topic. Clicking anywhere else on the form, or pressing the Esc key, closes the History window.

Browse

The Browse Left {ewc hmew2,ewBitmap2,BRWSLEFT.BMP} and Browse Right {ewc hmew2,ewBitmap2,BRWSRGHT.BMP} Browse buttons will be enabled and disabled to reflect the ability to Browse in either direction based on the Browse sequences set up in the Browse function while designing the help system.

Utility Help Buttons

The Utility Help Buttons, found in the Utility Area, provide the Search, Back, and History functions. They also provide access to the Browse and Links From forms. In Windows 95 support mode, Help Magician replaces these buttons with a default window definition for each topic.

SHED (SHG) File

If an image in the editor is a representation of an SHED file (Segmented Hypergraphics file created with SHED.EXE), double clicking on the image will display a table of the hot links contained in the file. The hot links can be Jumps, Popups, or macros.

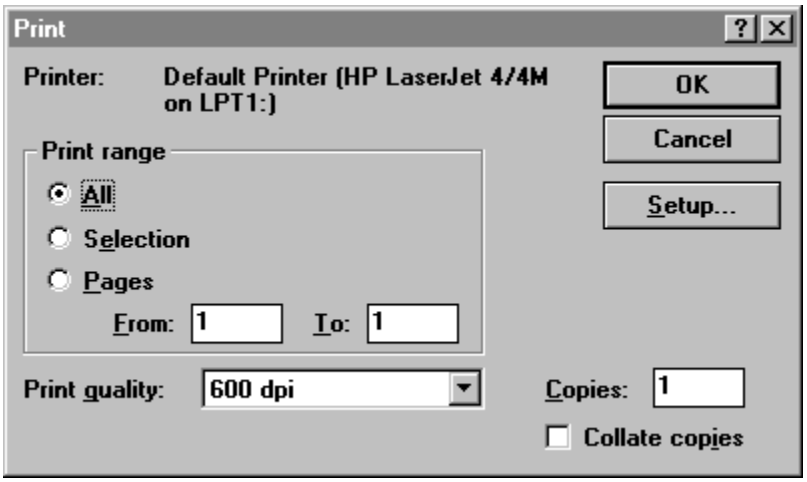
The Help Magician does not distinguish between the hot link types contained in an SHG file. If the help topic title exists, double clicking on the hot link name or clicking the accept button will cause the Help Magician to jump to that page. An error message will be displayed if the title does not exist in the current help source file. If the title exists, the Help Magician will jump to the page and History window and Back button will be updated.

Clicking on the SHED button will launch the SHED editor. Any changes to the image hotspots will be reflected next time you double-click on the SHED image file.

Creating Reports

Print

Selecting the ***Print*** sub menu from the ***File*** menu brings up the Print dialog.



Select the range of help topic pages to be printed, the print quality, the number of copies, and click on ***OK*** to print the selected topics. Text and graphics will be printed approximately the way they appear in the editor.

See Also

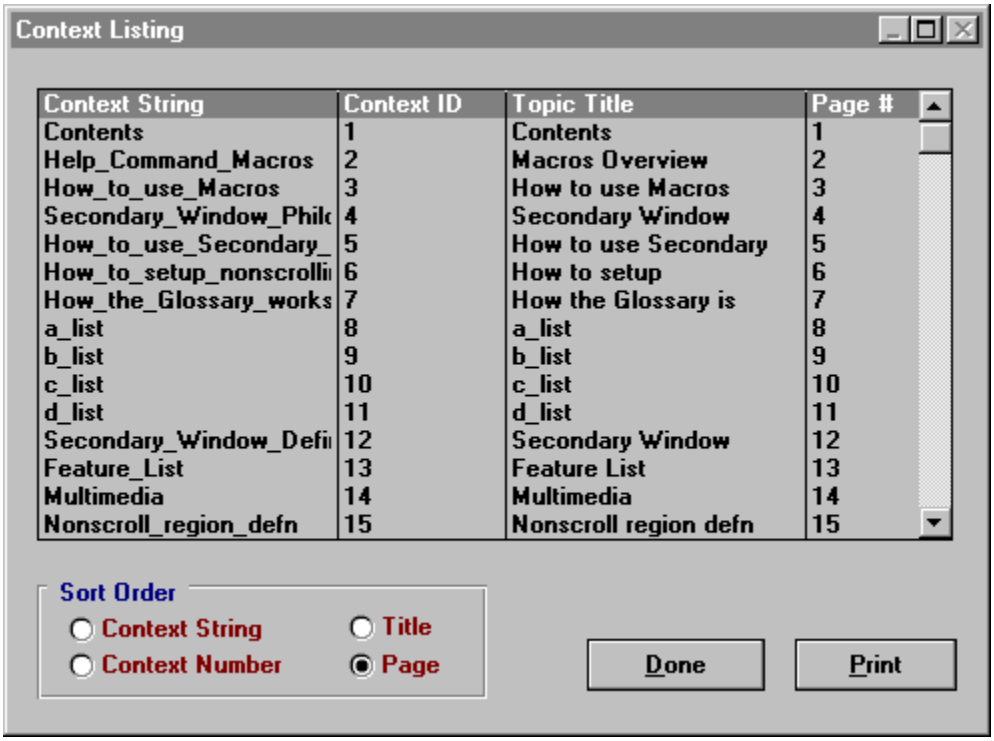
- Browse Sequences
- Links
- View

View

There are three options available from the File ... ***View*** menu: Context Relations, Links, and Browse Sequences.

Context Relations

Selecting Context Relations will bring up the Context Listing form.



Display

The Context Listing form can be re-sized and/or maximized to fit more information in the window. The vertical dividers cam be moved with the mouse to expand or contract any of the fields.

Sort Order

The listing can be sorted by Context String, Context Number, Title, or Page number.

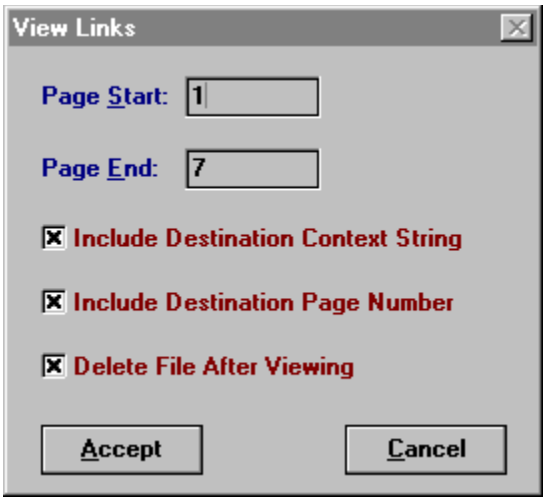
Print

Click on the ***Print*** 

button to send a copy of the current listing to the default printer.

Links

Selecting Links will bring up the View Links form.



Report File

The Context Relations report function will write a file in Windows Write format for viewing/printing with WRITE.EXE.

Page Start / Page End

Enter the range of pages to include in the Context Listing report.

Include Destination Context String

Check this box to include the destination Context String for Jumps and Popups in the report.

Include Destination Page Number

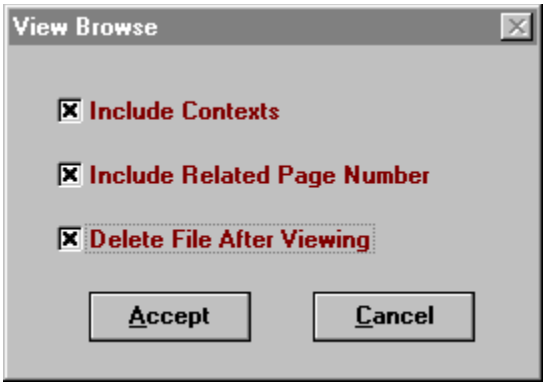
Check this box to include the destination page number for Jumps and Popups in the report.

Delete File After Viewing

If this box is checked, the temporary file will be deleted when you close Windows Write and return to the Help Magician. The temporary file is constructed from the help source filename. The last two characters of the filename are replaced or appended, as necessary with the characters, "VL" and the extension is .WRI. For instance, the report file for TUTOR.HLX would be TUTORVL.WRI.

Browse Sequences

Selecting Browse Sequences will bring up the View Browse form.



Include Contexts

If this option is checked, the Context String and Context Number of the related topic will be included in parenthesis after the topic title.

Include Related Page Number

If this option is checked, the page number of the related topic will be included in parenthesis after the topic title.

Delete File After Viewing

If this box is checked, the temporary file will be deleted when you close Windows Write and return to the Help Magician. The temporary file is constructed from the help source filename. The last two characters of the filename are replaced or appended, as necessary with the characters, "VB" and the extension is .WRI. For instance, the report file for TUTOR.HLX would be TUTORVB.WRI.

See Also

Print

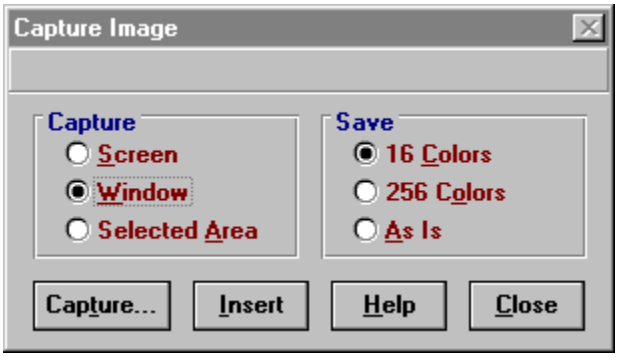
Auto TOC

The **Auto TOC** will generate a Table of Contents from all of the titles in the current help file. The table will be placed at the top of the current page. All of the entries will have the jumps to their respective topics assigned.

Default titles, those beginning with "T_" will not be included in the table.

Capture Image

The Capture Image tool will capture of an entire screen, a selected window, or a selected area of the screen and automatically insert the saved image into the Help Magician's editor.



Help Magician Window

The Help Magician's main window will be minimized while the Capture tool is in use.

Capture

Screen

When the **Screen** option is selected, the entire screen will be captured. Note that this will be an extremely large bitmap.

Window

The Capture Image tool defaults to **Window** as the area to be captured. As the mouse is moved around the screen, a frame is drawn around the window under the mouse cursor. To select a window to capture, move the cursor to the window and click on it with the left mouse button. Note that the Capture and Insert buttons will not be enabled until a window has been selected.

As the mouse cursor is moved, the Window ID and some or all of the Window text is displayed in a frame at the top of the Capture image form.

Selected Area

To capture a selected region of the screen, select the **Selected Area** option, move the mouse cursor to the top left corner of the desired area, hold down the left mouse button, drag the cursor to the bottom right corner of the desired area, and release the mouse button. Note that the Capture and Insert buttons will not be enabled until an area has been selected.

As the mouse cursor is moved, the X and Y coordinates of the cursor are displayed in a frame at the top of the Capture image form.

Save

Select the color depth for the saved image. By default, the image will be saved in the color depth of the host system but images with a color depth of more than 256 colors are not supported by the help compiler or the Capture tool so you will have to save the image as 16 or 256 colors.

Controls

Capture Button

When the area to be captured has been defined, click on the Capture button. A file dialog will prompt for a filename for the bitmap. Enter the filename and click on OK. The image has been saved to disk.

Insert Button

Click on the **Insert** button to insert the image into the Help Magician's editor. The Help Magician's main window will be restored to its previous state and the Image Options dialog will pop up. Select the image options and click on OK. The image will be inserted into the editor. If the image was saved with 256 colors, the image will automatically be inserted as an embedded window.

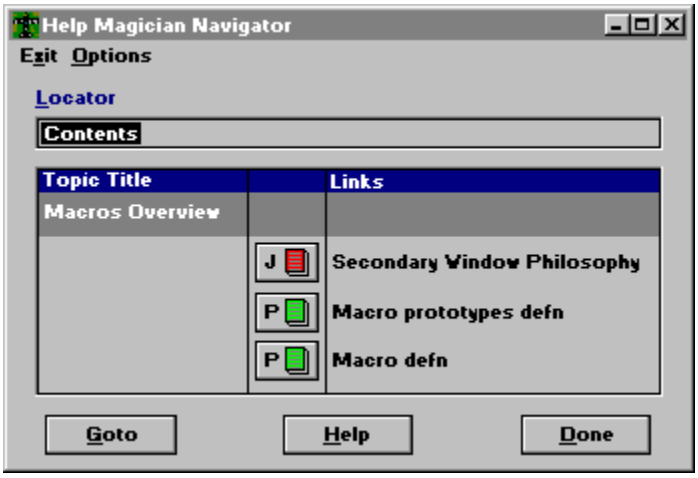
If embedded windows are used in your help system, **HMEW2.DLL** must be shipped with the compiled help file and installed to the **WINDOWS** directory. Any bitmaps displayed with **HMEW.DLL** must also be shipped with the help system.

Close Button

Click on the **Close** button to close the Capture tool without capturing or saving an image.

Navigator

The Navigator is a useful tool to view the tree structure of your help file and to traverse the help file with the use of the graphical representation of the file.



Selecting Navigator from the Tools menu will bring up the Navigator form.

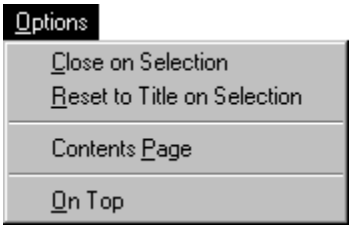
The Topic Title column displays all of the titles in the current help source file. The Links column displays all of the Jumps and Popups on that page. The type of link is displayed as an icon, shown later.

The form can be re-sized to fit more information in the window and the sizing and positioning is saved in the Help Magician's INI file so that the settings are duplicated when you use the Navigator again.

Scroll through the list of titles and select the topic to go to by clicking on the Goto button or double clicking on the title itself.

Click on the ***Options*** menu to drop down the Navigator Options sub menus.

Options



Click on the desired area of the Options Menu for specific Help

Legend

The type of link, in the Links cloumn is displayed as an icon.

Jump To Topic

Popup To Topic

Jump to File

Popup to File

Jump to Macro

Popup to Macro

Jump to Other Source

Popup to Other Source

Jump to Mid Topic Context String

Jump to Keyword

Jump to URL Address

Close on Selection

Click on this sub menu to toggle the state of the option. When there is a check mark before the menu item, it is in effect. When this option is selected, the Navigator window will close when you goto a title by clicking on the Goto button or by double clicking on the title in the list box.

Reset on Selection

Click on this sub menu to toggle the state of the option. When there is a check mark before the highlighted bar to the menu item, it is in effect. When this option is selected, the Navigator will reset the highlighted bar, in the Topic Title column, to the title selected in the Links column when you goto a title by clicking on the Goto button or by double clicking on the title in the list box.

Contents Page

Selecting this sub menu will cause the editor to display the Contents (or Index) page of the help file.

On Top

Click on this sub menu to toggle the state of the option. When there is a check mark before the menu item, it is in effect. Checking this sub menu will keep the Navigator window on top of the Help Magicians main window at all times.

Convert Images

You can convert the display state of all of the images on a page or in the entire help file. ***Convert Images*** is available from the ***Tools*** menu.

Display Image

Stores the image in memory and in the HLX file. This method uses the most memory and disk space but provides the fastest display of the image in the editor.

Display Filename

Stores the filename only in memory and in the HLX file. This method uses the least memory and disk space of all the display options.

Link to File

Stores only the the image filename in memory and in the HLX but displays the image in the editor. This method is a compromise between the two methods described above.

Glossary Wizard

The Auto Glossary function, available from the ***Tools*** menu, will automatically generate a Glossary shell in the last page of the current help file.

Support Files

Auto Glossary uses A-Z.SHG and CLOSE.BMP, both installed in the Help Magician \SOURCE directory. Both of these files must be in your help build 'ROOT' directory or in one of the paths listed in the Bitmap Directories on the Paths form. If your help source file is part of a project, then these files must exist in the same directory as your help source file. If these bitmaps are not available, you will get error messages when compiling. Copy these files to one of the appropriate directories before using the glossary wizard.

Glossary Topic

A topic will be created on the last page of the current help file. If a topic with the title or Context String, "Glossary" already exists, the Auto Glossary function will abort. If created, this topic will contain a Paragraph Marker (non-scrolling region), a Jump to a macro that will close the Glossary window, the A-Z.SHG file, and mid-topic Jumps for the letters A through Z.

Macros

The macros that will be created for the glossary:

Definition: Glossary_Close
Macro: CloseWindow("Glossary")
This is the macro that will close the Glossary window when the Close button is pressed.

Definition: HM_ProjectMacros:
Macro: CreateButton("Glossary_btn", "&Glossary", "JumpId(`THISFILE>Glossary', 'Glossary')")

This is the macro that creates the "Glossary" button on the main help window and opens the Glossary window when pressed.

Duplicate Macro definitions/macros

If the definitions/macros already exist, they will not be duplicated. The existing macros will be used.

Secondary Window

A secondary window called "Glossary" will be created, if it does not already exist. This is the window that will display the Glossary. The attributes of this window can be changed at any time in the Help Window Specifications form, available from the "Options" menu.

Fonts

Two paragraph styles are added to the defined styles: Glossary Heading and Glossary Letters, if they do not already exist. These paragraph styles can be changed at any time in the "Define Font Styles" form available from the Format menu.

Adding Glossary Items

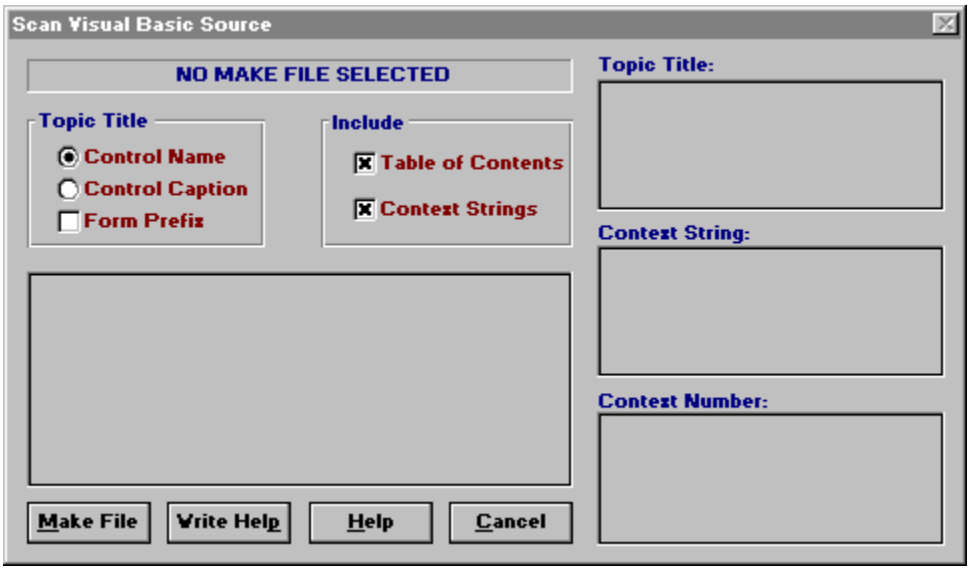
When the Glossary has been written, you can add items under the appropriate letters. Typically, these Glossary items would be Popups. Create a page for the contents of each definition and then establish a Popup link to this page from the Marker Menu or toolbar.

Scrolling

Note that WinHelp will not scroll the Glossary topic when near the bottom of the page unless there is more text than can be displayed in the window. With an empty Glossary, pressing the Z button will cause the display to start at about the letter V.

VB Help Wizard

Select ***VB Help Wizard*** from the ***Tools*** menu to access the Help Wizard form.



Scan VB Source

This function will scan all of the source files in a Visual Basic make file (.MAK) and create a help file shell in the Help Magician based on the HelpContextID properties assigned to the forms and controls in the source.

ALL OF THE FILES INCLUDED IN THE VISUAL BASIC MAKE FILE MUST HAVE BEEN "SAVED AS TEXT".

The titles, headings, and context strings will be no more coherent to your audience than the names assigned to them in the properties form in Visual Basic design mode. For this reason, assign names to the controls that will make sense to those reading your help file.

Control Name Parsing:

There are two control naming conventions that will be specially parsed by the VB Source Scanner.

Upper Case Convention

If control names are constructed from multiple words, with the first character of each word in upper case, a space will be inserted before each upper case character. The control name "ExitButton" will appear as "Exit Button" in the help file. The disadvantage to this method is that control names with repetitive upper case characters may not produce desirable results.

Underscore Convention

If control names are constructed from multiple words and the words are separated with underscore characters, the underscores will be replaced with spaces. The control name "Exit_Button" will appear as "Exit Button" in the help file. This naming convention provides the most flexibility and gives more control over the resulting help file title, etc.

Duplicate HelpContextID's

If HelpContextID's are duplicated throughout the Visual Basic source, the Table of Contents entry and the title for the topic will be taken from the first occurrence of the HelpContextID and subsequent occurrences will be appended to the topic header, separated with a comma and a space.

There are two sub menus available from the Scan VB Source menu: Entire File and Merge.

The Entire File option will write a entire new file from the VB source and the Merge option will create new topics for controls with HelpContextID's not already existing in the current help file.

Creating or Merging help from VB Source:

Step 1

Select the desired sub menu depending on whether you're creating a new help file or merging help with an existing file.

Step 2

When the "Scan Visual Basic Source" form appears, select from the available options:

Control Name

If the Control Name radio button is selected (default), the topic titles, heading, and context strings for the topics will be created from the control name as it appears in the properties form in Visual Basic.

Control Caption

If the Control Caption radio button is selected, the topic titles, heading, and context strings for the topics will be created from the control caption as it appears in the properties form in Visual Basic. If the control does not have a caption, the control name will be used.

Form Prefix

If the Form Prefix check box is checked, the form name or caption, depending on the Control Name/Control Caption selection, will precede each topic title, heading, and context string, separated with a space, a dash, and another space (Form - Control).

Table of Contents

If this check box is checked a table of contents will be generated on the first page of the help file. The title, heading, and context string will be "Contents". A Jump will be created for every topic that is written to the file.

Context Strings

If this box is checked, Context Strings will be written to each page of the help file.

Fonts

The font used for the "Glossary" heading will be font definition #1, "Topic Heading" by default. The font used for the alphabetical headings will be font definition #1, "Emphasized Help Text", by default.

Step 3

Press the "Make File" button to select the Visual Basic make file to use.

Step 4

View the three list boxes on the right side of the form. They contain all of the topics that will be written to the help file shell.

Step 5

Press the "Write Help" or "Merge Help" button, depending on the mode selected from the "Scan VB

Source" menu. Focus will be returned to the Help Magician editor with the new, or modified, help file in memory.

Step 6

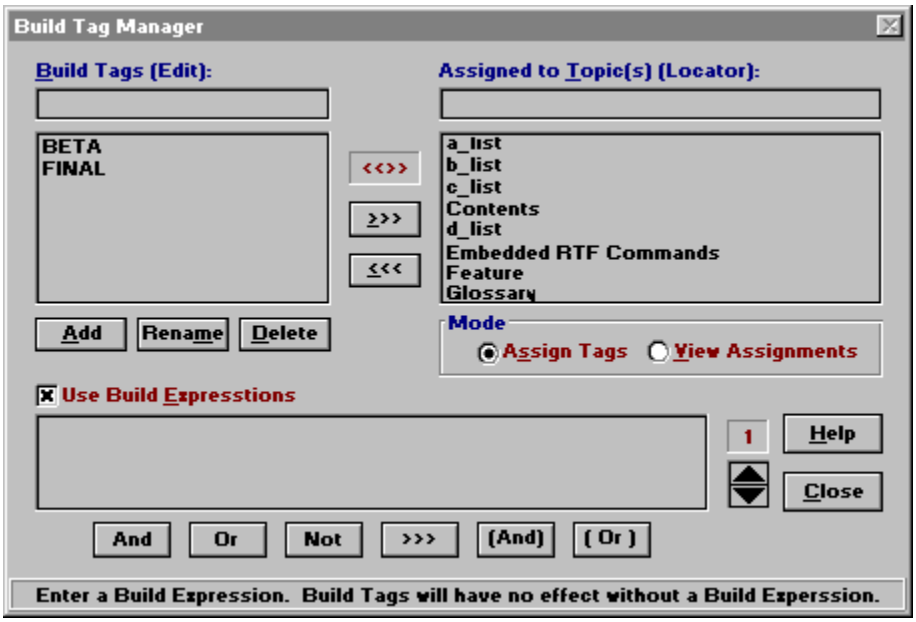
Review the help file with the test facilities of the Help Magician and/or print out the Context Relations, available from the File/Print menu.

Step 7

Save the file. If you selected the "Merge" sub menu, you will be required to perform a "Save As...". Compile the help file (Build/Rebuild All) and view it with WinHelp (Build/Call Winhelp).

Build Tag Manager

Select ***Build Tags*** from the ***Build*** menu to access the ***Build Tag Manager*** form.



Build Tag Overview

The BUILD option determines which topics containing build tags are included or excluded in a build. This is useful for conditionally compiling a help file. For example, you may be developing a help file that will be used for both the demo and full release version of your software. Use this option only if you've created build tags in your topic files. The BUILD option has the following syntax:

BUILD = expression

Expression is a logical statement that specifies which build tags to include or exclude. The tags must be included in the [BUILDTAGS] section, which is described later in the topic. The statement can include any of the following symbols, listed in order of importance.

( )	Parentheses
&	AND operator
|	OR operator
~	NOT operator
tag	Build tag

The compiler evaluates each expression from left to right using the following order:

Expressions with parentheses are evaluated first.
Expressions with a NOT operator (~) are evaluated next.
Expressions with a logical AND operator (&) are evaluated next.
Expressions with a logical OR operator (|) are evaluated last.

For example, if you created build tags named FINAL, INCOMPLETE, and BETA in a topic file, you could include any one of the BUILD expressions listed in the [OPTIONS] section.

BUILD = BETA	Topic that have the BETA build tag and topics with no build tags.
BUILD = INCOMPLETE & BETA	Topics that have both the INCOMPLETE and BETA build tags, plus topics that have no build tags.
BUILD = INCOMPLETE | BETA	Topics that have the INCOMPLETE or BETA build tags, plus topics that have no build tags.
BUILD = (INCOMPLETE | BETA) & FINAL	Topics that have either the INCOMPLETE or BETA build tag and also have the FINAL build tag.
BUILD = ~ FINAL	Topics that don't have a FINAL tag and topics with no tags at all.

Using the Build Tag Manager

Build Tags (Edit)

The text box under the label, Build Tags (Edit) is used to enter new Build Tags or to rename existing ones.

To add a Build Tag, type the desired Build Tag name and press Return. The new Build Tag will be inserted into the list box.

To rename a Build Tag, select the name in the list, then click on the text box above the list. Edit the name and press the Rename button. The new Build Tag will replace the old name in the list box.

Delete

To delete a Build Tag from the list, select the Build Tag by highlighting it in the list box and press the Delete button.

Assigned to Topic(s) (Locator)

The text box under this label is used to quickly locate items in the list by typing some of the characters in the help topic or macro. The list will automatically scroll to the item most closely matching the entry in the text box.

Topic List Box

This list box contains the titles of all the topics in your help file.

Mode

Select the appropriate radio button to assign Build Tags to topics or to view the current assignments.

Direction Indicator

The Direction Indicator  (shown in assign mode) displays the direction of the link(s) between the Build Tag list box and the Topic Title list box in "View Assignments" mode.

Click in Build Tag List Box

When the indicator is pointing to the right  topics are highlighted that have the selected tag assigned to them.

Click in Topic Title List Box

When the indicator is pointing to the left  Build Tags are highlighted that are assigned to the

selected topic.

Assign Tags

The Assign Tags button  assigns the tag(s) selected in the Build Tag list box to the title(s) selected in the Topic Title list box.

Remove Tags

The Remove Tags button  removes the assignment of the tag(s) selected in the Build Tag list box from the title(s) selected in the Topic Title list box.

Use Build Expression

Check this check box to use the selected Build Expression in the build of the help file. If this box is unchecked, the Build Expression will have no effect during the build.

Build Expression List

Up to ten Build Expressions can be stored with the source and any one of them can be active. Click on the up or down arrows on

the spin button  to enter, edit, or select the desired Build Expression. The number of the Build Expression is displayed in the label above the spin button.

Build Expression Editor

The text box below the Use Build Expression check box is used to edit Build Expressions.

The "BUILD=" will be added during the build process. Don't include this in the editor. Enter only the expression.

Build Expression Buttons

Inserts the And symbol '**&**' with the appropriate spacing into the editor.

Inserts the Or symbol '**|**' with the appropriate spacing into the editor.

Inserts the Not symbol '**~**' with the appropriate spacing into the editor.

Insert the single Build Tag selected in the Build Tag list box into the editor. This button can only be selected if a single Build Tag is selected in the Build Tag list box.

Insert two Build Tags, in parenthesis, separated with the And symbol '**&**' into the editor with the appropriate spacing (BETA & FINAL). This button can only be selected if two Build Tags are selected in the Build Tag list box.

Insert two Build Tags, in parenthesis, separated with the Or symbol '**|**' into the editor with the appropriate spacing (BETA | FINAL). This button can only be selected if two Build Tags are selected in the Build Tag list box.

Build Options

Click on the desired area of the Options Menu for specific Help

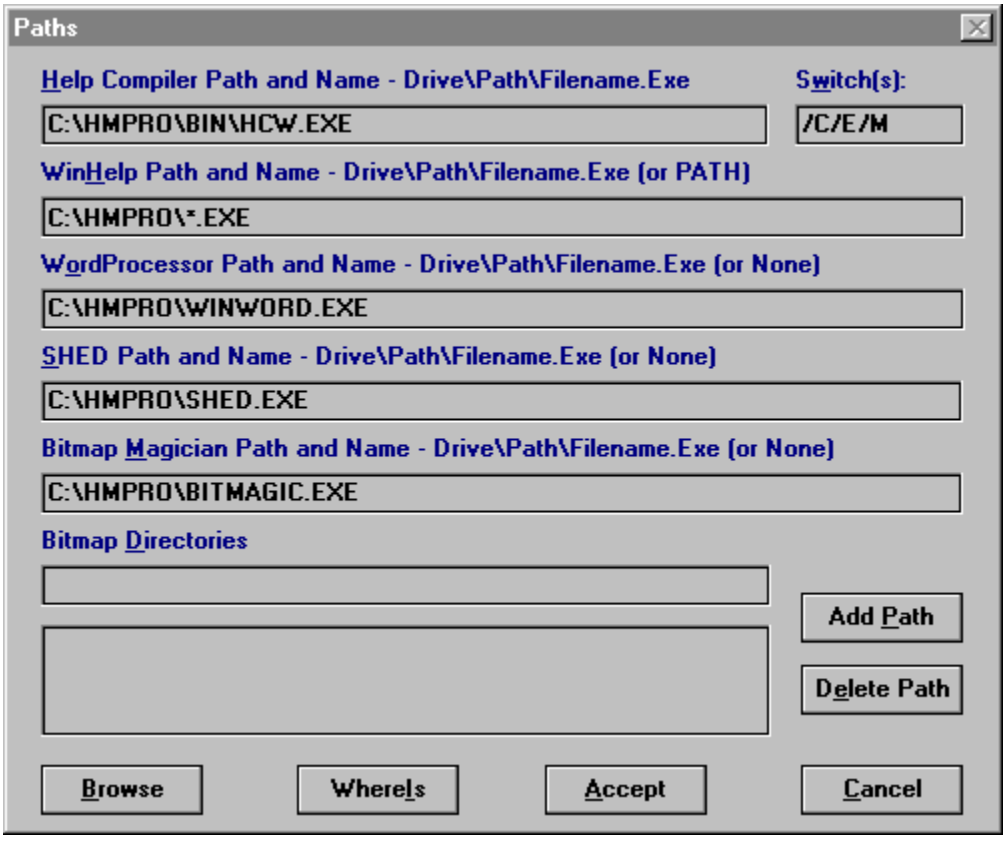
The Options Menu allows you to customize parts of the Help Magician and your Help file as well.

Display History Window...

When in Windows 95 compiler mode (Compiler Options, Compiler Tab), the History button is not in the WinHelp bar but it is available from the Options menu just as it would be when viewing the file with WinHelp.

Paths

Selecting the ***Paths*** sub menu option displays the ***Paths*** form. Type in the full path and name, where required, in the text field or type in the word "None" to signify no path.



Compiler Path

Enter the full drive, and path, to your Windows help compiler. The path defaults to C:\HMPRO30\ upon installing Help Magician but it can be changed at any time. The Compiler Path is stored in the HLPMAGIC.INI file. This is a change from previous versions of the Help Magician that stored the compiler path in each help file.

Important

If a help compiler *name* is included with the path (both the filename and .EXE), it will be used in place of the HC.EXE or HC31.EXE which is set by the Version radio boxes for versions 3.0 and 3.1 respectively. This provision accommodates help compilers whose executable name is unknown at the time of this writing (such as HCP.EXE found on Compuserve).

Word Processor

Enter the full drive, path, and filename to the word processor that you will use to view the RTF files generated by the Help Magician. This path is stored in the HLPMAGIC.INI file which is read when you open the Help Magician and written when you close it. "See Call Word Processor" later for more detailed information.

SHED

Enter the full drive, path, and filename to the SHED (Segmented Hypergraphics Editor) that you will use

to modify bitmaps to use as hotspots in your help file. The path defaults to C:\HMPRO30\ upon installing Help Magician but it can be changed at any time. This path is stored in the HLPMAGIC.INI file which is read when you open the Help Magician and written when you close it. Refer to "Help System Overview" and SHED.EXE, for more detailed information.

Bitmap Magician

Enter the full drive, path, and filename to the Bitmap Magician that you will use to modify font families to you're own specifications. This path is stored in the HLPMAGIC.INI file which is read when you open the Help Magician and written when you close it.

Bitmap Directories

Type the paths to any bitmaps used in your Help file in the drop down box. Add as many paths as you need. The Help compiler will search these directories for the bitmaps used in your Help file. Please do not include literal root directory paths, i.e. "C:\" as this can cause the Help compiler to malfunction. Note: If you are developing a help file with other team developers on a network, you should make sure all the bitmaps will be available in the same directory as your current help source and project files (.hlx and .hmp). If you leave the bitmap directories blank, the current help source directory will be used in locating the bitmaps.

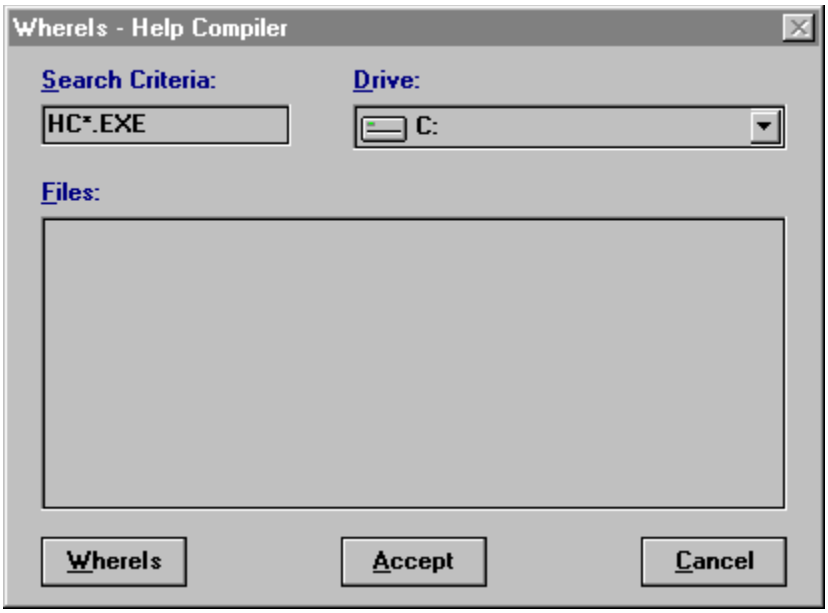
Browse Button

Pressing the Browse button will pop up a file dialog to automate the entry of the path and/or filename for the current field.

WhereIs Button

Pressing the WhereIs button will pop up the WhereIs dialog. This dialog will search the selected drive for files matching the specified criteria. The criteria is automatically entered for each of the fields but can be edited, if necessary.

If file(s) are found, highlight the desired filename in the list and it will be entered into the appropriate field in the Paths form.



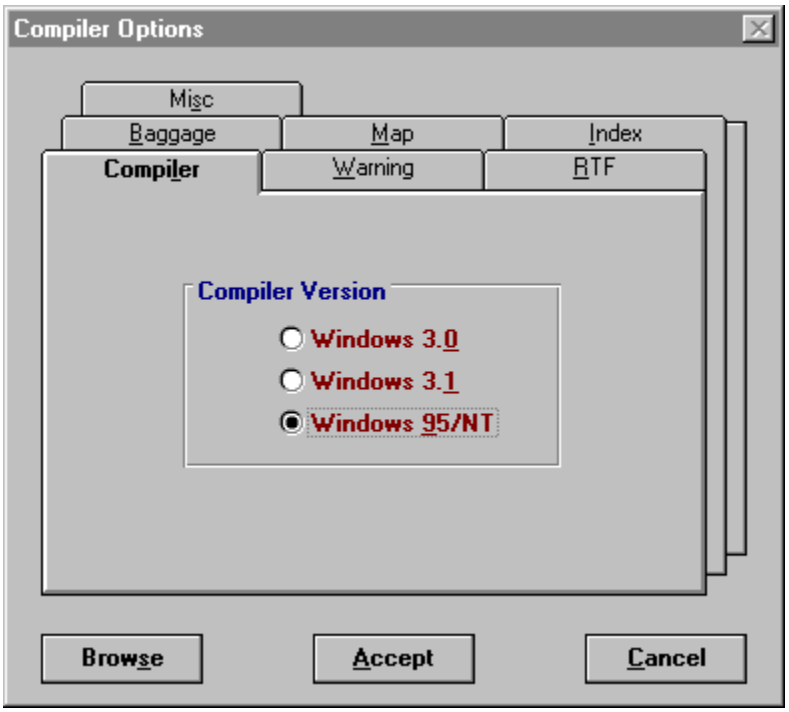
See Also

Call Word Processor

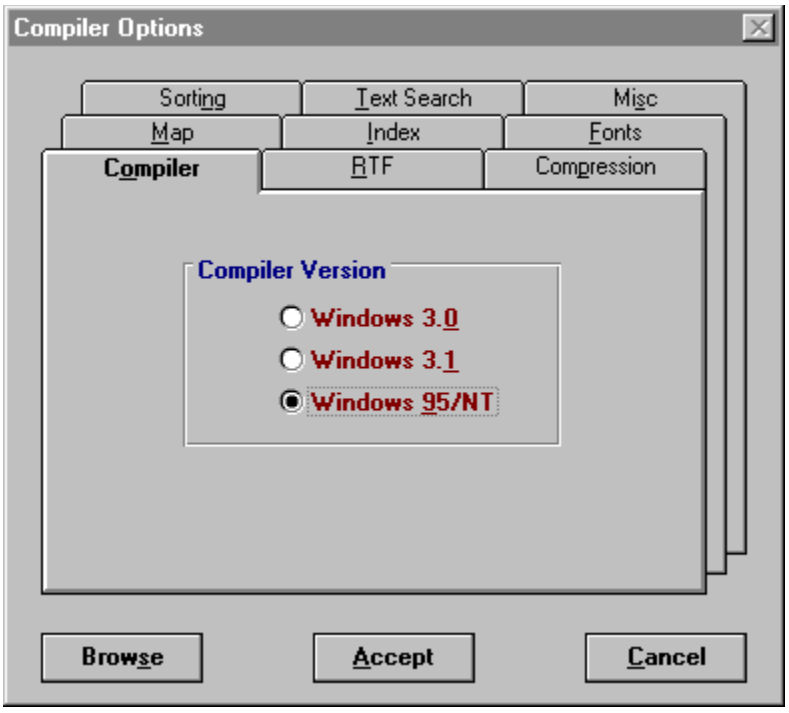
Compiler Options

Selecting the Compiler sub menu option displays the Compiler Options window. The ***Compiler Options*** form allows setting various help compiler options along with the RTF (Rich Text Format) version that the Help Magician generates for the compiler.

The Win31 Compiler Options form is shown here.



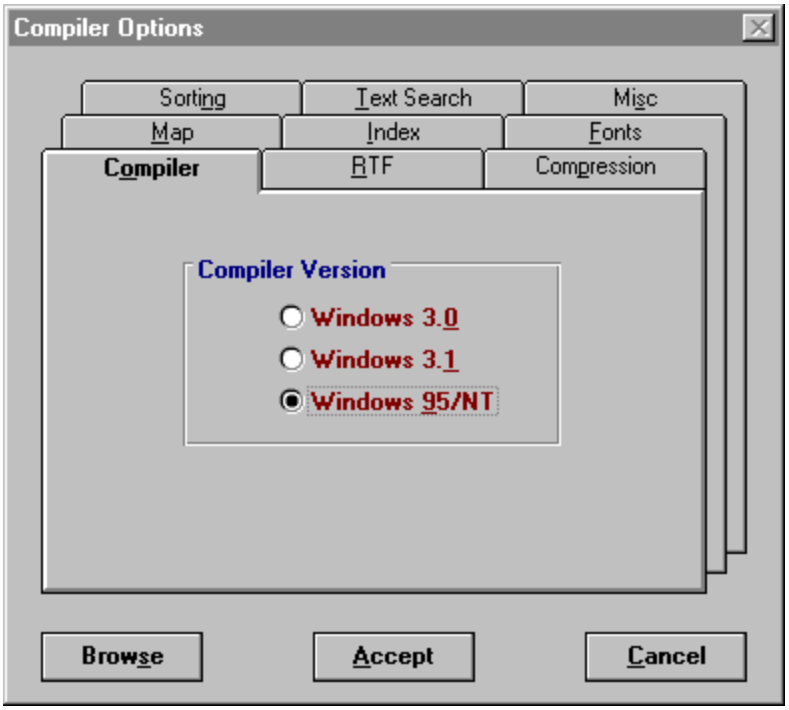
The Win95 Compiler Options form is shown here.



Click on the desired tab of the Compiler Options bitmap for specific Help

Compiler

The Windows 3.1 and the Windows 95 Compiler Options, Compiler Tab is shown here.

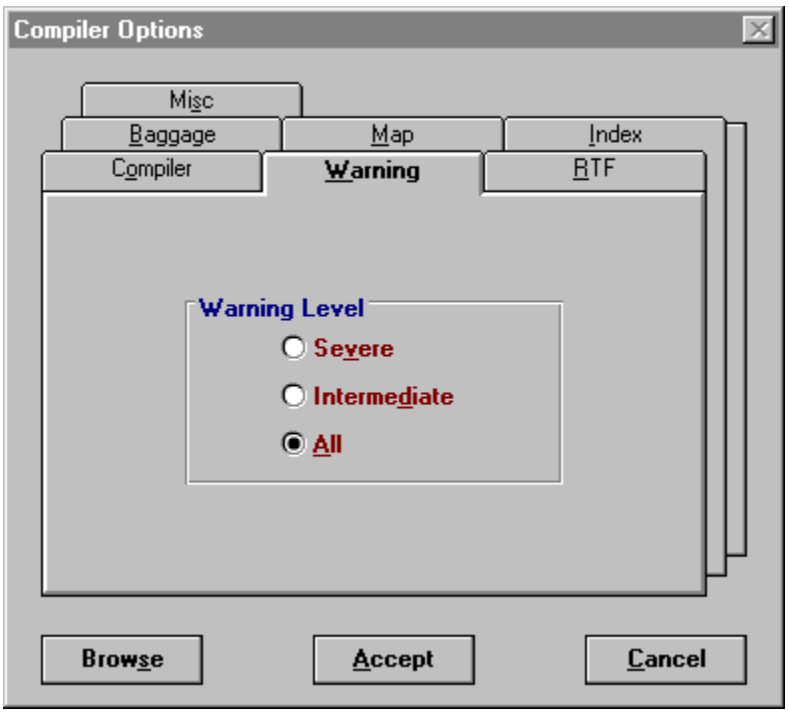


Compiler Tab

Setting the help compiler version enables/disables certain Help Magician features not available in older versions of the Windows Help Compiler. For instance, version 3.1 supports macros, which are not available in the 3.0 version.

Warning Tab

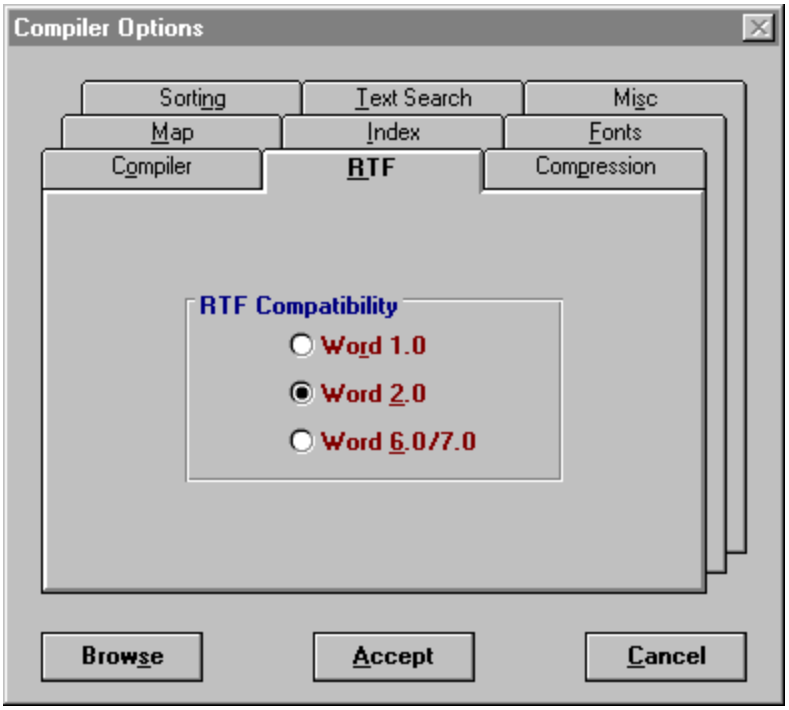
The Win31 Compiler Options, Warning Tab is shown here.



This option sets the level of error message reporting the compiler displays during the build process. Select Severe to display only critical errors, Intermediate for an intermediate level of warnings and All to see all errors reported by the compiler.

RTF Tab

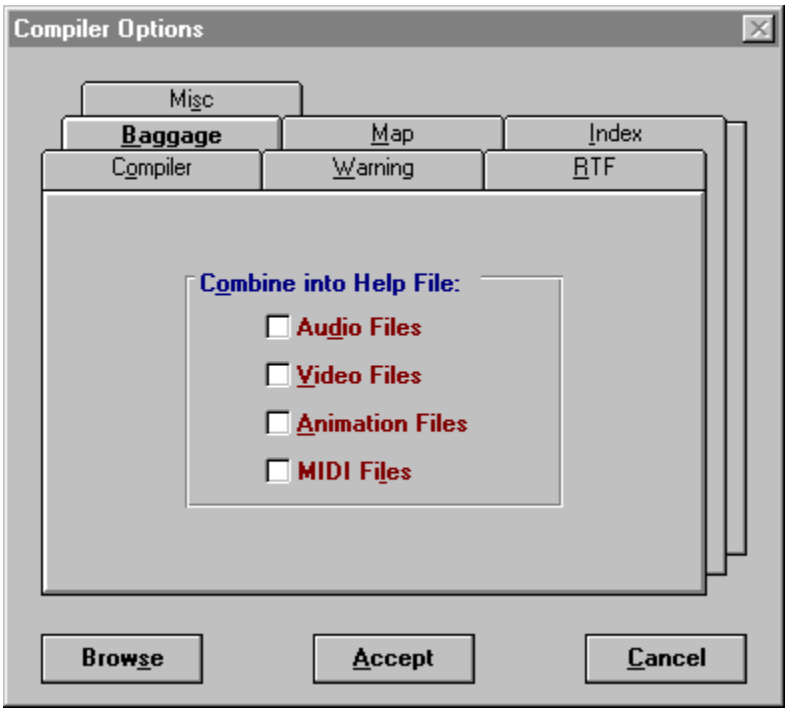
The Windows 3.1 and the Windows 95 Compiler Options, RTF Tab is shown here.



This feature lets you choose which version of Word for Windows the RTF file is compatible with. This is useful when exporting a Help Magician RTF file to Word for Windows. There are three versions of Word supported: version 2.0, 6.0 and 7.0.

Baggage Tab

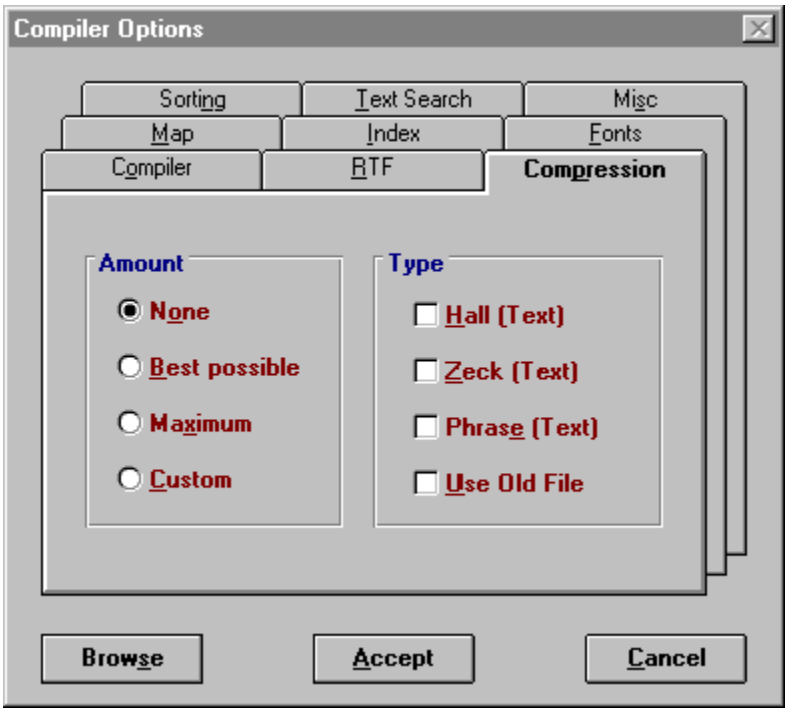
The Win31 Compiler Options, Baggage Tab is shown here.



Put an X in the check boxes next to the types of files you want included in the [BAGGAGE] section of the project file. This will include the code from the selected multimedia elements into the compiled help file and the individual multimedia elements will not have to be shipped with the help system.

Compresstion Tab

The Win95 Compiler Options, Compresstion Tab is shown here.



This option instructs the compiler to compress the compiled help file to make it smaller. This feature is useful when the help file is large, has many sparse paragraphs of text and recurring phrases, and contains bitmaps. You may select from several different types of compression to match the style of help file you are building.

None

Select this to turn off compression.

Best Possible

The help compiler determines the optimal compression.

Maximum

Instructs the help compiler to make the help compiler as small as possible.

Custom

Allows you to mix and match compression levels.

Hall - if this box is checked, the compiler uses Hall compression, which is more effective than Phrase compression for files that are greater than 100K in size. If you use this type of compression, your Help file will compile more quickly than if Phrase compression without an existing phrase file is used. Hall compression is more effective when used with Zeck compression. If you intend to ultimately compress your help file with a utility such as PKZIP for distribution purposes, then the utility may achieve higher compression if only Hall compression is used.

Zeck - if this box is checked, the compiler uses Zeck compression. When used with Hall the greatest

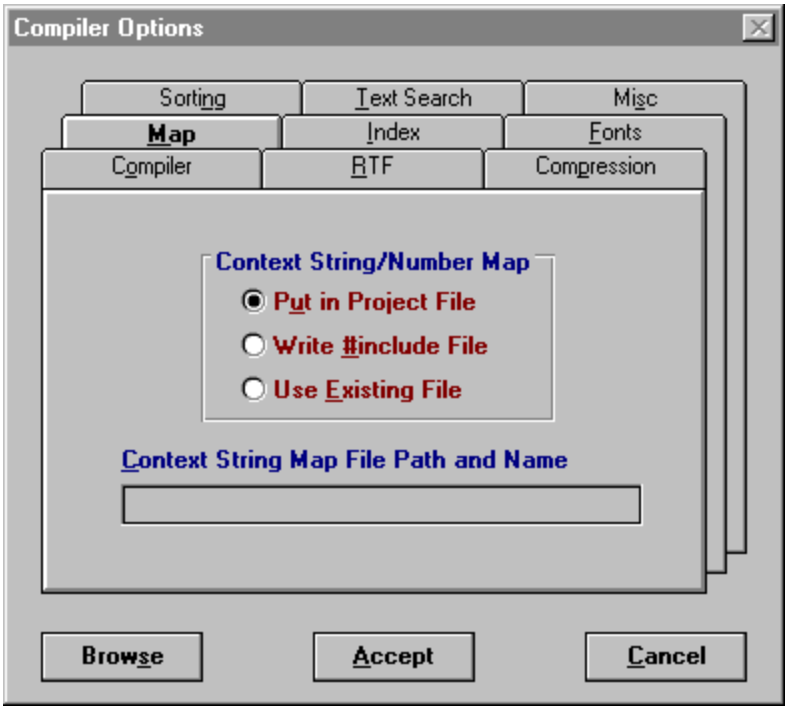
effective compression can be attained. You can use Zeck compression by itself to reduce compile time, but doing so will increase the size of the compiled Help file.

Phrase - if this box is checked, the compiler uses Phrase compression, which is more effective than Hall compression for files that are less than 100K in size. The compiler creates a phrase table file with a .PH extension if one does not already exist in the ROOT directory. This will decrease the compile time significantly if little has changed between compiles. Because the phrase table file speeds up the time to compile the Help file when little text has been changed since the last rebuild, you may want to keep the .PH file around between compiles. Check the "Use Old File" box to instruct the compiler to use the old phrase table file. To achieve maximum compression before a build, you will need to delete the .PH file before compiling the Help file. Note that Phrase compression cannot be used with Hall compression.

If you are embedding graphics, you may want to try various compression settings for better results.

Map Tab

The Windows 3.1 and the Windows 95 Compiler Options, Map Tab is shown here.



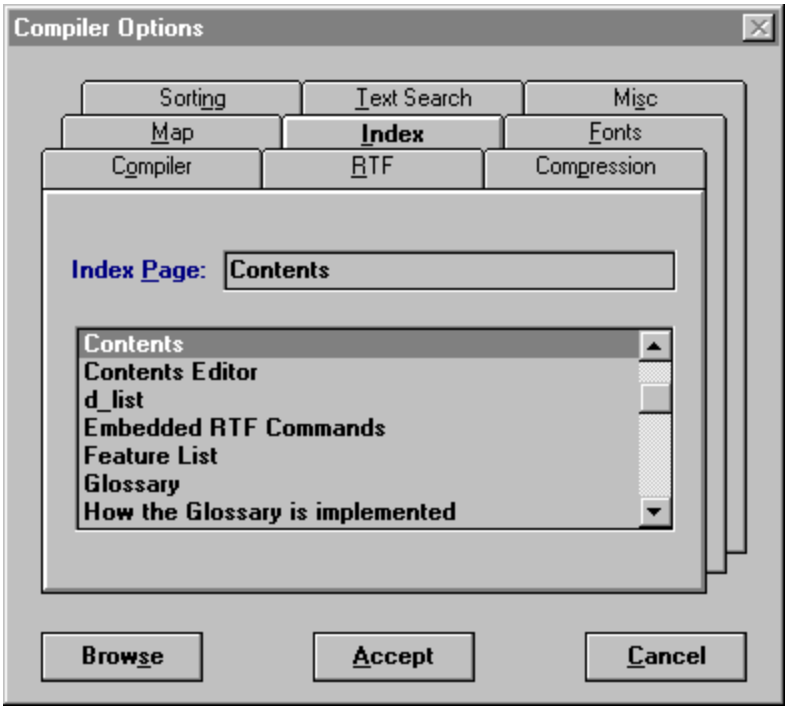
This option lets you specify where the context string map will be generated. Select "**Put in Project File**" to have the map included in the Help Project (.HPJ) File. Select "**Write #include File**" to have the Help Magician create a separate 'C' style #include file. The include file will be written any time an RTF is generated for the compiler. Select "**Use Existing File**" to have the Help Magician use a separate #include file. For the last option, you must enter the #include file name into the "Context String Map File" text box. See Visual C Support for detailed information on establishing context sensitivity between Visual C and the Help Magician.

Project Management Notes

If your help source file is a member of a multiple file project, the Put in Project File option will be disabled. If you choose the Use Existing File option, then just enter the name of the Context String Map file. Your existing map file should be located in the same directory as your project file (.hmp).

Index Tab

The Windows 3.1 and the Windows 95 Compiler Options, Index Tab is shown here.



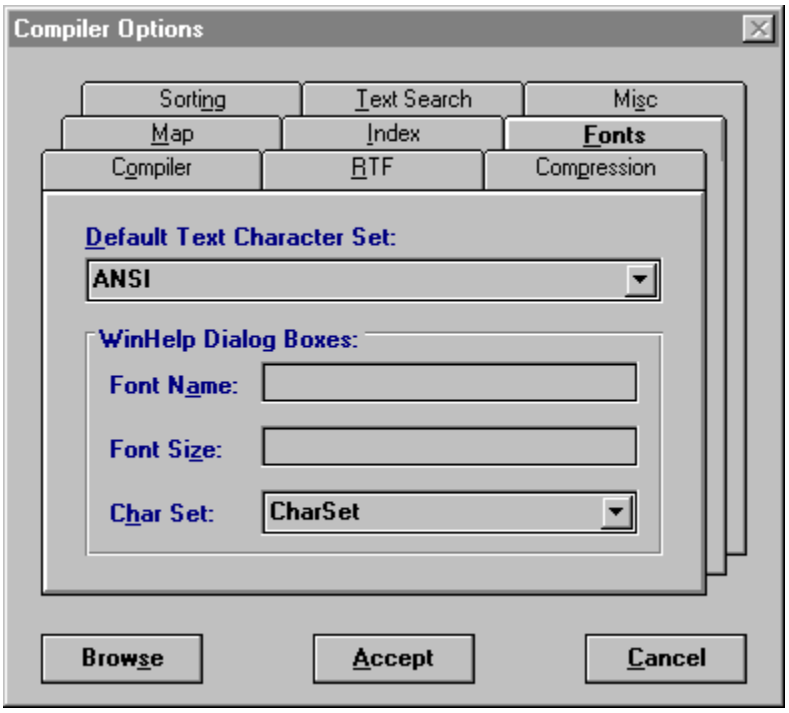
The Index Page option sets the topic that will be first displayed when Windows Help is called and when the "Contents" button is selected in Windows Help. If a page is not selected, the Help Magician will use page one as the help index page.

Project Management note

You MUST set the Index Page when doing a multiple file Project.

Fonts Tab

The Win95 Compiler Options, Fonts Tab is shown here.



Default Text Character Set

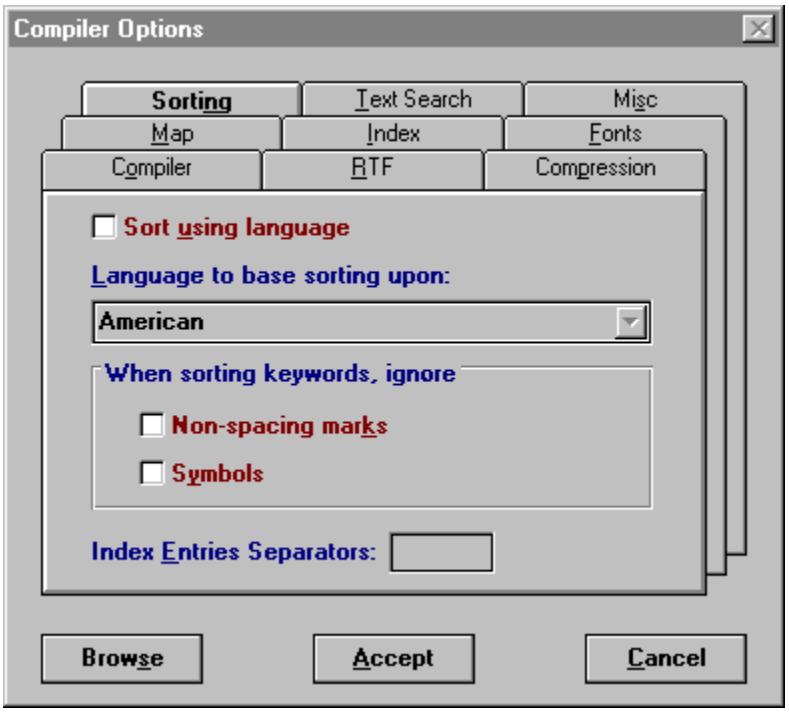
Use this drop down box to select the character set that will be used for the fonts in the compiled Help file.

WinHelp Dialog Boxes

Set the font name, size and character set for the dialog boxes used by WinHelp for you Help file.

Sorting Tab

The Win95 Compiler Options, Sorting Tab is shown here.



Language to base sorting on

If you intend to use a language other than English in your Help file, set that language here. This information is used when sorting keywords in the index.

When sorting keywords, ignore

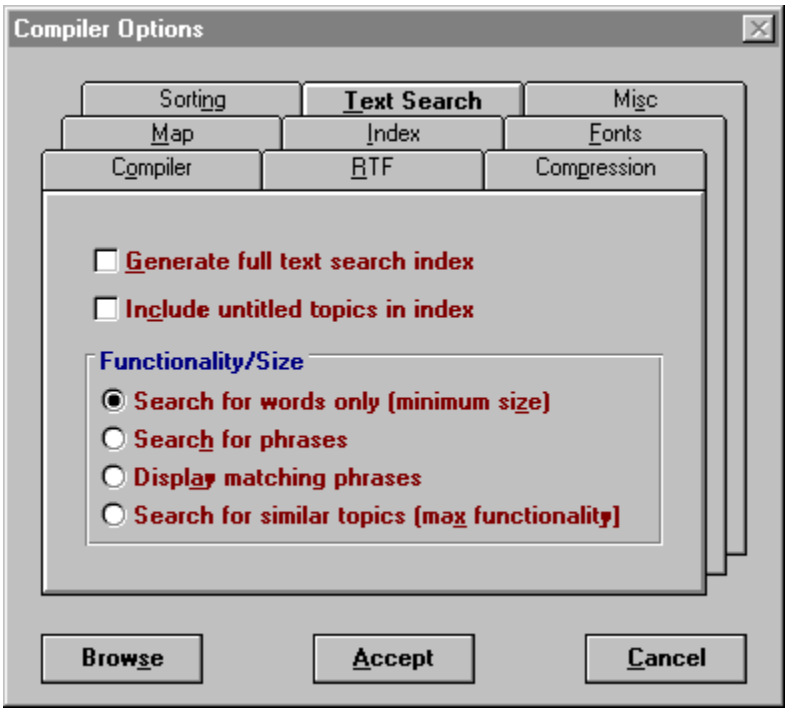
Check the "Non-spacing marks" to ignore spacing and "Symbols" to ignore symbols.

Index Entries Separators

Enter the character used to separate keywords in an entry (for example, "keywords, entering") and the character used to separate keywords in a list. If this option is not used the default comma, semicolon and colon are used.

Text Search

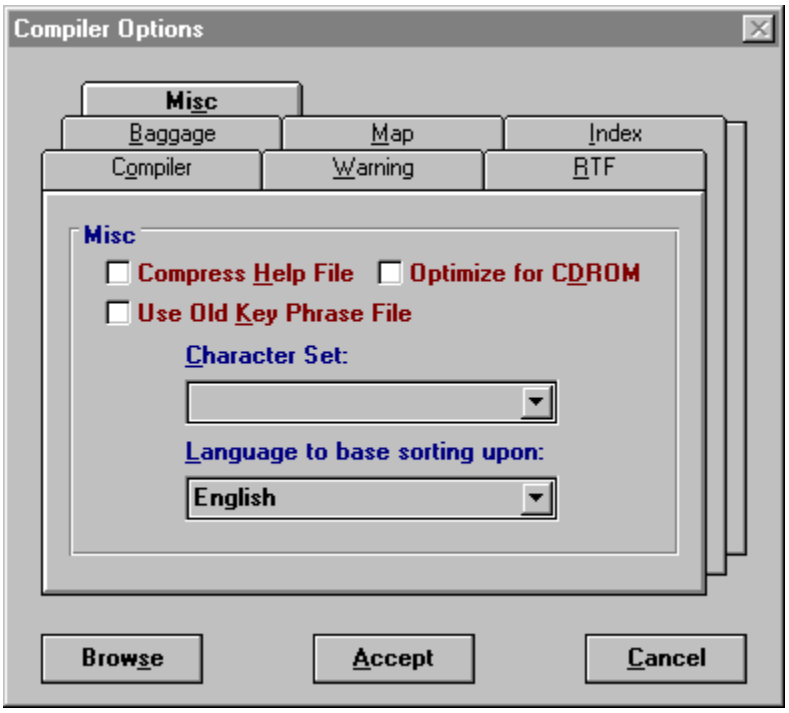
The Win95 Compiler Options, Text Search Tab is shown here.



In this section you may instruct the compiler to generate an index file for WinHelp to use when the end user requests full-text searching from the Find tab in the Help Topics dialog box. This is useful if the help file is large and the index might take a considerable amount of time to create. The index file would need to be included with the Help file at the time it is distributed. You may select from several options to instruct the compiler what to include in the text search.

Misc Tab (3.1)

The Win31 Copmiler Options, Misc Tab is shown here.



Compress Help File

Select this option to force the Help Compiler to compress the help file. The compiler uses two forms of compression: block and key-phrase. Block compression compresses the topic data into pre-defined units known as blocks. Key-phrase compression combines duplicate phrases found within the RTF file. If the "Use Old Key Phrase File" is not selected, the compiler creates a phrase-table file with the .PH extension. The .PH file can speed up the compression process when little text has changed since the last compilation, so you might want to keep the phrase file if you compile the same Help file several times with compression. To do this, select the "Use Old Key Phrase File" option. *However, you will get maximum compression if you don't use this option.*

Use Old Key Phrase File

When selected, this option forces the compiler to use an old key phrase file, if one exists, otherwise one will be created. If this option is not selected, the compiler will recreate the key phrase file each time the Help file is built.

Optimize for CDROM

Select this option to instruct the compiler to optimize the Help file for maximum performance when accessed on a CDROM.

Language Sort Order

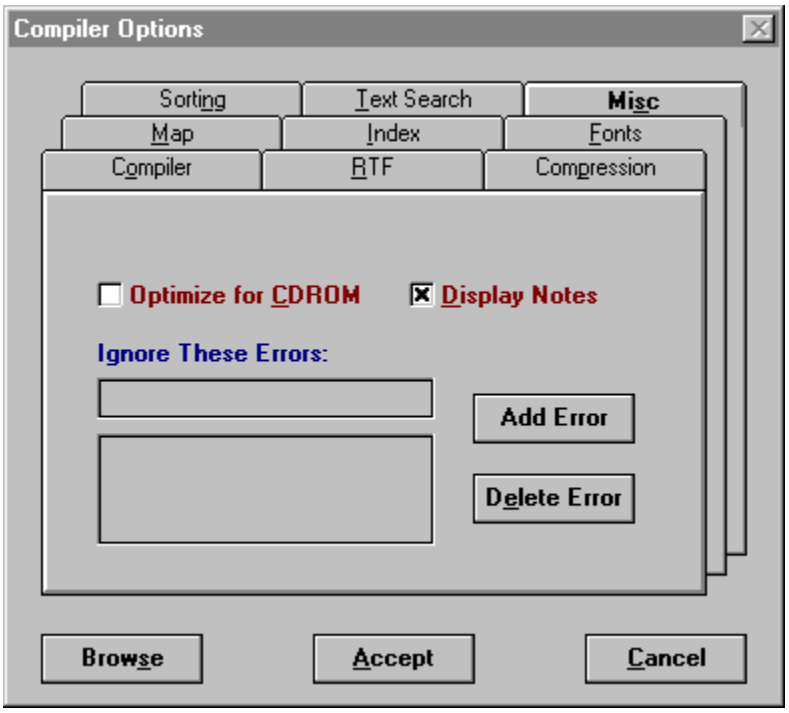
Use this option to set the sorting order for keywords in the Search dialog box. There are only two options currently available: English and Scandinavian.

Character Set

Use this option to select which character set to use for the final help system. The options are: ANSI, Apple Macintosh, OEM (code page 437), International English (code page 850), and Windows character sets.

Misc (95)

The Windows 95 Compiler Options, Misc Tab is shown here.



Optimize for CDROM

Select this option to instruct the compiler to optimize the Help file for maximum performance when accessed on a CDROM.

Display Notes

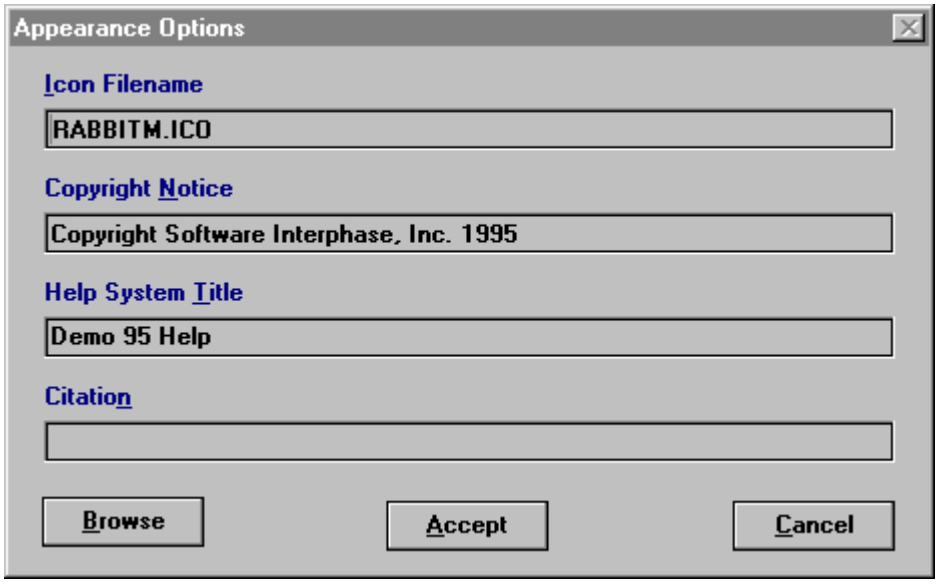
Select this option to have the compiler display note-level error messages while compiling. Note-level error messages are informational messages to the Help author.

Ignore These Errors

Enter any error message numbers here that you want the compiler to ignore during a build. Key in an error number in the text box and click on the Add Error button to add it to the list. To delete an error number from the list, simply select it in the list and click on the Delete Error button.

Appearance

Selecting the Appearance sub menu item displays the Appearance Options window. The Appearance option is used to set how certain aspects of the final help system will be displayed. Specifically, the icon that will be displayed when the help system is minimized can be defined as well as the copyright notice that is displayed in the About window, the help system title and the character set.



Icon Filename

The icon file name refers to the bitmap that will be displayed as the icon when the help file being displayed with WinHelp is minimized.

Copyright Notice

The text entered here will be displayed as the copyright notice in the About window in the help system. Note that you cannot change any other text in the About box (limitation set by Microsoft).

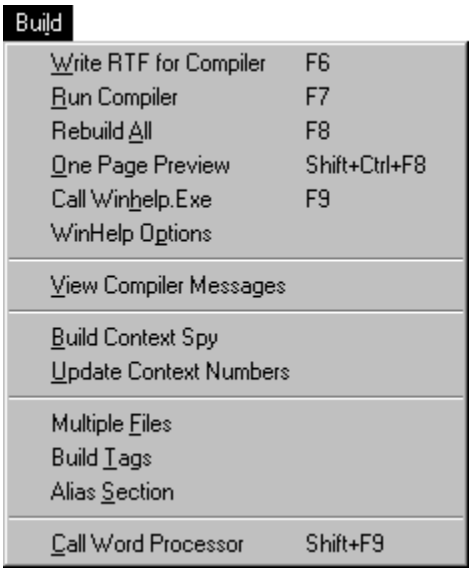
Help System Title

The text entered here will be displayed at the top of the help window when the help system is viewed with WINHELP.EXE.

Citation

The citation text is appended to the end of any text that is copied through WinHelp. A citation may contain a copyright, warning, or whatever message you want.

Building Your Help File



Click on the desired area of the Build Menu for specific Help

Build Menu

The Build main menu item selections allows you to build your Help Magician help system into a Windows Help file and view it. Several other options are provided to give you greater control over the build process.

Write RTF for Compiler

This option will write the RTF file, needed by the Windows help compiler, to the 'ROOT' directory for the current help build. If the help file has not been saved since the last edit, you will have the option to save it. You will be informed if the RTF file is up to date and the Help Magician will not rewrite the file if it is current. You can over-ride time and date stamp checking by selecting Rebuild All from the Build Menu.

If an Index Page has not been selected, then a warning message will appear indicating that page #1 of your help file will be used as a default index. Select OK to continue or Cancel to return to the editor.

If there are missing delimiters within numbered indents, warning messages will be displayed as well. Select OK to continue or Cancel to return to the editor.

Run Compiler

Calls the Windows help compiler, in the path specified in the Compiler Path option (this can be edited by selecting the Options Menu and choosing Paths...). If the current help file has not been saved since the last edit, you will have the option to save it. If the RTF file needs to be updated, you will have the option to update it at this point. To prevent a mismatch between the help file and the compiled help system, the Help Magician will not compile the files until all of the components are up to date. Again, you will be informed if the help system is up to date and, if so, it will not be re-compiled. You can over-ride time and date stamp checking by selecting Rebuild All from the Build Menu.

Messages

In version 3.0 of the Windows help compiler, the output messages are directed to a file and the Help Magician displays the contents of the file in a window when compilation is complete. In version 3.1, the ERRORLOG option is used in the project file and the file is displayed in a window when compilation is complete. This allows scrolling back and forth through the messages instead of trying to view them as they scroll by on the screen. The messages can be sent to the default printer by clicking on the Print button.

Version 3.00b of the Windows help compiler does not support redirection or the use of the ERRORLOG option. Therefore, it is not possible to display the messages in a window.

Rebuild All

Rebuilds all files in the current help system without regard for the time and date stamp on the files.

One Page Preview

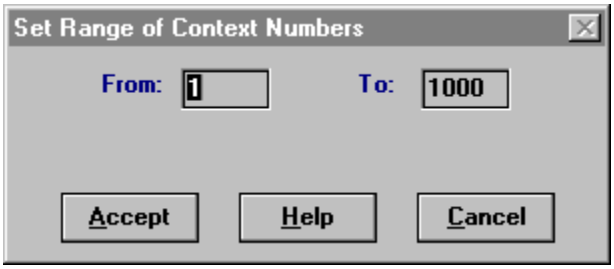
This option allows you to pick one topic page and compile it quickly for viewing in Windows Help. All hotspots are referenced to a single page especially created for the one page preview process, so that those individual topics need not be compiled. To use this option, display the desired page in the Help Magician's editor and select "One Page Preview" from the Build menu. The compiler will be invoked to compile the page. Note: no compiler errors will be reported during this stage. When compilation is complete, WINHELP.EXE will be called to display the single topic.

View Compiler Messages

This sub menu will display the messages from the Help Compiler during the last build, if any. If there has not been a build, the sub menu will be grayed out and cannot be selected.

Build Context Spy

Selecting the Build Context Spy sub menu brings up a form to enter the range of Context Numbers to include in the Context Spy help file.



Overview

A Context Spy help file can be used to determine the Context Numbers accessed by an application calling WinHelp to display specific help topics. This information can then be used to establish context sensitivity between the application and the help system.

Help File Name

The name of the Context Spy help file must be the same as the name of the help file called by the application and must be accessible to WinHelp in the path or working directory. If necessary, rename the original help to save it from being overwritten by the Context Spy help file and to prevent it from being accessed by WinHelp instead of the Context Spy help file.

When the calling application accesses the help file, a WinHelp window will appear as seen in the sample bitmap below. In this case, the calling application used Context Number 762 when it called WinHelp.

Update Context Numbers

The sub menu is only available if "External Map" is selected in the Compiler Options form, Map tab. Selecting this sub menu will update the relationship between the context numbers and context strings as provided by the external map file.

An automatic update is performed when the file is compiled and it is only necessary to use this function if changes have been made to the context numbers and/or context strings in the file in memory or in the external map file.

Multiple Files

The Multiple Files function of the Help Magician has been maintained in the Pro version for compatibility with earlier versions of the Help Magician. Project Management is much more powerful than Multiple File build and it is recommended that Multiple File projects be converted to Project Management projects.

See Project Management for detailed information on converting and maintaining Projects.

The Help Magician can compile and build a help system from multiple source files provided that all the information is written to the project file at the time of the build.

Step One

Open each of the files to be included in the build and select Compiler from the Options Menu. Click on the Write #include File radio button and press the Accept button. Select Write RTF from the Build Menu. This will write the RTF file and the context string map.

Step Two

Select one of the files to be the Master source file and load it into the Help Magician.

Step Three

Select Multiple Files from the Build Menu. Add the names of the RTF files that will be included in the build in the text box under the [FILES] label. Do not add the RTF file name for the Master file.

Step Four

Add the names of the map files that will be included in the build into the text box under the [MAP] label. Do not add the Map file name for the Master file.

The #include and the <> symbols will be added to the map file name and written to the project file when the file is compiled.

Step Five

Compile the Master help file and all of the files listed in the [FILES] Section will be included in the build.

Context Numbers and Context Strings may not be duplicated between multiple source files that will be combined into one help system.

Assign a range of Context Numbers for each of the members of the build.

One method that would ensure that all Context Strings were unique between the source file would be to assign a prefix for all Context Strings for each of the members of the build.

The Help Magician does not validate duplication of Context Strings or Context Numbers between multiple source files. You will receive error messages from the help compiler if there are duplicates.

A large file can be split into two or more files using the Save Range function.

If you export an RTF file to Word for Windows and change the file, the Help Magician detects the fact that the file was changed and asks if you want to re-import the file when you return. If the file is a member of a Multiple File build, press No. Use the Import RTF / Include Project sub menu available from the File Menu to re-import the file.

Team Help

You can use the Multiple Files capabilities of the Help Magician to build a help system as a team effort.

Each member of the team would be assigned a specific section of the complete help system and would write their individual sections maintaining unique Context Numbers and Context Strings as described above.

Networks

Microsoft's help compiler will not cross a network drive to compile a source file. If any of the source files are on network drives, copy them to a local drive before compiling, preferably the directory that contains the Master file (described above). Be sure that the path for the RTF file and the map file are accurate in the Multiple Files form.

See Also
Project Management

Call Word Processor

Select Call Word Processor to invoke the Word Processor specified in the Path options form window, with the RTF file pertaining to the current help file.

The file can be read back into the Help Magician as an RTF file with a few precautions and exceptions. The user should be aware that not all word processor features are supported by the Help Compiler or the Help Magician and may consequently be ignored. See the section on Importing RTF files and RTF Technical Specifications.

Note that the specification in the Paths option form must include the complete drive, path, and executable filename, such as:

C:\WINWORD\WINWORD.EXE.

Your word processor must be able to accept a filename with a .RTF extension as a command line argument. If it does not, load your word processor and read in the RTF file by selecting the appropriate options from the File Menu.

If the word processor is open at the time and a copy of the file is one of the windows, the word processor may open a second version of the RTF file. To avoid this, either close the windows containing the RTF file or close the word processor each time you return to the Help Magician.

See Also

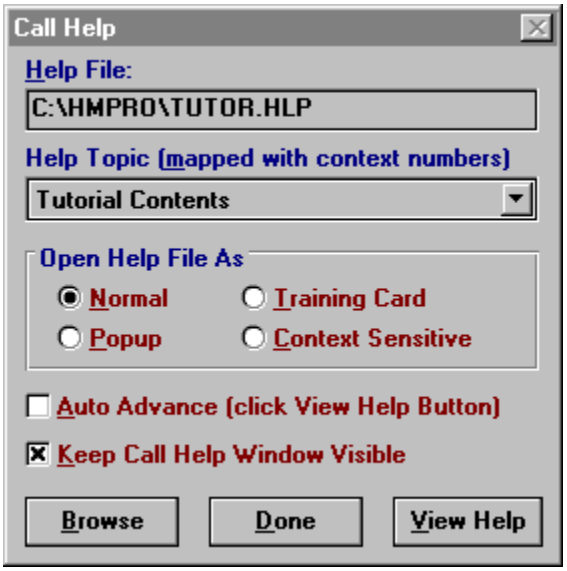
- Importing RTF Files
- RTF Technical Specifications

Call WINHELP.EXE

This will call WINHELP.EXE and display the compiled help system as your end users will view it. The Help Index will be displayed. If you have recently re-compiled the help system, you may encounter a "Help File Changed" message from Windows Help. Select OK and then select Call WINHELP again.

WinHelp Options

This will display the Call Help form illustrated below. There are several ways of calling WinHelp 4.0 in Win95 and they are all available from the Call Help form.



Help File

The currently loaded HLX filename should appear here with an HLP extension. You may change the name in the text box to another help file or search for one using the Browse button. Simply click in the text box, select the Browse button and choose a help file using the file directory dialog.

Help Topic

If any help topics have had context numbers assigned, they will appear in the Help Topic drop down box. Select one of the help topics to have WinHelp display it when called.

Open Help File As

The Call Help dialog can simulate a call to WinHelp from an application, a training card, as a popup window or as a stand alone application.

- Select the Normal radio button to call WinHelp as if it were a stand alone application (i.e. run by double-clicking an icon).
- Select the **Popup** radio button to call WinHelp as if it were called from an application as a popup window.
- Select the **Training Card** radio button to call WinHelp as if it were called from an application running the training cards feature.
- Select the **Context Sensitive** radio button to simulate calling WinHelp from an application using context sensitive help (i.e. using Context Numbers).

Auto Display Next Topic

Use this feature to view consecutively mapped topics. After the Help file is displayed, you may see the next mapped topic by repeatedly selecting the View Help button.

View Help

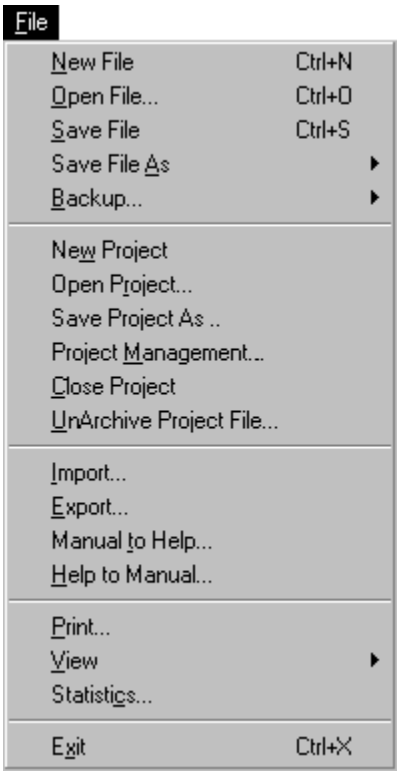
Select this button to view the Help file.

Select the Done button to close the Call Help dialog window.

File Functions

Single File Development vs. Project Management

If you will be maintaining a single source file, use the menu functions New File, Open File, Save File, and Save File As. If you plan on developing a help system with multiple source files and/or working on a team development effort on a network, use the project management functions New Project, Open Project, Save Project As, Project Management, and Close Project. If you do develop a single file, you can add it to a multiple file project later using the Project Management functions. However, once a single file has been added to a project, it can no longer be edited as a single file.



Click on the desired area of the File Menu for specific Help

New File

Select New from the File Menu to start a new help file. All the information from the previous file, if any, will be erased. If the previous help was edited but not saved, you will be warned and you will have the option to save the file before the information is erased. Editing begins at page one. The file name will be DEFAULT.HLX. Note that Help Magician will not let you save a file with the name DEFAULT.HLX so you should use Save As and give it a name.

If Project Management is active, New File will close the project and set up a new single file build. Select Project Management, New File to open a new file under the current project.

ROOT Directory Philosophy

The 'ROOT' directory for the current help system is any directory that you choose to open a help source file from or save a help source file to. This directory will be recorded in the help project file and all files pertaining to the current help file must be in this directory. Bitmaps must also be included unless the directories are entered in the Bitmap Directories list box in the Paths form found under the Options Menu.

Open File

Select Open to read in an existing help file with a .HLX extension. All of the keywords, Jumps, Popups, defined fonts, etc. that were saved with the file will be read in with the help text. The 'ROOT' directory for the current help build will be set to the directory from which you load the help file.

The last directory accessed for a .HLX file is stored in the HLPMAGIC.INI file and is restored when the Help Magician is run. If you are opening an older Help Magician file prior to version 3.0, the file will be converted to the new format.

If Project Management is active, Open File will close the project and set up a single file build. Select Project Management, and double-click on one of the listed files in the File Selection listbox to open a file under the current project.

Save File

Save will save the current help file without asking for a file name unless a file name has not yet been established. The default file name, DEFAULT.HLX, is not considered to be a designated file name and you will be asked to provide a file name if you select Save from the File Menu. The 'ROOT' directory for the current help build will be set to the directory to which you save the help file. When in Project Management mode, Save File will save the currently opened source file to disk as a member of the project.

Save File As

Save As gives you the option of specifying a file name for the current help file before saving. As described in Save, above, Save As will be called if a file name has not yet been assigned to the current help file. You can also use Save As to save a help file under a different file name than the name that is currently assigned to the file. Note that if you are using Help Magician in Project Management Mode, a newly saved file will added to the project member list (see Project Management on adding files).

The last directory accessed for a .HLX file is stored in the HLPMAGIC.INI file and is restored when the Help Magician is run. When using Save As in Project Management mode, the new filename you give to the source file will be updated in the File Selection listbox on the Project Management form.

Extension

The file extension for a Help Magician file must be .HLX. If you enter an extension other than .HLX, it will be changed to .HLX when saved.

More Info

See the section on Importing Text, later in the Help file, for a more detailed explanation of the format of text files to be imported into the Help Magician.

Save Range

The Save Range function allows you to save a range of pages to split a large file into two or more files.

You can then use the Multiple File capabilities of the Help Magician to build a help system from more than one file.

Range

Enter the starting and ending pages in the Save Range form and press Accept. A Save Range operation always functions as a Save As... and you will be required to enter a name for the file to be saved.

Convert Links to "Other Source"

Check this box to convert internal links to external links, referenced by Context String rather than by topic number, so that the links will be effective when the file(s) are included in a multiple file build or a project. Links referencing pages included in the saved range will not be converted.

See Also

- Building Your Help File
- Importing Text Files
- Multiple Files
- Project Management

Backup

The Backup sub menu has two sub options, Backup and Restore. Backup will write a backup copy of the current help file with the same file name and a .HLK extension. If you should lose a copy of your original HLX file, simply rename the backup HLK to HLX. Restore will restore a source file with the .HLK extension.

Import

Import RTF

The Help Magician will read RTF files generated by Word for Windows, Ami' Pro or Word Perfect for Windows version 6.0a.

Import Text

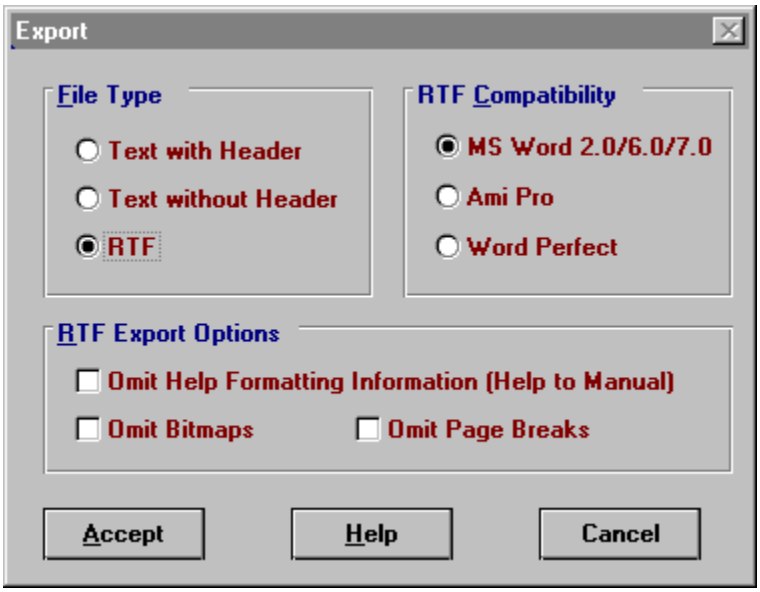
ASCII text can be imported in the Help Magician. The text can be written out from a word processor but some formatting is required before it can be read into the Help Magician.

See

Importing RTF Files
Importing Text Files

Export

Selecting Export from the File menu brings up the Export Form.



File Type

Export Text

The current help file can be exported as ASCII text without formatting. This is provided so that the help file can be read into a word processor that does not read RTF files. This special formatted file can be read back into the Help Magician, but none of the formatting will be retained.

Text With Header

The title, keywords, multiple keywords, context number, Browse group and the topics included in the Browse group will be written to the file before the help text on every page. Each section is preceded by a header (\Title:, \Keywords:, \Multiple Keywords:, \Context Number:, and \Browse:). The keywords and multiple keywords are separated with commas. The Browse group is separated from the included topics by a colon. See Importing Text Files for more information.

Text Without Header

Without the header, only the caption, Title: and the title are included for each page.

RTF

The Export RTF option will write an RTF file for the purpose of importing into a word processor.

HTML

The Export HTML option will write an Hyper Text Markup Language (HTML) file compatible with version 2.0 of the HTML specification. This file type is used for Internet World Wide Web documents. The filename will have a .HTM extension.

RTF Export Options

If the ***Omit Help*** *Formatting Information* check box is checked, the help formatting such as the hyperlinks

for Jumps and Popups will not be written to the RTF file. The resulting RTF file is more suitable for converting a help file to a manual.

If the ***Omit Bitmaps*** check box is checked, the images in the help source file will not be written to the RTF file.

RTF Compatibility

Select the RTF Compatibility for your word processor, Word for Windows 2.0/6.0/7.0, Ami' Pro, or Word Perfect for Windows.

See Also
Importing Text Files

Exit

This will exit the Help Magician and return control to Windows. You will be warned if the current file has been edited since the last save and you will have the option of saving it before exiting, or to cancel the exit and return to the editor.

Project Management Overview

What is Project Management?

Project Management is a feature of the Help Magician that helps you manage all the multiple source files within a help project. It takes care of handling clashes of context strings, context numbers, and topic titles across all members of a project. Entire projects can be archived, copied, moved, or deleted. Context relations for the entire project can be viewed or printed. Project Management is an excellent option if you are doing multiple file help authoring or doing team/workgroup development on a network.

Why use Project Management?

Project Management is so much more powerful than the Multiple Files feature on previous versions of Help Magician. For single help authors, managing multiple source files is easier. No need to keep track of a "master file" by entering associated RTF or MAP files. For simultaneous team development work on a network, Project Management makes sure that each help author doesn't interfere with each other or duplicate the context strings, context ids, and topic titles of other project members. This checking takes place instantly as the help author enters a context string, context number, or topic title.

Storing your Project-Related Files

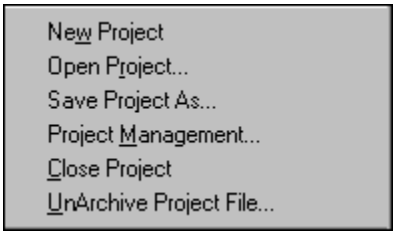
It is highly recommended that you create a unique directory for each unique help development project that you work on with the Help Magician. This directory becomes the "build" directory for the project file and project management. You can use the file manager in Windows to create this unique directory. A single directory containing ALL project-related files (.hlx source files, bitmaps, shed files, and multimedia) is especially needed for team/workgroup development on a network.

Doing a Build with Project Management

It's simple to make a final help file with Project Management.

1. Open a project file.
2. Make sure all .rtf and .h files are up to date (see status in File Selection Listbox). If they're not, open each source file and select Build...Write RTF for Compiler option or use the Build All button to automatically do this procedure.
3. Open the source file that you want to be your master file. It should contain your contents page.
4. From the Build Menu, select Rebuild All.

The following File menu items deal with project management.



Click on the desired area of the menu for specific help.

See Also

File Selection Listbox

New Project

Selecting New Project will clear the editor and prompt you for the new project's filename. If one exists by the name you select, you will be asked if you wish to overwrite it. Next, the Project Management Form will appear. The next step is adding an existing HLX file to the project, creating a new HLX file, or importing existing text into a new HLX file. Refer to the section below for more options on this form.

This command sets Help Magician into Project Management Mode.

Tech Tip: All project-level information such as the individual component files (.hlx) that you add to the project, window and compiler definitions, macros, and external file references are stored in the project file (with an extension of .hmp). A database containing fields for context numbers, context strings, and topic titles for the entire project is also created. The database uses the files with the .isd, .isf, and .ism extensions.

Open Project

Choosing Open Project will bring up a file dialog box with a list of existing project files (with a .hmp extension). After selecting an existing project to open, you will be presented with the Project Management Form. Refer to the section below for more options on this form.

This command sets Help Magician into Project Management Mode.

When you open a project file, other users on the network will not be able to use the Global Project commands Delete or Move from the Project Management Form.

Save Project As

The Save Project As option allows you to save the currently opened project file (.hmp) under a different name. The 3 database files (.isd, .isf, and .ism) are also saved under the new name. The Save Project As option will not allow you to save the project file to a different directory. Use the Project Management Form's Global Copy or Global Move commands to accomplish this.

Close Project

Selecting this menu item will close the current project file and clear the editor. If you have work in the editor that has not been saved, you will first be prompted if you wish to save your work.

Closing a Project file sets Help Magician back into Single File Mode.

UnArchive Project File

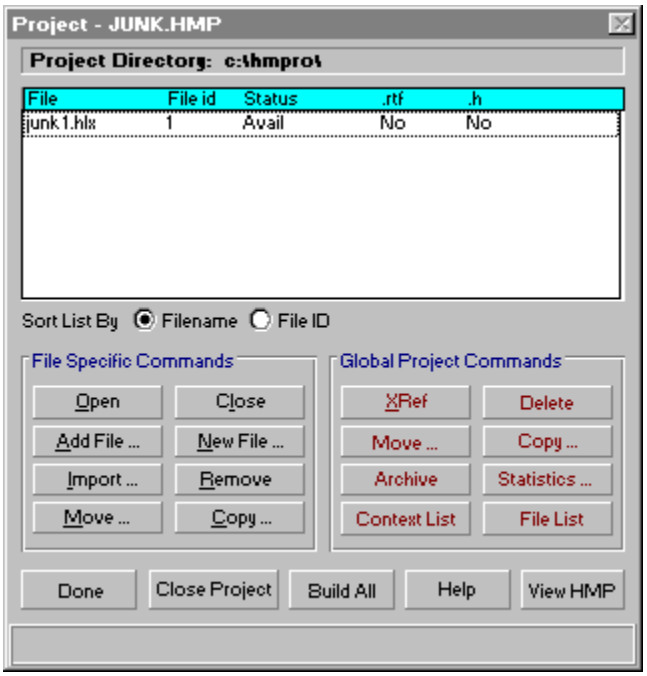
The option allows you to unarchive a Help Magician project file and all its components (stored in a .HZP file) to a subdirectory you choose. When the file is unarchived, the new subdirectory becomes the current help file "ROOT" directory for the build. All references to paths are converted to the new subdirectory after unarchiving.

You can archive a project file by clicking on the Archive button in the Project Management Form.

Archiving is a handy way of backing up your entire project into a single compressed file for moving to a different machine. All the project-related files are stored in a HZP file, which has the same structure as the industry standard ZIP compression format. The archive and unarchive features of the Help Magician require a support program ARCHIVE.EXE to exist in the same directory as HLPMAGIC.EXE - the Help Magician program.

Project Management

Selecting New Project, Open Project, or Project Management menu option will bring up the Project Management form. Here you can add files to the project; delete them from the project; globally move, copy, or delete files; archive files; view and print context relations of all member files; and refresh the project database of all context strings, context numbers, and topic titles. The Project Management form is shown below. Click on the preferred area for more information.



Click on the desired area of the Project Management form for specific Help

See Also

Project Management Overview

File Selection Listbox

The File Selection Listbox provides a list of Help Magician source files (.hlx) that are part of the currently opened project. Double-click on one of the source files to edit it. Files may be added to this list by clicking on Add File, New File, or Import. Files may be removed from this list by highlighting the file and clicking on Remove. The current directory of the Project is displayed just above the File Selection Listbox.

The listbox contains other information relative to the individual source file:

File ID is a number that the project manager assigns to the file when the file is added to the list. This number is unique and is never duplicated, even if the source file is removed. All references in the database to this file is done internally through this file id.

Status reports the status of the current file. One of three file status modes could be shown. "Avail" if the file is available for editing. "In Use Locally" if the file is currently being edited on your system. "In Use Remotely" if the file is currently being edited by someone else on the network.

.rtf reports the status on the companion RTF file that is needed for the final project build. A "Yes" will be shown if the RTF file exists and is up to date (ready for the final project build). An "Old" status will be shown if the RTF file does exist, but is older than the source file. A "No" will be shown if an RTF file doesn't exist. If "Old" or "No" is displayed, you should open the source file and select Write RTF for Compiler under the Build menu when you are ready to build your final help file. Executing this function will also update the MAP file.

.h reports the status of the companion MAP file that is needed for the final project build. A "Yes" will be shown if the MAP file exists and is up to date (ready for the final project build). An "Old" status will be shown if the MAP file does exist, but is older than the source file. A "No" will be shown if a MAP file doesn't exist. A "???" will be shown if the MAP file is an external file defined in Options...Compiler...Map. The project manager doesn't know in this case if the file is up to date.

View HMP

Views the contents of the HMP file, when in project mode.

Build All

This feature loads each HLX file in the list one at a time and writes the RTF file for the compiler for each HLX file. This is useful when you have a lot of HLX files to rebuild and want to automate the process.

Open

Opens a currently selected source file member of the project for editing. You could also double-click on a file in the File Selection Listbox to open it. When you are editing this file, other users on the network cannot access this file. Note that you can only edit one source file member at a time, if it's available. When you open a source file member, any other one that may be in the editor will be closed and the new one will be opened.

Close

Closes the current source file member that you are editing and clears the editor. Note that you cannot close someone else's file. Closing the file also makes it available to other help authors in a network environment.

Add File

Add File brings up a file dialog box prompting you for a Help Magician source file (.hlx) that will be added to the project. If you select a file that exists in another path that is different from the current project's path, the file will be moved to the current project's path and tagged as a member of a project. Once a file has been added to a project, it cannot be edited as a single file.

The project manager requires that all source file members of a project (.hlx) be in the same directory as the project file (.hmp). If the project is part of a team development effort on a network, all bitmaps and multimedia files need to be copied into the current project's path, so that everyone on the network will have access to these files- independent of the workstation's drive/path assignments.

When a source file is added to a project, all context string/context number/topic title references are added to the project database. If any duplicates to other members of the project are found, a report is generated which you can view or print.

In addition, the project manager scans the source file and extracts certain project information that will be added to the project file (.hmp). This information includes main and secondary window info, macros, compiler settings, and external files. Only new information is added to the project file (nothing is replaced). For example, if you have two secondary windows named "Glossary" and "Contents" in your source file and the project file already has "Contents" defined, only the configuration for the "Glossary" secondary window will be added to the project file. Thus, if you are adding files to the project, pick the first file that contains the project settings that you want to use.

Any files that are added to the project, will all share the same project information described in the previous paragraph. Any additions or deletions to this project information will affect other members of the project. Help Magician menu options that affect project information are Options...Paths (Bitmap Directories only), Options...Appearance, Options...Compiler, Options...Windows, and Macros.

New File

Selecting the New File button prepares the editor for a new file. If you had a modified source file in the editor, you will be asked if you wish to save it. All the project information discussed in Add File above (if any) becomes instantly available. Once the New File is "Saved File As", it is then added to the project file following the same procedure as Add File above.

Import

Clicking on the Import button brings up the Import Form. See topic on Importing for further information on this form. Once a file is imported and Saved As, it is then added to the project file following the same procedure as Add File above.

See Also

Import

Remove

Remove removes a currently selected source file in the File Selection listbox from the project. Any references to the file in the project database are also removed. You will be given an option to delete the source file from disk. Note that you cannot remove a file if it is currently in use. This command is password protected (see Statistics (Project) for setting password).

See Also

Statistics (Project)

Move

Move will move a currently selected source file to another directory of choice. The new full path will be displayed in the File Selection listbox. Note that if you are doing a team development effort, this file will no longer be available to other team members on the network (because it will be located in a directory different from the project's path). This command is password protected (see Statistics (Project) for setting password).

See Also

Statistics(Project)

Copy

Copy will copy a currently selected source file to another directory of choice. The new full path will be displayed in the File Selection listbox. Note that if you are doing a team development effort, this file will no longer be available to other team members on the network (because it will be located in a directory different from the project's path). This command is password protected (see Statistics (Project) for setting password).

See Also

Statistics (Project)

Xref

Xref scans all source file members of the project and refreshes the project database of all context strings, context numbers, and topic titles. It will find duplicates and produce a duplication report which you can view and print. You may need to do this if the database gets corrupted (through power failure or whatever) or you get DB errors. Note that this command cannot be executed if any of the project source file members is in use.

Delete

Clicking on delete will give you a choice of globally deleting the project file and optionally all its related files. Delete is one of those powerful commands that could be dangerous in the wrong hands- thus it can be password protected (see Statistics (Project) for setting password).

See Also

Statistics (Project)

Move

Move will globally move the project file and all its related files to another path. This command is password protected (see Statistics (Project) for setting password).

See also

Statistics (Project)

Copy

Copy will globally copy the project file and all its related files to another path. This command is password protected (see Statistics (Project) for setting password).

See Also

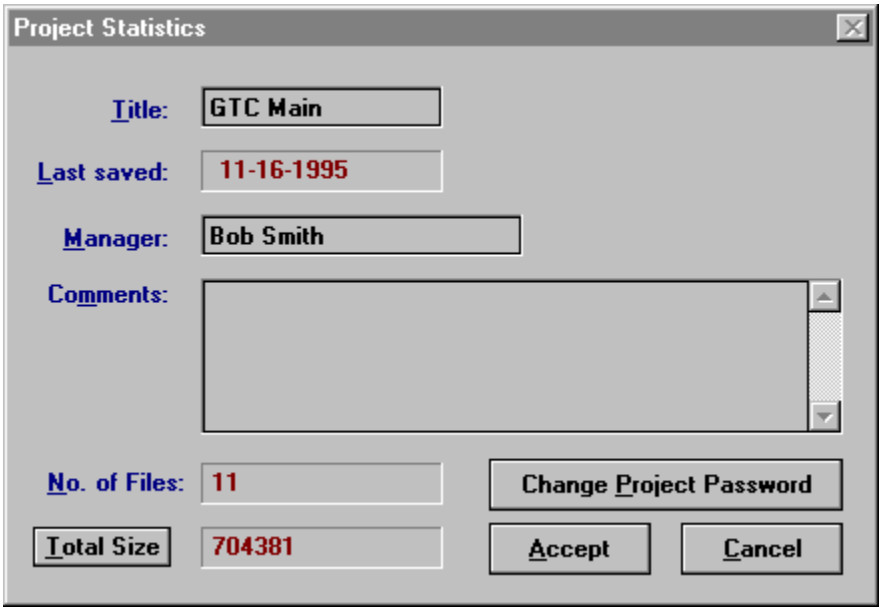
Statistics (Project)

Archive

Archiving is a handy way of backing up your entire project into a single compressed file for moving to a different machine. All the project-related files are stored in a HZP file, which has the same structure as the industry standard ZIP compression format. The archive and unarchive features of the Help Magician require a support program ARCHIVE.EXE to exist in the same directory as HLPMAGIC.EXE - the Help Magician program.

Statistics (Project)

The Statistics button allows you to set project-level textual information. It will also allow you to set the Supervisor password, which controls access to project moving, copying, and deleting.



Title

Title of the project.

Last Saved

Contains the date of the last time Project file (.hmp) was updated.

Manager

The name of the person managing the help project.

Comments

Any textual comments about the help project.

Number of Files

Reports the total number of files associated with the help project.

Total Size

Click on this button to calculate the total size of all the files associated with the help project.

Change Project Password

Click on this button to change the Supervisor password. If a password is already defined, you will be prompted for the old password. The password controls access to project-level copy, move, and delete functions.

Help to Manual

Selecting this sub menu brings up the **Export** form with the options set for **Help To Manual**.

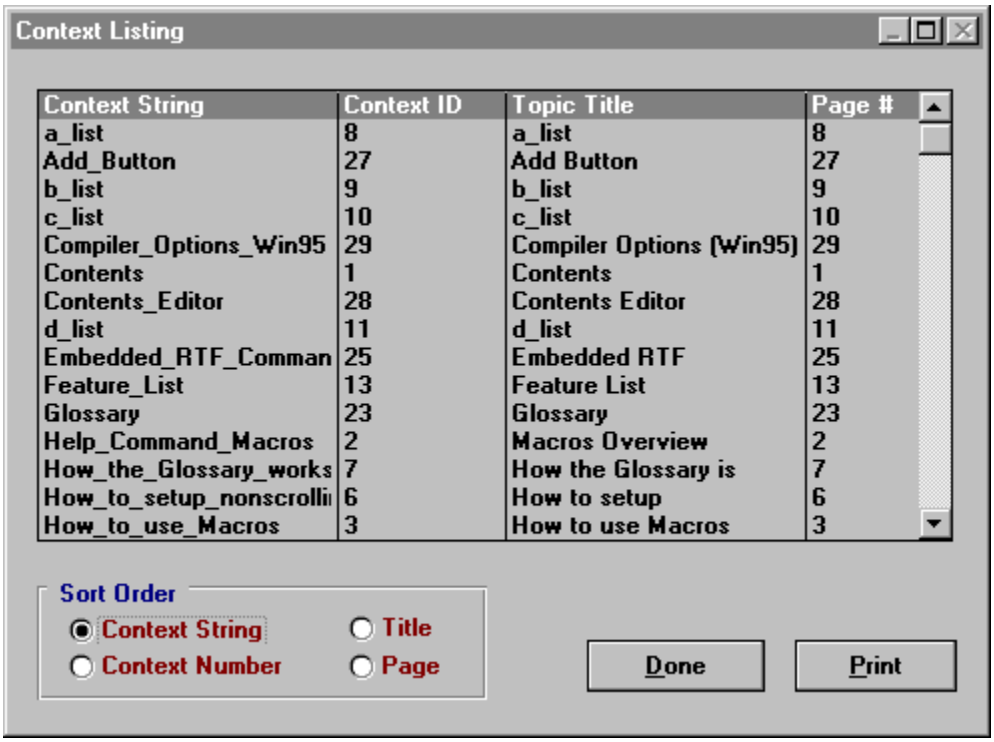
If the ***Omit Help*** *Formatting Information* check box is checked, the help formatting such as the hyperlinks for Jumps and Popups will not be written to the RTF file. The resulting RTF file is more suitable for converting a help file to a manual.

Statistics (File)

The Statistics dialog displays the title of the help file, the date and time of the last save, the author, comments, the number of pages and the word count, if desired.

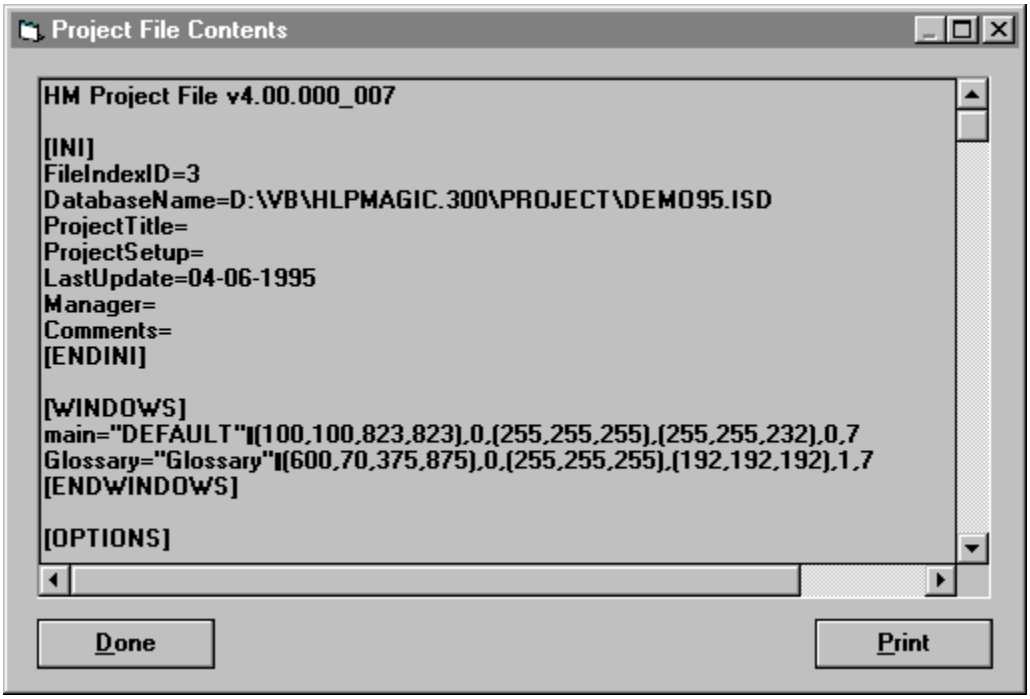
Context List (project management form)

The context list option reports a listing of all context strings, context numbers (ids), and topic titles for the entire project. You can view the database in any sorted order. You can also print the contents of the database in any sorted order.



File List

The File List button reports a list of all associated files to a help project. It will list project source files, database files, bitmaps, and multimedia files. You have the option of printing this list to a printer.



Manual to Help Conversion

The Manual To Help conversion function, available from the "Files" menu, allows you to create a help file from a manual or documentation. The manual must be created in a word processor such as Word for Windows or AmiPro and saved in Rich Text Format (RTF). By placing special "Import Commands" in the document, you can control how your manual will be converted to a help file.

How to Pass Project Options

When you convert a manual or document you might want to specify more advanced help system features such as secondary windows, macro definitions, copyright notices, etc. This can be accomplished by running the Help Magician, defining the appropriate options using the menus, and saving the file with the same name as your manual with a .HLX extension. It's that simple. When you convert your manual, you will have the option to enter a source for project options - enter the name of the file you saved in the Help Magician. The converter will use the options you selected in your "dummy" help file.

Keeping the Help File Updated

After conversion you should overwrite the dummy file with the converted file. From that point on everytime you update your manual and wish to recreate the help file, simply specify the old help file name for the Project Options Source. Any changes you might have made to the project options will automatically be re-used in the new help system.

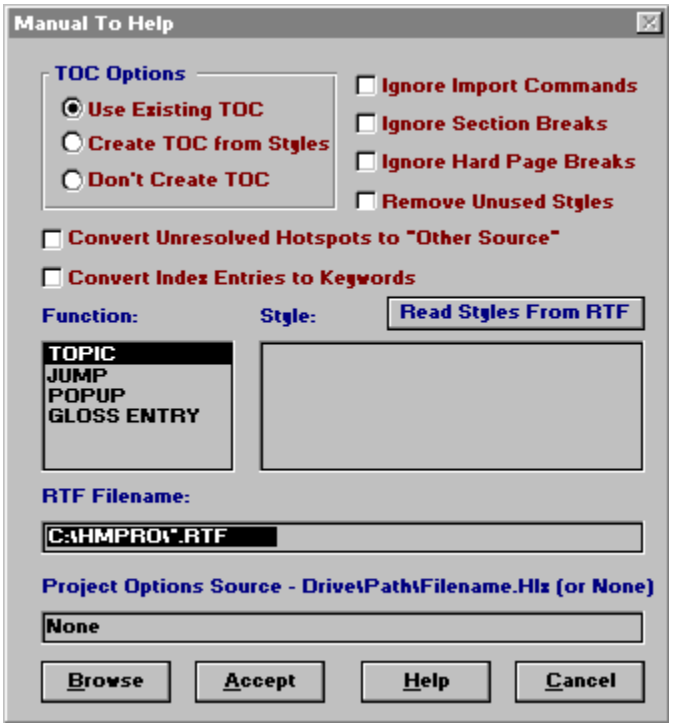


Table of Contents (TOC) Options

If you have a Table of Contents in your document, you have the option of converting it to a main "contents" topic, where each TOC entry will be converted to a jump to its respective topic. The "Use Existing TOC" is selected by default.

If your document does not have a TOC, you may choose to create a main contents topic from scratch by selecting "Create TOC from Styles". You must first enter the name of your document, previously saved in RTF format, in the "RTF Filename" text box. You may use the Browse button to search for your document.

After you have done this, select "TOPIC" from the Function list and then select a style or styles from the Style list that you want the Help Magician to recognize as topic headings. Every time one of the selected styles is encountered during the conversion process, a new topic will be created. and entered into the main contents topic.

You may choose to ignore any TOC and not have a main contents topic created by selecting "Don't create TOC".

Import Commands

If you have entered special import commands into your manual or document, you may tell the Help Magician to ignore those commands by selecting "Ignore Commands". See "Formatting Import Commands" for more information.

Ignoring Section and Hard Page Breaks

These features can be used when you have defined your own page breaks using the RTF Import Commands and wish to ignore any section and hard page breaks in the manual.

Removing Unused Styles

This feature is useful when there are alot of unused styles defined in the manual you are converting and you don't wish to have them added to the styles list in the Help Magician.

Converting Unresolved Hotspots

You can have the Help Magician convert unresolved hotpots that it finds to "Other Source" rather than leaving the hotspot unresolved.

Converting Index Entries

If you created an index in your manual and defined index entries throughout the text, you may have the Help Magician convert these entries into keywords. As the Help Magician finds index entries it will add the text to the list of keywords for the page the entries appear on.

Read Styles From RTF

The Manual to Help converter can take advantage of styles you have created in your document by converting the text they mark into topics, hotspots, TOC, or glossary entries.

RTF Filename

Enter the name of your document. It should have been previously saved in RTF format from your word processor.

Browse

Pressing the Browse button will pop up a file dialog to automate the entry of the path and filename for the current field.

Project Options Source

Enter the name of a Help Magician file (.HLX) that will be the source of the Help system project options. Refer to the section above on "How to Save Project Options".

See Also

Formatting Import Commands

Formatting Import Commands

You can control the conversion of your document into a help system by inserting special Import Commands in your document. You can simply type a command into your document using the format described below and then format the command as **invisible or hidden text**. These commands will not be displayed when the manual is printed and they can be hidden while viewing the document in your word processor by selecting the appropriate options from the menus. Each command has a short hand form you may use for ease of entry. Do not embed or group these commands inside one another. They may be concatenated one after the other as long as each command is complete and is paired with its corresponding ending command, if it needs one.

The commands and their respective functions are listed below:

[OMITPAGE] or [OPG]
Place this command at the top of any page you do not want in your help file.

[OMITSECTION] or [OS]
Place this command at the top of any section you do not want in your help file. Sections are defined in Microsoft Word using section breaks instead of page breaks.

[OMITPARAGAPH] or [OP]
Place this command at the beginning of any paragraph you do not want in your help file.

[BROWSEGROUP groupname:sequence] or [BG groupname:sequence]
Place this command at the top of any topic that you wish to assign to a browse group. Enter the group name and then the sequence number separated by a colon. Do not place any spaces between the colon and the group name or sequence. The sequence should be a decimal number, unique for the topic it is assigned and contain leading zeros to make it three digits long. You may only assign one browse group command per topic.

Examples:
[BROWSEGROUP commands:020]
[BG edit:012]

[KEYWORDS keyword1;keyword2;] or [KW keyword1;keyword2;]
Place this command at the top of any topic that you wish to assign keywords. Separate keywords as show with semicolons. Each keyword must have a terminating semicolon. Note spaces are not required between keywords and semicolons.

Examples:
[KEYWORDS Making Entries; Editing Keys; Editing;]
[KW Making Entries;Editing Keys;Editing;]

[MULTIKEYWORDS keyword1;keyword2;] or [MK keyword1;keyword2;]
Place this command at the top of any topic that you wish to assign multiple keywords. Separate keywords as show with semicolons. Each keyword must have a terminating semicolon. Note spaces are not required between keywords and semicolons.

Examples:
[MULTIKEYWORDS Keyboard layout;]
[MK Keyboard layout; Keys; Atl-A;]

[BUILDTAGS tag1;tag2;] or [BT tag1;tag2;]

Place this command at the top of a topic that you want to assign build tags. Separate build tags as show with semicolons. Each buildtag must have a terminating semicolon. Note spaces are not required between tags and semicolons.

there is a limit of 16,383 build tags total.

[JUMPTOPIC secwin fmtcmd title] or [JT secwin fmtcmd title]
The JUMPTOPIC and ENDJUMP commands are used together in pairs to mark text as a hypertext jump hotspot. Place the JUMPTOPIC command at the beginning of a text segment that you want to be a jump hotspot. The following fields are mandatory except for the secwin field, which may be omitted:

"secwin" field:
Enter a secondary window name in the "secwin" field to display the topic in or leave it blank to display the topic in the main window. The secondary window must be defined in the project options using the method described above in the "How to Pass Project Options" section.

"fmtcmd" field:
To format the way the hotspot will appear in the help file, enter one of the following options in the "fmtcmd" field:

NORMAL Display hotspot with color and underlining.
NOCOLOR Hotspot will appear with underline only. No color will be displayed.
NONE Hotspot will not be formatted. No color or underlining will appear.

"title" field:
Enter the title of the topic in the "title" field that will be displayed when the hotspot is selected by the user.

Examples:
The solenoid is a [JUMPTOPIC NONE Electrical Parts]electromechanical[ENDJUMP] device.
[JT bluewin NONE Sale Types]Periodic[EJ] sales dropped in the third quarter.
Using the [JT NOCOLOR Main Topic]Help Magician[EJ] is easy and fun.

[JUMPMACRO fmtcmd macdefn] or [JM fmtcmd macdefn]
The JUMPMACRO and ENDJUMP commands are used together in pairs to mark text as a hypertext macro jump hotspot. Place the JUMPMACRO command at the beginning of a text segment that you want to be a jump hotspot to a macro. The following fields are mandatory except for the secwin field, which may be omitted:

"fmtcmd" field:
To format the way the hotspot will appear in the help file, enter one of the following options in the "fmtcmd" field:

NORMAL Display hotspot with color and underlining.
NOCOLOR Hotspot will appear with underline only. No color will be displayed.
NONE Hotspot will not be formatted. No color or underlining will appear.

"macdefn" field:
Enter the name of a Macro Definition that was defined using the method described in the "How to Pass Project Options" section. The macro(s) defined in the macro definition will be executed when the hotspot is selected by the user.

Examples:
The solenoid is a [JUMPMACRO NORMAL PlayAudio]electromechanical[ENDJUMP] device.
[JM NONE ShowAbout]Periodic[EJ] sales dropped in the third quarter.
Using the [JM NOCOLOR AddButton]Help Magician[EJ] is easy and fun.

PlayAudio, ShowAbout and AddButton are examples of macro definitions. A macro definition is a user created name in the Help Magician for a group of macros or just one macro that control the WinHelp environment. Refer to the section on Help Macros for more information on controlling help files with macros.

[JUMPFILE secwin fmtcmd filename contextstring] or [JF secwin fmtcmd filename contextstring]
The JUMPFILE and ENDJUMP commands are used together in pairs to mark text as a hypertext jump to another help file. Place the JUMPFILE command at the beginning of a text segment that you want to be a jump hotspot to another help file. The following fields are mandatory except for the secwin field, which may be omitted:

"secwin" field:
Enter a secondary window name in the "secwin" field to display the topic in or leave it blank to display the topic in the main window. The secondary window must be defined in the project options using the method described above in the "How to Pass Project Options" section.

"fmtcmd" field:
To format the way the hotspot will appear in the help file, enter one of the following options in the "fmtcmd" field:

NORMAL Display hotspot with color and underlining.
NOCOLOR Hotspot will appear with underline only. No color will be displayed.
NONE Hotspot will not be formatted. No color or underlining will appear.

"filename" field:
Enter the DOS filename of the help file that you want the hotspot to jump to.

We advise not entering a path with the filename to avoid forcing end users of the help system to have specific paths on their systems. For example if you specified a path such a D:\HELP\MYHELP.HLP, and the user did not have a "D": drive on their system, WinHelp would report an error and the user would not be able to correct the situation. Try to avoid using hard coded paths in help systems.

"contextstring" field:
Enter the context string associated with the topic in the help file that you are jumping to.

Examples:
Solenoids are [JUMPFILE NORMAL solenoid.hlp Solenoids] electromechanical [ENDJUMP] devices.
[JF NONE schedule.hlp Period]Periodic[EJ] sales dropped in the third quarter.
The [JF helpwin NOCOLOR hlpmagic.hlp Contents]Help Magician[EJ] is easy and fun.

[JUMPSOURCE secwin fmtcmd contextstring] or [JS secwin fmtcmd contextstring]
The JUMPSOURCE and ENDJUMP commands are used together in pairs to mark text as a hypertext jump to another source file within a multiple file build. Place the JUMPSOURCE command at the beginning of a text segment that you want to be a jump hotspot to another topic in a multiple file build. The following fields are mandatory except for the secwin field, which may be omitted:

"secwin" field:
Enter a secondary window name in the "secwin" field to display the topic in or leave it blank to display the topic in the main window. The secondary window must be defined in the project options using the method described above in the "How to Pass Project Options" section.

"fmtcmd" field:
To format the way the hotspot will appear in the help file, enter one of the following options in the "fmtcmd" field:

NORMAL Display hotspot with color and underlining.
NOCOLOR Hotspot will appear with underline only. No color will be displayed.
NONE Hotspot will not be formatted. No color or underlining will appear.

"contextstring" field:
Enter the context string associated with the topic in the help file within the build that you are jumping to.

Examples:
Tires are [JUMPSOURCE NORMAL Tires]round[ENDJUMP] wheels.
[JS NONE Period]Periodic[EJ] sales dropped in the third quarter.
The [JS helpwin NOCOLOR Contents]Help Magician[EJ] is easy and fun.

[JUMPMIDTOPIC secwin fmtcmd contextstring] or [JMT secwin fmtcmd contextstring]
The JUMPMIDTOPIC and ENDJUMP commands are used together in pairs to mark text as a hypertext jump to a specified point in another topic. That point is defined by setting a MIDTOPIC command in the destination topic at the position you want to jump to. Place the JUMPMIDTOPIC command at the beginning of a text segment that you want to be a jump hotspot to that position. The following fields are mandatory except for the secwin field, which may be omitted:

"secwin" field:
Enter a secondary window name in the "secwin" field to display the topic in or leave it blank to display the topic in the main window. The secondary window must be defined in the project options using the method described above in the "How to Pass Project Options" section.

"fmtcmd" field:
To format the way the hotspot will appear in the help file, enter one of the following options in the "fmtcmd" field or leave it blank to default to NORMAL mode:

NORMAL Display hotspot with color and underlining.
NOCOLOR Hotspot will appear with underline only. No color will be displayed.
NONE Hotspot will not be formatted. No color or underlining will appear.

"contextstring" field:
Enter the context string that entered in the CONTEXTSTRING command in the topic that you want to jump to.

Examples:
The solenoid is a [JUMPMIDTOPIC NORMAL Electrical Parts] electromechanical [ENDJUMP] device.
[JMT bluewin NONE Sale Types]Periodic[EJ] sales dropped in the third quarter.
Using the [JMT NOCOLOR Main Topic]Help Magician[EJ] is easy and fun.

[ENDJUMP] or [EJ]
Used with the JUMPTOPIC, JUMPMACRO, JUMPFILE, JUMPSOURCE and JUMPMIDTOPIC command to mark text as a jump hotspot. Place this command at the end of a text segment you want to become a jump.

Examples:
The editor keys may be used to [JUMPTOPIC NONE Editor]edit[ENDJUMP] text in the window.
By selecting the [JT redwin NORMAL Keys]edit key[EJ], text can be formatted by the user.

[POPUPTOPIC fmtcmd title] or [PT fmtcmd title]
The POPUPTOPIC and ENDPOPUP commands are used together in pairs to mark text as a hypertext popup hotspot. Place the POPUPTOPIC command at the beginning of a text segment that you want to be a popup hotspot.

"fmtcmd" field:
To format the way the hotspot will appear in the help file, enter one of the following options in the "fmtcmd" field:

NORMAL Display hotspot with color and underlining.
NOCOLOR Hotspot will appear with underline only. No color will be displayed.
NONE Hotspot will not be formatted. No color or underlining will appear.

"title" field:
Enter the title of the topic in the "title" field that will be displayed when the hotspot is selected by the user.

Examples:
The solenoid is a [POPUPTOPIC NORMAL Electrical Parts] electromechanical [ENDPOPUP] device.
[PT NONE Sale Types]Periodic[EP] sales dropped in the third quarter.

[POPUPFILE fmtcmd filename contextstring] or [PF fmtcmd filename contextstring]
The POPUPFILE and ENDPOPUP commands are used together in pairs to mark text as a hypertext popup to another help file. Place the POPUPFILE command at the beginning of a text segment that you want to be a popup hotspot to another help file.

"fmtcmd" field:
To format the way the hotspot will appear in the help file, enter one of the following options in the "fmtcmd" field:

NORMAL Display hotspot with color and underlining.
NOCOLOR Hotspot will appear with underline only. No color will be displayed.
NONE Hotspot will not be formatted. No color or underlining will appear.

"filename" field:
Enter the DOS filename of the help file that you want the hotspot to popup.

We advise not entering a path with the filename to avoid forcing end users of the help system to have specific paths on their systems. For example if you specified a path such a D:\HELP\MYHELP.HLP, and the user did not have a "D:" drive on their system, WinHelp would report an error and the user would not be able to correct the situation. Try to avoid using hard coded paths in help systems.

"contextstring" field:
Enter the context string associated with the topic in the help file that you want to popup.

Examples:
The solenoid is a [POPUPFILE NORMAL Solenoids]electromechanical[ENDPOPUP] device.
[PF NONE Period]Periodic[EP] sales dropped in the third quarter.
Using the [PF NOCOLOR Contents]Help Magician[EP] is easy and fun.

[POPUPSOURCE secwin fmtcmd contextstring] or [PS secwin fmtcmd contextstring]
The POPUPSOURCE and ENDPOPUP commands are used together in pairs to mark text as a hypertext popup to another source file within a multiple file build. Place the POPUPSOURCE command at the beginning of a text segment that you want to be a popup hotspot to another topic in a multiple file build.

"fmtcmd" field:
To format the way the hotspot will appear in the help file, enter one of the following options in the "fmtcmd" field:

NORMAL Display hotspot with color and underlining.
NOCOLOR Hotspot will appear with underline only. No color will be displayed.
NONE Hotspot will not be formatted. No color or underlining will appear.

"contextstring" field:
Enter the context string associated with the topic in the help file within the build that you are popuping to.

Examples:
The solenoid is a [POPUPSOURCE NORMAL Electrical_Parts] electromechanical [ENDPOPUP] device.
[PS bluewin NONE SaleTypes]Periodic[EP] sales dropped in the third quarter.
Using the [PS NOCOLOR Main_Topic]Help Magician[EP] is easy and fun.

[ENDPOPUP] or [EP]
Used with the POPUPTOPIC, POPUPFILE and POPUPSOURCE command to mark text as a popup hotspot. Place this command at the end of a text segment you want to become a popup.

Examples:
The editor keys are used to [POPUPTOPIC NORMAL Editor]edit[ENDPOPUP] text in the window.
By selecting the [PT redwin NOCOLOR Keys]edit key[EP], text can be formatted by the user.

[CONTEXTSTRING contextstring] or [CS contextstring]
Place the CONTEXSTRING command at the top of a topic to assign it a context string. This is useful when using popups or jumps to other sources. Do not use this command for mid topic jumps, rather use the MIDTOPIC command to assign a context string to mark a specific position in a topic.

[TOPICTITLE title] or [TT title]
Place the TOPICTITLE command at the top of a topic to assign it a title. This command is ignored by the converter if you are generating a table of contents and the topic has a title generated for it automatically. Use this command to specify a title for a topic that otherwise would not have one generated from styles or needs a specific title to go with the keywords and see in the WinHelp search dialog.

[MIDTOPIC contextstring] or [MT contextstring]
Place the MIDTOPIC command at a desired position in a topic to assign it a mid-topic context string. This is useful when using the mid-topic jump command, JUMPMIDTOPIC.

Example:
....
[MT Glossary_J]J
James
Jackson
Jones

[MT Glossary_K]K
Karlan
Kraft
...

[GLOSSARYSTART] or [GS]
The GLOSSARYSTART and GLOSSARYEND commands are used together in pairs to mark text as an entry into a glossary topic. Place the GLOSSARYSTART command at the beginning of a text segment that you want to be an entry into the glossary. The text will appear in the glossary exactly as it appears in the document. No hotspot will be generated for the entry.

[GLOSSARYEND] or [GE]
Used with the GLOSSARYSTART commands to mark text as an entry into a glossary topic. Place this command at the end of a text segment you want to become an entry into the glossary.

Examples:
[GS]Anchors are devices that help prevent a vessel from drifting.[GE]

Turn on your [GS2 tubes]CRT - Cathode Ray Tube[GE] and wait for it to come on.

[GLOSSARYSTART1 title] or [GS1 title]
The GLOSSARYSTART1 and GLOSSARYEND commands are used to mark a word or phrase in your document that you want to appear in the glossary as a popup to another topic. Place the GLOSSARYSTART1 command at the beginning of a word or phrase that you want to become a popup in the glossary. Enter the title of the topic that you wish to popup from the glossary.

Example:
John used a [GS1 Spade Definition]spade[GE] to dig up the garden.

[GLOSSARYSTART2 title] or [GS2 title]
The GLOSSARYSTART2 and GLOSSARYEND commands are used to mark a word or phrase in your document that you want to appear in the glossary as a jump to another topic. Place the GLOSSARYSTART2 command at the beginning of a word or phrase that you want to become a jump in the glossary. Enter the title of the topic that you wish to "jump to".

Example:
Carla used a [GS2 Back Hoe Definition]back hoe[GE] to dig a hole in the road.

[GLOSSARYSTART3 definition] or [GS3 definition]
The GLOSSARYSTART3 and GLOSSARYEND commands are used to mark a word or phrase in your document that you want to appear in the glossary as a popup to a topic that will be created containing the definition your entered in the command. Place the GLOSSARYSTART3 command at the beginning of a word or phrase that you want to become a popup in the glossary. Enter the definition text that you wish to popup from the glossary. A topic will be created containing the definition you entered.

Examples:
The user should enter his [GS3 employee identification number]EID[GE] in the box.
Place the flat portion of the [GS3 A scanner used by technicians]HandiScan[GE] on the unit to be tested.

The glossary topic will have a picture of the letters A through Z that are actually mid-topic jumps to respective areas further down the page. The page consists of 26 areas each marked with a letter of the alphabet. As the Manual To Help converter finds a piece of text marked with glossary commands, it will make a copy of the text and place it in the appropriate area in the glossary topic. The converter will use the first letter of the text to determine where to place the text in the glossary. If GS1, GS2 or GS3 are used, then the converter will make the appropriate hotspot on the text. A topic will be created in the case of GS3 containing the user entered definition text.

- Use the GS and GE commands when you have plain text to be included in the glossary. No hotspots will be generated for the text.

- Use the GS1 and GE commands when you have a word or phrase that you want included in the glossary as a popup to an existing topic.

- Use the GS2 and GE commands when you have a word or phrase that you want included in the glossary as a jump to an existing topic.

- Use the GS3 and GE commands when you have a word or phrase that you want included in the glossary as a popup to another topic that will be created for you with a definition you supply.

Tools

The Tools menu for Win95 is shown here.

Click on the desired area of the Tools Menu for specific Help

Tools

When you select one of the tools from the Tools Menu, the particular program is run independently of the Help Magician. This allows you to remain in the Help Magician and run one or more tools at the same time.

Renumber Context #'s (Numbers)

This function will renumber all of the in the current help source file. If Context Numbers already exist, you will be warned that they will be overwritten. A dialog will ask for the base value from which to start numbering. Context Numbers will begin at the base value and continue sequentially to the end of the file.

SHED.EXE

SHED is an acronym for **Segmented Hypergraphics Editor.** SHED is used to create hotspots within bitmaps. Refer to SED.EXE for more information.

Paintbrush

Paintbrush is a popular Windows application that allows you to create and edit bitmaps for use in the Help Magician and SHED.

Bitmap Magician

The Bitmap Magician is a program that allows you to modify font families to you're own specifications. With the Bitmap Magician you can convert special and decorative fonts to bitmaps for use within the help system. You can create customized bullets and superscripted/subscripted text. The Bitmap Magician is installed with the Help Magician and an icon is included in the Help Magician Pro program group.

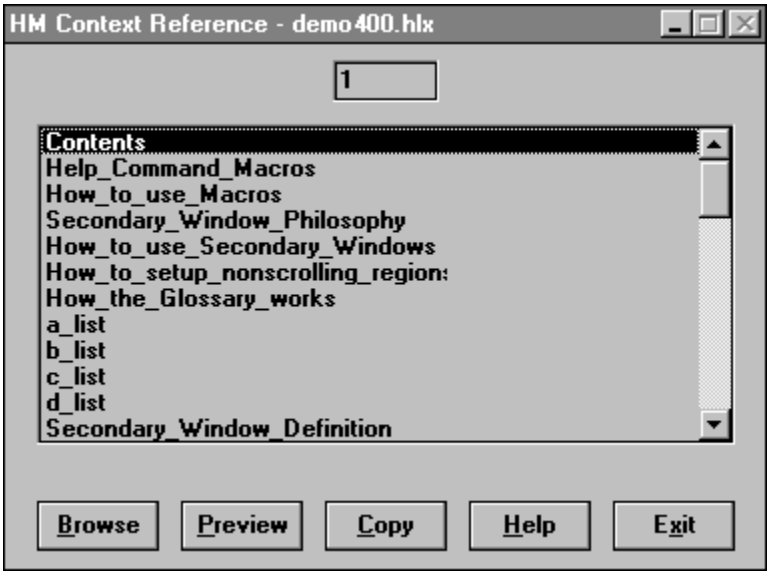
Sound Recorder

This sub menu item will call the Windows sound recorder which you can use to record wave (.WAV) files to be played in your help system.

Context Reference

Selecting the Context Reference sub menu will call the HM Context Reference executable. This utility can also be called from its icon in the Help Magician Pro program group.

The purpose of the Context Reference utility is to view the Context Strings in any HLX file on disk. This is particularly useful when working in SHED.EXE as a reference to the Context Strings to be linked to Jumps and Popups to be entered into the Attributes dialog.



Browse

When the Context Reference utility is first called, a file selection dialog will pop up from which you will select the HLX file to read. At any other time, press the Browse button to bring up the file selection dialog.

The selected HLX file must have been saved with version 3.0 or later of the Help Magician. If it was not, a message will inform you that the file was not indexed.

Preview

Press the Preview button to view the topic that is highlighted in the list box.

Copy

The Copy button will copy the Context String to the Clipboard for pasting into the Attributes dialog in SHED.EXE.

Help

On-line help for the Context Reference utility.

Exit

Close the Context Reference utility.

User Defined...

Select the User Defined sub menu to add your own tools to the Tools menu. A file selection dialog will pop up from which you will select the executable (.EXE) to add to the menu. When the file has been added to the menu, you will be able to invoke the program by clicking on its sub menu.

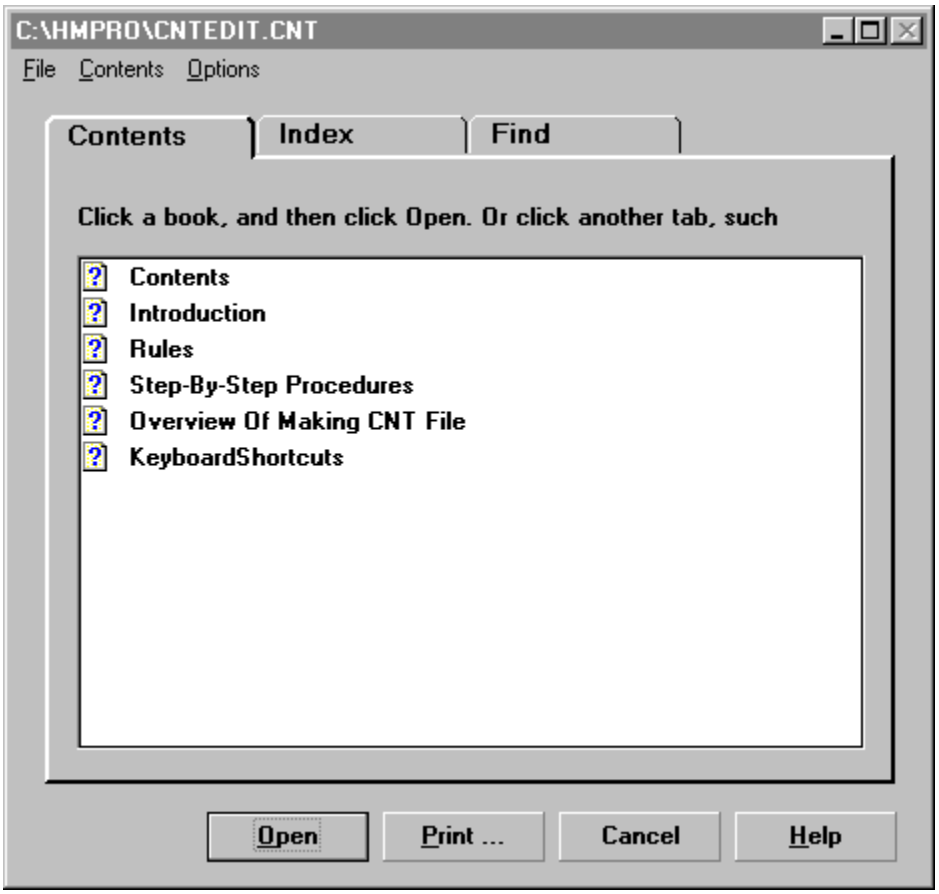
When prompted, enter the window title bar text, as it will appear when the program is executed. This is the text at the top of the window, usually the program name, sometimes followed by the filename of the file currently being worked on. For example, Word for Windows title bar contains "Microsoft Word - FILENAME.DOC". In this instance, "Microsoft Word" should be entered as the title bar text because it is the only consistent part of the text.

The title bar text is used to look for an instance of the program in the task list to avoid invoking another instance of the program. If found, the current instance of the program will be activated.

To delete a user defined tool from the Tools menu, press and hold the Ctrl key while clicking on the sub menu.

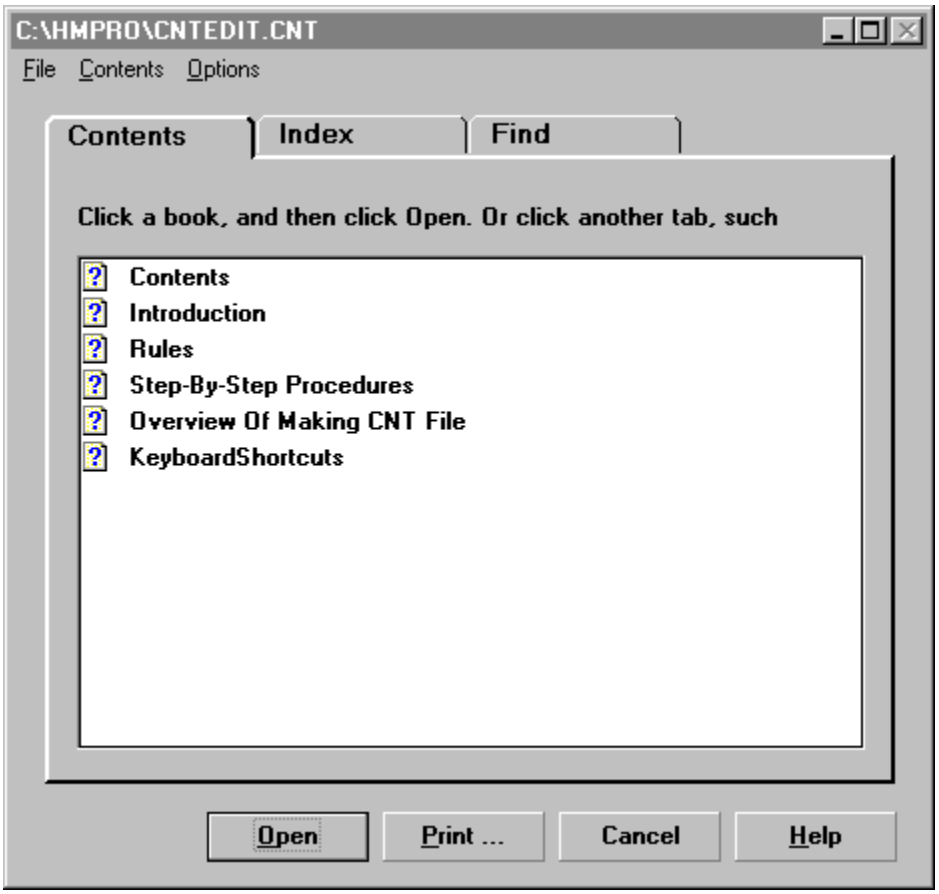
Help Topics Browser

Help Magician Pro 95 emulates the WinHelp 95 Help Topics Browser Window that contains the Contents Tab, Index Tab, and the Find Tab. To show the Help Topics browser window, simply click on the "Help Topics" button on the WinHelp Button Bar just above the Help Magician's editor. The WinHelp 95 Help Topics Browser window containing the Contents Tab, Index Tab, and Find Tab, will appear. Note that if you are in WinHelp 3.x emulation, then the "Contents" button will take you to the contents page of your help file and not show the special WinHelp 95 Help Topics browser window.



Click on the desired button of the Help Topics Browser bitmap for specific Help

WinHelp 95 Contents Tab



During development of your help file, Help Magician Pro 95 can display the Contents File (.CNT) in a special Help Topics window the same way WinHelp 95 does. The CNT file is a companion file for Windows 95 help that defines how topics will appear in this Help Topics browser window (Contents Tab portion). Books containing topics are shown in a hierarchical tree or outline format.

The CNT file can be created with our WinHelp 95 Contents Editor utility. This utility allows you to create books, and drag-and-drop topics into books, as well as visually rearrange your contents file. The WinHelp 95 Contents Editor utility can be run at any time from the Tools...WinHelp 95 Contents Editor menu.

The CNT file should exist in the same directory as the Help Magician source (HLX) files.

If no CNT file exists when the Contents Tab is shown, you will have the option of allowing Help Magician Pro to create a CNT file that simply lists all the topics in the order they appear in the Help Magician Pro editor, or running the WinHelp 95 Contents Editor for more advanced CNT authoring. If you let Help Magician create the simple CNT file, without using the WinHelp 95 Contents Editor, the CNT file will be created and the Help Topics Browser window will appear.

Tip: It is recommended that you develop the CNT file after you've outlined the structure of your help file and created the topic pages. The WinHelp 95 Contents Editor gives you the option of importing existing topics from a Help Magician single source file (.hlx) or a project source file (.hmp)- which contain multiple .hlx files.

Navigating using the Contents Tab

Once the Contents Tab is displayed, you can click on books to open them and show other books or topics. To show a topic (listed in the Contents Tab) in the Help Magician's editor window, double-click on a topic or select the topic and click on the Display button. To print a topic, select the topic and click on the Print button. The topic will be displayed or printed provided that the topic exists in the currently opened source file. The Help Topics browser window will stay open and on top until you close it.

Note: Any topic printed from Help Magician may not look exactly like the way the topic would be printed from WinHelp 95, because of some limitations within the print handler in one of the 3rd party controls that Help Magician uses for printing.

See Also

- Find Tab
- Index Tab

WinHelp 95 Index Tab

The Index Tab replaces the Search dialog box in WinHelp 3.1. It provides a comprehensive index to the topics in one or more Help files. The index contains a list of keywords that you specify for each topic. A user can scroll to the desired keyword either by clicking the scroll arrows or by typing the word. If more than one topic is associated with the keyword, WinHelp (and Help Magician simulated Help Topics dialog) displays the Topics Found dialog box that lists the associated topics. WinHelp 95 and Help Magician Pro 95 now supports second level index entries.

Tip: You can use Help Magician Pro's powerful Keyword Management feature to create keywords for the Index Tab. Access it from the Links...Keyword Management menu. For more information, see the section on Keyword Management.

Using Help Magician Pro's Simulated Index Tab

The Index Tab in Help Magician works the same way it does in WinHelp 95. The keywords will be read from Help Magician's internal keyword database and displayed in the keyword list. Any related keywords (subentries) will be displayed indented under a listed keyword. You can search the keyword list by typing the first few letters of the word in the textbox at the top of the Index Tab. As you type, the keyword with the closest match will be highlighted in the keyword list. Click on Display to view the topics that are associated with the selected keyword. From that topic list, you can select a topic and then click on Display to display the topic in the Help Magician editor (provided it is a topic that is already part of your opened source file).

See Also

- Contents Tab
- Find Tab

WinHelp 95 Find Tab

Help Magician Pro simulates most of the functionality of the Find Tab in the Help Topics window in Windows 95. When you click on the Find Tab, the Find Tab window will appear.

Using Help Magician Pro's Find Tab

Follow the instructions on the Find Tab form. Start by entering the word(s) you want to search for in the textbox at the top. You can clear your typing by pressing Clear. Each time you enter text in the textbox, your entry will be added to the list. To select one of your past entries, click on the down arrow and select an entry.

Next click on the Find Now button. This may take a while depending on the size of your currently opened help file. Once Help Magician finishes scanning the text on all your help pages, a list of topics where the text was found will appear in the listbox. To go to that topic, simply double click on the topic, or highlight the topic and click on the Display button. You can print a selected topic by clicking on the Print button.

Help Magician supports some of the methods that WinHelp 95 uses to do full text search. You can search for a word, phrase, partial word, or group of words appearing in a topic. You can even tell Help Magician to treat lowercase and uppercase letters the same.

See Also

- Contents Tab
- Index Tab

On Line Help

Click on the desired area of the Help Menu for specific Help

Help on Help

Calls the Microsoft Help on Help file with instructions on how to use Windows help files.

Contents

Clicking on the Contents sub menu will display the Help Magician's own help system, at the table of Contents page. From this page, you can get more information on any topic by clicking on the desired word or phrase. You can also select Search to search for help on a topic by keyword.

Search

Selecting Search from the Help Menu will bring up the help system in search mode.

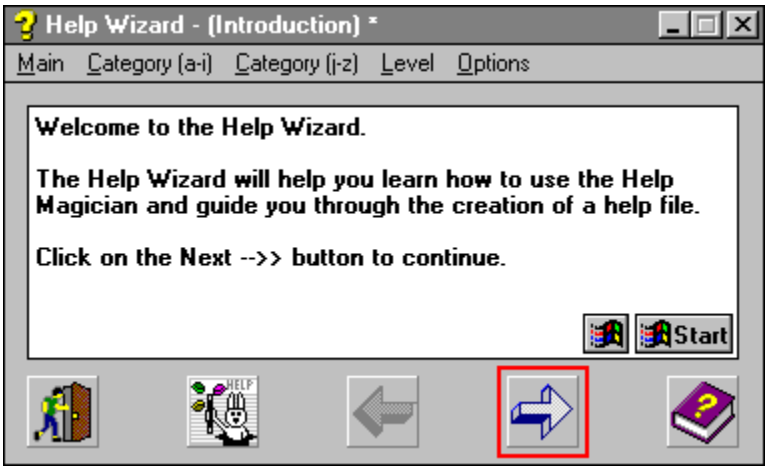
Keyboard

Selecting Keyboard from the Help Menu will bring up the help system with keyboard definitions as the current topic.

About

Clicking on the About sub menu will display the application logo, the Windows operating mode, the amount of free memory available to Windows, and the version number of the Help Magician.

Help Wizard



The **Help Wizard** will help familiarize you with help systems in general, it will help familiarize you with the **Help Magician**, and it will assist you with almost any of the functions used to create a help system.

Categories

If you are new to help systems or to the **Help Magician**, the **Introduction**, the **Tutorial**, and the **New Help File** categories will be of particular interest.

The **Tutorial** walks you through the creation of a simple help file and does all of the work for you. You can quickly learn some of the basics while the help system is being created.

The **New Help File** category will step you through setting up all of the necessary components of a help system.

After the categories described above, there are two basic types of categories, **Tours** and **Assist** categories.

The **Tour** categories display the selected form and explain all of its controls.

The **Assist** categories actually walk you through the process of the selected feature while you fill in the information for your own help system.

The **Help Wizard** is interactive and user friendly and requires little explanation. When you begin to use it, it will become apparent that no further documentation is necessary to complete the selected function.

Level

Novice vs Expert

In **Novice** mode, the **Help Wizard** displays more explanatory information than in **Expert** mode. If you are already familiar with help systems, the **Expert** mode is probably more appropriate for you. If you are new to help systems, the **Novice** mode will provide more information and guidance.

Options

Show Wizard at Startup

By default, the **Help Wizard** pops up when the **Help Magician** is started. You can turn this feature off by unchecking the **Show Wizard at Startup** sub menu, available from the **Options** menu.

Sound

A sound is generated when the **Help Wizard** types for you. You can turn off this feature by unchecking the **Sound** sub menu.

Quick Help

A small window is displayed when the cursor is moved over the five toolbar buttons with a description of the functionality of the button. You can turn off this feature by unchecking the **Quick Help** sub menu.

Cursor Speed

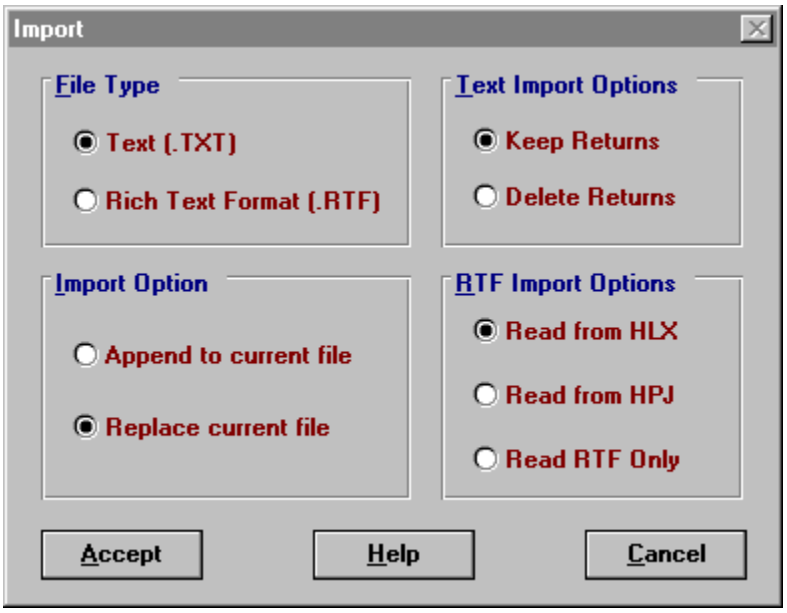
As the **Help Wizard** demonstrates a feature, it moves the mouse cursor to the location of the control being discussed. There are many factors involved in the determination of the speed of the cursor in addition to personal preference.

To adjust the speed, select the **Options** menu, **Cursor Speed** sub menu and click on the desired speed, from one (slowest) to six (fastest).

Windows Version

The **Help Wizard** will display information and provide assistance based on the version of Windows on your system and the **Compiler Version** as selected on the **Compiler Options** form.

Importing Text Files



Import Text

ASCII text can be imported in the Help Magician. The text can be written out from a word processor but some formatting is required before it can be read into the Help Magician. Select the Import sub menu from the file menu. This will bring up the Import form. On the ***Import*** form, click on the ***Text (.TXT)*** radio button.

Import Option

Select whether the imported text will replace the current file or be appended to the end of the file.

DDE

Note that text can also be "pasted" into the editor with DDE (Dynamic Data Exchange). While the new Windows standard keystrokes for DDE are Ctrl X, Ctrl C, and Ctrl V for Cut, Copy and Paste, the Help Magician uses Shift Del, Ctrl Ins, and Shift Ins respectively. See Environment Options for alternate keyboard configuration for Cut, Copy, and Paste.

Page Breaks

When a word processor file is exported as text, there are no hard page breaks. The Help Magician needs hard page breaks to determine where on help topic ends and another begins.

If you write out an ASCII file from the Help Magician, using Save As Text, each page begins with the heading, Title:. The Help Magician considers this to be a page break and will begin a new topic. Optionally, you can enter the sequence, \Page, or a literal character 12, into your ASCII file as a page delimiter. If no page break is included in the ASCII file, the Help Magician will try to place all of the text into the first page, probably resulting in an "out of string space" error.

Formatting

Other formatting can be used to facilitate importing ASCII files into the Help Magician. If an ASCII file is formatted in the same manner as the Help Magician's Save As Text command, the information following the headers (\Title:, \Keywords:, \Multiple \Keywords:, \Context Number:, and \Browse:) will be written to their appropriate positions in the help file database as if they had been entered from the Help Magician.

The headers must be at the left margin with no characters, spaces, or tabs before the header.

TUTOR.TXT

The Tutorial file, TUTOR.HLX, was Saved As Text as TUTOR.TXT. The following example listing of the third page, in bold type, shows the exact format of the text file:

\Title: Cutting Text
\Keywords: Clipboard, Cut, Edit
\Multiple Keywords:
\Context Number: 0
\Browse: Edit: Cutting Text, Copying Text, Pasting Text

Cutting Text

Select the text to be cut by holding the left mouse button and dragging over the text. Then select Cut from the Edit Menu or press Shift Del. The text is cut to the Windows Clipboard.

Conversions

Most word processors can save a file in ASCII, or text only format. To convert a word processor document to a Help Magician file, modify a version of the document, save the document as text, and import it into the Help Magician with the Import Text function available from the File Menu (or Project Management's Import button). Be sure to include one of the page break delimiters (described earlier in Page Breaks) in the text file. You can include any of the other headers shown in the TUTOR.TXT file above or import the text file and enter the information in the Help Magician's editor.

Keep Returns

If possible, don't word wrap the lines in the text file with hard returns. This will cause lines of fixed length in the Windows help system, defeating the automatic word wrapping done by WINHELP.EXE. The Keep Returns option will import the text file exactly as it was written.

If you don't want certain paragraphs to word wrap, select the ***Define Styles*** sub menu from the ***Format*** menu and define a style for this purpose. Click on the alignment tab on the ***Paragraph*** form and check the ***Force word wrap off*** check box.

Delete Returns

If hard returns are present, select the Delete Returns option from the Import Text Menu. This will remove all hard returns except those at paragraphs, or double spaced lines.

See Also
Environment Options

Importing RTF Files

Overview

The Help Magician will read RTF files generated by Word for Windows, Ami' Pro or Word Perfect for Windows version 6.0a. When we mention a "word processor" we mean specifically Word for Windows, Ami Pro or Word Perfect for Windows version 6.0a.

Backup

It is important that you backup up your RTF file before reading it into the Help Magician and writing it out again. Because of the filtering that is necessary to import the RTF file into the Help Magician, the file will be changed.

Exporting

To export an RTF from Word for Windows, select "Save As..." from the File Menu. Click on the drop down box labeled **"Save File as Type:"** and select **"Rich Text Format (*.rtf)".** Change the file extension to .RTF in the **"File Name:"** box. This distinguishes the file from a Word for Windows .DOC file and makes it easier to import into the Help Magician because the default extension expected is .RTF. Follow the same type of procedures for Ami Pro and Word Perfect for Windows.

Importing

To import an RTF file into the Help Magician, select Import from the File Menu. The file extension defaults to .RTF. Under "RTF Import Options", select a source of the help project options. You may choose a to read them from a WinHelp project file (.HPJ), a Help Magician file (.HLX) or just read the RTF file only.

The default path for the Help Magician will not be changed to the path from which the RTF file was imported. When you import an RTF file and save the Help Magician's file, you can select another path as the 'ROOT' path for the help build so that the original RTF file will not be overwritten when you build the help system.

The Help Magician will read RTF files generated by Word for Windows, Lotus' Ami Pro, or WordPerfect for Windows. RTF files written by the Help Magician can be read into Word for Windows, edited, and read back into the Help Magician.

Merging RTF Files

If you have several RTF files that you would like to import into the Help Magician as a single source file, merge the files in Word for Windows before importing. The files can be merged as .DOC files or RTF files. Merge the files as follows:

1. Open the first file in Word for Windows.
2. Place the cursor at the end of the file.
3. Select the **Insert** menu and the **File** sub menu. Select the file to be merged (select the file type if the file is not in Word format) and click on OK.
4. Repeat for all files to be merged.

RTF Import Options

Much of the information that the Help Magician maintains for a help system is not contained in the RTF file written for the help compiler. These "Project Options" include the Compiler Options, the Appearance Options, the bitmap directories, macros, and window definitions.

If an RTF file was exported from the Help Magician, this information can be retrieved from either the project file (.HPJ) or the Help Magician source file (.HLX).

Read from HLX

Use this option to read the "Project Options" from the HLX file from which the RTF file was written.

Read from HPJ

Use this option to read the "Project Options" from the HPJ file for the help system.

Read RTF Only

Use this option if this is a first time import and neither an HLX or a .HPJ file exist or to ignore the "Project Options".

Default Path

The default path for the Help Magician will not be changed to the path from which the RTF file was imported. When you import an RTF file and save the Help Magician's file, you should select another path as the 'ROOT' path for the help build so that the original RTF file will not be overwritten when you build the help system.

Filter

The Help Magician supports most of the RTF commands supported by the Windows help compiler. Commands not supported will be filtered out and written to a report file consisting of the same name as the RTF file, with a .RPT extension, for your reference. Any errors that occur during the importing will also be written to this file.

Unsupported Commands

Most of the commands listed in the report file as Unsupported Commands are filtered out because they are not necessary to the help compiler. The Help Magician supports almost all of the RTF commands that are supported by the help compiler.

Origin

There are some differences in the way the RTF file is handled depending on the origin of the help file. In older versions of The Help Magician, additional fields were written into the footnotes, that are not used by the help compiler. The Help Magician uses this information when the RTF file is read back in. The following paragraphs describe the differences.

Word Processor Files

Help files originally written in Word for Windows, Ami Pro or Word Perfect for Windows will be read in and the help file specific data, such as Titles, Browse Sequences, Context Strings, etc. will be stored in the Help Magician's database.

Context #'s

Since Context Numbers are not actually part of the help file, but are included in the project file or an include file, help files originating in a word processor will not contain Context Numbers. If the help file originated in word processor, you will have to enter the Context Numbers in the Help Magician.

If the help file originated in an older version of the Help Magician, the Context Numbers would have been written to an additional footnote field and will be read in and properly stored by the Help Magician.

Context $'s

The Help Magician generates new Context Strings for every topic. If you enter Context Strings in the context string footnote field, in a word processor, they will be used by the Help Magician instead of constructing new Context Strings.

Titles

The Help Magician requires that every page have a title. When pages are read in from an RTF file, if a page does not have a title, a title is created for the page. The temporary title is constructed from a unique

ID assigned to the help file when is is created.

Font Colors

If the file originated in another word processor other than Word for Windows, colors will be re-mapped to Word for Windows colors.

Browse

If Browse Sequences originated in a word processor, the sequence numbers don't necessarily have to be sequential, they can be incremented by any value. When imported into the Help Magician, they will be renumbered so that they are sequential. If the help file originated in an older version of the Help Magician, the sequence numbers will already be sequential.

Version 3.1

While importing an RTF file, if the Help Magician detects codes specific to features supported only by Windows 3.1, the Help compiler version, stored in the Compiler Options, will be set to version 3.1.

Likewise, if the Help Magician detects codes specific to features supported only by Windows 95, the Help compiler version will be set to version 4.0.

See Also
Manual to Help Conversion

RTF Technical Specifications

RTF Format

The Help Magician's RTF Reader supports both Word for Windows, Ami Pro and Word Perfect for Windows (version 6.0a) RTF formats at this time. When we mention "a word processor" we mean specifically Word for Windows, Ami Pro or Word Perfect for Windows version 6.0a. If you are a Word Perfect for Windows user we strongly recommend upgrading to version 6.0a because of the greatly improved RTF converter. We cannot guarantee that the Help Magician will properly import RTF files created in earlier versions of Word Perfect for Windows.

The Compiler Options Menu offers an option to select the version of Word for Windows RTF file written by the Help Magician. The option defaults to version 2.0, however the user may choose between versions 1.0, 2.0, 6.0 and 7.0.

Select version 2.0 to write RTF files compatible with Ami Pro and Word Perfect for Windows version 6.0a.

Version 4.0

The RTF Reader performs both WinHelp 3.1 and WinHelp 4.0 syntax checking.

Warnings

There are potential translation problems converting back and forth from Word for Windows versions 1.0, 2.0 and 6.0 RTF formats. The RTF formats sometimes get garbled and may become unreadable by the RTF Reader.

Also, Word Perfect RTF file format is drastically different than Word for Windows and Ami Pro. Consequently these word processors cannot reliably read each other's RTF files.

Word Perfect for Windows v6.0 and v6.0a has a bug where bullets are not written to their RTF files.

Errors

RTF Reader errors, unsupported features and unknown or unsupported RTF commands will be written to a report file using the original filename with ".RPT" extension.

Unsupported Commands

Most of the commands listed in the report file as Unsupported Commands are filtered out because they are not necessary to the help compiler. The Help Magician supports almost all RTF commands that are supported by the help compiler.

Critical errors will be displayed on screen as well as written to the report file.

Any invalid references in the original RTF by hot links (jumps or popups) to non-existent topics and/or context strings will be ignored to the extent that an unresolved hot link will be created..

Browse

Browse sequences that have been formatted as text instead of numbers will be converted to numbered sequences and the original text will be lost. **Please format browse sequences numerically if initial development is done in a word processor.** Also, numerically formatted browse sequences will be renumbered sequentially and the original numbers will **not** be saved.

Browse sequences created in Word for Windows with no group title, i.e. "null list browse sequences", will be assigned to a group entitled "Null" by the RTF Reader.

Fonts

The fonts used in Word for Windows RTF will be converted to Help Magician Styles and put in a table. If a table already exists for the project, it will be used and any additional font styles discovered will be added to the table.

Macros

The macros used in a word processor RTF will be converted to Help Magician Macro Definitions and put into a table. If a table already exists for the project, it will be used and any additional macro definitions discovered will be added to the table.

ROOT

Please keep all associated files for a particular project in the 'ROOT' directory, with the exception of bitmaps, which may be stored in the 'BMROOT' directory. This includes font style files, bitmaps (if possible) and all the Help Magician created files (HLX, etc.).

Formatting

Please use one continuous underline, double underline or strikeout formatting for popups and jumps, respectively, in Word for Windows. Breaking up the formatting for hot links may cause problems in translation from Word for Windows versions 1.0, 2.0, 6.0 and 7.0 conversion process. Likewise, fonts should be treated in the same way.

Please do not format help text and footnotes ACROSS pages in a word processor.

Context $'s

The Help Magician handles Context Strings invisibly to the user. If Context Strings are created in the Help Magician, they are created from the topic title. However, in version 1.1 and later, when they are included in a footnote, they are read into the Help Magician and preserved.

Duplicate context string names will be adjusted by renaming one with a "_000" tagged onto the end of the string, where the actual number used refers to the particular help page the context string references.

Missing Context Strings will be constructed using the unique file ID created when the file is imported. Duplicates will be adjusted using the aforementioned procedure.

Titles

Missing help page titles will be constructed using the unique file ID created when the file is imported.

Hot Links

Improperly formatted hot links, i.e. jumps, popups, etc., will be ignored by the RTF Reader and will appear as they did in the original document.

Version

The RTF Reader will detect the appropriate version of the Help Compiler to be used for the file being read. It is up to the user to verify the Help Compiler version selected in Compiler Options.

Web Authoring

HTML Export Options

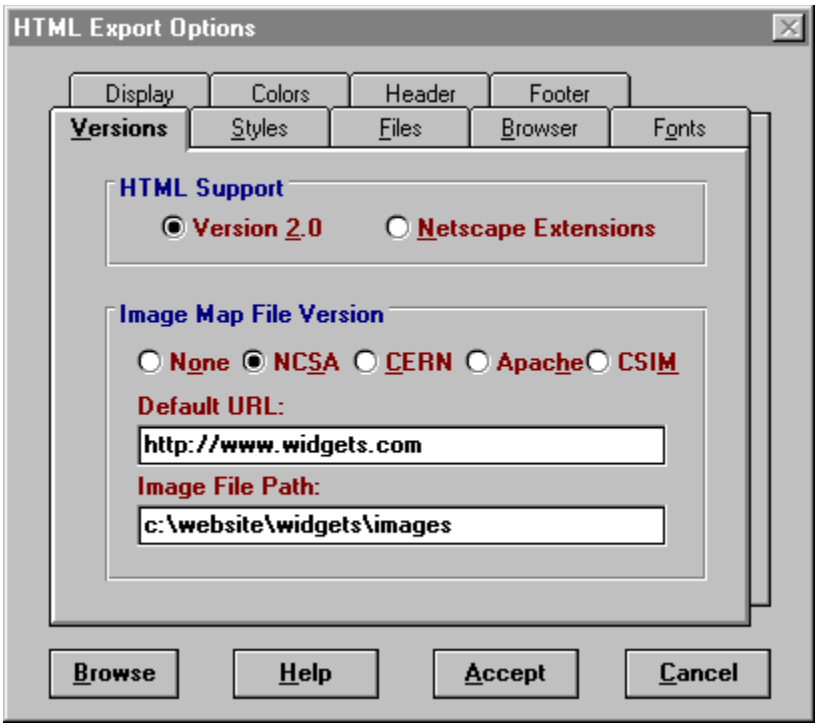
The Help Magician allows you to create pages that can be displayed on the Internet's World Wide Web (WWW). This is a particularly useful feature for Help authors who wish to display their Help files on the Internet. This can be accomplished by using the Help Magician to convert their Help system into an HTML format, which is the standard format for viewing information on the Internet by using one of the many Web browsers such as Netscape Navigator, Spry Mosaic, Microsoft Internet Explorer or Chameleon's Websurfer.

In order to display HTML pages (sometimes referred to as "home pages" or "web pages") on the Internet, most people have a company that provides Internet access, typically called an Internet Service Provider (ISP), to house their web pages for them. To do this you will need to find out what type of web server your ISP is running. This is important if you want to convert the SHED bitmaps in your Help system to image maps, which are dependent upon the type of web server they will used with. Certain web servers require specific formats for image map files. Some examples of web servers are CERN, Apache, NCSA, and WebSite.

Another consideration in creating web pages from your Help system is to determine what version of HTML you wish to support. When the Internet was first becoming popular, web browsers supported version 1.0 of HTML. Very few text formatting features were supported in version 1.0 so version 2.0 was released which supported more features. Version 2.0 became the standard for all browsers at that time and now version 3.2 is the popular version. Unfortunately (or fortunately depending on your view of the subject), Netscape Corporation decided to add features that were not supported in the 2.0 HTML specification. But that caused problems for people using browsers that didn't understand the new HTML codes in that they couldn't properly view some of the pages intended to be viewed by the Netscape browser. These extra features became known as the Netscape Extensions.

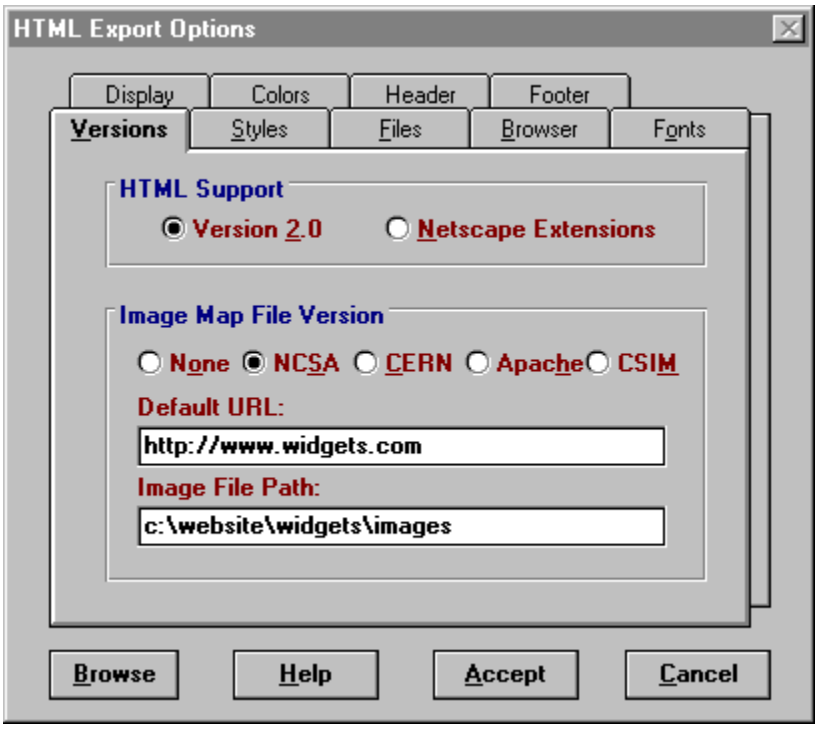
To solve this dilemma many web page authors create their pages using the HTML 2.0 specification so that the greatest cross section of web browsers can view their pages as they intended to be viewed. Most pages that were intended to viewed by Netscape browsers are labeled with "This page can be best viewed with Netscape". Microsoft's "Internet Explorer" has followed in the footsteps of Netscape's "Netscape Navigator" and supports all of the features of the Netscape HTML extensions. The Help Magician allows you to select which version of HTML you wish to use to create your web pages with.

Note: You may wish to use the built in HTML styles that are provided with the Help Magician. You must, however, include these styles before editing your help file.



Click on the desired tab of the HTML Export Options bitmap for specific Help

Versions



Versions

Sets the version level of the HTML code and compatibility of the image map file to a particular server.

HTML Support

As previously discussed, this options allows you to select which version of HTML that will be used to create web pages from your help file. Remember that not all web browsers support the Netscape extensions.

Image Map File Version

To have all SHED bitmaps in your Help system converted to mapped images in HTML pages, select the type of web server your web pages will be used with. You will need to contact your Internet Service Provider or your system administrator for this information. The choices are: NCSA, CERN, or Apache. If you don't know this information, you can try NCSA as a test.

The CSIM options stands for "Client Side Image Map" which means that the user's browser interprets the image map, not the web server. The Help Magician will place the map information in the HTML pages it creates and the user's browser will be required to display the image map properly. This feature is supported by most newer browsers and web servers.

Default URL:

Enter a valid URL such as "http:\\www.server.com\index.html" to be displayed when the user selects an area in the image that has not been mapped. This feature is optional.

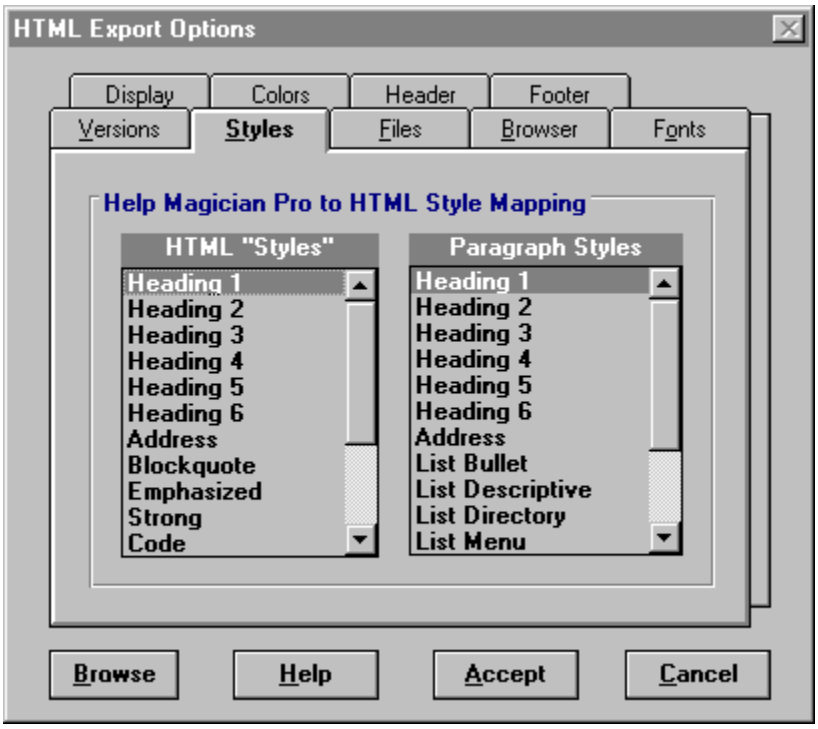
Image File Path:

Enter a valid path where the images will be stored on the server. This path will be added to all images so that the web server can locate them. This feature is optional if the images are stored with the HTML files.

See Also

- Browser Tab
- Colors Tab
- Display Tab
- Files Tab
- Fonts Tab
- Footer Tab
- Header Tab
- Styles Tab

Styles



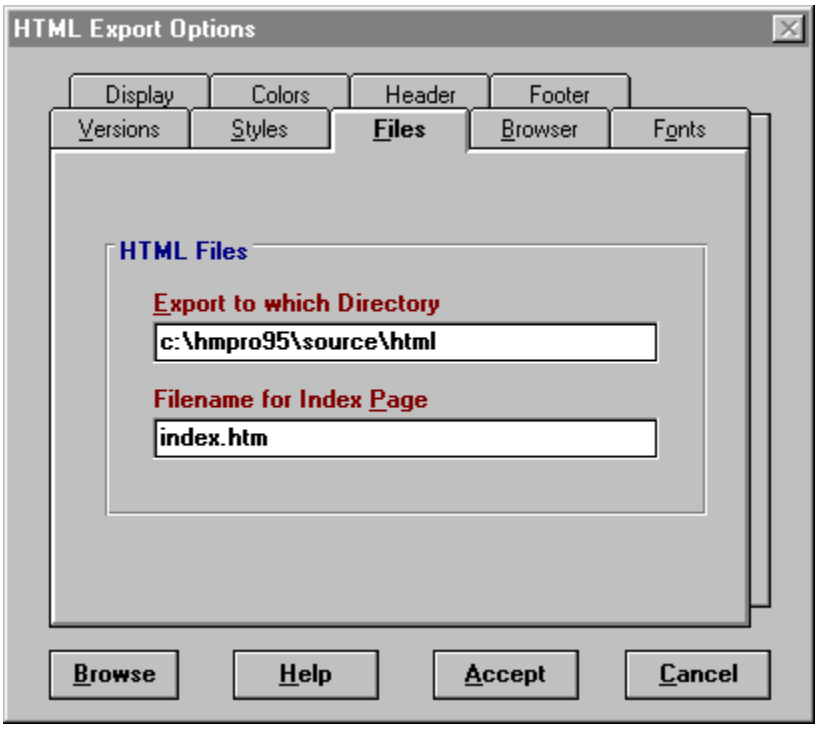
Styles

The HTML specification provides for six default styles and several ad hoc styles that you can take advantage of using Help Magician's style mapping utility. To map a style you have used in your help file to an HTML style, select the HTML style first and then the corresponding paragraph style on the right hand list box. Any paragraphs in your help file that are formatted using this style, will be converted to the HTML style. You will probably need to view the results in your browser to get comfortable with the way HTML styles are displayed versus help styles in WinHelp.

NOTE: Not all browsers display HTML pages in the same manner. On top of that, each browser can be reconfigured by the end user to display fonts and styles using their preferences. In other words, what you see in your browser might not and probably will not be what the end users see in their browsers. The rule of thumb is to use HTML styles and headers sparingly.

See Also
- Browser Tab
- Colors Tab
- Display Tab
- Files Tab
- Fonts Tab
- Footer Tab
- Header Tab
- Versions Tab

Files



HTML Files

Enter the path where you want the HTML files to be written. Use the Browse button to search for an existing path on your system.

Export to which Directory

Enter a valid path on your computer where you wish to store the HTML files generated from your help system. You may use the Browse button to search for a path on one of your drives.

Start The Help Magician marks the HTML files it writes to disk with a comment in the header section of each file. This is done as an added measure of protection to prevent the Help Magician from overwriting files from other sources. If the Help Magician attempts to overwrite an index file for instance, a warning message will be displayed if the file isn't recognized with the Help Magician comment mark.

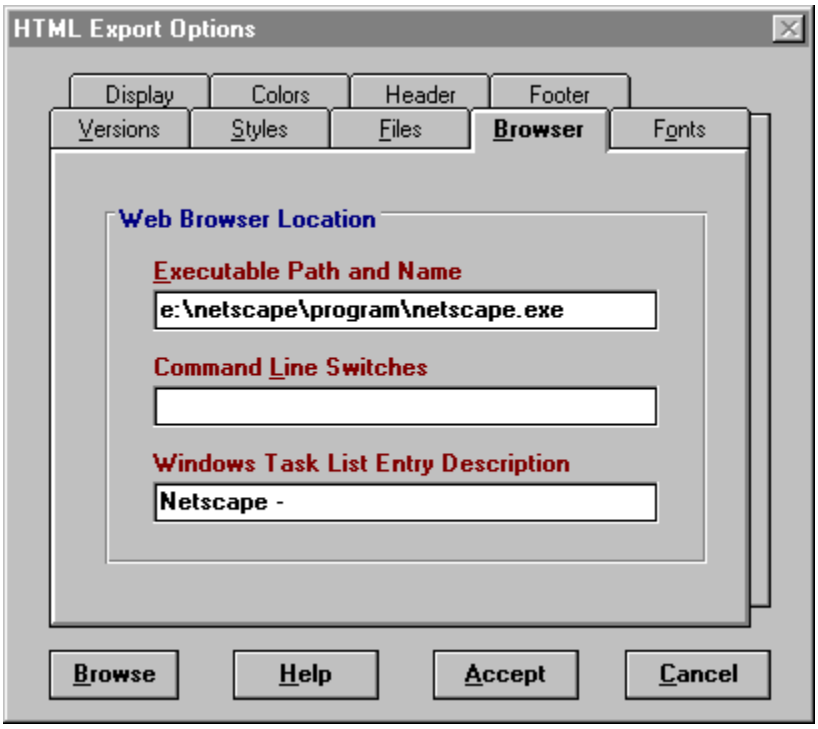
Filename for Index Page

Enter a valid filename (usually "index.htm") for the Index Page of your web pages. The Index Page is equivalent to the Contents Page in the help system.

See Also

- Browser Tab
- Colors Tab
- Display Tab
- Fonts Tab
- Footer Tab
- Header Tab
- Styles Tab
- Versions Tab

Browser



Web Browser Location

Enter the path and name of your web browser on your computer. You may also include command line switches if your browser requires them. Refer to the manual for your web browser for more information. You may use the Browse button to search for your browser on your system.

Command Line Switches

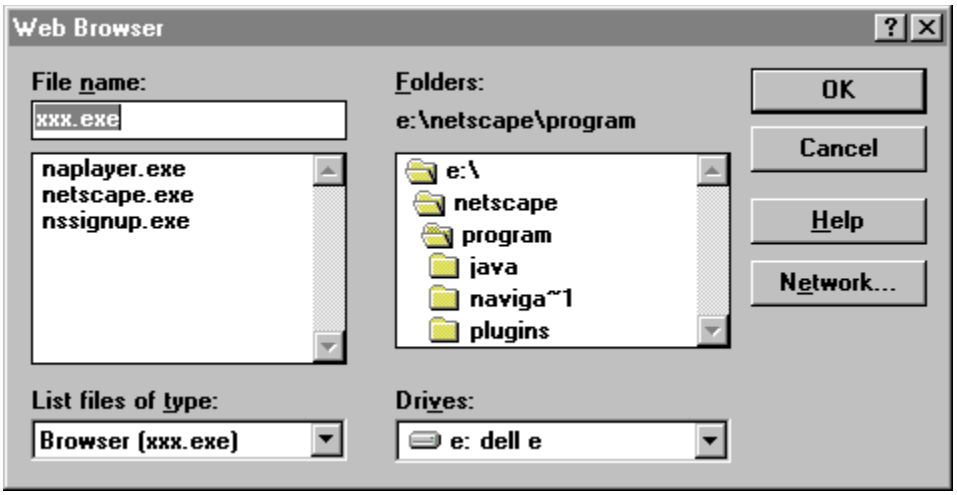
Enter any command line switches that your web browser may require to start up. Refer to your Web Browser's documentation or help files for more information.

Windows Task List Entry Description

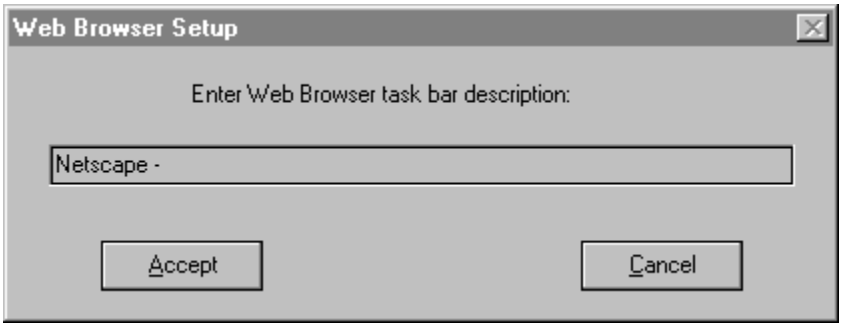
Enter the text that appears in the Windows task list when the browser is running. This will help prevent multiple versions of the browser from being executed each time the Web browser is called. A sample will be entered for you after you enter a web browser location. You can modify this field to match the text in the Windows task list if the sample text needs changing.

Calling Your Web Browser

To call your web Browser from the Help Magician, select the Web main menu item and then the Call Web Browser sub menu item. If you haven't already entered a path and filename for your web browser in the Web Options form (mentioned above), then you will be prompted to do so before continuing:

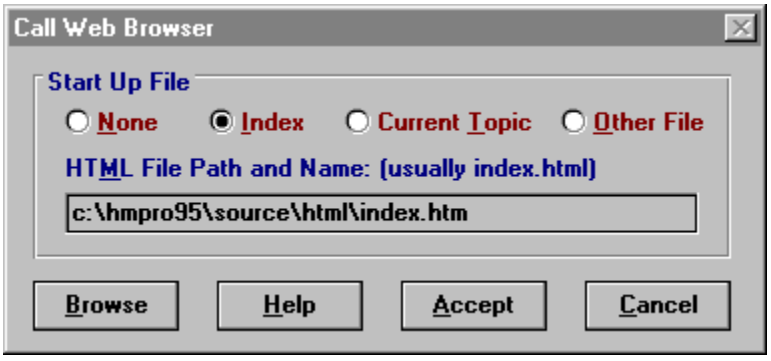


After selecting your web browser using the file dialog, you will be prompted to enter the text that appears on the Windows task bar. This will help prevent multiple instances of the web browser from executing:



Once you have completed this procedure, you will not be prompted for this information again. You can change your web browser in the Web Options dialog.

The next window displayed will be the one that you should normally see after selecting Call Web Browser menu item:



Start Up File

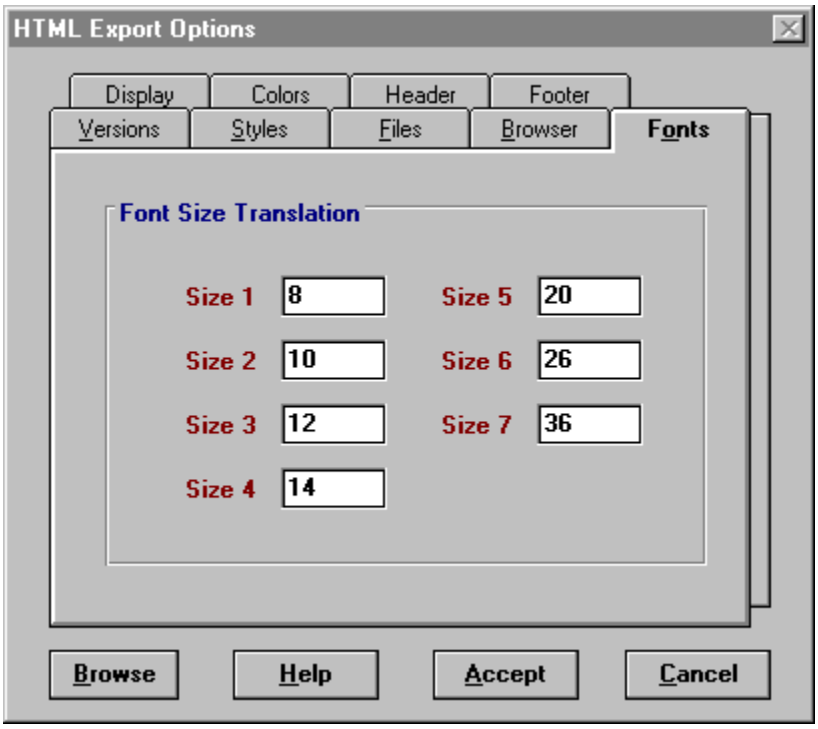
Select the file that you want to be displayed in your browser after it starts up. Choose "None" to have no file loaded at startup, "Index" to load the index file, "Current Topic" to display the HTML file that corresponds to the current file, and "Other File" to display the HTML file of your choice. Click on the Browse button to search for a particular file via a file dialog. Click on Accept when you have finished

selecting an option and call your web browser.

See Also

- Colors Tab
- Display Tab
- Files Tab
- Fonts Tab
- Footer Tab
- Header Tab
- Styles Tab
- Versions Tab

Fonts



Fonts

Font size support was introduced with Netscape's HTML extensions. There are seven font sizes defined in the document. The Help Magician allows you to map the fonts sizes you have used in your Help file to each of the seven font sizes used by the Netscape extensions. Some default values have been entered for you to start with.

To change a font size, pick a HTML font size, say Size 1, and enter a number into the adjacent text box. Enter only one number. In the example in font size 8 has been entered for HTML size 1. The Help Magician will automatically map the number entered and all sizes smaller to HTML font size 1. Likewise, the Help Magician will map all font sizes greater than 8 up to and including size 10 to HTML font size 2. There are seven predefined font sizes used in HTML pages. You have the option of specifying a range of help font sizes to convert to each HTML font size. The Help Magician defaults to the following conversions:

Help Font Size Range	HTML Font Size
up to 8 point	Size 1
more than 8 up to 12 point	Size 2
more than 12 up to 14 point	Size 3
more than 14 up to 20 point	Size 4
more than 20 up to 26 point	Size 5
more than 26 up to 36 point	Size 6
greater than 36 point	Size 7

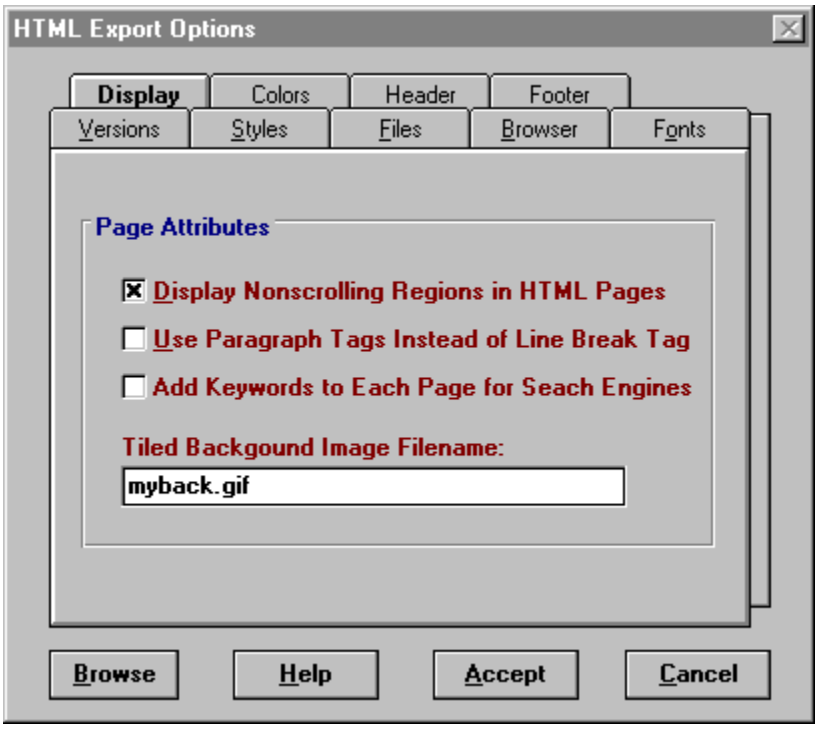
To exclude an HTML font size, enter a zero into the text box to the left of it.

See Also

- Browser Tab
- Colors Tab

Display Tab
Files Tab
Footer Tab
Header Tab
Styles Tab
Versions Tab

Display



Display

The Display tab allows you to control the appearance of the HTML pages. Any options set on this tab are applied to all HTML pages.

Display Nonscrolling Regions in HTML Document

Select this option to include any nonscrolling regions you have defined in your help file in your HTML files. The default is to include nonscrolling regions in the HTML documents.

Use Paragraph Tags Instead of Line Break Tags

Some browsers ignore paragraphs that are defined by line break tags in the HTML code. Select this option to use paragraph tags instead of line break tags for optimal display in browsers with this symptom. You may need to experiment with this option to suit your needs.

Add Keywords to Each Page for Search Engines

Select this option to have the keywords that are assigned to each Help topic added to its corresponding HTML page. The Help Magician will create a special META HTML tag that lists the keywords in the heading of each HTML document. This will allow most search engines to use these documents in searches performed on your server or over the Internet. You may wish to discuss this option with your Internet provider or system administrator. The default is not to include the keywords.

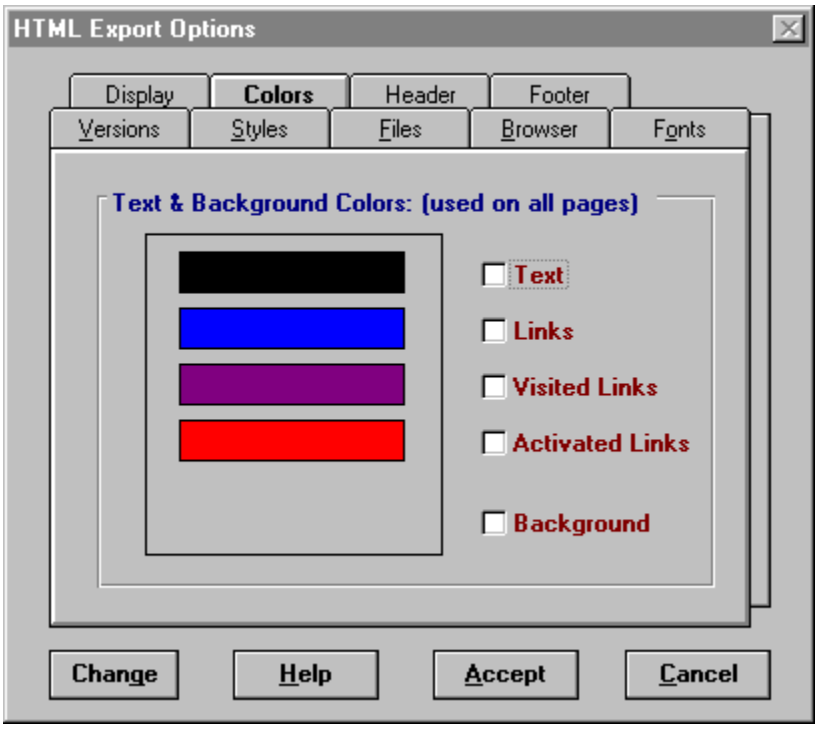
Tiled Background Image Filename

Enter the filename of a GIF file to be displayed in the background of the HTML documents. The image will be "tiled" or copied over the whole background of the page. Use the Browse button to search for a specific file via a file dialog. This is a relatively new feature and might not be displayed properly on older versions of some browsers.

See Also

- Browser Tab
- Colors Tab
- Files Tab
- Fonts Tab
- Footer Tab
- Header Tab
- Styles Tab
- Versions Tab

Colors



Colors

The colors of the text and background that appears in the HTML page when viewed in a web browser can be modified to better match a color scheme you have chosen for your HTML web pages. The colors selected apply to all web pages written with the Help Magician.

Text

Check the box to override the default color (black) of normal text. Select the Change button or double click over the bar to the left of the check box to select a different color from a color palette.

Links

Check the box to override the default color (blue) of text that represents a link to another document. Select the Change button or double click over the bar to the left of the check box to select a different color from a color palette.

Visited Links

Check the box to override the default color (purple) of a link that has been previously selected. Select the Change button or double click over the bar to the left of the check box to select a different color from a color palette.

Activated Links

Check the box to override the default color (red) of a link that is being activated (clicked on with a mouse). Select the Change button or double click over the bar to the left of the check box to select a different color from a color palette. To test the change, click and hold down the left mouse button over one of the color bars to the left of the Links" or "Visited Links"
check boxes.

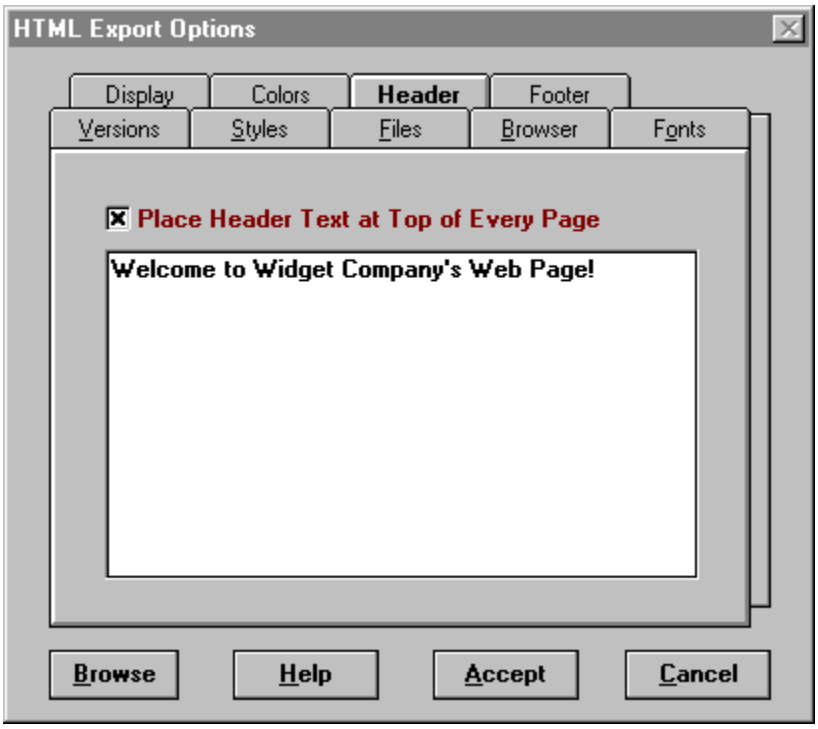
Background

Check the box to override the default background color (gray). Select the Change button or double click over the bottom region of the outlined box to the left of the check box to select a different color from a color palette.

See Also

- Browser Tab
- Display Tab
- Files Tab
- Fonts Tab
- Footer Tab
- Header Tab
- Styles Tab
- Versions Tab

Header



Header

Check the box to place text at the top of every HTML document written by the Help Magician. Enter the header text in the white text box. This can be used for welcome messages, company information, etc.

See Also

- Browser Tab
- Colors Tab
- Display Tab
- Files Tab
- Fonts Tab
- Footer Tab
- Styles Tab
- Versions Tab

Footer

Footer

Check the box to place a text at the bottom of every HTML document written by the Help Magician. Enter the footer text in the white text box. This can be used for salutations, copyright notices, etc.

See Also

- Browser Tab
- Colors Tab
- Display Tab
- Files Tab
- Fonts Tab
- Header Tab
- Styles Tab
- Versions Tab

Help Compiler Notes

Symbol Font

The Windows help compiler 3.0 does not support the Symbol font. (Problem ID: WIN9101004, Microsoft Knowledge Base).

The Symbol font problem was fixed in version 3.00b.

Extended

The newest versions of the Help Compilers for Windows 3.1 require extended or expanded memory. The actual file name for these compilers varies but they can be identified when executed. The word "(extended)" is displayed after the version number. This compiler cannot be run from Windows in Standard mode. An earlier version of the help compiler must be used if you are running Windows in Standard mode. The most reliable help compiler, at the time of this printing, is version 3.10.504 (extended), which is included with the Help Magician.

Root Directory

The Help Compiler apparently does not support the use of 'ROOT' or 'BMROOT' directories that are the drive root directory, i.e. "C:\" or "D:\". This will generate a compiler error and should be avoided.

Out of Memory

If you get out of memory errors while trying to compile your help file, make sure you haven't exceeded the limits below. Also, a version of the Help Compiler (HCP.EXE), which was built on top of Phar Lap's DOS extender, makes use of extended memory that is provided by a VCPI or DPMI-compatible memory manager. This version is included with the Help Magician for use with Windows 3.1.

Help Compiler Limits for HC31.EXE

The following is a list of limitations published in the Microsoft Developer's CD ROM:

Maximum Paragraph Size (including embedded bitmaps) = about 32,000 bytes
Maximum string length for topic title = 127 characters
Maximum topics per RTF source file = about 32,700
Maximum topics per help file = 42 million
Maximum by-reference bitmaps per help file = 32,767
Maximum help file size = 2 gigabytes
Maximum keywords per help file = limited only by file size
Maximum keyword length = 255 bytes
Maximum topics per keyword = 32,767
Maximum topics in Search dialog box = 400
Maximum browse sequence length = limited by number of topics
Maximum bookmarks per help file = limited by file size
Maximum annotations per topics = 1
Maximum annotations per help file = 32,767
Maximum history list length = 40 topics
Maximum text that can be copied to clipboard = limited by available RAM
Maximum macro length = 255
Main windows per topic = 1
Secondary window types per topic = 5

Help Compiler Limits for HCW.EXE

The following is a list of limitations published in the Microsoft Developer's CD ROM:

Project Files and WinHelp()

The Help Magician, when requested to write an RTF, will also create a project file (.HPJ) required by the Windows help compiler. This file contains information to provide the help compiler with various options selected in the Options main menu. The options that appear in the project file depend upon the compiler version selected in Compiler, Appearance and Windows options menus. As a help author, you do not need to be concerned with the contents of the HPJ file. Help Magician takes care of this for you. The text presented here is for informational purposes. The following is a sample listing of a typical WinHelp 3.1 project file, titled MYHELP.HPJ, created from a help file titled MYHELP:

```
[OPTIONS]
COMPRESS = FALSE
INDEX = Help_Topic_One
MULTIKEY = M
ROOT = C:\HC\MYHELP
BMROOT=C:\MYBMPS
TITLE = MyHelp
WARNING = 3

[FILES]
MYHELP.RTF

[BAGGAGE]
AVIFILE.AVI
MIDIFILE.MID
SOUND.WAV
MOVIE.MMM

[BITMAPS]
FLOWER.BMP
CLOWN.BMP

[CONFIG]
BrowseButtons()
[MAP]
Help_Topic_One          1     ;Help Topic One
Help_Topic_Two          2     ;Help Topic Two
Help_Topic_Three        45    ;Help Topic Three

[WINDOWS]
main="MyHelp",(123,123,256,256),0,(0,0,255),(255,0,0),1
```

The **[OPTIONS]** section sets various compiler options selected from the Project Options Menu in the editor. The TITLE option sets the title of the help system for your application. MULTIKEY sets an internal parameter for the Help Compiler to properly interpret multiple keywords in the RTF file.

The **[FILES]** section specifies the name of the RTF file to be compiled into the Windows Help Resource file, MYHELP.HLP in this case.

The **[BAGGAGE]** section will contain the file names of any of the multimedia elements used in the help system if the corresponding check box was checked on the Multimedia form. When the file names are included in the [BAGGAGE] section, the code from the media element is included in the compiled help file. This will result in a longer compile time and the help file will be larger but you won't have to ship the individual media element files with the compiled help system

The **[BITMAPS]** section specifies bitmaps included by your help system. If the complete path is not included with the filename (in the help file) they must be in the ROOT path or the path must appear in the Bitmap Paths list in the Paths menu under the Options main menu selection, or warning messages will be generated by the compiler indicating that the bitmaps could not be found in the ROOT path or bitmap paths. If the path is included with the file name, the path and file name will be included in the project file.

The **[ALIAS]** section will be written to the Project file if the information is entered in the **Project [ALIAS] Section**, available from the **Build** Menu, **Alias** Section sub menu. See Alias Section for more information.

The **[MAP]** section is where each page and its context number (one will be assigned if the context number is blank) will be mapped to a context string that the compiler uses internally. These context numbers can be used to employ context sensitive help from your application and also "segmented hypergraphics". The context strings are generated by Help Magician and are simply the topic titles with all non-alphanumeric characters mapped to valid characters acceptable to the Help Compiler (see Context Strings). The original page titles will be added as a comment at the end of each line using a semicolon delimiter.

If you specified a Context String Map File in the Compiler option form, only the Map file name will be included here as "#include <filename.ext>" and a standard 'C' header file will be generated with "#define" statements for the context strings and numbers, if you selected "Write #define File". You may also specify an external existing #define file by selecting "Use Existing File" in the Compiler Options form, and specifying a file name in the Map File text box.

Word for Windows Users: You need not worry about creating context strings anymore! The Help Magician takes care of context strings for you internally. You now only need the context number that you assigned to each topic page in the help file and use it in your calls to WINHELP from your application. Context strings are available for use in the SHED utility to edit bitmaps for hypergraphics, however. A complete printout of the context strings is available by selecting Context Relations from the File/Print Menu.

For example, if you would like the user to have context sensitive help available on a topic called "Edit" in your application program, create a help page for "Edit" in your help file and assign it a context number of your own desire. In your application program, you can call WINHELP() API procedure using the context number as the data argument and HELP_CONTEXT as the WINHELP command to access the "Edit" help topic immediately.

For more information on calling WINHELP() from your application, programming your application for help or using the SHED utility, refer to Microsoft's Help Compiler Guide and the SDK 3.1 documentation.

See Also

Context Strings

Context Strings

Context strings are used by the Help Compiler to uniquely identify each topic in the help file. The Help Magician automatically generates context strings for you by using the topic heading of each page and converting it to an acceptable format. Context strings may only contain numbers, letters, periods and/or underscore characters. Since topic headings may include any ANSI character, invalid characters must be mapped to valid ones. This is done internally by the Help Magician using the following table:

Context Strings are now refered to as Topic ID's in the Windows 95 environment.

Invalid	Valid	Invalid	Valid	Invalid	Valid
!	A	,	L	\	b
"	B	-	M	]	c
#	C	/	O	^	d
$	D	:	P	`	f
%	E	;	Q	{	g
&	F	<	R	|	h
'	G	=	S	}	i
(	H	>	T	~	j
)	I	?	U	space	_
*	J	@	V		
+	K	[	a		

A complete printout of the context strings is available by selecting View Context Relations from the File Menu. Context strings can be used for setting up segmented hypergraphics using the SHED utility provided by Microsoft. The Context String is entered in the Attributes dialogue of the SHED editor, available from the Edit/Attributes menu.

WARNING: Since the context strings are derived from the topic titles, if the topic title is changed, the reference to the Context String must be changed using SHED.EXE.

SHED.EXE

Hot Spots

The Windows Help Magician supports multiple hot spots, as provided for by the help compiler for version 3.1 of Windows.

SHED

SHED.EXE, Microsoft's HotSpot Editor, is supplied with The Microsoft Software Development Kit (SDK), Visual Basic 2.0/3.0 professional version, and the Help Magician.

In SHED.EXE, import the .BMP file to be converted and define the desired hot spots by holding the left mouse button and 'dragging' to outline the hot spot area. Select Attributes from the Edit Menu and enter the Context String associated with the page to be linked. Save the file with a .SHG extension.

In the Help Magician, insert the .SHG file, where the Bitmap is to be placed, in the editor. During compilation, the help compiler will make the appropriate links from the SHG file, using the Context Strings as a reference.

The Context Strings, associated with each page can be printed from Help Magician's File View...Context Relations menu. The Context Strings are also listed in the project (.HPJ) file, generated by the Help Magician.

The Help Magician constructs Context Strings from the topic titles. If there are no illegal characters in the title, the only change from the title is that spaces are replaced with an underscore (Context Strings). Remember that, as the title is changed, the Context String is also changed and any references to the original Context String, in SHED.EXE, must also be changed.

See Also

Context Strings

Calling WinHelp()

C Language

The following is an extract from Microsoft's help file for Quick C for Windows 3.1:

Syntax

BOOL WinHelp(hWnd, lpHelpFile, wCommand, dwData)

This function invokes the Windows Help application and passes optional data indicating the nature of the help requested by the application. The application specifies the name and, where required, the directory path of the help file which the Help application is to display.

Parameter Description

hWnd

HWND type. Identifies the window requesting help.

LpHelpFile

LPSTR type. Points to a null-terminated string containing the directory path, if needed, and the name of the help file which the Help application is to display.

WCommand

WORD type. Specifies the type of help requested. It may be any one of the following values:

Value Meaning

HELP_CONTEXT Displays help for a particular context identified by a 32-bit unsigned integer value in dwData.

HELP_HELPONHELP Displays help for using the help application itself. If the wCommand parameter is set to **HELP_HELPONHELP,** WinHelp ignores the lpHelpFile and dwData parameters.

HELP_INDEX Displays the index of the specified help file. An application should use this value only for help files with a single index. It should not use this value with HELP_SETINDEX.

HELP_KEY Displays help for a particular key word identified by a string pointed to by dwData.

HELP_MULTIKEY Displays help for a key word in an alternate keyword table.

HELP_QUIT Notifies the help application that the specified help file is no longer in use.

HELP_SETINDEX Sets the context specified by the dwData parameter as the current index for the help file specified by the lpHelpFile parameter. This index remains current until the user accesses a different help file. To help ensure that the correct index remains set, the application should call WinHelp with wCommand set to HELP_SETINDEX (with dwData specifying the corresponding context identifier) following each call to WinHelp with wCommand set to HELP_CONTEXT. An application should use this value only for help files with more than one index. It should not use this value with HELP_INDEX.

DwData

DWORD type. Specifies the context or key word of the help requested. If wCommand is HELP_CONTEXT, dwData is a 32-bit unsigned integer containing a context-identifier number. If

wCommand is HELP_KEY, dwData is a long pointer to a null-terminated string that contains a key word identifying the help topic. If wCommand is HELP_MULTIKEY, dwData is a long pointer to a MULTIKEYHELP data structure. Otherwise, dwData is ignored and should be set to NULL.

Return Value

The return value specifies the outcome of the function. It is TRUE if the function was successful. Otherwise it is FALSE.

Comments

The application must call WinHelp with wCommand set to HELP_QUIT before closing the window that requested the help. The Help application will not actually terminate until all applications that have requested help have called WinHelp with wCommand set to HELP_QUIT.

See Also
Examples
HELP Context ID Property
Visual Basic

Calling WinHelp Version 4.0

The following is an extract from Microsoft's help file for Help Workshop for Windows 95:

Syntax

BOOL WinHelp(HWND hwnd, LPCTSTR lpszHelpFile, UINT fuCommand, DWORD dwData)
The WinHelp function starts Windows Help (WINHELP.EXE) and passes additional data indicating the nature of the help requested by the program.

Parameter Description

hwnd

Handle of the window requesting Help. The WinHelp function uses this handle to keep track of which programs have requested Help. If fuCommand specifies HELP_CONTEXTMENU or HELP_WM_HELP, hwnd identifies the control requesting Help.

lpszHelpFile

Address of a null-terminated string containing the path, if necessary, and the name of the Help file that WinHelp is to display.

The filename may be followed by an angle bracket (>) and the name of a secondary window if the topic is to be displayed in a secondary window rather than in the primary window. The name of the secondary window must have been defined previously in the [WINDOWS] section of the Help Project (.HPJ) file.

fuCommand

Type of help requested. For a list of possible values and how they affect the value to place in the dwData parameter, see the Comments section.

dwData

Additional data. The value used depends on the value of the fuCommand parameter. For a list of possible values, see the Comments section.

Return Value

If successful, the return value is TRUE; otherwise, it is FALSE.

Comments

The program specifies the name and, where required, the directory path of the Help file to display.

Before closing the window that requested Help, the program must call WinHelp with the fuCommand parameter set to HELP_QUIT. Until all programs have done this, WinHelp will not terminate. Note that calling WinHelp with the HELP_QUIT command is not necessary if you used the HELP_CONTEXTPOPUP command to start Help.

See Also

Calling WinHelp()
#defines list
Examples
FuCommand Parameters
HELP Context ID Property
The HELPINFO Structure
The HELPWININFO Structure
The MULTIKEYHELP Structure

The WM TCARD Message
Visual Basic

FuCommand Parameters

The following table shows the possible values for the fuCommand parameter and the corresponding formats of the dwData parameter:

fuCommand	**Action**	**dwData**
HELP_COMMAND name of the specifies multiple separated by short form of because WinHelp	Runs a Help macro or macro string.	Address of a string that specifies the Help macro(s) to run. If the string macro names, the names must be colons or semicolons. You must use the the macro name for some Macros does not support the long name.
HELP_CONTENTS	Displays the topic specified by the Contents option in the [OPTIONS] section of the .HPJ file. This command is for backward compatibility. New programs should provide a .CNT file and use the HELP_FINDER command.	Ignored; set to 0.
HELP_CONTEXT identifer [MAP] section of the .HPJ file.	Displays the topic identified by the	Unsigned long integer specified context defined containing the context in the identifier for the topic.
HELP_CONTEXTMENU The first double word second is a context	Displays the Help menu for the selected window, and then displays (in a pop-up window) topic for the selected control.	Address of an array of double word pairs. in each pair is a control identifier, and the number for a topic.
HELP_CONTEXTPOPUP context identifier for a	Displays, in a pop-up window, the topic identified by the specified context identifier defined in the [MAP] section of the .HPJ file.	Unsigned long integer containing the topic.
HELP_FINDER	Displays the Help Topics dialog box.	Ignored; set to 0.
HELP_FORCEFILE	Ensures that WinHelp is displaying the correct Help file. If the incorrect Help file is being displayed, WinHelp opens the correct one; otherwise, there is no action.	Ignored; set to 0.
HELP_HELPONHELP	Displays Help on how to use WinHelp, if the WINHLP32.HLP file is available.	Ignored; set to 0.
HELP_INDEX	Displays the topic specified by the CONTENTS option in the [OPTIONS] section of the .HPJ file. This command is for backward compatibility. New programs should provide a .CNT file and use the HELP_FINDER command.	Ignored; set to 0.
HELP_KEY keywords must be	Displays the topic in the keyword table that matches the specified keyword, if there is an exact match. If there is more than one match, this command displays the	Address of a keyword string. Multiple separated by semi-colons.

	Topics Found list box.		
HELP_MULTIKEY	Displays the topic specified by a keyword in an alternative keyword table.	Address of a MULTIKEYHELP structure that specifies a table footnote character and a keyword.	
HELP_PARTIALKEY	Displays the topic in the keyword string. Multiple keywords must be separaed by	Address of a keyword table keyword if there is an exact match. If there is more than one match, semi-	
colons.	this command displays the Topics Found dialog box. To display the Index without passing a keyword, you should use a pointer to an empty string.		
HELP_QUIT	Informs the WinHelp program that it is no longer needed. If no other programs have asked for Help, Windows closes the WinHelp program.	Ignored; set to 0.	
HELP_SETCONTENTS	Specifies the Contents topic.	Unsigned long integer	The
WinHelp program displays	containing the context identifier for the Contents file does not have an associated .CNT file.	this topic when the user clicks the Contents button if the Help	topic.
HELP_SETPOPUP_POS	Sets the position of the subsequent pop-up window.	Address of a POINTS structure. The pop-up	
	window is positioned as if the specified point when invoked.	the mouse cursor were at the pop-up window is	
HELP_SETWINPOS	Displays the Help window, if it is minimized or in memory, and sets	Address of a HELPWININFO structure	its size
and position as specified.	that specifies the size and or secondary Help window.	position of either a primary	
HELP_TCARD	Indicates that a command is for a	Depends on the command	training
card instance of the	with which the flag is combined. combines the HELP_TCARD flag with other commands by using the bitwise OR operator.	WinHelp program. A program	
HELP_WM_HELP	Displays, in a pop-up window, the	Address of an array of	topic for
the control identified by	double word pairs. The first second is a context	hwnd. double word in each pair is a control identifier, and the identifier for a topic.	

See Also

- Calling WinHelp()
- Calling WinHelp Version 4.0
- #defines list
- Examples
- HELP Context ID Property
- The HELPINFO Structure
- The HELPWININFO Structure
- The MULTIKEYHELP Structure
- The WM TCARD Message
- Visual Basic

#defines list

Comments

The following table shows the complete list of **#defines** for **WinHelp** commands.

wCommand	**Hexadecimal value**
#define HELP_CONTEXT	0x0001
#define HELP_QUIT	0x0002
#define HELP_INDEX	0x0003 (Windows Help version 3.0)
#define HELP_CONTENTS	0x0003
#define HELP_HELPONHELP	0x0004
#define HELP_SETINDEX	0x0005 (Windows Help version 3.0)
#define HELP_SETCONTENTS	0x0005
#define HELP_CONTEXTPOPUP	0x0008
#define HELP_FORCEFILE	0x0009
#define HELP_KEY	0x0101
#define HELP_COMMAND	0x0102
#define HELP_POPUPID	0x0104
#define HELP_PARTIALKEY	0x0105
#define HELP_CLOSEWINDOW	0x0107
#define HELP_CONTEXTNOFOCUS	0x0108
#define HELP_MULTIKEY	0x0201
#define HELP_SETWINPOS	0x0203

The following values may also be used, however, since they are not in windows.h, you must add them to your own header file:

wCommand	**Hexadecimal value**
#define HELP_FORCE_GID	0x000e
#define HELP_TAB	0x000f

fuCommand	**Action**	**dwData**
HELP_FORCE_GID	Changes to the .GID file associated with the help file passed in as the lpszHelpFile parameter.	Ignored; set to 0.
HELP_TAB	Opens the Help Topics dialog	Zero-based index of the box, displaying
the specified	extensible tab to display (0	extensible tab. is the first tab, 1 the
second,		etc.).

See Also

- Calling WinHelp()
- Calling WinHelp Version 4.0
- Examples
- FuCommand Parameters
- HELP Context ID Property
- The HELPINFO Structure
- The HELPWININFO Structure
- The MULTIKEYHELP Structure
- The WM TCARD Message
- Visual Basic

The HELPWININFO Structure

```
typedef struct {              // hwi
    int    wStructSize;
    int    x;
    int    y;
    int    dx;
    int    dy;
    int    wMax;
    TCHAR rgchMember[2];
} HELPWININFO;
```

The HELPWININFO structure contains the size and position of either a primary or a secondary Help window.

Parameter	Description
wStructSize	Size of this structure, in bytes.
x	X-coordinate of the upper left corner of the window, in screen coordinates.
y	Y-coordinate of the upper left corner of the window, in screen coordinates.
dx	Width of the window, in pixels.
dy	Height of the window, in pixels.
wMax	Value specifying how to show the window. This member must be one of these values:

Value	Meaning
SW_HIDE	Hides the window and passes activation to another window.
SW_MINIMIZE	Minimizes the specified window and activates the top-level window in the Z order.
SW_RESTORE	Same as SW_SHOWNORMAL.
SW_SHOW	Activates a window and displays it in its current size and position.
SW_SHOWMAXIMIZED	Activates the window and displays it as a maximized window.
SW_SHOWMINIMIZED	Activates the window and displays it as an icon.
SW_SHOWMINNOACTIVE	Displays the window as an icon. The window that is currently active remains active.
SW_SHOWNA	Displays the window in its current state. The window that is currently active remains active.
SW_SHOWNOACTIVATE	Displays a window in its most recent size and position. The window that is currently active remains active.
SW_SHOWNORMAL	Activates and displays the window. Whether the window is minimized or maximized, Windows restores it to its original size and position.
rgchMember	Name of the window.

Comments

WinHelp divides the display into 1024 units in both the x- and y-directions. To create a secondary window that fills the upper left quadrant of the display, for example, a program would specify zero for the x and y members and 512 for the dx and dy members.

A program can set the size and position information by calling the WinHelp function with the HELP_SETWINPOS value.

Training Card Help

Using training card help, an application can display a sequence of instructions to guide the user through the steps of a task. A training card typically consists of text that explains a particular step and authorable buttons associated with TCard macros that allow the user to tell the application what to do next. Training cards may only be displayed in secondary windows and must not contain hot links to other topics in the help file.

An application initiates the training card instance of Windows Help by calling the WinHelp function and specifying the HELP_TCARD command in combination with another command such as HELP_CONTEXT. Subsequently, when the user clicks an authorable button in the training card, clicks a hot spot assigned to the TCard macro, or closes the training card, Windows Help notifies the application by sending it a WM_TCARD message. The *wParam* parameter identifies the button or user action, and the *lParam* parameter contains additional data, the meaning of which depends on the value of *wParam*.

See Also

- Calling WinHelp()
- Calling WinHelp Version 4.0
- #defines list
- Examples
- FuCommand Parameters
- HELP Context ID Property
- The HELPINFO Structure
- The MULTIKEYHELP Structure
- The WM TCARD Message
- Visual Basic

The MULTIKEYHELP Structure

```
typedef struct tagMULTIKEYHELP {       // mkh
    DWORD mkSize;
    BYTE   mkKeylist;
    TCHAR szKeyphrase[1];
} MULTIKEYHELP;
```

The MULTIKEYHELP structure specifies a keyword table and an associated keyword to be used by the WinHelp program.

Parameter	Description
mkSize	Size of this structure, in bytes.
mkKeylist	A single character that identifies the keyword table to search.
szKeyphrase	A null-terminated text string that specifies the keyword to locate in the keyword table.

The WM_HELP message:

lphi = (LPHELPINFO) lParam;

The WM_HELP message is sent whenever the user presses the F1 key.

Parameter	Description
lphi	Address of a HELPINFO structure that contains information about the menu item, control, dialog box, or window for which Help is requested.

Default Action

The DefWindowProc function passes WM_HELP to the parent window of a child window, or to the owner of a top-level window.

Comments

If a menu is active when F1 is pressed, WM_HELP is sent to the window associated with the menu. Otherwise, WM_HELP is sent to the window that has the keyboard focus. If no window has the keyboard focus, WM_HELP is sent to the currently active window.

See Also

- Calling WinHelp()
- Calling WinHelp Version 4.0
- #defines list
- Examples
- FuCommand Parameters
- HELP Context ID Property
- The HELPINFO Structure
- The HELPWININFO Structure
- The WM TCARD Message
- Visual Basic

The HELPINFO Structure

```
typedef   struct   tagHELPINFO {          // hi
    UINT     cbSize;
    int      iContextType
    int      iCtrlId;
    HANDLE hItemHandle;
    DWORD   dwContextId;
    POINT   MousePos;
} HELPINFO, FAR *LPHELPINFO;
```

The HELPINFO structure contains information about an item for which context-sensitive Help has been requested.

Parameter	Description
cbSize	Size of this structure, in bytes.
iContextType	Type of context for which Help is requested. This member can be one of these values:

Value	Meaning
HELPINFO_MENUITEM	Help requested for a menu item.
HELPINFO_WINDOW	Help requested for a control or window.
iCtrlId	Identifier of the window or control if the iContextType parameter is HELPINFO_WINDOW, or identifier of the menu item if the iContextType parameter is HELPINFO_MENUITEM.
hItemHandle	Identifier of the child window or control if iContextType is HELPINFO_WINDOW, or identifier of the associated menu if iContextType is HELPINFO_MENUITEM.
dwContextId	Help context identifier of the window or control.
MousePos	A POINT structure that contains the screen coordinates of the mouse cursor. This parameter is useful for providing Help based on the position of the mouse cursor.

See Also

- Calling WinHelp()
- Calling WinHelp Version 4.0
- #defines list
- Examples
- FuCommand Parameters
- HELP Context ID Property
- The HELPWININFO Structure
- The MULTIKEYHELP Structure
- The WM TCARD Message
- Visual Basic

The WM TCARD Message

```
idAction = wParam;          // user action
dwActionData = lParam;      // action data
```

The WM_TCARD message is sent to a program that has initiated a training card with the WinHelp program.

Parameter	Description
idAction	Value of wParam. Indicates the action that the user has taken. This parameter can be one of the following values:

Value	Meaning
IDABORT	The user clicked an authorable Abort button.
IDCANCEL	The user clicked an authorable Cancel button.
IDCLOSE	The user closed the training card.
IDHELP	The user clicked an authorable Help button.
IDIGNORE	The user clicked an authorable Ignore button.
IDOK	The user clicked an authorable OK button.
IDNO	The user clicked an authorable No button.
IDRETRY	The user clicked an authorable Retry button.
IDYES	The user clicked an authorable Yes button.
HELP_TCARD_DATA	The user clicked an authorable button. The lParam parameter contains a long integer specified by the Help author.
HELP_TCARD_NEXT	The user clicked an authorable Next button.
HELP_TCARD_OTHER_CALLER	Another program has requested training cards.

dwActionData	Value of lParam. If idAction specifies HELP_TCARD_DATA, this parameter is a long integer specified by the Help author. Otherwise, this parameter is zero.

Return Value

The return value is ignored; use zero.

Comments

The message informs the program when the user clicks an authorable button or hotspot that runs the TCard macro. A program initiates a training card by specifying the HELP_TCARD command in a call to the WinHelp function.

WinHelp can run only one training card at a time. If another program requests a training card, the program currently using training cards receives a WM_TCARD message with wParam == HELP_TCARD_OTHER_CALLER. The program receiving this message should either terminate the training card (HELP_QUIT | HELP_TCARD), or the next time it gets the focus, it should call WinHelp with a HELP_FORCEFILE | HELP_TCARD command.

See Also

Calling WinHelp()
Calling WinHelp Version 4.0
#define list
Examples
FuCommand Parameters
HELP Context ID Property
The HELPINFO Structure
The HELPWININFO Structure
The MULTIKEYHELP Structure
Visual Basic

Visual Basic

Declaration

To call WINHELP.EXE from a Visual Basic application, first declare the API function in the global module of your source:

Declare Function WinHelp Lib "User" (ByVal hWnd As Integer, ByVal lpHelpFile As String, ByVal wCommand As Integer, dwData As Any) As Integer

Enter the declaration on one line. Do not enter carriage returns.

Constants

Paste CONSTANT.TXT into your global module, or enter the following lines:

```
'Help Constants
Global Const HELP_CONTEXT = &H1
        'Display topic in ulTopic
Global Const HELP_QUIT = &H2
        'Terminate help
Global Const HELP_INDEX = &H3
        'Display index
Global Const HELP_CONTENTS = &H3
Global Const HELP_HELPONHELP = &H4
        'Display help on using help
Global Const HELP_SETINDEX = &H5
        'Set the current Index for multi index help
Global Const HELP_SETCONTENTS = &H5
Global Const HELP_CONTEXTPOPUP = &H8
Global Const HELP_FORCEFILE = &H9
Global Const HELP_KEY = &H101
        'Display topic for keyword in offabData
Global Const HELP_COMMAND = &H102
Global Const HELP_PARTIALKEY = &H105
        'call the search engine in WinHelp
```

The comments are shown here on the line following the declarations because of space constraints. They can follow the declaration, on the same line.

See Also

- Calling WinHelp()
- Calling WinHelp Version 4.0
- #define list
- Examples
- FuCommand Parameters
- HELP Context ID Property
- The HELPINFO Structure
- The HELPWININFO Structure
- The MULTIKEYHELP Structure
- The WM TCard Structure

Examples

Contents

To call your help system, opening with the contents page (the calls are shown on two lines because of space constraints - enter the calls on one line in your code):

```
dwData& = 1                'Assumes Contents page 1
ReturnValue% = WinHelp(hWnd, "myhelp.hlp", HELP_CONTEXT, ByVal dwData&)
```

Specific Topic

To call a specific topic in the help file:

```
dwData& = 58               'Specific Topic Context Number
ReturnValue% = WinHelp(hWnd, "myhelp.hlp", HELP_CONTEXT, ByVal dwData&)
```

Help on Help

To call the help file, provided by Microsoft, that gives the end user help on using help systems:

```
ReturnValue% = WinHelp(hWnd, "myhelp.hlp", HELP_PARTIALKEY, ByVal dwData&)
```

Close

To close the help system, before closing the window that requested the help:

```
ReturnValue% = WinHelp(hWnd, "myhelp.hlp", HELP_QUIT, ByVal dwData&)
```

See Also

- Calling WinHelp()
- Calling WinHelp Version 4.0
- #define list
- FuCommand Parameters
- HELP Context ID Property
- The HELPINFO Structure
- The HELPWININFO Structure
- The MULTIKEYHELP Structure
- The WM TCard Structure
- Visual Basic

HELP Context ID Property

VB 2.0/3.0

Visual Basic version 2.0/3.0 has a built in mechanism to employ the use of context sensitive help.

Help File

To make use of the built in help file functions, you must first declare the name of the help file for the application:

App.HelpFile = "myhelp.hlp"

Place this declaration in the Form_Load procedure of your main form.

Context ID

In Visual Basic version 2.0/3.0, most controls have a HelpContextID property. To make use of the built in context sensitivity, set the HelpContextID property of each control to the Context Number of the page that provides help on that control.

When the user presses the F1 key, the help system will be called, with the appropriate page displayed. If no Context ID is assigned to the active control, the help system will start with the index page.

Constants

Constants can be declared, assigning the value of the Context Number to a string descriptor. Search Visual Basic's help on the word HelpContextID for more information.

See Also

- Calling WinHelp()
- Calling WinHelp Version 4.0
- #define list
- Examples
- FuCommand Parameters
- The HELPINFO Structure
- The HELPWININFO Structure
- The MULTIKEYHELP Structure
- The WM TCard Structure
- Visual Basic

Compiler Errors 3.1

The following compiler errors are for Help compiler version 3.1 and earlier.

Message numbers	Type of errors	Message numbers	Type of errors
1019-1536	Source file errors	2771-2932	Other options errors
2010	Project file errors	3011-3178	Build tag footnote and expression errors
2030-2214	Syntax errors	3511-3652	Macro errors
2273-2331	General section errors	4011-4196	Context string errors
2341-2372	ALIAS and MAP section errors	4211-4312	Footnote errors
2391-2501	WINDOWS section errors	4331-4393	Topic title errors
2511-2532	OPTIONS section errors	4412-4452	Keyword errors
2550-2570	Root option errors	4471-4492	Build tag errors
2591-2632	Font range option errors	4551	Entry macro errors
2651-2672	Forcefont errors	4616-4813	Topic file errors
2691-2752	Multikey errors	5035-5115	Miscellaneous errors

Source File Errors

1019 **Project file extension cannot be .HLP or .PH.**
You cannot specify a project file with an .HLP or .PH extension. Project files must use the .HPJ extension.
Rename the Help project file and then recompile.

1030 **File name exceeds limit of 259 characters.**
The combined length of the path and file name is more than the MS-DOS limit of 259 characters.
Shorten the path and then recompile.

1079 **Out of file handles.**
The Help Compiler does not have enough available file handles to continue the build.

1100 **Cannot open file *filename*: permission denied.**
You do not have the required file privileges to open the requested file.

1150 **Cannot overwrite file *filename.***
The Help file cannot overwrite the specified file because the file has a read-only attribute.
Rename the Help project file or change the read-only attribute.

1170 **File *filename* is a directory.**
A subdirectory in the Help project root directory has the same name as the requested Help file. This is a MS-DOS file error.
Move or rename the subdirectory and then recompile.

1190 **Cannot use reserved MS-DOS file *filename.***
A file has been referred to by a reserved MS-DOS name such as COM1, LPT2, or PRN. This is a MS-DOS file error.
Rename the file and then recompile.

1230 **File *filename* not found.**
The specified file could not be found or is unreadable. This is a MS-DOS file error or an out-of-memory condition.
Check to see if the file exists and also check the amount of available memory.

1292 **File *filename* is not a valid bitmap.**
The specified bitmap file could not be found or is not in a recognizable bitmap format. This is a MS-DOS file error or an out-of-memory condition.
Check to see if the file exists, and if it does, check its format. If necessary, save the file again in your paint or draw program, and then recompile.

1319 **Disk full.**
The Help file could not be written to disk.
Create more space on the destination disk and then recompile.

1513 **Bitmap name *filename* duplicated.**
The [BITMAPS] section contains duplicate bitmap names. The Help Compiler uses the first occurrence of the name.
Rename the duplicate bitmap file names and then recompile.

1536 **Not enough memory to check and compress bitmap *filename.***
The specified bitmaps cannot be compressed because of insufficient memory. If any of the specified bitmaps are segmented hypergraphics, the context strings stored in them are not checked for validity during the build.

Project File Errors

2010 Include statements nested more than 5 deep.
The **#include** statement on the specified line has exceeded the maximum of five include levels. Do not nest **#include** statements more than 5 deep.

Syntax Errors

2030 **Comment starting at line *linenumber* of file *filename* unclosed at end of file.**
The Help Compiler has unexpectedly come to the end of the Help project file. There may be an open comment in the .HPJ file or in an included file.

2050 **Invalid #include syntax.**
The correct **#include** syntax is as follows:
#include <filename>
Correct the syntax and then recompile.

2091 **Bracket missing from section heading [sectionname].**
The right bracket (]) is missing from the specified section heading.
Insert the bracket and then recompile.

2111 **Section heading missing.**
The section heading on the specified line is not complete. This error is also reported if the first entry in the Help project file is not a section heading.

2131 **Invalid OPTIONS syntax: option=value' expected.**
Check the syntax of the options in the [OPTIONS] section.

2141 **Invalid ALIAS syntax: context=context' expected.**
Check the syntax of the entries in the [ALIAS] section.

2151 **Incomplete line in [*sectionname*] section.**
The entry on the specified line is incomplete. The Help Compiler skips the line.

2171 **Unrecognized text.**
There is unrecognizable text following valid text.
The Help Compiler ignores the line.

2191 **Section heading [*sectionname*] unrecognized.**
A section heading that is not supported by the Help Compiler has been used.
The Help Compiler ignores the line.

2214 **Line in .HPJ file exceeds length limit of 2047 characters.**
There is a line in the .HPJ file that exceeds the maximum length of 2047 characters.

General Section Errors

2273 **[OPTIONS] should precede [FILES] and [BITMAPS] for all options to take effect.**
It is recommended that the [OPTIONS] section be the first section of the .HPJ file so that all the options will take effect. Also, if the **ERRORLOG** option is used, it should be the first line in the [OPTIONS] section.

2291 **Section *sectionname* previously defined.**
A duplicate section has been found in the Help project file.
The Help Compiler ignores the lines under the duplicated section and continues from the next valid section heading.

2305 **No valid files in [FILES] section.**
The file section is either empty or contains only invalid files.

2322 **Context string *context_name* cannot be used as alias string.**
A context string that has been assigned an alias cannot be used later as an alias for another context string. That is, you cannot map a = b and then c = a in the [ALIAS] section.
The Help Compiler ignores the attempted reassignment on this line.

2331 **Context number already used in [MAP] section.**
The context number on the specified line in the Help project file was previously mapped to a different context string.
The Help Compiler ignores the line.

ALIAS and MAP Section Errors

2341 **Invalid or missing context string.**
The specified line is missing a context string before an equal sign.

2351 **Invalid context identification number.**
The context number on the specified line is empty or contains invalid characters.

2362 **Context string *context_name* already assigned an alias.**
A context string can only have one alias. That is, you cannot map a = b and then a = c in the [ALIAS] section. The specified context string has already been assigned an alias in the [ALIAS] section.
The Help Compiler ignores the attempted reassignment on this line.

2372 **Alias string aliasname already assigned.**
You can't alias an alias. An alias string cannot, in turn, be assigned another alias. That is, you cannot map a = b and then b = c in the [ALIAS] section.
The Help Compiler ignores the attempted reassignment on this line.

WINDOWS Section Errors

2391 **Limit of 6 window definitions exceeded.**
The maximum number of window definitions is one main window definition and five secondary window definitions.

2401 **Window maximization state can assume any of ten different values.**

0 SW_HIDE	Hides the window and passes activation to another window.
1 SW_SHOWNORMAL	Activates and displays a window. If the window is minimized or maximized, Windows restores it fo its original size and position.
2 SW_SHOWMINIMIZED	Activates the window and displays it as an icon
3 SW_SHOW	Activates the window and displays it as a maximized window.
4 SW_SHOWNOACTIVATE	Display's at it's most recent size and position. The window that is currently active remains active.
5 SW_SHOW	Activates a window and displays it in it's current size and position.
6 SW_MINIMIZE	Minimizes the specified window and activates the top-level window in the window-manager's list.
7 SW_SHOWMINOACTIVE	Displays the window as iconic. The window that is currently active remaing active.
8 SW_SHOW	Displays the window in it's current state. The window that is currently active remains active.
9 SW_RESTORE	Same as SW_SHOWNORMAL

The value of the fMax variable in the window attribute specification in the .HPJ file is invalid.
Correct the entry and then recompile.

2411 **Invalid syntax in window color.**
The correct syntax for the text and background color is: .mono (rrr, ggg, bbb).endmono.
Correct the syntax and then recompile. For information on setting the window color, please see "Defining Help Windows" in the Help Compiler Guide, "Building your Help File."

2421 **Invalid window position.**
The correct syntax to indicate the predefined window position is: .mono (x, y, DX, dy).endmono.
The Help Compiler ignores this line, and the window does not have a predefined position.
For more information, please see "Defining Help Windows" in the Help Compiler Guide, "Building your Help File."

2431 **Missing quote in window caption.**
The value of the caption attribute for the window definition in the .HPJ file is not enclosed in quotation marks.
Correct the syntax and then recompile.

2441 **Window name *windowname* is too long.**
The window name exceeds the maximum length of 8 characters.

2451 **Window position value out of range 01023.**
One or more of the window position coordinates (X, Y, X + dX, or [Y+dY]) exceed the maximum position value of 1023.

2461 **Window name missing.**
The window specification in the .HPJ file is missing the window name.

2471 **Invalid syntax in [WINDOWS] section.**
The entry for a main or secondary window is incorrect.
The Help Compiler ignores the window entry.
Check the window entry syntax and then recompile.

2481 **Secondary window position required.**
The X, Y, dX, and dY entries for the secondary window definitions must be specified in the .HPJ file.

2491 **Duplicate window name *windowname.***
There are duplicate window names in the .HPJ file.
Check the uniqueness of each member name and then recompile.

2501 **Window caption *windowcaption* exceeds limit of 50 characters.**

The caption for the window exceeds the limit of 50 characters.

OPTIONS Section Errors

2511 **Unrecognized option *optionname* in [OPTIONS] section.**
An option has been used that is not supported by the compiler.
The Help Compiler skips this line.

2532 **Option *optionname* previously defined.**
The specified option has been defined on a previous line.
The Help Compiler ignores the attempted redefinition.

Root Option Errors

2550 **Invalid path pathname in *optionname* option.**
The Help Compiler cannot find the path specified by the ROOT option.
The Help Compiler uses the current working directory.

2570 **Path in *optionname* option exceeds number of characters.**
The path specified by the ROOT option exceeds the MS-DOS maximum limit.
The Help Compiler ignores the path and uses the current working directory.

Font Range Option Errors

2591 **Invalid MAPFONTSIZE option.**
The font range syntax is invalid.
The correct syntax is m[-n]:p.

2612 **Maximum of 5 font ranges exceeded.**
The maximum number of font ranges that can be specified is five.
The Help Compiler ignores any additional ranges.

2632 **Current font range overlaps previously defined range.**
A font size range overlaps a previously defined mapping.
The Help Compiler ignores the second mapping.
Adjust one or both of the font ranges to remove any overlaps.

Forcefont Errors

2651 **Font name exceeds limit of 20 characters.**
Font names cannot exceed 20 characters.
The Help Compiler ignores this line.

2672 **Unrecognized font name *fontname* in FORCEFONT option.**
The Help Compiler has encountered a font name that it does not support.
The Help Compiler ignores the font name and uses the default Helvetica font.

Multikey Errors

2691 **Invalid MULTIKEY syntax.**
The Help Compiler does not recognize the syntax used in a **MULTIKEY** option. The valid syntax is **MULTIKEY** = char, where char is any capital letter other than "K."

2711 **Maximum of 5 keyword tables exceeded.**
The limit of five keyword tables has been exceeded.
The Help Compiler ignores the additional tables.
Reduce the number of tables and then recompile.

2732 **Character already used.**
A character used for indicating the keyword table (**MULTIKEY** = char) was previously used.
The Help Compiler ignores the entry.

2752 **Characters K' and k' cannot be used.**
These characters are reserved for Help's normal keyword table.
Choose another character, and then recompile.

Other Options Errors

2771 **REPORT option must be ON' or OFF'.**
The **REPORT** option must be either **True**, **1, ON, YES, False, 0, OFF**, or **NO.** Correct the entry and then recompile.

2811 **OLDKEYPHRASE option must be ON' or OFF'.**
The OLDKEYPHRASE option must be either **True, 1, ON, YES, False, 0, OFF**, or **NO.**
Correct the entry and then recompile.

2832 **COMPRESS option must be OFF', MEDIUM' or HIGH'.**
The **COMPRESS** option must be either **1, YES, ON, True, HIGH, MEDIUM, 0, NO, OFF, or False.**
Correct the entry and then recompile.

2842 **OPTCDROM option must be TRUE' or FALSE'.**
The **OPTCDROM** option must be either **True** or **False.**
The Help Compiler defaults to False.
Correct the entry and then recompile.

2852 **Invalid TITLE option.**
The **TITLE** option defines a string that is empty or contains more than 32 characters.
The Help Compiler truncates the title.

2872 **Invalid LANGUAGE option.**
You have specified an ordering that is not supported by the compiler.
The Help Compiler defaults to U.S. sort ordering.

2893 **Warning option must be 1, 2, or 3.**
The warning reporting level can be set only to **1**, **2**, or **3.**
The Help Compiler defaults to full reporting (level 3).

2911 **Invalid icon file *filename.***
The Help Compiler cannot find the icon file specified in the **ICON** option, or the file is not a valid icon file.

2932 **Copyright string exceeds limit of 50 characters.**
The maximum length of the copyright string in the About box is 50 characters.
The Help Compiler truncates the string.

Build Tag Footnote and Expression Errors

3011 **Maximum of 32 build tags exceeded.**
The maximum number of build tags that can be defined is 32.
The Help Compiler ignores the additional tags.

3031 **Build tag length exceeds 32 characters.**
The build tag on the specified line exceeds the maximum of 32 characters.
The Help Compiler skips this entry.

3051 **Build tag *tagname* contains invalid characters.**
Build tags can contain only alphanumeric characters or the underscore (_) character.
The Help Compiler skips this line.

3076 **[BUILDTAGS] section missing.**
The **BUILD** option declared a conditional build, but there is no [BUILDTAGS] section in the Help project file.
The Help Compiler includes all topics in the build.

3096 **Build expression too complex.**
The build expression has too many expressions ("~", "|" or "&") or is too deeply nested.

3116 **Invalid build expression.**
The syntax used on the specified line of the build expression contains one or more logical or syntax errors.

3133 **Duplicate build tag in [BUILDTAGS] section.**
A build tag in the [BUILDTAGS] section has been repeated unnecessarily.

3152 **Build tag *tagname* not defined in [BUILDTAGS] section.**
The specified build tag has been assigned to a topic, but not declared in the Help project file.
The Help Compiler ignores the tag for the topic.

3178 **Build expression missing from project file.**
The topics have build tags, but there is no **BUILD** = *expression* in the .HPJ file.
The Help Compiler includes all topics in the build.

Macro Errors

3511 **Macro *macrostring* exceeds limit of 254 characters.**
The macro string exceeds the maximum limit of 254 characters.

3532 **Undefined function in macro *macroname.***
The specified macro is not on the list of macros supported by the compiler, nor is it specified in the **RegisterRoutine().**
The Help Compiler nonetheless passes the macro to the .HLP file.

3552 **Undefined variable in macro *macroname.***
The macro contains a variable that is not recognized by the compiler.

3571 **Wrong number of parameters to function in macro *macroname.***
There are too many or too few parameters in the macro.

3591 **Syntax error in macro *macroname.***
The syntax of the macro is invalid.

3611 **Function parameter type mismatch in macro *macroname.***
There is a type mismatch (string or numeric) in the function call.

3631 **Bad macro prototype.**
The prototype string passed to **RegisterRoutine** is invalid.

3652 **Empty macro string.**
The "!" footnote or a hidden text starting with "!" does not contain a macro.

Context String Errors

4011 **Context string *contextname* already used.**
The specified context string was previously assigned to another topic.
The Help Compiler ignores the latter string, and the topic has no identifier.

4031 **Invalid context string contextname.**
The context string footnote contains non-alphanumeric characters or is empty.
The Help Compiler does not assign the topic an identifier.

4056 **Unresolved context string specified in CONTENTS option.**
The Contents topic defined in the Help project file could not be found.
The Help Compiler uses the first topic in the build as the Contents.

4072 **Context string exceeds limit of 255 characters.**
The context string hidden text cannot exceed 255 characters.
The Help Compiler ignores the string.

4098 **Context string(s) in [MAP] section not defined in any topic.**
The Help Compiler cannot find a context string listed in the [MAP] section in any of the topics in the build.
The Help Compiler ignores the entry.

4113 **Unresolved jump or pop-up *contextname.***
The specified topic contains a context string that identifies a nonexistent topic.
Check the topic for spelling errors in the context string, and also check to see if the requested topic is included in the build.

4131 **Hash conflict between *contextname* and *contextname.***
The hash algorithm has generated the same hash value for both of the listed context strings.
Change either of the context strings and then recompile.

4151 **Invalid secondary window name *windowname.***
The window name for the secondary window is "main" or another disallowed member name.

4171 **Cannot use secondary window with pop-up.**
The hidden text defining the pop-up identifier contains a secondary window name.

4196 **Jumps and lookups not verified.**
Due to the low memory conditions, the build is continued without the jump and keyword validity verification.

Footnote Errors

4211 **Footnote text exceeds limit of 1023 characters.**
The footnote text cannot exceed the limit of 1023 characters.
The Help Compiler ignores the footnote.

4231 **Footnote text missing.**
The specified topic contains a footnote that has no characters.

4251 **Browse sequence not in first paragraph.**
The browse-sequence footnote is not in the first paragraph of the topic.
The Help Compiler ignores the browse sequence.

4272 **Empty browse sequence string.**
The browse-sequence footnote for the specified topic contains no sequence characters.

4292 **Missing sequence number.**
A browse-sequence number ends in a colon (:) for the specified topic.
Remove the colon, or enter a "minor" sequence number and then recompile.

4312 **Browse sequence already defined.**
There is already a browse-sequence footnote for the specified topic.
The Help Compiler ignores the latter sequence.

Topic Title Errors

4331 **Title not in first paragraph.**
The title footnote ($) is not in the first paragraph of the topic.
The topic will not have a topic title string.

4352 **Empty title string.**
The title footnote for the specified topic contains no characters.
The Help Compiler does not assign the topic a title.

4372 **Title defined more than once.**
There is more than one title footnote in the specified topic.
The Help Compiler uses the first title string.

4393 **Title exceeds limit of 128 characters.**
The title for the specified topic exceeds the limit of 128 characters.
The Help Compiler ignores the additional characters.

Keyword Errors

4412 **Keyword string exceeds limit of 255 characters.**
The keyword string exceeds the maximum limit of 255 characters.

4433 **Empty keyword string.**
There are no characters in the keyword footnote.

4452 **Keyword(s) defined without title.**
The topic has a keyword assigned to it, but no title.
The topic will appear as ">>Untitled Topic<<" in the history list and in the keyword search dialog.

Build Tag Errors

4471 **Build tag footnote not at beginning of topic.**
The build tag footnote marker, if used, has to be the first character in the topic.

4492 **Build tag exceeds limit of 32 characters.**
A build tag for the specified topic exceeds the maximum of 32 characters.
The Help Compiler ignores the tag for the topic.

Entry Macro Errors

4551 Entry macro not in first paragraph.
The "!" footnote (for running a macro) is not in the first paragraph of the topic. The Help Compiler ignores the macro.

Topic File Errors

4616 **File *filename* is not a valid RTF topic file.**
The specified file is not an RTF file.
Check to make sure that you have saved the topic file as RTF from your word processor.

4639 **Error in file *filename* at byte offset 0x%lX.**
The specified file contains unrecognized RTF at that byte offset.
This message should not appear if you are using Microsoft Word for Windows, Microsoft Word for MS-DOS, or Microsoft Word for the Macintosh..
Check the RTF syntax and then recompile. If you are using Microsoft Word for the Macintosh, transfer the file to the PC again, and then recompile.

4649 **File *filename* contains more than 32767 topics.**
The maximum number of topics allowed in one RTF file is 32767.

4652 **Table formatting too complex.**
The Help Compiler encountered a table with borders, shading, or right justification.
Remove the unsupported formatting and then recompile.

4662 **Side-by-side paragraph formatting not supported.**
The side-by-side paragraph formatting is not supported in WinHelp in Windows version 3.1. The Help Compiler ignores the side-by-side text.
If you are using WinHelp in Windows 3.1, use the table feature.

4671 **Table contains more than 32 columns.**
The maximum number of columns in one table is 32. Some word processors may have different limits for the number of columns supported.

4680 **Font *fontname* in file *filename* not in RTF font table.**
A font not defined in the RTF header has been entered into the topic.
The Help Compiler uses the default system font.

4692 **Unrecognized graphic format.**
The Help Compiler supports only Windows bitmaps and metafiles.
The Help Compiler ignores the graphic.
Make sure that you have not used Macintosh picture formats.

4733 **Hidden page break.**
The page break was found as a part of the hidden text. A page break formatted as hidden text will not separate two topics.

4753 **Hidden paragraph.**
A paragraph marker was found in the hidden text.
The Help Compiler ignores the paragraph marker.

4763 **Hidden carriage return.**
A carriage return was found in the hidden text.
The Help Compiler ignores the carriage return.

4774 **Paragraph exceeds limit of 64K.**
A single paragraph has more than 64K of text or 64K of graphics.

4792 **Nonscrolling region defined after scrolling region.**
A paragraph that was authored as "keep with next" is not the first paragraph in the topic.
The Help Compiler ignores the "keep with next" attribute, and the paragraph is treated as regular text and will be part of the regular topic text.

4813 **Nonscrolling region crosses page boundary.**
The "keep with next" paragraph formatting crosses a page break boundary.

Miscellaneous Errors

5035 **File *filename* not created.**
There are no topics to compile, or the build expression is False for all topics.
The Help Compiler does not create a Help file.

5059 **Not enough memory to build Help file.**
To free up memory, unload any unneeded applications, device drivers, and memory-resident programs.

5075 **Help Compiler corrupted. Please reinstall HC31.EXE.**
The virus checking code has detected a corruption in the Help Compiler.
Reinstall the Help Compiler from the original source disk.

5098 **Using old key-phrase table.**
Maximum compression can only result by deleting the .PH file before each recompilation of the Help topics or by setting the **OLDKEYPHRASE** option to **0.**

5115 **Write failed.**
Write to disk failed.
Contact Microsoft Product Support Services.

Compiler Errors 95

The following compiler errors are for Help compiler version 4.0 and later.

Message numbers	Type of errors
1000-1024	Notes
3000-3079	Warnings
3080-4014	Warnings 2
5000-6008	Errors

Notes

1000 Note: A keyword footnote has been specified without a keyword.

Problem: There are no characters in the keyword footnote in the topic (.rtf) files.
Result: Help Workshop ignores the keyword footnote. If there are no additional keyword footnotes, the topic will be inaccessible from the Index tab in the Help Topics dialog box.
Solution: Add one or more keywords to the keyword footnote, and then recompile.

1001 Note: The keyword [ ] is already defined in this topic.

Problem: The same keyword has be defined more than once in a topic.
Result: Help Compiler ignores the duplicate keyword.
Solution: Open the topic and remove the duplicate.

1002 Note: Using existing phrase table: [ ].

Problem: Help Compiler found an existing phrase (.ph) table for the Help file you are building, and the project (.hpj) file does not specify not to use the table.
Result: Help Compiler uses the phrase table that was generated from an earlier build instead of creating a new table.
Solution: To achieve the best phrase compression when compiling your Help file, you must delete the old key-phrase table before each recompilation, or clear the "Use old phrase file" option in the Compression tab in the Options dialog box of Help Magician.

1003 Note: A paragraph mark is formatted as a hidden character.

Problem: A paragraph mark has been formatted as hidden text. This often occurs when a hotspot is the last word in a paragraph.
Result: Help Compiler ignores the paragraph mark, so the two paragraphs will run together.
Solution: Reformat the paragraph marker as plain text in the topic file, and then recompile.

1004 Note: A previous instance of [ ] does not contain [ ].

Problem: A {bmx} command specifying only one resolution version of a bitmap is followed later in the topic files by a {bmx} command specifying that version plus one or more additional versions.
Result: WinHelp will treat the earlier single-resolution command as if it were the same as the command specifying multiple resolutions.
Solution: Replace the single resolution command with a copy of the multiple-resolution command.

1005 Note: [ ] is not an unsigned number for the macro [ ]. The sign will be ignored.

Problem: The parameter for this macro should be unsigned, but the number has been prefixed with a minus character.
Result: Help Compiler will remove the minus sign when it creates the Help file.
Solution: Remove the minus sign from the macro definition, and then recompile.

1006 Note: The ExecProgram macro has been used instead of ExecFile.

Problem: The ExecProgram macro has been used.
Result: The ExecProgram macro ignores the restricted programs that can be set in Windows 95. It will not search the registry from programs, so some programs cannot be launched with this command.

Solution: Replace instances of ExecProgram macro commands with ExecFile macro commands, and then recompile.

1007 Note: The include tag [ ] has been specified in the [EXCLUDE] section.

Problem: You have specified the same tag in the [EXCLUDE] section as you have in the [INCLUDE] section.
Result: Help Compiler ignores the tag in the [EXCLUDE] section.
Solution: Remove the tag from either the [EXCLUDE] or the [INCLUDE] section.

1008 Note: Ignoring the transparent flag on the monochrome bitmap [ ].

Problem: The transparent flag has been specified for a monochrome bitmap. It may only be used with a 16-color bitmap.
Result: Help Compiler ignores the flag.
Solution: Remove the transparent flag.

1009 Note: The map entry for [ ] has text after the number: [ ].

Problem: Text occurs after the numeric value specified in a map entry.
Result: Help Compiler ignores the text following the number in the entry.
Solution: Use Help Magician to make sure the map entry is correctly entered in the project file. (Comments must be preceded by a semicolon, which Help Magician adds automatically.) Recompile after making any changes.

1010 Note: The following mapped topic IDs are not used in any topic: [ ].

Problem: Entries appear in the [MAP] section for which you have no topics.
Result: If a program tries to use these mapped IDs, WinHelp will display an error message.
Solution: Check the listed entries to make sure they were not mistyped. Remove any IDs that are not actually used, and add topics for those that are.

1011 Note: The right-justified table row style is not supported.

Problem: A table style has been specified that WinHelp does not support.
Result: Help Compiler ignores the style.
Solution: Remove the right-justified style.

1012 Note: Table cell borders are not supported.

Problem: A table format has been specified that WinHelp does not support.
Result: Help Compiler ignores table cell borders.
Solution: Remove table cell borders in the topic file(s), and then recompile.

1013 Note: The "[ ]" section is missing a right bracket.

Problem: A section in your project file begins with a left bracket, but does not end with a right bracket.
Result: Help Compiler ignores the section heading on this line. The following lines will be processed as if they are part of the previous section which may result in errors.
Solution: Insert the right section heading bracket or remove the line entirely, and then recompile.

1014 Note: The multikey value [ ] has already been defined.

Problem: A character used for indicating an alternate keyword table has been used already.

Result: Help Compiler ignores the duplicate keyword table assignment on this line in the project file.
Solution: Remove the duplicate character, and then recompile.

1015 Note: A page break is formatted as a hidden character.

Problem: A page break has been formatted as hidden text.
Result: Help Compiler ignores the page break. If the page break was used to separate two topics, the topics will appear as one.
Solution: Reformat the page break in the topic file as plain text, and then recompile.

1016 Note: A carriage return is formatted as a hidden character.

Problem: A carriage return has been formatted as hidden text.
Result: Help Compiler ignores the carriage return and does not create a new line.
Solution: Reformat the carriage return in the topic file as plain text, and then recompile.

1017 Note: A nonscrolling region crosses a page break.

Problem: The Keep With Next paragraph formatting used for specifying a non-scrolling region crosses a page break boundary.
Result: The topic beginning with the page break will have a nonscrolling region, whether or not one is intended.
Solution: Remove the Keep With Next paragraph formatting from the page break, and then recompile.

1019 Note: The Language option is obsolete. It should be replaced with the LCID option.

Problem: The LANGUAGE option has been used. This option has been made obsolete by the LCID option which should be used instead.
Result: Help Compiler will attempt to convert the LANGUAGE option into an appropriate LCID option.
Solution: Check the Compiler version in the Compiler Options dialog. If the problem persists, call technical support.

1024 Note: The [ ] option has been specified more than once.

Problem: The same option has been specified more than once.
Result: Help Compiler uses the last definition for the option and ignores the others.
Solution: Rebuild help file. If problem persists, call technical support.

Warnings (3000-3079)

3000 Warning: The keyword [ ] is longer than [ ] characters.

Problem: A single keyword exceeds 255 characters.
Result: Help Compiler ignores the keyword.
Solution: Check for a missing semicolon or shorten the length of the keyword string, and then recompile.

3001 Warning: The file [ ] does not contain a valid icon.

Problem: You have specified an icon using the ICON option in the [OPTIONS] section of your project file, however, the file does not contain a valid icon image.
Result: Help Compiler ignores icon.
Solution: Since WinHelp 4.0 doesn't use the icon anyway, simply remove the ICON option from your project file and recompile..

3002 Warning: [ ] is a number but should be a string for the macro [ ].

Problem: The parameter for the should be a string, but it starts with a numeric character.
Result: Help Compiler ignores the rest of the macro. WinHelp will not be able to run the macro.
Solution: If the parameter really is a string that begins with a number, then you must place the string in quotations marks (`').

3003 Warning: The macro name [ ] is invalid.

Problem: Help Compiler does not recognize the macro you have entered.
Result: WinHelp may not be able to run the macro.
Solution: If you meant to use a WinHelp macro, confirm that you have spelled it correctly. Note that macro names are always entered in English no matter what language the text of the topic file is in. If the macro is supplied by a dynamic-link library (DLL), make certain you have registered the macro name using the RegisterRoutine macro in the [CONFIG] section of your project file.

3004 Warning: The variable used in the macro [ ] does not match the macro's argument type.

Problem: You have used a string for a numeric parameter or a number for a string parameter.
Result: WinHelp will not be able to run the macro.
Solution: Replace the incorrect parameter types, and then recompile. If the macro is defined in a custom dynamic-link library (DLL), check the RegisterRoutine macro in the project file for the correct parameter types.

3005 Warning: The macro variable [ ] is undefined.

Problem: The specified macro contains a variable that is not recognized by Help Compiler.
Result: Help Compiler ignores the rest of the macro. WinHelp will not be able to run the macro.
Solution: Replace incorrect variables, and then recompile.
The following is a list of valid macro variables:

- hwndContext
- hwndApp
- qchPath
- qError
- TopicNo
- hfs
- coForeground

- coBackground

3006 Warning: Missing comma in authorable button command.

Problem: You have specified a {button } command without specifying a comma followed by the macro to run when the button is clicked.
Result: Help Compiler ignores the button command.
Solution: Look up the {button } command in Related Topics and correct the syntax.

3007 Warning: [ ] is an invalid display-state parameter for the macro [ ].

Problem: A non-supported display state has been specified for a macro that launches a program.
Result: Help Compiler ignores the rest of the macro. WinHelp may not be able to run the macro.
Solution: Look up the macro in the Related Topics, and use one of the documented display states.

3008 Warning: Missing double quotation mark in macro [ ].

Problem: There is an unmatched double quotation mark in the macro.
Result: Help Compiler ignores the rest of the macro. WinHelp cannot run the macro.
Solution: Edit the macro with syntax checking turned off and then recompile.

3009 Warning: Missing end quotation mark (') in macro [ ].

Problem: A start quotation mark has been specified without an end quotation mark.
Result: Help Compiler ignores the rest of the macro. WinHelp cannot run the macro.
Solution: Turn off syntax checking, add the missing single quotation mark to the macro definition and then recompile.

3010 Warning: The macro name [ ] is undefined.

Problem: The macro name you have used is not a valid WinHelp macro and has not been defined by a RegisterRoutine macro.
Result: Help Compiler ignores the rest of the macro. WinHelp cannot run the macro.
Solution: If this is a macro that calls a dynamic-link library (DLL) function, add the RegisterRoutine macro to the project file that defines this function name. Otherwise, you have probably misspelled the macro name.

3011 Warning: The accelerator [ ] has already been added.

Problem: The accelerator you have used in the AddAccelerator macro has already been used.
Result: Help Compiler ignores the rest of the macro. You may get a syntax error if WinHelp tries to run the macro.
Solution: Use another accelerator key. Using the same format for accelerators will help avoid duplication. For example, 'A', 65, and 0x41 are all the same accelerator.

3012 Warning: The macro [ ] does not include a window name.

Problem: The macro requires a window name, and you have not specified one.
Result: Help Compiler ignores the rest of the macro. WinHelp cannot run the macro.
Solution: Supply the window name in the macro.

3013 Error: [ ] is not a valid numeric parameter for the macro [ ].

Problem: The parameter for the specified macro can only be a numeric value or a macro that

returns a numeric value, and you have specified a string. Some macros accept a limited type of string parameter in place of a number, but if you do not use one of those values, you get an error.
Result: Help Compiler ignores the rest of the macro. WinHelp cannot run the macro.
Solution: Change the macro parameter in the topic file to a numeric value, and then recompile.

3014 Warning: Window name [ ] has already been used.

Problem: The same window type has been specified more than once in the [WINDOWS] section of the project file.
Result: Only the first definition of the window is used.
Solution: Check for a duplicate window definition in the Windows section of the Options menu item. Make the correction, and then recompile. If the problem persists, call technical support.

3015 Warning: This window position is invalid: [ ].

Problem: An invalid window position has been used. The window position in a window definition consists of four numbers that define the window's location on the screen and its width and height. The four numbers must be given in WinHelp's 1024-by-1024 coordinate system, and they must be enclosed in parentheses and separated by commas.
Result: Help Compiler ignores the window definition.
Solution: Adjust the position as necessary using the Windows Definition option.

3016 Warning: Invalid syntax for authorable button: [ ].

Problem: The syntax you have used for an authorable button is invalid.
Result: Help Compiler ignores the {button} statement.
Solution: Correct the syntax, and then recompile. Typically the problem occurs by leaving out a comma before the macro definition. The correct syntax for an authorable button is:
{button button name, macro()}

3017 Warning: Topic ID for [ ] macro contains invalid characters: [ ].

Problem: There is an invalid character in the topic ID.
Result: Help Compiler ignores the rest of the macro. WinHelp cannot run the macro.
Solution: Replace the invalid characters in the topic ID, and then recompile.

3018 Warning: In the project file, an #include statement is specified without a filename.

Problem: An #include line has been specified in the project file, but the line did not include a filename.
Result: Help Compiler ignores the #include line.
Solution: Either delete the line or add a correct filename to the statement, and then recompile.

3019 Warning: Missing quotation mark or parenthesis in the macro [ ].

Problem: There is an unmatched quotation mark or parenthesis.
Result: Help Compiler ignores the rest of the macro. WinHelp cannot run the macro.
Solution: Add the missing quotation or right parenthesis to the macro definition in the topic file and then recompile.

3021 Warning: Cannot find the bitmap [ ].

Problem: Help Compiler is unable to find the bitmap you have specified.
Result: Help Compiler creates a temporary bitmap containing the filename of the missing bitmap.
Solution: If the bitmap name is valid, add the path it is contained in to the list of bitmap paths

(under the Options menu item in the Paths dialog).

3022 Warning: The bitmap [ ] has already been used as a non-transparent bitmap.

Problem: A bitmap cannot be transparent in one instance and non-transparent in another.
Result: The bitmap will always be displayed as non-transparent.
Solution: Change either bitmap command so that they are both transparent or both non-transparent. Alternatively, you can create a new file for a bitmap.

3023 Warning: The bitmap [ ] has already been used as a transparent bitmap.

Problem: A bitmap cannot be transparent in one instance and non-transparent in another.
Result: The bitmap will always be displayed as transparent.
Solution: Change either bitmap command so that they are both transparent or both non-transparent. Alternatively, you can create a new file for a bitmap.

3024 Warning: The window name [ ] contains more than eight characters and has been truncated to [ ].

Problem: The window name is too long.
Result: Help Compiler truncates the name to 8 characters. If this results in a duplicate window name, you will get another warning message.
Solution: Verify the window caption in the Window Definitions dialog under the Options menu item and then recompile the project file. If the problem persists, call technical support.

3025 Warning: Jump to undefined topic ID: [ ].

Problem: No topic has been defined with the topic ID specified in the hotspot.
Result: WinHelp displays the "Topic not found" error message when the user chooses a hotspot with the unresolved topic ID.
Solution: Check for spelling errors in the hotspot topic ID.

3026 Warning: The topic ID [ ] has already been defined in topic [ ] in file [ ].

Problem: Another topic exists with an identical topic ID.
Result: Help Compiler ignores the duplicate topic ID, so the duplicate topic has no identifier.
Solution: Change the duplicate topic ID so that it is unique, and then recompile.

3027 Warning: The following alias line does not contain an '=' character separating the topic IDs: [ ].

Problem: An invalid line has been entered in the [ALIAS] section of the project file.
Result: Help Compiler ignores the alias.
Solution: Make the correction in the Help Magician and then recompile. If the problem persists, call technical support.

3028 Warning: The bitmap file [ ] is corrupted.

Problem: Either the file is corrupt, or it has not been saved as an uncompressed bitmap (.bmp) file.
Result: Help Compiler ignores this bitmap, and WinHelp displays the "Unable to display picture" message in the topic instead of the bitmap.
Solution: Replace the bitmap file with one that is not corrupt, or remove the bitmap reference in the topic file.

3029 Warning: The file format of the bitmap file [ ] is unrecognized or unsupported.

Problem: Either the file is corrupt, or it has not been saved as an uncompressed bitmap (.bmp) file.
Result: Help Compiler ignores this bitmap, and WinHelp displays the "Unable to display picture" message in the topic instead of the bitmap.
Solution: Convert the bitmap to a supported format, and then recompile.

3031 Warning: Help Compiler does not support compressed BMP files: [ ].

Problem: The bitmap was saved with RLE compression.
Result: Help Compiler ignores this bitmap, and WinHelp displays the "Unable to display picture" message in the topic instead of the bitmap.
Solution: Save the bitmap without compression.

3032 Warning: The bitmap file [ ] is truncated.

Problem: The size of the bitmap does not match the size of the file. The bitmap file is corrupt.
Result: Help Compiler ignores this bitmap, and WinHelp displays the "Unable to display picture" message in the topic instead of the bitmap.
Solution: Resave or recreate the specified bitmap file.

3033 Warning: The topic ID [ ] contains invalid characters.

Problem: There is an invalid character in the topic ID footnote.
Result: Help Compiler does not assign an identifier to the topic.
Solution: In the topic file, change the characters in the topic ID so that it is valid, and then recompile.

3034 Warning: Invalid parameter for TCard macro: [ ].

Problem: The parameter specified for the TCard macro is invalid.
Result: Help Compiler ignores the macro, and WinHelp returns an error when it tries to run the macro.
Solution: Correct the invalid TCard macro definition in the topic file, and then recompile.

3035 Warning: Map entry does not specify a numeric value: [ ].

Problem: Help Compiler expected a numeric value, and encountered text instead.
Result: Help Compiler ignores the entry.
Solution: Change the entry to a numeric value.

3037 Warning: The map value for [ ] is the same as the map value for [ ].

Problem: A context number in the [MAP] section of the project (.hpj) file was previously mapped to a different topic ID.
Result: Help Compiler ignores the line with the duplicate context number.
Solution: Verify that all context numbers in the project file are unique. Remove or reassign any duplicate context numbers, and then recompile.

3038 Warning: The following topic IDs were not defined in the [MAP] section of the project file.

Problem: Help Compiler cannot find the specified topic IDs listed in the [MAP] section of the project (.hpj) file.
Result: Help Compiler ignores the lines in the project file. If the program uses that number in the

WinHelp function, WinHelp displays the "Help topic not found" error message.
Solution: Verify that all topic IDs listed in the [MAP] section of the project file are assigned to topics included in the build and that the topic IDs are spelled correctly. Then recompile the project file.

3041 Warning: The phrase file [ ] exceeds 64K. Phrase compression turned off.

Problem: The phrase file is invalid and cannot be used.
Result: Help Compiler cannot perform phrase compression on the Help file.
Solution: Delete the phrase file, and then recompile. If the problem still occurs, switch to Hall or Maximum compression.

3042 Warning: The copyright string exceeds 255 characters. It has been truncated to the following: [ ].

Problem: The copyright string is too long.
Result: Help Compiler truncates the copyright string at 255 characters.
Solution: In Help Magician, edit the text in the Copyright information box on the Appearance dialog under the Options menu item, and then recompile.

3043 Warning: The title string exceeds 127 characters. It has been truncated to the following: [ ].

Problem: The title string is too long.
Result: Help Compiler truncates the title string.
Solution: In Help Magician, edit the text in the Help Title box on the Appearance dialog under the Options menu item, and then recompile.

3044 Warning: Hotspot exceeds [ ] characters: [ ].

Problem: The hidden text portion of the hotspot is too large.
Result: Help Compiler does not make the selected text or graphic a hotspot.
Solution: Shorten the hotspot in the topic file, and then recompile.

3045 Warning: Footnote exceeds [ ] characters: [ ].

Problem: The footnote is too large.
Result: Help Compiler ignores the footnote.
Solution: Reduce the length of the footnote text in the topic file, and then recompile. For some footnotes, such as the K-footnote, you can use several smaller footnotes instead of one larger footnote.

3046 Warning: The [ ] file is not an RTF (Rich Text Format) file. It appears to have been saved as a Microsoft Word document.

Problem: You have saved the topic file in Word format instead of RTF format.
Result: Help Compiler ignores the file.
Solution: Save the document as a Rich-Text Format file, and then recompile.

3047 Warning: No name was specified for the bitmap command.

Problem: The bitmap command ({bmc}, {bml}, or {bmr}) was specified in a topic without a bitmap filename.
Result: Help Compiler ignores the bitmap command.
Solution: In the topic file, specify a bitmap file to include, or remove the bitmap command. Then recompile.

3048 Warning: Nonscrolling region is defined after scrolling region.

Problem: The Keep With Next attribute was applied to a paragraph after a paragraph that did not have this attribute.
Result: Help Compiler ignores the Keep With Next attribute; the paragraph is treated as plain text and is displayed as part of the topic text displayed in the scrolling region of the topic.
Solution: Set the Keep With Next attribute for the first paragraph in the topic if you want a nonscrolling region, and remove the Keep With Next attribute from following paragraphs. Then recompile the Help file.

3049 Warning: Invalid topic IDs in hypergraphic: [ ].

Problem: A topic ID in the shed file contains invalid characters.
Result: The hotspot is not active in the Help file.
Solution: Edit the hypergraphic and ensure that the topic IDs for the hotspots are valid and spelled correctly, and then recompile.

3050 Warning: Missing topic IDs in hypergraphic: [ ].

Problem: A hotspot was specified in the shed file, but no actual topic ID was specified.
Result: The hotspot is not active in the compiled Help file.
Solution: Edit the hypergraphic and ensure that each hotspot calls a valid topic ID, and then recompile.

3051 Warning: Cannot jump to window [ ]. No windows have been defined in the project file.

Problem: There is no [WINDOWS] section in the project file, so you cannot jump to any window type other than main.
Result: The hotspot is not active in the Help file.
Solution: Use Help Magician to define the specified secondary window, and then recompile.

3052 Warning: The window name [ ] has not been defined in the project file.

Problem: You have attempted to display a topic in a window type that has not been defined in your project file.
Result: The hotspot is not active in the Help file.
Solution: Use Help Magician to define the specified secondary window, and then recompile.

3053 Warning: The project file contains more than 255 window definitions.

Problem: You have defined more than 255 window types in your project file.
Result: Help Compiler ignores the additional window definitions. If you attempt to jump to a topic in one of the window definitions beyond the first 255, you will get an error message about the window type not being defined.
Solution: Remove the extra window definitions from the project file, and then recompile.

3054 Warning: Build tag contains invalid characters. The following is invalid: [ ].

Problem: A build tag in the [BUILDTAGS] section contains something other than a alphanumeric character or the underscore (_) character.
Result: Help Compiler ignores the line containing invalid characters.
Solution: Edit the build tag so that it contains only valid characters. Make any necessary changes in topics containing the specified build tag, and then recompile. Alternatively, specify the build tags to include/exclude in the Build Tags tab of the Options dialog box. This form of

including/excluding topics places no restrictions on the characters, length, or number of build tags, and does not use the [BUILDTAGS] section.

3055 Warning: The build tag [ ] has already been used.

Problem: The build tag in the [BUILDTAGS] section of your project file has already been declared previously in the same section.
Result: Help Compiler uses the first build tag and ignores the duplicate build tag.
Solution: Remove (or rename) the duplicate build tags in the project file, and then recompile.

3056 Warning: The window name [ ] in the macro [ ] contains more than eight characters and has been truncated to [ ].

Problem: The window name you have specified in the macro is invalid due to its length.
Result: Help Compiler ignores the window definition in this macro.
Solution: Specify a valid window name in the macro, and then recompile.

3057 Warning: A hotspot is defined with a macro (using !), but the macro is not specified.

Problem: You started a hotspot with a '!' character, but did not follow it with a macro.
Result: Help Compiler ignores the macro call from the hotspot.
Solution: Specify a valid macro in the hotspot, or remove the '!' character from the hotspot definition. Make certain that all text from the '!' to the end of the macro is marked as hidden.

3058 Warning: Topic ID for hotspot contains invalid characters: [ ].

Problem: An invalid topic ID has been used in a hotspot.
Result: The hotspot is not active in the Help file.
Solution: Correct the topic ID and recompile.

3059 Warning: A hotspot is specified without a macro or topic ID.

Problem: A hotspot was specified without specifying either a topic ID or a macro.
Result: The hotspot is not active in the Help file.
Solution: Edit the hotspot in the topic file and ensure that the topic ID or macro for the hotspot is valid, spelled correctly, and styled as hidden text. Then recompile.

3060 Warning: Window name is specified for a pop-up jump.

Problem: You have specified a window name in a pop-up hotspot.
Result: The hotspot is not active in the Help file.
Solution: Remove the right angle bracket and secondary window name from the pop-up hotspot, or change the single underline to double underline (for a topic jump). Then recompile the project file.

3062 Warning: Window definition is missing an '=' character: [ ].

Problem: The window definition in the [WINDOWS] section is invalid.
Result: Help Compiler ignores the line.
Solution: Use Help Magician to edit the window definition, and then recompile.

3063 Warning: Window definition does not contain anything after the '=' character: [ ].

Problem: The window definition in the [WINDOWS] section is invalid.
Result: Help Compiler ignores the line.

Solution: Use Help Magician to edit the window definition, and then recompile.

3064 Warning: Window name is not defined before the '=' character: [ ].

Problem: The window definition in the [WINDOWS] section is invalid.
Result: Help Compiler ignores the line.
Solution: Use Help Magician to edit the window definition, and then recompile.

3065 Warning: A closing quotation mark () is missing in window caption: [ ].

Problem: The window definition in the [WINDOWS] section is invalid.
Result: Help Compiler ignores the window definition.
Solution: Use Help Magician to edit the window definition, and then recompile.

3066 Warning: Window caption contains more than 50 characters: [ ].

Problem: The window definition in the [WINDOWS] section contains a title that is too long.
Result: Help Compiler ignores the window definition.
Solution: Use Help Magician to edit the window definition, and then recompile.

3067 Warning: The [ ] command in the [OPTIONS] section of the project file does not specify a value after the '=' character.

Problem: You have specified an invalid command in the [OPTIONS] section of your project file.
Result: Help Compiler ignores the line.
Solution: Use Help Magician to edit the option, and then recompile.

3068 Warning: The following line in the [OPTIONS] section of the project file does not contain an '=' character: [ ].

Problem: You have specified an invalid command in the [OPTIONS] section of your project file.
Result: Help Compiler ignores the line.
Solution: Use Help Magician to edit the option, and then recompile.

3069 Warning: The [ ] option in the [OPTIONS] section of the project file is not recognized.

Problem: You have specified an invalid command in the [OPTIONS] section of your project file.
Result: Help Compiler ignores the line.
Solution: Use Help Magician to edit the option, and then recompile.

3070 Warning: the major browse string exceeds 50 characters: [ ].

Problem: The name to the left of the colon in a browse footnote contains too many characters.
Result: Help Compiler truncates the string to 50 characters. The browse sequence may be affected.
Solution: Shorten the browse string.

3071 Warning: One or more browse sequences are set, but browse buttons are not enabled in any window.

Problem: You have specified a browse footnote, but none of the windows defined for the Help file contains browse buttons.
Result: The browse information is stored in the Help file (taking up space) but the information can never be accessed by the user.
Solution: Either add browse buttons or remove all browse footnotes.

3072 Warning: The macro [ ] does not include a control panel name.

Problem: The macro expected a control panel name, but you didn'
t specify one.
Result: Help Compiler ignores the ControlPanel macro.
Solution: Specify a control panel name in the topic file macro, and then recompile.

3073 Warning: No section is defined for the line: [ ].

Problem: The section heading on the specified line is not complete, or the first entry in the project (.hpj) file is not a section heading.
Result: Help Compiler ignores the section heading on this line and all succeeding lines until it encounters a valid section heading.
Solution: Edit the specified section in the project file. Then recompile the project file.

3074 Warning: The [ ] section is not recognized by this version of Help Compiler.

Problem: Your project file contains a section that Help Compiler does not recognize.
Result: Help Compiler ignores the section heading.
Solution: Verify that the section headings in the project file are valid and spelled correctly, and then recompile.

3075 Warning: The [ ] section does not follow the [OPTIONS] section in the project file.

Problem: You have specified a section that requires the [OPTIONS] section before you have specified the [OPTIONS] section.
Result: Help Compiler may not be able to find some files, or the files may be compiled incorrectly.
Solution: Call technical support.

3079 Warning: The alias string [ ] has already been aliased [ ].

Problem: A topic ID can have only one alias.
Result: Help Compiler ignores this line.
Solution: Use Help Magician to correct the alias string mapping, and then recompile.

Warnings (3080-4014)

3080 Warning: The string [ ] has already been aliased [ ].

Problem: An alias string cannot be assigned another alias.
Result: Help Compiler ignores this line.
Solution: Use Help Magician to correct the alias string mapping, and then recompile.

3081 Warning: Both alias and topic ID are identical: [ ].

Problem: The alias is the same as the topic ID.
Result: Help Compiler ignores this line.
Solution: Use Help Magician to correct the alias string mapping, and then recompile.

3082 Warning: The map number [ ] is not a valid number: [ ].

Problem: A context number in the [MAP] section of the project (.hpj) file contains invalid characters.
Result: Help Compiler ignores this line.
Solution: Use Help Magician to correct the mapping, and then recompile.

3083 Warning: Invalid syntax for window color: [ ].

Problem: The syntax specified for the windows color in the [WINDOWS] section of your project file is invalid.
Result: Help Compiler ignores the window definition.
Solution: Use Help Magician to edit the window definition, and then recompile.

3084 Warning: Invalid window syntax: [ ].

Problem: The syntax you have specified for the window in the [WINDOWS] section of your project file is invalid.
Result: Help Compiler ignores the invalid window definition.
Solution: Use Help Magician to edit the window definition, and then recompile.

3086 Warning: Window position is out of range: [ ].

Problem: One or more of the window position coordinates exceed the limit of 1024.
Result: Help Compiler ignores the invalid window definition.
Solution: Use Help Magician to edit the window definition, and then recompile.

3087 Warning: The following filename exceeds 259 characters: [ ].

Problem: You have specified a filename that contains too many characters.
Result: Help Compiler ignores the specified file.
Solution: Shorten the filename. Then recompile the project file.

3088 Warning: More than 20 font ranges are mapped.

Problem: You have specified too many font ranges.
Result: Help Compiler ignores additional font ranges.
Solution: Remove the excess ranges, and then recompile. Alternatively, remove the font ranges and instead use the font substitution in the Fonts tab of the Compiler Options dialog box in Help Magician.

3089 Warning: Invalid font range: [ ].

Problem: The font range is invalid.
Result: Help Compiler ignores the option.
Solution: Edit the font range, and then recompile. Alternatively, remove the font ranges and instead use the font substitution in the Fonts tab of the Compiler Options dialog box in Help Magician.

3090 Warning: Current font range overlaps previously defined range: [ ].

Problem: Two or more font ranges overlap each other.
Result: Help Compiler uses the first range and ignores the second mapping.
Solution: Edit the font range, and then recompile. Alternatively, remove the font ranges and instead use the font substitution in the Fonts tab of the Compiler Options dialog box in Help Magician.

3091 Warning: Unrecognized forced font name: [ ].

Problem: You have tried to force all font names to an invalid font.
Result: Help Compiler ignores the font name and uses the default MS Sans Serif font.
Solution: Change the font name in the project file to a supported font, and then recompile. Alternatively, remove the font ranges and instead use the font substitution in the Fonts tab of the Compiler Options dialog box in Help Magician.

3092 Warning: This version of the project file is not supported by this Help compiler.

Problem: The project file was created with a more recent version of Help Compiler than you are currently using.
Result: Help Compiler may remove options or sections.
Solution: Get a more recent version of Help Compiler.

3093 Warning: Keyword type is not a letter or number. [ ] is invalid.

Problem: An invalid character has been used for the MULTIKEY option.
Result: Help Compiler ignores the attempted keyword table assignment on this line.
Solution: Edit the MULTIKEY option in the project file, and then recompile.

3094 Warning: More than three keyword types (besides 'K' and 'A') defined.

Problem: More than three keywords have been defined using the MULTIKEY option.
Result: Help Compiler ignores the additional keyword types.
Solution: Remove the extra MULTIKEY options from the project file, and then recompile.

3095 Warning: Multikey values of 'K' or 'A' cannot be defined.

Problem: The MULTIKEY option has been used to specify either '
K' or 'A' as a multikey value. These characters are reserved by Help Compiler and WinHelp.
Result: Help Compiler ignores the attempted keyword table assignment on this line.
Solution: Edit the MULTIKEY option in the project file, and then recompile.

3096 Warning: The font name [ ] is longer than 31 characters.

Problem: The font name is invalid because it is too long.
Result: Help Compiler ignores the option on this line.

Solution: Use Help Magician to shorten the font name, and then recompile.

3097 Warning: The [BUILDTAGS] section is missing from the project file.

Problem: A BUILD option was specified in the project file, but no [BUILDTAGS] section was specified.
Result: Help Compiler includes all topics in the build.
Solution: Add a build section to the project file, and then recompile. Alternatively, specify the topics to include or exclude using the Build Tags Option dialog box of Help Magician.

3098 Warning: Invalid build tag expression.

Problem: The BUILD option specifies an invalid expression.
Result: Help Compiler ignores the line with the invalid expression.
Solution: Edit the build tags in the project file, and then recompile. Alternatively, specify the topics to include or exclude using the Build Tags Option dialog box of Help Magician.

3099 Warning: Build expression too complex.

Problem: The build expression in the project (.hpj) file has too many expressions or is nested too deeply.
Result: Help Compiler includes all topics in the build.
Solution: Edit the build expression in the project file, and then recompile. Alternatively, specify the topics to include or exclude using the Build Tags Option dialog box of Help Magician.

3101 Warning: Unknown build error.

Problem: Help Compiler was not able to interpret the BUILD option correctly.
Result: Help Compiler includes all topics in the build.
Solution: Edit the build expression in the project file, and then recompile. Alternatively, specify the topics to include or exclude using the Build Tags Option dialog box of Help Magician.

3102 Warning: No macro is specified for the entry macro footnote.

Problem: You have specified an entry macro footnote, but the footnote did not contain any macros.
Result: Help Compiler ignores the entry macro footnote.
Solution: Add a macro to the entry macro footnote or remove the footnote marker. Then recompile the Help file.

3103 Warning: The entry macro footnote does not precede text.

Problem: The entry macro footnote was specified after text was specified for the topic.
Result: Help Compiler ignores the entry macro footnote.
Solution: Move the entry macro footnote to the beginning of the topic, and then recompile.

3104 Warning: The topic ID footnote (#) does not specify a topic ID.

Problem: You have a specified a topic ID footnote that does not contain any characters.
Result: Help Compiler does not assign the topic an identifier.
Solution: Add a valid topic ID to the footnote, and then recompile.

3105 Warning: The minor browse string exceeds 50 characters: [ ].

Problem: The browse string to the right of the colon is larger than 50 characters.

Result: Help Compiler truncates the string which may affect the browse sequence itself.
Solution: Shorten the browse string in the appropriate topics.

3106 Warning: Browse footnote (+) does not appear before any text.

Problem: The browse sequence footnote (+) does not precede the first paragraph of the topic.
Result: Help Compiler ignores the browse sequence footnote.
Solution: Move the browse sequence footnote so that is at the beginning of the topic, and then recompile.

3107 Warning: A browse sequence has already been defined for this topic.

Problem: The browse sequence footnote (+) has been specified twice in the same topic.
Result: Help Compiler uses the first browse sequence footnote and ignores the duplicate footnote.
Solution: Remove the duplicate browse sequence footnote from the topic, and then recompile.

3108 Warning: The title footnote ($) does not appear before any text.

Problem: The title footnote does not precede the first paragraph of the topic.
Result: Help Compiler ignores the title footnote, and the topic does not have any title.
Solution: Move the title footnote to the beginning of the topic, and then recompile.

3109 Warning: The title footnote ($) does not contain any text.

Problem: A title footnote was specified that does not contain any text.
Result: Help Compiler ignores the title footnote, and the topic does not have any title.
Solution: Add a title string next to the topic title footnote, and then recompile.

3110 Warning: A title has already been defined for this topic.

Problem: A title footnote has already been specified for the topic.
Result: Help Compiler uses the first title footnote and ignores the duplicate title footnote.
Solution: Remove the second title footnote from the topic, and then recompile.

3111 Warning: This topic contains keywords but no title.

Problem: One or more keywords have been specified for the topic, but a title footnote was not specified.
Result: If no title is specified for the topic, and the keyword appears in more than one topic (including other Help files), WinHelp will display "Untitled topic #n" in the Topics Found dialog box.
Solution: If this is not a pop-up topic, add a title using the title footnote ($), and then recompile. If the topic will only be displayed in a pop-up window, remove the keyword footnotes.

3112 Warning: The build footnote (*) is not the first footnote in the topic.

Problem: The build footnote does not precede all other text and footnotes in the topic.
Result: Help Compiler ignores the build tag footnote, and the topic is not assigned a build tag.
Solution: Move the build tag footnote to the very beginning of the topic, before any other footnotes, and then recompile.

3113 Warning: Build tag is longer than 32 characters: [ ].

Problem: The build tag is too long when used in conjunction with the BUILD option in the project file.

Result: Help Compiler ignores the build tag assigned to the topic.
Solution: Shorten the build tag to 32 or fewer characters, and then recompile. Alternatively, specify the build tags to include/exclude in the Build Tags tab of the Options dialog box. This form of including/excluding topics places no restrictions on the characters, length, or number of build tags.

3114 Warning: The bitmap [ ] has been used as part of another {bmx} command.

Problem: Two {bmx} commands that specify bitmaps for multiple color depths have a bitmap in common but are otherwise different or in a different order.
Result: WinHelp will treat the shared bitmap files as separate files, and this effectively doubles the space required for each of the specified bitmaps.
Solution: Make sure that bitmap commands that are intended to be identical include the same bitmaps in the same order.

3116 Warning: Table has more than [ ] columns.

Problem: The table has more columns than WinHelp is able to display.
Result: Help Compiler treats the additional columns as one table cell.
Solution: Reduce the number of table columns, and then recompile.

3117 Warning: Side-by-side paragraphs are not supported.

Problem: Side-by-side paragraph formatting has been used.
Result: Help Compiler ignores the side-by-side paragraph formatting.
Solution: Convert the side-by-side paragraphs to a table or a tabbed paragraph, and then recompile.

3118 Warning: Unrecognized RTF (Rich Text Format) graphics format.

Problem: An embedded bitmap contains a format that Help Compiler does not recognize. Help Compiler supports only Windows bitmaps (.bmp), Windows device-independent bitmaps (.dib), and Windows metafiles (.wmf)
Result: Help Compiler ignores the specified graphic.
Solution: Convert the graphic to a supported format, and then recompile.

3119 Warning: Hash conflict between [ ] and [ ]. One of these topic IDs must be changed.

Problem: The hash algorithm used by Help Compiler to convert a topic ID into an internal number that WinHelp uses, has generated the same hash value for both of the listed topic IDs.
Result: WinHelp displays the topic with the first topic ID whenever the user chooses a hotspot containing either topic ID.
Solution: Change one of the specified topic IDs, and then recompile.

3120 Warning: The LANGUAGE option [ ] is not supported.

Problem: The LANGUAGE option has been specified. This option is no longer supported.
Result: Help Compiler uses the default English (U.S.) sorting order.
Solution: Call technical support.

3121 Warning: Invalid version of Ftsrch.dll.

Problem: The version of Ftsrch.dll on your system is invalid.
Result: Neither Hall nor Maximum compression can be used and no index file is created for full text search.

Solution: Reinstall Ftsrch.dll from your Windows 95 or Windows NT installation disks, and then recompile.

3123 Warning: Missing '=' character in REPLACE option: [ ].

Problem: The syntax for the REPLACE option is invalid.
Result: Help Compiler ignores the entire option.
Solution: Call technical support.

3124 Warning: The TMP folder [ ] is invalid.

Problem: The folder name is invalid.
Result: Help Compiler uses the default folder for temporary files.
Solution: Use the Paths dialog box under the Options menu item to edit the folder name.

3126 Warning: The window (>) footnote does not appear after the topic ID (#) footnote.

Problem: The window footnote was placed before the topic ID footnote.
Result: Help Compiler ignores the window footnote.
Solution: Reposition the window footnote after the topic ID footnote, and then recompile.

4001 Warning: Cannot find or load Ftsrch.dll.

Problem: Help Compiler cannot find the Ftsrch.dll file.
Result: Neither Maximum nor Hall compression can be used and no index file is created for full text search.
Solution: Reinstall Ftsrch.dll from your Windows 95 or Windows NT installation disks, and then recompile.

4002 Warning: The RTF file [ ] is corrupted at offset [ ].

Problem: Help Compiler encountered an error while processing the topic (.rtf) file. This problem can occur when the number of opening and closing braces don't match, when the end of the file is reached before all columns of a table have been specified, or when the number specified for an RTF token is invalid.
Result: Help Compiler ignores the rest of the topic file.
Solution: Open the file and save it again as Rich Text Format (RTF), and then recompile. If the problem persists, look for errors in the topic file.

4003 Warning: An error occurred attempting to read the file [ ].

Problem: Help Compiler can not read the specified file. Another program may have locked the file, or if the file is on a network, the network may be down.
Result: Help Compiler ignores the file.
Solution: Check file and its path. Correct errors, and then recompile.

4004 Warning: File is not an RTF (Rich Text Format) file.

Problem: The specified file has not been saved as an RTF file.
Result: Help Compiler ignores the file.
Solution: Resave the file as RTF, and then recompile.

4005 Warning: The [ ] Help file has not been created.

Either there are no topics to compile, or the build expression is incorrect for all topics.

Problem: Help Compiler reached the end of all topic files without encountering any text.
Result: Help Compiler does not create a Help file.
Solution: Confirm that you have specified at least one topic file that contains text. If all topic contain build tags, you must specify at least one build tag to include (use the Build Tags Option dialog box of Help Magician).

4006 Warning: The folder [ ] specified for the [ ] option does not exist.

Problem: A nonexistent folder has been specified. Either the folder name was specified incorrectly, or it points to a network location that does not (currently) exist.
Result: Help Compiler uses the current working folder.
Solution: Use Help Magician to correct the folder name for the specified option, and then recompile.

4007 Warning: The [ ] option [ ] is not a valid value.

Problem: The value specified for the option is invalid.
Result: Help Compiler ignored the option.
Solution: Call technical support.

4008 Warning: Cannot find or load Ftsrch.dll. Hall compression turned off.

Problem: Help Compiler either cannot find the Ftsrch.dll file, or it is out of date or corrupted.
Result: Neither Maximum nor Hall compression can be used and no index file is created for full text search.
Solution: Reinstall Ftsrch.dll from Windows 95 or Windows NT installation disks, and then recompile.

4009 Warning: The topic ID specified in the project file as the default topic does not exist.

Problem: You have specified a default topic ID in the project file, but none of your topics contains that topic ID.
Result: Help Compiler uses the first topic in the build as the default topic.
Solution: Verify that the topic ID exists in the desired topic and that it is spelled correctly, and that the topic file is included in the [FILES] section of the project file, and then recompile.

4010 Warning: There are more opening braces than closing braces.

Problem: When Help Compiler reached the end of the file, the number of opening and closing braces did not match.
Result: The Help file may be corrupt.
Solution: Load and resave the topic (.rtf) file, and then recompile. If the problem persists, you will need to edit the topic file directly.

4011 Warning: There are 20 opening braces without intervening closing braces.

Problem: More opening braces without matching closing braces have been encountered than should ever occur in a normal topic (.rtf) file.
Result: The rest of the file is ignored.
Solution: Load and resave the topic file, and then recompile. If the problem persists, you will need to edit the topic file directly.

4012 Warning: The full-text search index cannot be created because neither Phrase nor Hall compression has been selected.

Problem: In order to create a full-text search index, you must compile with either Maximum or Hall compression.
Result: Help Compiler is unable to generate an full-text search (.fts) index file.
Solution: Edit the Compression Options in the Compiler Options dialog under the Options menu item and recompile.

4013 Warning: Invalid default font number in [ ]. Using [ ] as the default font.

Problem: The RTF token \deff specifies as a default font a font number that is not defined in the \fonttbl section.
Result: Help Compiler will ignore the specification and use the first font declared in the \fonttbl section as the default font.
Solution: In the topic (.rtf) file change the number after the \deff token to specify a font declared in the \fonttbl section. Then recompile.

4014 Warning: There is text after the closing brace in the RTF file.

Problem: Help Compiler encountered the closing brace that should signify the end of the topic (.rtf) file, and then encountered additional text.
Result: Help Compiler adds the text to the Help file, but the text cannot be accessed.
Solution: In the raw topic file, move or delete the text after the closing brace, and then recompile.

Errors

5000 Error: The filename [ ] is too long.

Problem: The specified filename is too long.
Result: The Help file is not created.
Solution: Use Help Magician to shorten the path, and then recompile.

5001 Error: File is a folder, not a file.

Problem: You have specified a filename, but the name specified is actually a folder.
Result: The Help file is not created.
Solution: Move or rename the folder or the file, and then recompile.

5002 Error: The text for the button in the CreateButton macro is too long.

Problem: You have specified more than 96 characters in the CreateButton macro.
Result: The Help file is not created.
Solution: Reduce to 96 or less the number of characters in the CreateButton macro, and then recompile.

5003 Error: Permission to open the file [ ] is denied. Another program has probably locked the file.

Problem: You do not have the required file privileges to open the requested file. This can happen if the file has been opened by another program. For example, if WinHelp has the Help file open, Help Compiler is not be able to write to that Help file.
Result: The Help file is not created.
Solution: If the file has been loaded in another program, switch to that program and either close the file or close the program. If you are trying to write a file to a networked drive, make certain you have write permission on that server.

5005 Error: The file [ ] cannot be found.

Problem: The specified file cannot be located.
Result: The Help file is not created.
Solution: Verify that you have specified the correct filename and that the file exists in the correct folder. Then recompile the project file.

5006 Error: Invalid RTF tokens for a table.

Problem: A \row command appeared before a \cell command in the topic (.rtf) file.
Result: The Help file is not created.
Solution: Change the order of commands in the topic file, and then recompile.

5007 Error: Invalid use of the WinHelp menu [ ] in the [ ] macro..

Problem: You have attempted to use a standard WinHelp menu ID.
Result: The Help file is not created.
Solution: Use a menu ID created with a call to the ExtInsertItem, ExtInsertMenu, or InsertMenu macros.

5009 Error: [ ] does not contain a comma separating the parameters in the macro [ ].

Problem: Either a parameter has been left out of the macro or parameters are not separated with a comma.
Result: The Help file is not created.
Solution: Add the missing comma or parameter to the macro in the topic file, and then recompile.

5010 Error: [ ] is an invalid parameter for the macro [ ].

Problem: You cannot use a standard WinHelp menu as a parameter for this macro.
Result: The Help file is not created.
Solution: Use the name of your own custom menu.

5011 Error: Cannot open the file [ ].

Problem: The file cannot be opened. This can occur when another program has the file locked, or the file is on a network and the network is down.
Result: The Help file is not created.
Solution: If another program has the file open, close the file. In some cases, it may be necessary to close the program.

5012 Error: Too many nested #include files. Cannot include the file [ ].

Problem: A file specified in a #include statement can contain an included file which can in turn contain an included file, up to five levels. This limit has been exceeded.
Result: The Help file is not created.
Solution: Merge the contents of the deepest #include file into its parent, and then recompile.

5013 Error: Invalid DBCS escape sequence: [ ].

Problem: The DBCS character is invalid.
Result: The Help file is not created.
Solution: If this is not a DBCS topic file, change the language of the Help file. Otherwise you will need to use a different word that does not use this DBCS character.

6000 Error: Help Compiler is out of memory. If you have any other programs running, close them and try compiling again.

Problem: There is insufficient memory and swap file space.
Result: The Help file is not created.
Solution: If you are running Windows 95, increase the amount of hard disk space available on the drive Windows is installed on. Otherwise, increase the size of your permanent swap file.

6001 Error: Out of disk space writing to the temporary file [ ]. Free up disk space on this drive or change your TMP environment variable.

Problem: There is insufficient space on the drive where Help Compiler is creating temporary files.
Result: The Help file is not created.
Solution: Either free space on the specified drive, or tell Help Magician to create temporary files on a different drive (or network connection).

6002 Error: An error occurred while reading the file [ ].

Problem: Help Compiler was not able to read the file. The file may be corrupt.
Result: The Help file is not created.
Solution: Confirm that the file is valid, replacing if not. If the file appears to be valid, run scandisk to verify that your hard disk does not have problems.

6003 Error: Out of file handles. Increase the FILES= line in CONFIG.SYS.

Problem: There are not enough file handles available.
Result: The Help file is not created.
Solution: If possible, increase the FILES setting in the CONFIG.SYS file on your computer to FILES=50 or greater. Reboot your computer, then recompile.

6004 Error: The file [ ] is a read-only file.

Problem: Help Compiler cannot write to the specified file.
Result: The Help file is not created.
Solution: Change the file's read-only attribute if you want Help Magician to overwrite the file. Otherwise, rename the file and try again to compile the project file.

6005 Error: [ ] is a folder, not a file.

Problem: The name you have specified for a file is actually a folder.
Result: The Help file is not created.
Solution: Move or rename the folder or the Help file, and then recompile.

6006 Error: [ ] is a device name and cannot be used as a filename.

Problem: The name you have specified for a file is a device.
Result: The Help file is not created.
Solution: Change the name to a file, not a device, and then recompile.

6007 Error: Cannot write to the file [ ].

Problem: An error occurred while Help Compiler was writing to a file.
Result: The Help file is not created.
Solution: Run scandisk to fix any problems on your hard drive. Or trying compiling on a different drive or network connection.

6008 Error: No files have been specified in the [FILES] section of the project file.

Problem: The [FILES] section of your project file did not specify any files.
Result: The Help file is not created.
Solution: Call technical support.

Close **Glossary**

A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

A

Alignment
AVI

B

Baggage
Bitmap
BMROOT
Browse
Browse definition
Browse group
Browse sequence
Build
Build Tags
Bulleted List

C

Context Number
Context-Sensitive Help
Context-Sensitive Topic
Context String

D

Default Font

E

Environment Options

F

File Extensions
Font
Font Attributes

G

Glossary

H

Help Compiler
Help Editor
Help File
Help System
Help Topic
Hot Link
Hot Spot
Hypergraphic

I

Index Page

J

Jump

K

Keyword

L

Link
Line Spacing

M

Macro
Macro Definition
Macro Definitions File (.HLM)
Map
Multiple Keywords

N

Numbered List

O

Options

P

Phrase Table
Popup
Project
Project Macro

Q

Quick Button

R

ROOT Directory
RTF

S

Secondary Window
Segmented Hypergraphics
Status Bar

T

Test Mode
Topic
Topic Heading
Topic Macro
Topic ID

U

Utility Area

V

VB Help Wizard

W

Web Authoring
WinHelp.Exe
WMF

X

Y

Z

Bitmap

A bitmap is an array of bits where one or more bits corresponds to each display pixel. It is a graphic image or picture stored as a series of numbers.

BMROOT

The directory list where one or more of the bitmaps used in the Help Magician may be stored.

Browse

The function in Windows Help to move back and forth between related topics.

Browse Definition

A browse group or groups that have been assigned user defined titles and have had the topics in each group logically ordered by the user.

Browse Group

A series of related topics that are assigned to a user defined name and will be used as a browse sequence when run in Windows Help.

Browse Sequence

A user defined ordered group of topics that will be used for browsing related topics in the final Windows Help system.

Build

The automated process of creating a help file. The steps include creating an RTF file from the help text file, compiling the RTF file into a .HLP file and reporting any errors that occurred during compilation.

Bulleted List

An indented list of sentences that have bullets (black marks) at the beginning each paragraph. Used to attract attention to important points, outline a list, etc.

Context Number

A number that the user may assign to help topics that they want to call directly from their application as context-sensitive help.

Context-Sensitive Help

The ability in the user's application to directly call a specific help topic.

Context-Sensitive Topic

A help topic that has a context number assigned to it and is intended by the user to be called from their application.

Context String

A unique alphanumeric string that is assigned to every help topic created in the Help Magician. Context strings are noew refered to as Topic ID's in the Windows 95 environment.

File Extensions

The various DOS file extensions that are used in this program are:

.BMP Bitmap file.
.CNT A Windows 95 Contents file.
.ICO Icon file.
.HLD Help Magician macro prototypes file.
.HLK Help Magician backup file extension.
.HLM Help Magician macro definitions file.
.HLP The standard Windows Help file.
.HLS Help Magician database of font styles.
.HLX Help Magician text file.
.HMP Help Magician Project file for Project Management Mode.
.HPJ Help Magician project file.
.HTM An HTML file for use in Web Authoring.
.INI Help Magician configuration file, created during installation.
.ISD, .ISM, .ISF, .ISL Topic/Project Database files.
.KSD, .KSM, .KSF, .KSL Keyword Database files.
.PH Phrase-table created during compile time with the COMPRESS option on.
.RPT Help Magician RTF error report file. Created while importing an RTF file.
.RTF Rich Text Format. Help compiler requires this to create a help file.
.SHG Hypergraphics file produced by the SHED utility.

Font

An assortment of characters all of one size and type, i.e. "Helvetica, 12 point, bold, italic, blue".

Font Attributes

Typeface, size, boldness, color, italic.

Help Compiler

A compiler supplied by Microsoft that will produce a Windows Help file (.HLP) from an RTF file.

Help Editor

A specialized text editor that provides features to create a Windows Help file and test the final help system. The Help Magician is considered to be a Help editor or Help Authoring Tool.

Help File

When referred to as a help file, this refers to the Help Magicians .HLX editor file. When referred to as a Windows help file, this refers to the Windows .HLP file.

Help System

A compiled help file, with a .HLP extension, viewed with WINHELP.EXE.

Help Topic

A user defined help subject that, when viewed in the Windows Help system, will aid the end user in understanding a particular subject or subjects pertaining to the application.

Hot Link

A Jump or Popup that could be a word, phrase or bitmap in the final help system. The word or phrase will be underlined for Jumps, dotted underlined for Popups, and colored green in the final Windows Help system. Hot Link bitmaps, or hyper-graphics are only supported in Windows Help 3.1.

Hot Spot

A portion of a bitmap that is defined as a hot link.

Hypergraphic

A hot link that is a bitmap. This is accomplished by entering a bitmap in the editor and then selecting the bitmap and creating a Jump or Popup markers from the Markers menu. This is a feature of Windows Help 3.1.

Index Page

The user defined "table of contents" or index page for Windows Help. This option can be set in Compiler Options.

Jump

A link between a word or phrase in one topic to a different help topic. In Windows Help, all Jumps will be green and underlined and when selected will display or "Jump" to another help topic.

Keyword

A word or phrase that is intended to aide the end user in finding help on a particular topic or topics. Each topic in the editor may have one or more keywords assigned to it that describe the subject material of that topic. Keywords are used by Windows Help during a Search and are listed for the user to quickly locate a general subject and then list the available related help topics.

Macro

A sort of "mini program" that executes in WinHelp and can control or 'program' the WinHelp environment.

Macro Definition

The name or identifier assigned to one or more macros. A group of macros.

Macro Definitions File (.HLM)

A file created by the Help Magician that contains macro definitions that may be used in other Help Magician projects.

Map

The part of the Project File where the context strings for each topic are listed with their corresponding context numbers. This is useful for designing applications requiring context-sensitive help and/or Hot Spots.

Multiple Keywords

Treat these the same as keywords except that they are used for alternate keyword lists. One possible example might be used in an editor where topics other than your own would be looked up.

Numbered List

An indented list of sentences that may have combinations of numbers or letters at the beginning each paragraph. Used to create a list of important points or subjects, outlining, etc.

Phrase Table

A file (.PH) that the help compiler generates when compressing a help file. It contains a table of duplicate phrases found throughout the help file.

Popup

A link between a word or phrase in one topic to a definition of that word or phrase. The definition should reside in its own separate help page, but does not require a context number since it would never be called directly for context-sensitive help. In Windows Help, all Popups will be green and dotted underlined, and when selected will display or "Popup" the definition for the highlighted word or phrase.

Project

Each separate help system will be viewed as a "project". A project file will be generated when an RTF file is created and used to provide option settings, file and path names, etc. to the help compiler. See Project Files and WinHelp() for more details on the Project File.

Project Macro

A macro definition that has been assigned to the whole project. The macros assigned to the macro definition will be executed in order upon loading the help system via WinHelp.

ROOT Directory

The path where all the help files can be found, with the exception of bitmaps, which may be located in the BMROOT directories.

RTF

Rich Text Format. The format of the intermediate file that the Help Magician creates for the help compiler to generate the Windows Help file.

Secondary Window

A Help window that can be displayed in WinHelp to show topics other than the ones displayed in the WinHelp main window. Only one secondary window may be displayed at a time.

Segmented Hypergraphics

Another name for a bitmap that has had several Hot Spots assigned to various areas of itself. This is the same as a hypergraphic bitmap, with the difference being several areas of one bitmap can represent different Jumps or Popups. These are created in the SHED bitmap editor supplied by Microsoft and is a feature of Windows Help 3.1.

Status Bar

The region directly below the text window in the Help Magician where various status' are displayed. For instance the current font style that the cursor is position over in the help text is displayed here.

Test Mode

Simulates the Windows Help system within the Help Magician without having to compile the help file.

Topic

Any discrete unit of information, such as a topic screen, a conceptual description, a set of instructions, a keyboard display, a glossary definition, a list of Jumps, a picture, etc. Within the Help Magician, a topic is also considered a page.

Topic Heading

This is the Title of the topic page. The topic headings or titles are used for reference in Jumps, Popups, etc. The context strings are internally built from these titles.

Topic Macro

A macro definition that has been assigned to a particular topic. The macros assigned to the macro definition will be executed in order anytime the topic is displayed in WinHelp.

Utility Area

This is the region above the main window in the Help Magician bar where the user may input keywords, multiple keywords, context numbers, etc.

WINHELP.EXE

The Windows Help system program, supplied by Microsoft with all copies of Windows 3.x. The user may call WinHelp() routine from the Help Magician to view the compiled Windows Help file, just like it would be from their own application.

Alignment

A paragraph property that determines the let, right, or center justification of the text.

AVI
A Microsoft video file.

Baggage

A term used to specify whether the contents of audio and/or video files should be embedded into the compiled help file or referenced externally.

Build Tags

Words assigned to topics that will be referenced by a Build Expression to determine if topics are to be included in the build.

Default Font

The font characteristics of text that is not assigned a paragraph tag (or style) or other character formatting.

Environment Options

Options specific to the environment of the Help Magician and its user interface.

Glossary

This Window! A list of terms with Popups supplying the definitions.

Options

In the Help Magician, the Options menu provides dialogs to set preferences for the help file and the Help Magician environment.

VB Help Wizard

A Help Magician utility that reads VB source code and creates a help file shell.

WMF
A form of graphic image, Windows Meta File.

Link

A term referring to the link between two topics where one topic Jumps or Pops up to another topic.

Line Spacing

A paragraph property that refers to the spacing before, between, and after lines of text.

Quick Button

The ***Create Button*** option available from the ***Macros*** menu.

Topic ID

See context string.

Web Authoring

Writing pages for display on the Internet.

Rabbit Bitmap
Courtesy of Harald Zoschke

Topic Map

- Table of Contents
- Introduction
  - Function Keys
  - Keyboard
  - System Requirements
  - Technical Info
  - Version 4.0 - New Features
- Getting Started
- Acknowledgements
- Alternate Keywords
- Appearance Options
- Authorable Button
- Build Tag Manager
- Building Your Help File
  - Build Context Spy
  - Call WINHELP.EXE
  - One Page Preview
  - Project [ALIAS] Section
  - Rebuild All
  - Run Compiler
  - View Compiler Messages
  - Write RTF for Compiler
- Call Word Processor
  - Importing RTF Files
  - RTF Technical Specifications
- Calling WinHelp()
  - Examples
  - HELP Context ID Property
  - Visual Basic
- Calling WinHelp Version 4.0
  - #defines
  - fuCommand Parameters
  - The HELPINFO Structure
  - The HELPWININFO Structure
  - The MULTIKEYHELP Structure
  - The WM TCARD Message
- Character Attributes
  - Alignment
  - Borders
  - Indents
  - Load/Save Styles
  - Merge Styles
  - Spacing
  - Tabs

- Compiler Errors 3.1
  - ALIAS and MAP Section Errors
  - Build Tag Errors
  - Build Tag Footnote and Expression Errors
  - Context String Errors
  - Entry Macro Errors
  - Font Range Option Errors
  - Footnote Errors
  - Forcefont Errors
  - General Section Errors
  - Keyword Errors
  - Macro Errors
  - Miscellaneous Errors
  - Multikey Errors
  - OPTIONS Section Errors
  - Other Options Errors
  - Project File Errors
  - Root Option Errors
  - Source File Errors
  - Syntax Errors
  - Topic File Errors
  - Topic Title Errors
  - WINDOWS Section Errors
- Compiler Errors 95
  - Notes
  - Warnings
  - Warnings 2
  - Errors
- Compiler Options
  - Baggage
  - Compiler
    - Compression
    - Fonts
  - Index
  - Map
  - Misc 3.1
    - Misc 95
  - RTF
    - Sorting
    - Text Search
  - Warning
- Convert Images
- Creating Browse Sequences
- Creating Reports
  - View
  - Links
  - Browse Sequences
- Defining Help Windows

- Buttons Tab
- Colors Tab
- General Tab
- Macros Tab
- Position Tab

- Jumps and Popups
- Paragraph Styles

◆ Embed Window

◆ Environment Options
- Context Numbers

- Display
- File Conversion
- File Save
- HLX Conversion
- Image Import
- Keyboard Options
- Mouse
- Tab Ruler
- Tips
- Warnings

- Word Delimiter

◆ File Functions
- Backup
- Exit
- Export
- Import
- New File
- Open File
- Project Management
- Save File
- Save File As
- Statistics (File)

◆ Glossary Wizard

◆ Help Compiler Notes

◆ Help Magician Window

◆ Help Topics Browser
- Contents Tab
- Find Tab
- Index Tab

◆ Help Wizard

◆ Importing RTF Files
- Manual to Help Conversion
- RTF Technical Specification

◆ Importing Text Files
- Environment Options

◆ Inserting Bitmaps/Pictures

- Integrated Test Mode
  - WinHelp Bar
- Jumps and Popups
  - Context Relations
  - Creating Browse Sequences
  - Defining Help Windows
  - Delete
  - Links From
  - Mid Topic Jump
  - Multiple Files
  - Multimedia with WinHelp
  - Unreferenced Topics
  - View/Modify
- Keyword Management
  - The Keyword Management Screen
  - View by Keywords with Topics Mode
  - View by Topics with Keywords Mode
  - Working with Keywords
- Macros, Creating
  - Create Button
  - Define Macros
  - Load Macros
  - Macro Form Controls
  - Macro/Keyword Association
  - Project Macros
  - Save Macros
  - Topic Macros
- Macro Form Controls
  - Definitions List
  - Display Macros
  - Editor
  - Editor Button
  - Editor Menu Add to Def
  - Editor Menu Clear All
  - Editor Menu Clear Arg
  - Editor Menu Copy
  - Editor Menu Paste
  - Exit Button
  - Help Button
  - Help Label
  - Hints/Macro Ref
  - Locator Fields
  - Macros Button
  - Macro Definitions
  - Macro List
  - Macro Status Indicators
  - Protoypes Button
  - Prototypes List
  - Syntax Checking

- Macro Reference
- Managing Topic Text
  - Bookmarks
  - Convert to Hard Spaces
  - Copying Text
  - Cutting Text
  - Delete Page
  - Deleting Text
  - Find
  - Goto
  - Insert Page
  - Pasting Text
  - Replace
  - Unlink Cursor
- Manual to Help Conversion
  - Formatting Import Commands
- Mid Topic Jump
  - Context Strings
- Multimedia with WinHelp
- Multiple Files
  - Save File As Range
- Navigator
  - Close on Selection
  - Contents Page
  - On Top
  - Reset on Selection
- On Line Help
  - Help About
  - Help Contents
  - Help Keyboard
  - Help on Help
  - Help Search
  - Help Wizard
- Options
  - Display History Window...
- Paragraph Styles
  - Alignment
  - Applying Styles
  - Borders
  - Bullets
  - Character Formatting
  - Defaults
  - Indents
  - Load/Save Styles
  - Merge Styles
  - Modify Character Attributes
  - Paragraph Formatting

- Spacing
- Tabs

- Path Options
  - Call Word Processor

- Project Files and WinHelp()
  - Context Strings

- Project Management
  - Close Project
  - File Selection Listbox
  - New Project
  - Open Project
  - Project Add File
  - Project Archive
  - Project Close
  - Project Context List
  - Project Copy
  - Project Delete
  - Project File List
  - Project Global Copy
  - Project Global Move
  - Project Import
  - Project Management Overview
  - Project Move
  - Project New
  - Project Open
  - Project Remove
  - Project View HMP
  - Project XRef
  - Save Project As
  - Statistics (Project)
  - UnArchive Project File

- Restore Page

- SHED.EXE
  - Context Strings

- Spell Checking

- Status Bar
  - Defining Styles

- Testing and Navigating

- Tool Bar

- Tools
  - Auto TOC
  - Bitmap Magician
  - Capture Image
  - Context Reference
  - Convert Images
  - Navigator
  - Paintbrush

- —Renumber Context #'s (Numbers)
- —Sound Recorder
- —Tools SHED.EXE
- —User Defined...

- —Utility Area
  - —Context Relations
  - —Context Strings
  - —SHED.EXE
  - —Unreferenced Topics

- —VB Help Wizard

- Web Authoring
  - Browser
  - —Colors
  - —Display
  - Files
  - Fonts
  - —Footer
  - —Header
  - Styles
  - Versions

- Working with Keywords
  - Keyword Management
  - The Keyword Management Screen

Visual C Support

Visual C

From the MFC Tech Notes #28:

The best way to add Help to your application is to check the "Context Sensitive Help" option in AppWizard's Options dialog before creating your application. That way AppWizard automatically adds the necessary message map entries to your CWinApp-derived class to support Help.

If you have already created your application without Help support and now wish to add it, see Chapter 10 of the Class Library User's Guide.

Build Application

When you build your application for the first time, Visual C will create a number of files to be used to build a help system. It will generate two RTF files (typically AFXCORE.RTF and AFXPRINT.RTF) that contain generic text for standard menus, etc., a .HPJ (help project) file, and bitmaps of the controls on the toolbar.

Prepare RTF files for import into the Help Magician

If you have a version of the Help Magician less than 3.0, perform the following steps to prepare the RTF files for import:

1. Open each of the RTF files in MS Word for Windows (if you want your help source to be in a single file, merge the RTF files in Word for Windows).
2. Locate each table and put the cursor on the table.
3. From the **Table** menu, click on **Select Table**.
4. From the **Table** menu, click on **Convert Table to Text** (select the **Tabs** option).
5. Save each file in **RTF** format.

Context Sensitivity

Visual C uses two header files for the Context Sensitive mapping between your application and the help file: **\mfc\include\afxhelp.hm** and **\yourapp\hlp\yourapp.hm**.

The header file, **yourapp.hm** must be updated each time the controls in your application are changed. Visual C writes a batch file called **makehelp.bat** that rewrites **yourapp.hm** and compiles the help file. The path where **makehm.exe** resides must be in your **DOS** path or you can edit the batch file and enter the path before each occurrence of **makehm**.

Note: to avoid the compilation of your help file while creating or updating **yourapp.hm**, comment out the line in **makehelp.bat** that calls the help compiler:

REM hc31 yourapp.hpj

Help Magician

Version

You must have version 3.00.030 or later of the Help Magician for full support for Visual C.

Root Directory

References to the ***Root*** directory in this document (as defined by Microsoft) refer to the directory that contains the source files, the project file, the RTF files, etc. for the help system. This is also the directory

to which the compiled help file will be written.

Build Type

Select one of the build types described below:

A) Single File Build
B) Multiple File Build
C) Project Management

Build Type C, Project Management describes selecting a ***Root*** directory and copying files to that directory before building a help system. The same strategy can be applied to Build Types A and B. If you don't want to build your help system in your Visual C directories, select a ***Root*** directory and copy the necessary file to that directory before proceeding with the instructions for the selected Build Type.

A) Single File Build

Follow these instructions if you merged the RTF files in Word for Windows:

1. Import the combined RTF file into the Help Magician. From the **File** menu, select **Import**. On the **Import** form select **RTF** as the **File Type**. Under **Import** Options, select **Replace Current File**, and under **RTF Import Options**, select **Read from HPJ**. When the file dialog appears, select the **hpj** file in your Visual C application sub directory.

   When the RTF file has been imported, the map filenames will automatically be entered into the Compiler Options.

2. Save the file as an HLX file with the **File/Save File As/New File** sub menu. Rename the file as desired in the **Save File As** dialog. This will become the name of your compiled help system.

3. Edit the file as needed and select **Rebuild All** from the **Build** menu to compile the help file.

B) Multiple File Build

Follow these instructions if you did not merge the RTF files in Word for Windows:

1. Import **afxprint.rtf** file into the Help Magician. From the **File** menu, select **Import**. On the **Import** form select **RTF** as the **File Type**. Under **Import** Options, select **Replace Current File**, and under **RTF Import Options**, select **Read from HPJ**. When the file dialog appears, select the **hpj** file in your Visual C application sub directory.

   When the RTF file has been imported, the map filenames will automatically be entered into the Compiler Options.

2. Save the file as an HLX file with the **File/Save File As/New File** sub menu. Rename the file as desired in the **Save File As** dialog.

3. Write the RTF file by selecting **Write RTF for Compiler** from the **Build** menu.

4. Import **afxcore.rtf** into the Help Magician as described for **afxprint** above.

5. From the **Build** menu, select **Multiple Files**. Enter **afxprint.rtf** into the **Files** list box by clicking on the **Add** button and selecting the file with the file dialog. You will not need to enter a map file in the [MAP] section because you will be using the external maps generated by Visual C.

6. Save the file as an **hlx** file with the **File/Save File As/New File** sub menu. Rename the file as desired

in the **Save File As** dialog. This is the "**Master"** file in the Multiple File build and will become the name of your compiled help system.

7. Edit the files as needed and select **Rebuild All** from the **Build** menu to compile the help file. Note that the RTF file must be written for the secondary file(s), **afxprint** in this case, before compiling the help system from the **Master** file.

C) Project Management

Follow these instructions if you did not merge the RTF files in Word for Windows and you want to use the Project Management features of the Help Magician:

1. Select a ***Root*** directory on your system for all of the help related files.

2. Copy the following files to the ***Root*** directory:

   \yourapp\yourapp.hpj
   \yourapp\hlp\afxcore.rtf
   \yourapp\hlp\afxprint.rtf
   \yourapp\hlp\yourapp.hm
   \yourapp\hlp*.bmp
   \visualc\mfc\include\afxhelp.hm

3. From the **File** menu, select **New Project**. In the file dialog, select your help build ***Root*** directory described in step one. Enter a valid filename as the name of the project, with a .**hmp** extension.

4. On the **Project Management** dialog, click on the **Import** button. Set **File Type** to **Rich Text Format (.RTF)**, and select **Read from HPJ** under **RTF Import** Options and click on **Accept**. In the file dialog, select **afxcore.rtf** from your ***Root*** directory. When the second file dialog appears, select the **yourapp.hpj** from your ***Root*** directory. Add the file to the project when prompted.

5. Repeat the steps above for **afxprint.rtf**.

6. Click on the **Done** button.

7. Select **Write RTF for Compiler** from the **Build** menu. Answer **Yes** to overwrite the existing RTF file.

8. Select **Project Management** from the **File** menu. Select **afxcore.hlx** in the file list and click on the **Open** button.

9. Edit the files as needed and select **Rebuild All** from the **Build** menu to compile the help file. Note that the RTF file must be written for members of the project, not currently in use, before compiling the help system from the **Master** file. The **Project Management** dialog shows the status of all the members of the project.

Notes

In the project file (.**hpj**) that Visual C writes, Compression is set to True. This results in a smaller help file but it takes much longer to compile. You may want to turn Compression off (**Options** menu, **Compiler Options**, **Misc** tab, **Compress Help File**) while developing the help file and turn it on again for final compile.

The first page in **afxprint**, as generated by Visual C, is blank. After importing the file(s) into the Help Magician, you can delete this page, if desired, by selecting **Delete Page** from the **Edit** menu.

Help Magician Macro Reference

About
AddAccelerator (AA)
Annotate
AppendItem
Back
BookMarkDefine
BookMarkMore
BrowseButtons
ChangeButtonBinding (CBB)
ChangeItemBinding (CIB)
CheckItem (CI)
CloseWindow
Contents
CopyDialog
CopyTopic
CreateButton (CB)
DeleteItem
DeleteMark
DestroyButton
DisableButton (DB)
DisableItem (DI)
EnableButton (EB)
EnableItem (EI)
ExecProgram (EP)
Exit
ExtAbleItem
ExtInsertItem
ExtInsertMenu
FileOpen
FloatingMenu
FocusWindow
GoToMark
HelpOn
HelpOnTop
History
IfThen
IfThenElse
InsertItem
InsertMenu
IsMark
JumpContents
JumpContext (JC)
JumpHelpOn
JumpId (JI)
JumpKeyWord (JK)
mciExecute
Next
Not
PopupContext (PC)
PopupId (PI)
PositionWindow (PW)
Prev

Print
PrinterSetup
RegisterRoutine (RR)
RemoveAccelerator (RA)
ResetMenu
SaveMark
Search
SetContents
SetHelpOnFile
sndPlaySound
UncheckItem (UI)

About

This macro displays the About dialog box (same as the About command on the Help menu).

Syntax

About()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

AddAccelerator (AA)

This macro assigns a Help macro to a shortcut key (or key combination) so that the macro is run when the user presses the shortcut key(s).

Syntax

AddAccelerator(key, shift-state, "macro")

Parameters

Argument	Definition
key	The Windows virtual-key value. For a list of these keys, select the **Acc Keys** hints button.
shift-state	A number specifying the combination of ALT, SHIFT, and CTRL keys used with the shortcut key: 0 (none), 1 (SHIFT), 2 (CTRL), 3 (SHIFT+CTRL), 4 (ALT), 5 (ALT+SHIFT), 6 (ALT+CTRL), or 7 (ALT+SHIFT+CTRL).
macro	The Help macro or macro string that is run when the user presses the shortcut key(s). The macro must appear in quotation marks. Multiple macros in a string must be separated by semicolons (;).

Example

The following macro starts the Windows Notepad program (provided in Windows version 3.1) when the user presses ALT+SHIFT+CTRL+F4:

```
AddAccelerator(0x73, 7, "ExecProgram(`notepad.exe', 1)")
```

Comments

The Help macro that is run by AddAccelerator might not work in secondary windows, or its use may be discouraged if the macro it runs is prohibited or discouraged in secondary windows. Check the usage notes for the macro before using AddAccelerator to run it in a secondary window.

Use RemoveAccelerator to remove an assigned accelerator key.

Annotate

This macro displays the Annotation dialog box (same as the Annotate command on the Edit menu).

Syntax

Annotate()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

If the Annotate macro is run from a pop-up window, the annotation is attached to the topic that contains the hot spot to the pop-up window.

AppendItem

This macro appends a menu item to the end of a menu you create with the InsertMenu macro.

Syntax

AppendItem("menu-id", "item-id", "item-name", "macro")

Parameters

Argument	Definition
menu-id	Name used in the InsertMenu macro to create the menu. This name must appear in quotation marks. The new item is appended to this menu.
item-id	Name that WinHelp uses internally to identify the menu item. This name is case-sensitive and must appear in quotation marks. Use this name in DisableItem, EnableItem, DeleteItem, ExtAbleItem, ChangeItemBinding macros if you want to disable/enable or remove the item, or change the operations that the item performs in certain topics.
item-name	Name that WinHelp displays on the menu for the item. This name is case-sensitive and must appear in quotation marks. Within the quotation marks, place an ampersand (&) before the character used as the macro's keyboard access key.
macro	Help macro or macro string that is run when the user chooses the menu item. The name must appear in quotation marks. Multiple macros in a string must be separated by semicolons (;).

Example

The following macro appends a menu item labeled "Index" to a menu that has an identifier "mnu_cards":

AppendItem("mnu_cards", "mnu_index", "&Index", "JI(`index.hlp', `index_topic')")

Choosing the menu item causes a jump to a topic with the context string "index_topic" in the INDEX.HLP file.

Comments

WinHelp ignores this macro if it is run in a secondary window.

Make sure that the keyboard access keys you assign to menu items are unique. If you assign a key that conflicts with other menu access keys, WinHelp displays the error message, "Unable to add item", and ignores the macro.

Back

This macro displays the previous topic in the Back list. The Back list includes the last 40 topics the user has displayed since starting WinHelp.

Syntax

Back()

Parameters

None

Comments

WinHelp ignores this macro if it is run in a secondary window.
If the Back macro is run when the Back list is empty, WinHelp takes no action.

BookmarkDefine

This macro displays the Define dialog box (same as the Define command on the Bookmark menu).

Syntax

BookmarkDefine()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

If the BookmarkDefine macro is run from a pop-up window, the bookmark is attached to the topic that invoked the pop-up window.

BookmarkMore

This macro displays the More dialog box (same as the More command on the Bookmark menu). The More command appears on the Bookmark menu if the user has defined more than nine bookmarks.

Syntax

BookmarkMore()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

BrowseButtons

This macro adds the Browse Back (<<) and Browse Forward (>>) buttons to the toolbar in WinHelp.

Syntax

BrowseButtons()

Parameters

None

Example

The following macros in the Project Macro Definition cause the Notepad button to appear immediately before the two browse buttons on the toolbar:

```
CreateButton("&Notepad", "ExecProgram(`notepad', 0)")
BrowseButtons()
```

Comments

WinHelp ignores this macro if it is run in a secondary window.

If the BrowseButtons macro is used with one or more CreateButton macros Project Macro Definition, the order of the browse buttons on the WinHelp toolbar is determined by the order of the BrowseButtons macro in relation to the other macros listed in the Project Macro Definition.

Note:

WinHelp version 3.1 doesn't automatically provide Browse Forward (>>) and Browse Back (<<) buttons. The Help Magician will automatically insert the BrowseButtons macro into the Project Macro Definition if you have created browse sequences.

ChangeButtonBinding (CBB)

This macro assigns a Help macro to a Help button.

Syntax

ChangeButtonBinding("button-id", "button-macro")

Parameters

Argument	Definition
button-id	Identifier assigned to the button in the CreateButton macro or, for a standard Help button, one of the following predefined button identifiers: btn_contents (Contents), btn_search (Search), btn_back (Back), btn_history (History), btn_previous (<<), or btn_next (>>). The button identifier must appear in quotation marks.
button-macro	Help macro run when the user chooses the button. The macro must appear in quotation marks.

Example

The following macro changes the function of the Contents button so that choosing it causes a jump to the Table of Contents topic (identified by the context string "dict_contents") in the DICT.HLP file:

ChangeButtonBinding("btn_contents", "JumpId(`dict.hlp', `dict_contents')")

Comments

WinHelp ignores this macro if it is run in a secondary window.

ChangeItemBinding (CIB)

This macro assigns a Help macro to an item that you add to a WinHelp menu using the AppendItem macro.

Syntax

ChangeItemBinding("item-id", "item-macro")

Parameters

Argument	Definition
item-id	Identifier assigned to the item in the AppendItem macro. The item identifier must appear in quotation marks.
item-macro	Help macro that is run when the user selects the item. The macro must appear in quotation marks.

Example

The following macro changes the menu item identified by "time_item" so that it starts the Windows Clock program:

ChangeItemBinding("time_item", "ExecProgram(`clock', 0)")

Comments

WinHelp ignores this macro if it is run in a secondary window.

CheckItem (CI)

This macro places a check mark beside a menu item added to a Help menu using the AppendItem macro.

Syntax

CheckItem("item-id")

Parameters

Argument	Definition
item-id	Identifier assigned to the item in the AppendItem macro. The item identifier must appear in quotation marks.

Example

The following macro places a checkmark beside the menu item named "syntax-item":

CheckItem("syntax-item")

Comments

Use the UncheckItem macro to clear the checkmark.
WinHelp ignores this macro if it is run in a secondary window.

CloseWindow

This macro closes the specified window, which is either the main WinHelp window or a secondary window.

Syntax

CloseWindow("window-name")

Parameters

Argument	Definition
window-name	The name of the window to close. The name "main" is reserved for the primary Help window. For secondary windows, the window name must be defined in the Help Windows Specifications form under the Options menu. This name must appear in quotation marks.

Example

The following macro closes the secondary window "keys":

CloseWindow("keys")

Comments

The secondary windows names can also be found in the [WINDOWS] section of the project (.HPJ) file. The project file is created every time the RTF file is written in the Help Magician.
If the window does not exist, WinHelp ignores the macro.

Contents

This macro displays the contents topic in the current Help file. The contents topic can be set in the Compiler Options form under the Options menu.

Syntax

Contents()

Parameters

None

Comments

The contents topic is defaulted to the first topic in the help file. If multiple files are used, the contents topic will come from the main help file.

CopyDialog

This macro displays the Copy dialog box (same as the Copy command on the Edit menu).

Syntax

CopyDialog()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

CopyTopic

This macro copies all the text in the currently displayed topic onto the Windows Clipboard.

Syntax

CopyTopic()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

This macro copies text only; it does not copy bitmaps or any other images in the Help topic.
A copyright notice can be set in the Appearance form under the Options menu.

CreateButton (CB)

This macro adds a new button to the WinHelp toolbar.

Syntax

CreateButton("button-id", "caption", "macro")

Parameters

Argument	Definition
button-id	Name that WinHelp uses internally to identify the button. This name must appear in quotation marks. Use this name in the DisableButton or DestroyButton macro if you want to remove or disable the button, or in the ChangeButtonBinding macro if you want to change the Help macro that the Name that WinHelp uses internally to identify the button. This name must appear in quotation marks. Use this name in the DisableButton or DestroyButton macro if you want to remove or disable the button, or in the ChangeButtonBinding macro if you want to change the Help macro that the button runs in certain topics.
caption	The text that appears on the button. This caption must appear in quotation marks. To designate a letter as a keyboard access key for this button, place an ampersand (&) before a letter in this text. The button caption is case-sensitive and can contain up to 29 characters, beyond which the caption is clipped.
macro	Help macro or macro string that is run when the user chooses the button. The macro must appear in quotation marks. Multiple macros in a macro string must be separated by semicolons (;).

Example

The following macro creates a new button labeled "Order Info" that jumps to a topic with the context string "orderform" in the BUYPROG.HLP file when the button is chosen:

CreateButton("btn_order", "&Order Info", "JumpId(`buyprog.hlp', `orderform')")

Comments

WinHelp ignores this macro if it is run in a secondary window.

WinHelp allows a maximum of 16 authored buttons. It allows a total of 22 buttons, including the standard browse buttons, on the toolbar.

If the BrowseButtons macro is used with one or more CreateButton macros in the Project Macro Definition, the order of the browse buttons on the WinHelp toolbar is determined by where the BrowseButtons macro is listed in relation to the other macros in the Project Macro Definition. The button order may be changed in the Macro Editor form available from the Define Macros menu item under the Macros menu.

DeleteItem

This macro removes a menu item that was added using the AppendItem macro.

Syntax

DeleteItem("item-id")

Parameters

Argument	Definition
item-id	The item identifier string used in the AppendItem macro. The item identifier must appear in quotation marks.

Example

The following macro removes the menu item "Tools" appended in the example for the AppendItem macro:

DeleteItem("mnu_tools")

Comments

WinHelp ignores this macro if it is run in a secondary window.

DeleteMark

This macro removes a text marker added with the SaveMark macro.

Syntax

DeleteMark("marker-text")

Parameters

Argument	Definition
marker-text	Marker text specified in the SaveMark macro. The marker text must appear in quotation marks.

Example

The following macro removes the marker "Saving Money" from the Financial Help file:

DeleteMark("Saving Money")

Comments

If the marker does not exist when the DeleteMark macro is run, WinHelp displays a "Topic not found" error message.

DestroyButton

This macro removes a button added with the CreateButton macro.

Syntax

DestroyButton("button-id")

Parameters

Argument	Definition
button-id	Identifier assigned to the button in the CreateButton macro. The button identifier must appear in quotation marks. The button identifier cannot duplicate an identifier used for one of the standard Help buttons. (See the ChangeButtonBinding macro for a list of these identifiers.)

Comments

WinHelp ignores this macro if it is run in a secondary window.

DisableButton (DB)

This macro disables and dims a button added with the CreateButton macro.

Syntax

DisableButton("button-id")

Parameters

Argument	Definition
button-id	Identifier assigned to the button in the CreateButton macro. The button identifier appears in quotation marks.

Comments

WinHelp ignores this macro if it is run in a secondary window.

A button disabled by the DisableButton macro cannot be used in the topic until an EnableButton macro is run.

DisableItem (DI)

This macro disables and dims a menu item added with the AppendItem macro.

Syntax

DisableItem("item-id")

Parameters

Argument	Definition
item-id	Identifier assigned to the menu item in the AppendItem macro. The item identifier must appear in quotation marks.

Comments

WinHelp ignores this macro if it is run in a secondary window.

A menu item disabled by the DisableItem macro cannot be used in the topic until an EnableItem macro is run.

EnableButton (EB)

This macro re-enables a button disabled with the DisableButton macro.

Syntax

EnableButton("button-id")

Parameters

Argument	Definition
button-id	Identifier assigned to the button in the CreateButton macro. The button identifier must appear in quotation marks.

Comments

WinHelp ignores this macro if it is run in a secondary window.

EnableItem (EI)

This macro re-enables a menu item disabled with the DisableItem macro.

Syntax

EnableItem("item-id")

Parameters

Argument	Definition
item-id	Identifier assigned to the menu item in the AppendItem macro. The item identifier must appear in quotation marks.

Comments

WinHelp ignores this macro if it is run in a secondary window.

ExecProgram (EP)

This macro runs a Windows-based application.

Syntax

ExecProgram("command-line", display-state)

Parameters

Argument	Definition
command-line	Command line for the application to be executed. The command line must appear in quotation marks. WinHelp searches for this application in the current directory, followed by the Windows directory, the user's path, and the directory of the currently displayed Help file.
display-state	A value indicating how the application is shown when executed. A value of 0 indicates normal, 1 indicates minimized, and 2 indicates maximized. For a complete list of display states, select the "Display" hints button.

Example

The following macro runs the Windows Clock program in its normal window size:

ExecProgram("clock.exe", 0)

Exit

This macro exits the WinHelp application (same as the Exit command on the File menu).

Syntax

Exit()

Parameters

None

ExtAbleItem

This macro enables or disables an existing menu item. It is functionally equivalent to the combination of the EnableItem and DisableItem macros.

Syntax

ExtAbleItem("item-id", enabled-state)

Parameters

Argument	Definition
item-id	Identifier assigned to the menu item in the AppendItem or ExtInsertItem macros. The item identifier must appear in quotation marks.
enabled-state	An integer specifying the initial state of the menu item. Use a '0' to enable the item and '1' to disable the item.

Example

The following macro disables the menu item "mnu_tools":

ExtAbleItem("mnu_tools", 1)

Comments

WinHelp ignores this macro if it is run in a secondary window.

ExtInsertItem

This macro functions identically to the InsertItem macro with one exception: it has an extra argument to enable or disable the item on startup.

Syntax

ExtInsertItem("menu-id", "item-id", "item-name", "macro", position, enabled-state)

Parameters

Argument	Definition
menu-id	Either a standard WinHelp menu name or the name used in the InsertMenu macro to create the menu. Standard menu names are mnu_file (File menu), mnu_edit (Edit menu), mnu_bookmark (Bookmark menu), and mnu_helpon (Help menu). The menu identifier must appear in quotation marks. The new item is inserted into this menu.
item-id	Name that WinHelp uses internally to identify the menu item. The item identifier must appear in quotation marks. Use this name in DisableItem, EnableItem, DeleteItem, ExtAbleItem, ChangeItemBinding macros if you want to disable/enable or remove the item, or change the operations that the item performs in certain topics.
item-name	Name that WinHelp displays on the menu for the item. This name is case-sensitive and must appear in quotation marks. Within the quotation marks, place an ampersand (&) before the character used for the item's keyboard access key.
macro	Help macro or macro string that is run when the user chooses the menu item. The macro must appear in quotation marks. Multiple macros in a string must be separated by semicolons (;).
position	An integer specifying the position in the menu where the new item will appear. Position 0 is the first or topmost position in the menu.
enabled-state	An integer specifying the initial state of the menu item. Use a '0' to enable the item and '1' to disable the item.

Example

The following macro inserts a menu item labeled "Tools" as the third item (0,1,2) on a menu that has an identifier "mnu_books":

ExtInsertItem("mnu_books", "mnu_tools", "&Tools", "JI(`tools.hlp', `first_topic')", 3, 0)

Selecting the menu item causes a jump to a topic with the context string "first_topic" in the TOOLS.HLP file.

Comments

WinHelp ignores this macro if it is run in a secondary window.

Make sure that the keyboard access keys you assign to menu items are unique. If you assign a key that conflicts with other menu access keys, WinHelp displays the error message Unable to add item and ignores the macro.

Use "mnu_main" for the menu-id to place an item directly on the WinHelp main menu bar.

Use "mnu_floating" to place an item in the floating menu. Remember to execute the FloatingMenu() macro to enable the floating menu option.

ExtInsertMenu

This macro functions similarily to the InsertMenu macro with a couple of enhancements:

1. It has an extra argument to allow placing the menu under other menus as a sub-menu.
2. It has an extra argument to enable or disable the item on startup.

Syntax

ExtInsertMenu("parentmenu-id", "submenu-id", "submenu-name", submenu-position, enabled-state)

Parameters

Argument Definition

parentmenu-id Either a standard WinHelp menu name or the name used in the InsertMenu macro to create the menu. Standard menu names are . The menu identifier must appear in quotation marks. The new item is inserted into this menu. If "mnu_main" is used, the menu will be inserted into WinHelp's main menu bar.

submenu-id Name that WinHelp uses internally to identify the submenu item. The item identifier must appear in quotation marks. Use this name in DisableItem, EnableItem, DeleteItem, ExtAbleItem, ChangeItemBinding macros if you want to disable/enable or remove the item, or change the operations that the item performs in certain topics.

submenu-name Name that WinHelp displays on the submenu for the item. This name is case-sensitive and must appear in quotation marks. Within the quotation marks, place an ampersand (&) before the character used for the submenu's keyboard access key.

submenu-position An integer specifying the position in the parent menu where the new submenu will appear. Position 0 is the first or topmost position in the parent menu.

enabled-state An integer specifying the initial state of the menu item. Use a '0' to enable the item and '1' to disable the item.

Example #1

The following macro inserts a submenu item labeled "Tools" as the third item (0, 1, 2) on a parent menu that has an identifier "mnu_books":

ExtInsertMenu("mnu_books", "mnu_tools", "&Tools", 2, 0)

Example #2

The following macro inserts a menu labeled "Charts" as the second item (0, 1) on WinHelp's main menu:

ExtInsertMenu("mnu_main", "mnu_chts", "&Charts", 1, 0)

Example #3

The following macros insert a submenu labeled "File" as the first item on WinHelp's floating menu and adds four menu items to it:

```
ExtInsertMenu("mnu_floating", "mnu_ffile", "&File", 0, 0)
AppendItem("mnu_ffile", "itm_file1", "&Open", "FileOpen()")
AppendItem("mnu_ffile", "itm_file2", "&Print Topic", "Print()")
AppendItem("mnu_ffile", "itm_file3", "P&rint Setup...", "PrinterSetup()")
AppendItem("mnu_ffile", "itm_file4", "&Exit", "Exit()")
```

To see this example function, click the right mouse button to display the floating menu.

Comments

WinHelp ignores this macro if it is run in a secondary window.

Make sure that the keyboard access keys you assign to menu items are unique. If you assign a key that conflicts with other menu access keys, WinHelp displays the error message Unable to add menu and ignores the macro.

Use "mnu_main" for the menu-id to place a menu directly on the WinHelp main menu bar.

Use "mnu_floating" to place a sub-menu in the floating menu. Remember to execute the FloatingMenu() macro to enable the floating menu option.

FileOpen

This macro displays the Open dialog box (same as the Open command on the File menu).

Syntax

FileOpen()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

FloatingMenu

This macro displays the floating menu, providing items have been added to it.

Syntax

FloatingMenu()

Parameters

None

Comments

Menu items should be added to the floating menu before calling this macro.

Typically the user would click the right mouse button to display the floating menu, but this macro provides the help author (you) a way to display it from a button or a hotspot, for example.

Example

The following macros insert a submenu labeled "File" as the first item on WinHelp's floating menu and adds four menu items to it:

```
ExtInsertMenu("mnu_floating", "mnu_ffile", "&File", 0, 0)
AppendItem("mnu_ffile", "itm_file1", "&Open", "FileOpen()")
AppendItem("mnu_ffile", "itm_file2", "&Print Topic", "Print()")
AppendItem("mnu_ffile", "itm_file3", "P&rint Setup...", "PrinterSetup()")
AppendItem("mnu_ffile", "itm_file4", "&Exit", "Exit()")
Floatingmenu()
```

Click the right mouse button to see an example of the floating menu.

FocusWindow

This macro changes the focus to the specified window, which is either the main WinHelp window or a secondary window.

Syntax

FocusWindow("window-name")

Parameters

Argument	Definition
window-name	The name of the window to have the focus. The name "main" is reserved for the primary Help window. For secondary windows, the window name must be defined in the Help Windows Specifications form under the Options menu. This name must appear in quotation marks.

Example

The following macro changes the focus to the secondary window "glossary":

FocusWindow("glossary")

Comments

If the window does not exist, WinHelp ignores the macro.

GoToMark

This macro jumps to a marker set with the SaveMark macro.

Syntax

GoToMark("marker-text")

Parameters

Argument	Definition
marker-text	Marker text specified in the SaveMark macro. The marker text must appear in quotation marks.

Example

The following macro jumps to the marker "Saving Money" in the Financial Help file:

GoToMark("Saving Money")

HAnimateCommand

This is currently a non-supported Viewer macro.
Its inclusion at this point is strictly for future version compatibility.

HAudioCommand

This is currently a non-supported Viewer macro.
Its inclusion at this point is strictly for future version compatibility.

HAudioDialog

This is currently a non-supported Viewer macro.
Its inclusion at this point is strictly for future version compatibility.

HDisplayBitmap

This is currently a non-supported Viewer macro.
Its inclusion at this point is strictly for future version compatibility.

HelpOn

This macro displays the Using Help file for the WinHelp application (same as the Using Help command on the Help menu).

Syntax

HelpOn()

Parameters

None

HelpOnTop

This macro forces the Help window to always stay on top of other applications so that it is always visible.

Syntax

HelpOnTop()

Parameters

None

Comments

Use of this macro with the main window is discouraged. To set the "on top" feature for the main window, check the "Always On Top" menu item off the Help menu in WinHelp. Windows Help dows not provide a way to monitor the state of the "on top" feature so it is up to the user to track the status of it.

History

This macro displays the Windows Help History window, which shows the last 40 topics the user has viewed since opening a Help file in WinHelp. It has the same effect as choosing the History button on the WinHelp toolbar.

Syntax

History()

Parameters

None

Comments

WinHelp ignores this macro if it is run in a secondary window.

IfThen

This macro runs a Help macro if a given marker exists. It uses the IsMark macro to make the test.

Syntax

IfThen(IsMark("marker-text"), "macro")

Parameters

Argument	Definition
marker-text	Marker text tested by the IsMark macro. The marker text must appear in quotation marks.
macro	The Help macro or macro string that is run if the marker exists. The macro must appear in quotation marks. Multiple macros in a macro string must be separated by semicolons (;).

Example

The following macro jumps to the topic with the context string "bal_chk" if a marker named "Balance Checkbook" has been set by the SaveMark macro:

IfThen(IsMark("Balance Checkbook"), "JI(`finance.hlp', `bal_chk')")

IfThenElse

This macro runs one of two Help macros, provided a marker exists. It uses the IsMark macro to make the test.

Syntax

IfThenElse(IsMark("marker-text"), "macro1", "macro2")

Parameters

Argument	Definition
marker-text	Marker text tested by the IsMark macro. The marker text must appear in quotation marks.
macro1, macro2	WinHelp runs macro1 if the marker exists and macro2 if it does not. Both macros must appear in quotation marks. Multiple macros in either macro string must be separated by semicolons (;).

Example

The following macro jumps to the topic with the context string "bal_chk" if a marker named "Balance Checkbook" has been set by the SaveMark macro. If the marker does not exist, it jumps to the contents screen for the FINANCE.HLP file:

IfThenElse(IsMark("Balance Checkbook"), "JI(`finance.hlp', `bal_chk')","JumpContents(`FINANCE.HLP')")

InsertItem

This macro inserts a menu item at a given position on an existing menu. The menu can be either one you create with the InsertMenu macro or one of the standard WinHelp menus.

Syntax

InsertItem("menu-id", "item-id", "item-name", "macro", position)

Parameters

Argument	Definition
menu-id	Either a standard WinHelp menu name or the name used in the InsertMenu macro to create the menu. Standard menu names are mnu_file (File menu), mnu_edit (Edit menu), mnu_bookmark (Bookmark menu), and mnu_helpon (Help menu). The menu identifier must appear in quotation marks. The new item is inserted into this menu.
item-id	Name that WinHelp uses internally to identify the menu item. The item identifier must appear in quotation marks. Use this name in DisableItem, EnableItem, DeleteItem, ExtAbleItem, ChangeItemBinding macros if you want to disable/enable or remove the item, or change the operations that the item performs in certain topics.
item-name	Name that WinHelp displays on the menu for the item. This name is case-sensitive and must appear in quotation marks. Within the quotation marks, place an ampersand (&) before the character used for the item's keyboard access key.
macro	Help macro or macro string that is run when the user chooses the menu item. The macro must appear in quotation marks. Multiple macros in a string must be separated by semicolons (;).
position	An integer specifying the position in the menu where the new item will appear. Position 0 is the first or topmost position in the menu.

Example

The following macro inserts a menu item labeled "Tools" as the third item on a menu that has an identifier "mnu_books":

InsertItem("mnu_books", "mnu_tools", "&Tools", "JI(`tools.hlp', `first_topic')", 3)

Selecting the menu item causes a jump to a topic with the context string "first_topic" in the TOOLS.HLP file.

Comments

WinHelp ignores this macro if it is run in a secondary window.

Make sure that the keyboard access keys you assign to menu items are unique. If you assign a key that conflicts with other menu access keys, WinHelp displays the error message Unable to add item and ignores the macro.

InsertMenu

This macro adds a new menu to the WinHelp menu bar.

Syntax

InsertMenu("menu-id", "menu-name", menu-position)

Parameters

Argument	Definition
menu-id	Name that WinHelp uses internally to identify the menu. The menu identifier must appear in quotation marks. Use this identifier in the AppendItem macro to add commands to the menu.
menu-name	Name for the menu that WinHelp displays on the menu bar. This name is case-sensitive and must appear in quotation marks. Within the quotation marks, place an ampersand (&) before the character used for the menu's keyboard access key.
menu-position	Number telling WinHelp which position on the menu bar the new menu name will have. Positions are numbered from left to right, with position 0 being the leftmost menu.

Example

The following macro adds a menu named "Utilities" to WinHelp:

InsertMenu("menu_util", "&Utilities", 3)

The label "Utilities" appears as the fourth menu on the WinHelp menu bar. The user presses ALT+U to display the menu and its commands.

Comments

WinHelp ignores this macro if it is run in a secondary window.

Make sure that the keyboard access keys you assign to menus are unique. If you assign a key that conflicts with other menu access keys, WinHelp displays the error message "Unable to add menu" and ignores the macro.

IsMark

This macro determines whether a marker set by the SaveMark macro exists. It is used as a parameter to the conditional macros IfThen and IfThenElse.

Syntax

IsMark("marker-text")

Parameters

Argument	Definition
marker-text	Marker text tested by the IsMark macro. The IsMark macro returns a True value if the mark exists and a False value if it does not. The marker text must appear in quotation marks.

Example

The following macro jumps to the topic with the context string "bal_chk" if a marker named "Balance Checkbook" has been set by the SaveMark macro:

```
IfThen(IsMark("Balance Checkbook"), "JI(`finance.hlp', `bal_chk')")
```

Comments

The Not macro can be used to reverse the results of the IsMark macro.

JumpContents

This macro jumps to the contents topic of a specified Help file. The contents topic can be set in the Compiler Options form under the Options menu.

Syntax

JumpContents("filename")

Parameters

Argument	Definition
filename	The name of the destination file for the jump. The file name must appear in quotation marks. If WinHelp cannot find this file, it displays an error message and does not perform the jump.

Example

The following macro jumps to the contents topic of the PROGMAN.HLP file:

JumpContents("PROGMAN.HLP")

Comments

If the CONTENTS option is not specified, WinHelp jumps to the first topic in the Help file.
WinHelp ignores this macro if it is run in a secondary window.
Use the word "THISFILE" to refer to the current help file.

JumpContext (JC)

This macro jumps to a topic identified by a context number. The context number must be entered in the "Context #" field in the Help Magician.

Syntax

JumpContext("filename", context-number)

Parameters

Argument	Definition
filename	The name of the destination file for the jump. The file name must appear in quotation marks. If WinHelp cannot find this file, it displays an error message and does not perform the jump.
context-number	Context number of the topic in the destination file. The context number must assigned for the topic to be recognized. If the context number does not exist or cannot be found, WinHelp jumps to the contents topic or the first topic in the file instead, and displays an error message.

Example

The following macro jumps to the topic mapped to the context number 1501 in the PROGMAN.HLP file:

JumpContext("PROGMAN.HLP", 1501)

Comments

Use the word "THISFILE" to refer to the current help file.

JumpHelpOn

This macro jumps to the contents topic of the "Using Help" file. The "Using Help" file is either the default WINHELP.HLP file or the Help file designated by the SetHelpOnFile macro.

Syntax

JumpHelpOn()

Parameters

None

Example

The following macro jumps to the contents topic of the designated Using Help file:

JumpHelpOn()

Comments

If WinHelp cannot find the specified Help file, it displays an error message and does not perform the jump.

JumpId (JI)

This macro jumps to the topic with the specified context string in the Help file.

Syntax

JumpId("filename", "context-string")

Parameters

Argument	Definition
filename	Name of the Help file (.HLP) containing the context string. The file name must appear in quotation marks. If WinHelp does not find this file, it displays an error message and does not perform the jump.
context-string	Context string of the topic in the destination file. The context string must appear in quotation marks. If the context string does not exist, WinHelp jumps to the contents topic for that file instead.

Example

The following macro jumps to a topic with "second_topic" as its context string in the Help file SECOND.HLP:

JumpId("second.hlp", "second_topic")

Comments

Use the word "THISFILE" to refer to the current help file.

JumpKeyword (JK)

This macro opens the indicated Help file (.HLP), searches through the keyword table, and displays the first topic containing the keyword specified in the macro.

Syntax

JumpKeyword("filename", "keyword")

Parameters

Argument	Definition
filename	The name of the .HLP file that contains the desired keyword table. The file name must appear in quotation marks. If this file does not exist, WinHelp displays an error message and does not perform the jump.
keyword	The keyword that the macro searches for. The keyword must appear in quotation marks. If WinHelp finds more than one match, it displays the first matched topic. If it does not find any matches, it displays a "Not a keyword" message and the contents topic of the destination file.

Example

The following macro opens the first topic that has "hands" as an index keyword in the Help file CLOCK.HLP:

JumpKeyword("clock.hlp", "hands")

Comments

Use the word "THISFILE" to refer to the current help file.
WinHelp searches through the 'K' keyword table, the default table that the Help Magician generates. Multiple keyword tables are ignored.

mciExecute

Basic Commands

The following list summarizes the basic commands. The use of these messages by a device is optional.

Command	Description
load	Recalls data from a disk file.
pause	Pauses playing or recording.
play	Starts transmitting output data.
record	Starts recording input data.
resume	Resumes playing or recording on a paused device.
save	Saves data to a disk file.
seek	Seeks forward or backward.
set	Sets the operating state of the device.
status	Obtains status information about the device. The status command is also listed in the group of required commands. In the basic group, options are added for devices that use linear media with identifiable positions.
stop	Stops playing or recording.

Comments

If a driver supports a basic command, it must also support a standard set of options for the command.

This is not a standard WinHelp macro and needs to be registered using the RegisterRoutine macro should you be creating the project file manually.

Next

This macro displays the next topic in the browse sequence for the Help file. It has the same effect as choosing the Browse Forward (>>) button.

Syntax

Next()

Parameters

None

Comments

If the current topic is the last of a browse sequence, this macro does nothing.

WinHelp ignores this macro if it is run in a secondary window.

Not

This macro reverses the True or False result returned by the IsMark macro. It is used with the IsMark macro as a parameter to the conditional macros IfThen and IfThenElse.

Syntax

Not(IsMark("marker-text")

Parameters

Argument	Definition
marker-text	Marker text tested by the IsMark macro. The Not macro returns a False value if the mark exists or a True value if it does not. The marker text must appear in quotation marks.

Example

The following macro jumps to the topic with context string "Settle Accounts" if a marker named "Balance Checkbook" has not been set by the SaveMark macro:

IfThen(Not(IsMark("Balance Checkbook")), "JI(`finance.hlp', `Settle Accounts')")

PopupContext (PC)

This macro displays a topic identified by a context number. The context numbers are entered via the Help Magician's "Context #" field.

Syntax

PopupContext("filename", context-number)

Parameters

Argument	Definition
filename	The name of the file that contains the topic to be displayed in the pop-up window. The file name must appear in quotation marks. If WinHelp cannot find this file, it displays an error message.
context-number	Context number of the topic to be displayed in the pop-up window. The context number must entered in the "Context #" entry field in the Help Magician. If the context number does not exist or cannot be found, WinHelp displays the contents topic or the first topic in the file instead.

Example

The following macro displays in a pop-up window the topic mapped to the context number 1501 in the file PROGMAN.HLP:

PopupContext("progman.hlp", 1501)

Comments

Use the word "THISFILE" to refer to the current help file.

PopupId (PI)

This macro displays a topic from a specified file in a pop-up window.

Syntax

PopupId("filename", "context-string")

Parameters

Argument	Definition
filename	The name of the file that contains the pop-up window topic. The file name must appear in quotation marks. If this file does not exist, WinHelp displays an error message.
context-string	Context string of the topic in the destination file. The context string must appear in quotation marks. If the requested context string does not exist, WinHelp displays the contents topic or the first topic in the file in the pop-up window. Context strings can be entered in the "Context $" entry field in the Help Magician.

Example

The following macro displays in a pop-up window a topic identified by the context string "second_topic" in the file SECOND.HLP:

PopupId("second.hlp", "second_topic")

Comments

Use the word "THISFILE" to refer to the current help file.

PositionWindow (PW)

This macro sets the size and position of the main Help window or an existing secondary window.

Syntax

PositionWindow(x, y, width, height, state, "window-name")

Parameters

Argument	Definition
x, y	X and Y coordinates of the upper-left window corner. Positions are defined in terms of WinHelp's 1024 x 1024 coordinate system.
width, height	Gives the default width and height of the window. Window sizes, like positions, are defined in terms of WinHelp's coordinate system.
window-state	Specifies how the window is sized. This parameter is 0 for normal size and 1 for maximized. If the parameter is 1, WinHelp ignores the x, y, width, and height parameters.
window-name	The name of the window to position. The name "main" is reserved for the primary Help window. Secondary windows must be defined in the Help Windows Specifications form under the Options menu in the Help Magician. This name must appear in quotation marks.

Example

The following macro positions the secondary window "Samples" in the upper-left corner (100, 100) with a width and height of 500 (in WinHelp coordinates):

PositionWindow(100, 100, 500, 500, 0, "Samples")

Comments

If the window to be positioned does not exist, WinHelp ignores the macro.

Prev

This macro displays the previous topic in the browse sequence for the Help file. It has the same effect as choosing the Browse Back (<<) button.

Syntax

Prev()

Parameters

None

Comments

If the currently displayed topic is the first topic of a browse sequence, this macro does nothing. WinHelp ignores this macro if it is run in a secondary window.

Print

This macro sends the currently displayed topic to the printer.

Syntax

Print()

Parameters

None

Comments

This macro should be used only to print topics in windows other than the main Help window. For example, it can be used to print topics displayed in secondary windows, provided the user doesn't have a dialog box open at the time of printing.

Use of this macro in secondary windows is discouraged.

PrinterSetup

This macro displays the Print Setup dialog box (same as the Print Setup command on the File menu).

Syntax

PrinterSetup()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

RegisterRoutine (RR)

This macro registers a function within a DLL as a Help macro. Registered functions can be used in macro hot spots, topic macros, or project macros, just as standard Help macros are used. (It should be noted that the Help Magician performs all necessary macro registrations for you automatically when the RTF file is written.)

Syntax

RegisterRoutine("DLL-name", "function-name", "format-spec")

Parameters

Argument	Definition
DLL-name	The file name of the DLL being called. The file name must appear in quotation marks. If WinHelp cannot find the DLL, it displays an error message and does not perform the call.
function-name	The name of the function to be executed in the designated DLL. The function name must appear in quotation marks.
format-spec	A string specifying the formats of parameters passed to the function. The format string must appear in quotation marks. Characters in the string represent C parameter types: "u" for unsigned short, "U" for unsigned long, "i" for short int, "I" for long int, "s" for string (near char *), "S" for string (far char *), or "v" for void. WinHelp automatically makes sure these formats match the parameter types specified in the function prototype.

Example

The following DLL call registers a routine "RetString" in the DLL named HELPLIB.DLL. RetString takes arguments of types far char *, short int, and unsigned long.

RegisterRoutine("HELPLIB", "RetString", "S=iU")

Comments

This is an advanced level macro and its use is discouraged for novices.

RemoveAccelerator

This macro removes a Help macro from a shortcut key that had been previously assigned.

Syntax

RemoveAccelerator(key, shift-state)

Parameters

Argument	Definition
key	The Windows virtual-key value. For a list of these keys, select the **Acc Keys** hints button.
shift-state	A number specifying the combination of ALT, SHIFT, and CTRL keys used with the shortcut key: 0 (none), 1 (SHIFT), 2 (CTRL), 3 (SHIFT+CTRL), 4 (ALT), 5 (ALT+SHIFT), 6 (ALT+CTRL), or 7 (ALT+SHIFT+CTRL).

Example

The following macro removes the macro assigned to the key sequence Ctrl-P:

RemoveAccelerator(0x50, 2")

Comments

WinHelp does not display an error message if you try to remove an unassigned accelerator key.
Use the AddAccelerator macro to assign a macro to a key combination.

ResetMenu

This macro returns the WinHelp menus to the default configuration.

Syntax

ResetMenu()

Parameters

None

SaveMark

This macro saves the location of the currently displayed topic and file and associates a text marker with that location. The GoToMark macro can then be used to jump to this location.

Syntax

SaveMark("marker-text")

Parameters

Argument	Definition
marker-text	Text used to identify the topic location. The marker text must appear in quotation marks, and it must be unique. If the same text is used for more than one marker, WinHelp recognizes only the most recently entered marker.

Example

The following macro saves the marker "Balance Checkbook" in the current topic in the Financial Help file:

SaveMark("Balance Checkbook")

Comments

In addition to GoToMark, WinHelp offers the following other macros for use with text markers:

DeleteMark removes any defined marker.

IsMark tests whether a given marker has been set in the Help file. Not negates the result of this test.

IfThen and IfThenElse run one or more Help macros if a given marker has been set. These use the IsMark (and optional Not) macro to test whether the marker is set.

Text markers are not saved if the user exits and then restarts WinHelp.

Search

This macro displays the dialog box for the Search button, which allows users to search for topics using keywords defined in the Help Magician. It has the same effect as choosing the Search button.

Syntax

Search()

Parameters

None

Comments

WinHelp ignores this macro if it is run in a secondary window.

SetContents

This macro designates a specific topic as the contents topic within the Help file.

Syntax

SetContents("filename", context-number)

Parameters

Argument	Definition
filename	The name of the Help file that contains the desired contents topic. The file name must appear in quotation marks. If WinHelp cannot find the file, it displays an error message and does not perform the jump.
context-number	Context number of the topic in the specified file. The context number must be assigned in the destination Help file. If the context number does not exist or cannot be found in the destination help file, WinHelp displays an error message.

Example

The following macro sets the topic mapped to the context number 1501 in the PROGMAN.HLP file as the contents topic:

SetContents("PROGMAN.HLP", 1501)

After running this macro, pressing the Contents button causes a jump to the specified topic.

Comments

Use the word "THISFILE" to refer to the current help file.

SetHelpOnFile

This macro designates the specific Help file that replaces WINHELP.HLP, the default Using Help file in the Windows environment.

Syntax

SetHelpOnFile("filename")

Parameters

Argument	Definition
filename	The name of the replacement Using Help file. The file name must appear in quotation marks. If WinHelp cannot find this file, it displays an error message.

Example

The following macro sets the Using Help file as MYHELP.HLP:

SetHelpOnFile("myhelp.hlp")

Comments

If this macro appears within a topic in the Help file, the replacement file is set after execution of the macro. If this macro appears in the Project Macro Definition, the replacement file is set when the Help file is opened.

Comments

Use the word "THISFILE" to refer to the current help file.

sndPlaySound

This macro plays a sound (.WAV) file.

Syntax

sndPlaySound("filename", play-flag)

Parameters

Argument	Definition
filename	The name of the sound (.WAV) file. The file name must appear in quotation marks. If WinHelp cannot find this file, it displays an error message.
play-flag	A flag that determines how long the sound plays for. Enter a '0' to terminate normally, '8' to play continuosly until the macro is called again with the '0' play-flag set.

Example

The following macro plays the "CHIMES.WAV" file:

sndPlaySound("chimes.wav", 0)

Comments

This is not a standard WinHelp macro and needs to be registered using the RegisterRoutine macro should you be creating the project file manually.

UnCheckItem

This macro removes a check mark beside a menu item added to a Help menu using the AppendItem macro.

Syntax

UncheckItem("item-id")

Parameters

Argument	Definition
item-id	Identifier assigned to the item in the AppendItem macro. The item identifier must appear in quotation marks.

Example

The following macro removes a checkmark beside the menu item named "syntax-item":

UncheckItem("syntax-item")

Comments

Use the CheckItem macro to set the checkmark.
WinHelp ignores this macro if it is run in a secondary window.

Load (Basic)

Syntax

load device_id [filename] [notify] [wait]

The load command loads a device element from disk.

Parameters

You can specify the following optional parameter:

filename

Specifies the source path and file.

Example

The following command loads a file into the "vidboard" device:

load vidboard c:\vid\fish.vid notify

The notify flag tells MCI to send a notification message when the loading completes.

pause (Basic)

Syntax

pause device_id [notify] [wait]

The pause command pauses playing or recording. Most drivers retain the current position, allowing playback or recording to continue at the current position.

Example

The following command pauses the "mysound" device:

pause mysound

play (Basic)

Syntax

play device_id [parameters] [notify] [wait]

The play command starts playing the device.

Parameters

You can specify one or more of the following optional items for parameters:

from position

Specifies a starting position for the playback. If the from paramer is not specified, playback begins at the current position.

to position

Specifies an ending position for the playback. If the to parameter is not specified, playback ends at the end of the media.

Comments

Before issuing any commands that use position values, you should set the desired time format using the set command.

Example

The following command plays the mysound device from position 1000 through position 2000, sending a notification message when the playback completes:

play mysound from 1000 to 2000 notify

record (Basic)

Syntax

record device_id [parameters] [notify] [wait]

The record command starts recording data. All data recorded after a file is opened is discarded if the file is closed without saving it.

Parameters

You can specify one or more of the following optional items for parameters:

insert

Specifies that new data is added to the device element at the current position.

from position

Specifies a starting position for the recording. If the from parameter is not specified, the device starts recording at the current position.

to position

Specifies an ending position for the recording. If the to parameter is not specified, the device records until it receives a stop or pause command.

overwrite

Specifies that new data will replace data in the device element.

Comments

Before issuing any commands that use position values, you should set the desired time format using the set command.

Example

The following command starts recording data into the "newsound" device at the current position:

record newsound

The recording stops when a stop or pause command is issued.

resume (Basic)

Syntax

resume device_id [notify] [wait]

The resume command continues playing or recording on a paused device.

Example

The following command continues playing or recording the "newsound" device:

resume newsound

save (Basic)

Syntax

save device_id [filename] [notify] [wait]

The save command saves the MCI element.

Parameters

You can specify the following optional item:

filename

Specifies the destination path and file.

Comments

The filename parameter is required if the device was opened using the new device ID.

Example

The following command saves the data in the "newsound" device to C:\SOUNDS\NEWSND.WAV:

save newsound c:\sounds\newsnd.wav

seek (Basic)

Syntax

seek device_id parameter [notify] [wait]

The seek command moves to the specified position and stops.

Parameters

Specify one of the following items for parameter:

to position

Specifies the position to stop the seek.

to start

Seeks to the start of the media.

to end

Seeks to the end of the media.

Comments

Before issuing any commands that use position values, you should set the desired time format using the set command.

Example

The following command seeks to the start of the media file associated with the "mysound" device:

seek mysound to start

set (Basic)

Syntax

set device_id parameters [notify] [wait]

The set command establishes control settings for the driver.

Parameters

Specify one or more of the following items for parameters:

audio all off

Disables audio output.

audio all on

Enables audio output.

audio left off

Disables output to the left audio channel.

audio left on

Enables output to the left audio channel.

audio right off

Disables output to the right audio channel.

audio right on

Enables output to the right audio channel.

door closed

Loads the media and closes the door if possible.

door open

Opens the door and ejects the media if possible.

time format milliseconds

Sets the time format to milliseconds. All commands that use position values will assume milliseconds. You can abbreviate milliseconds as ms.

video off

Disables video output.

video on

Enables video output.

Example

The following command sets the "mysound" device to use milliseconds as the time format:

```
set mysound time format ms
```

status (Basic)

Syntax

status device_id parameter [notify] [wait]

The status command gets status information for the device. This command is also listed as a required command. As a basic command, status adds options for devices with linear media.

Parameters

Specify one of the following items for parameter:

current track

Returns the current track.

length

Returns the total length of the media.

length track track_number

Returns the length of the track specified by track_number.

number of tracks

Returns the number of tracks on the media.

position

Returns the current position.

position track track_number

Returns the position of the start of the track specified by track_number.

ready

Returns true if the device is ready to play.

start position

Returns the starting position of the media.

time format

Returns the current time format.

Comments

Before issuing any commands that use position values, you should set the desired time format using the set command.

Example

The following command returns the time format used by the "mysound" device:

status mysound time format

stop (Basic)

Syntax

stop device_id [notify] [wait]

The stop command stops playback or recording.

Example

The following command stops playback or recording on the "mysound" device:

stop mysound

predefined WinHelp menu id's:

mnu_file (File menu)
mnu_edit (Edit menu)
mnu_bookmark (Bookmark menu)
mnu_help (Help menu)
mnu_helpon (Help sub-menu)
mnu_helpontop (Help sub-menu)
mnu_main (WinHelp main menu)
mnu_floating (Floating menu)

Unknown Prototype

This prototype is not a standard WinHelp macro.

Example Syntax

SampleMacro(arg1, "arg2")

arg1 - Numerical argument. Enter any integer number in decimal or hex (ex. 3 or 0xF7).

arg2 - String argument. Enter any combination of text within the double (or single) quotes. Some characters need to be preceded with a backslash (\) such as an opening or closing parenthesis "()", single or double quotes, and the backslash.

File names

File naming conventions in the Windows environment use the last three characters after the period, called the file name extension or extension for short, as a way of identifying the file type. Below is a list of files used in various WinHelp macros along with their functions and extensions:

Audio Files
These files contain sound information and carry the extension of **.WAV**.

Animation Files
Animation files contain information used in multimedia animation files. They have **.MMM** as an extension.

Image Files
Image files contain picture or bitmap information in the "DIB" or "BMP" format and have **.DIB** or **.BMP** for their extensions, respectively.

Help Files
Help files, like this one, have **.HLP** as an extension.

Dynamic Link Libraries (DLL's)
Dynamic link libraries are files that contain programs that can be called from WinHelp macros. They typically exist in the Windows directory or in the parent directory of the application, but may exist in the default path set in the AUTOEXEC.BAT file using the PATH statement. They have **.DLL** as an extension.

Executable Files
These files are the actual application programs and can also be called from WinHelp macros such as ExecProgram. They typically exist in the Windows directory or in the parent directory of the application, but may exist in the default path set in the AUTOEXEC.BAT file using the PATH statement. They have **.EXE** as an extension.

COMMENTS:

All files with the exception of executables and DLL's should be located in the ROOT path or BMROOT (in the case of bitmaps and DIB's). Executables and DLL's should be located in the Windows directory, the SYSTEM directory, the default DOS PATH, or preferably in the ROOT path set in the Help Magician.

Note: The Macro Editor hints button will allow you to select a file name with an associated path, but will strip the path from the file name before inserting it into an argument. This is because the Help Magician expects all files to be located in the locations mentioned in the paragraph above. You may include a path with a file name in an argument at anytime, but beware that WinHelp will expect that file to be in that particular path when the Help file is opened. This can cause unexpected behavior in WinHelp if the file is not found, like in an targeted end user's environment for instance. It is safest to keep all the support files together in the ROOT directory and to advise your end users to do the same.

ROOT

The ROOT directory, for a help build, is any directory designated as containing the files necessary to build and compile the Windows help file. To the Help Magician, this is the directory from which the help file is opened or the directory to which the help file is saved.

BMROOT

The BMROOT directory, for a help build, is any directory or directories designated as containing the bitmap and/or DIB files necessary to build and compile the Windows help file. These paths can be set in the Paths form from the Options menu in the Help Magician.

Help Magician Pro 95 Macro Reference

About
AddAccelerator (AA)
ALink
Annotate
AppendItem
Back
BookMarkDefine
BookMarkMore
BrowseButtons
ChangeButtonBinding (CBB)
ChangeEnable
ChangeItemBinding (CIB)
CheckItem (CI)
CloseSecondarys
CloseWindow
Compare
Contents
ControlPanel
CopyDialog
CopyTopic
CreateButton (CB)
DeleteItem
DeleteMark
DestroyButton
DisableButton (DB)
DisableItem (DI)
EnableButton (EB)
EnableItem (EI)
EndMPrint
ExecFile
ExecProgram (EP)
Exit
ExtAbleItem
ExtInsertItem
ExtInsertMenu
FileExist
FileOpen
Finder
FloatingMenu
Flush
FocusWindow
GoToMark
HelpOn
HelpOnTop
History
IfThen
IfThenElse
InitMPrint
InsertItem
InsertMenu
IsBook
IsMark

IsNotMark
JumpContents
JumpContext (JC)
JumpHelpOn
JumpId (JI)
JumpKeyWord (JK)
KLink
mciExecute
Menu
MPrintHash
MPrintID
Next
NoShow
Not
PopupContext (PC)
PopupId (PI)
PositionWindow (PW)
Prev
Print
PrinterSetup
RegisterRoutine (RR)
RemoveAccelerator (RA)
ResetMenu
SaveMark
Search
SetContents
SetHelpOnFile
SetPopupColor
ShellExecute
ShortCut
sndPlaySound
TCard
Test
TestALink
TestKLink
UncheckItem (UI)
UpdateWindow

About

This macro displays the About dialog box (same as the About command on the Help menu).

Syntax

About()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

AddAccelerator (AA)

This macro assigns a Help macro to a shortcut key (or key combination) so that the macro is run when the user presses the shortcut key(s).

Syntax

AddAccelerator(key, shift-state, "macro")

Parameters

Argument	Definition
key	The Windows virtual-key value. For a list of these keys, select the **Acc Keys** hints button.
shift-state	A number specifying the combination of ALT, SHIFT, and CTRL keys used with the shortcut key: 0 (none), 1 (SHIFT), 2 (CTRL), 3 (SHIFT+CTRL), 4 (ALT), 5 (ALT+SHIFT), 6 (ALT+CTRL), or 7 (ALT+SHIFT+CTRL).
macro	The Help macro or macro string that is run when the user presses the shortcut key(s). The macro must appear in quotation marks. Multiple macros in a string must be separated by semicolons (;).

Example

The following macro starts the Windows Notepad program (provided in Windows version 3.1) when the user presses ALT+SHIFT+CTRL+F4:

```
AddAccelerator(0x73, 7, "ExecProgram(`notepad.exe', 1)")
```

Comments

The Help macro that is run by AddAccelerator might not work in secondary windows, or its use may be discouraged if the macro it runs is prohibited or discouraged in secondary windows. Check the usage notes for the macro before using AddAccelerator to run it in a secondary window.

Use RemoveAccelerator to remove an assigned accelerator key.

ALink (Win95 Only)

This macro searches for matching A-keywords.

Syntax

ALink("keywords", type, "context-string", "window-name")

Parameters

Argument	Definition
keywords	One or more keywords separated by semicolons (;).
shift-state	Specifies the action to perform if one or more keywords are found. If this parameter is not specified or is zero, the default action is always to display the Topics Found dialog box containing the topic title. This parameter may specify one or more of the following values, separated by spaces.

Value	Meaning
JUMP (1)	Specifies that if only one topic matches any of the keywords, WinHelp should jump directly to that topic.
TITLE (2)	Specifies that if a keyword is found in more than one Help file, WinHelp should display the title of the Help file (as specified in the contents [.cnt] file) beside the topic title in the Topics Found dialog box.
TEST (4)	Specifies that the macro should return a value indicating whether or not there is at least one match. The TestALink macro is converted by Help Workshop into an ALink macro with this parameter.

context-string	Specifies the context string of the topic to display in a pop-up window if no matches are found. If this parameter is not specified, WinHelp displays a message box with the text "No additional information is available". To specify a topic in a different Help file, the topic ID should end with an '@' character and the name of the Help file.
window-name	Specifies the window in which to display the topic. If this parameter is not specified, the window that is specified for a topic (if one is defined) is used, or the default or current window is used. If this macro results in an interfile jump, the window must be defined in the project file for the Help file that is being jumped to.

Example

The following macro searches for the A-keyword "Engines":

ALink("Engines")

Comments

The ALink macro searches for A-keyword matches in the current Help file. If the Help file is associated with a contents file, WinHelp searches all Help files specified in the contents file (by the :Index and :Link

commands) for matching A-keywords.

The ALink macro is identical to the KLink macro except that it searches for A-keywords instead of K-keywords.

See also: KLink Macro

Annotate

This macro displays the Annotation dialog box (same as the Annotate command on the Edit menu).

Syntax

Annotate()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

If the Annotate macro is run from a pop-up window, the annotation is attached to the topic that contains the hot spot to the pop-up window.

AppendItem

This macro appends a menu item to the end of a menu you create with the InsertMenu macro.

Syntax

AppendItem("menu-id", "item-id", "item-name", "macro")

Parameters

Argument	Definition
menu-id	Name used in the InsertMenu macro to create the menu. This name must appear in quotation marks. The new item is appended to this menu.
item-id	Name that WinHelp uses internally to identify the menu item. This name is case-sensitive and must appear in quotation marks. Use this name in DisableItem, EnableItem, DeleteItem, ExtAbleItem, ChangeItemBinding macros if you want to disable/enable or remove the item, or change the operations that the item performs in certain topics.
item-name	Name that WinHelp displays on the menu for the item. This name is case-sensitive and must appear in quotation marks. Within the quotation marks, place an ampersand (&) before the character used as the macro's keyboard access key.
macro	Help macro or macro string that is run when the user chooses the menu item. The name must appear in quotation marks. Multiple macros in a string must be separated by semicolons (;).

Example

The following macro appends a menu item labeled "Index" to a menu that has an identifier "mnu_cards":

AppendItem("mnu_cards", "mnu_index", "&Index", "JI(`index.hlp', `index_topic')")

Choosing the menu item causes a jump to a topic with the context string "index_topic" in the INDEX.HLP file.

Comments

WinHelp ignores this macro if it is run in a secondary window.

Make sure that the keyboard access keys you assign to menu items are unique. If you assign a key that conflicts with other menu access keys, WinHelp displays the error message, "Unable to add item", and ignores the macro.

Back

This macro displays the previous topic in the Back list. The Back list includes the last 40 topics the user has displayed since starting WinHelp.

Syntax

Back()

Parameters

None

Comments

WinHelp ignores this macro if it is run in a secondary window.
If the Back macro is run when the Back list is empty, WinHelp takes no action.

BackFlush (Win95 Only)

Removes the back history list from the current window. This macro does not affect the history list displayed in the History window.

Syntax

BackFlush()

Parameters

None

Comments

None.

BookmarkDefine

This macro displays the Define dialog box (same as the Define command on the Bookmark menu).

Syntax

BookmarkDefine()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

If the BookmarkDefine macro is run from a pop-up window, the bookmark is attached to the topic that invoked the pop-up window.

BookmarkMore

This macro displays the More dialog box (same as the More command on the Bookmark menu). The More command appears on the Bookmark menu if the user has defined more than nine bookmarks.

Syntax

BookmarkMore()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

BrowseButtons

This macro adds the Browse Back (<<) and Browse Forward (>>) buttons to the toolbar in WinHelp.

Syntax

BrowseButtons()

Parameters

None

Example

The following macros in the Project Macro Definition cause the Notepad button to appear immediately before the two browse buttons on the toolbar:

```
CreateButton("&Notepad", "ExecProgram(`notepad', 0)")
BrowseButtons()
```

Comments

WinHelp ignores this macro if it is run in a secondary window.

If the BrowseButtons macro is used with one or more CreateButton macros Project Macro Definition, the order of the browse buttons on the WinHelp toolbar is determined by the order of the BrowseButtons macro in relation to the other macros listed in the Project Macro Definition.

Note:

WinHelp version 3.1 doesn't automatically provide Browse Forward (>>) and Browse Back (<<) buttons. The Help Magician will automatically insert the BrowseButtons macro into the Project Macro Definition if you have created browse sequences.

ChangeButtonBinding (CBB)

This macro assigns a Help macro to a Help button.

Syntax

ChangeButtonBinding("button-id", "button-macro")

Parameters

Argument	Definition
button-id	Identifier assigned to the button in the CreateButton macro or, for a standard Help button, one of the following predefined button identifiers: btn_contents (Contents), btn_search (Search), btn_back (Back), btn_history (History), btn_previous (<<), or btn_next (>>). The button identifier must appear in quotation marks.
button-macro	Help macro run when the user chooses the button. The macro must appear in quotation marks.

Example

The following macro changes the function of the Contents button so that choosing it causes a jump to the Table of Contents topic (identified by the context string "dict_contents") in the DICT.HLP file:

ChangeButtonBinding("btn_contents", "JumpId(`dict.hlp', `dict_contents')")

Comments

WinHelp ignores this macro if it is run in a secondary window.

ChangeEnable (Win95 Only)

Assigns a macro to a button bar button and enables that button.

Syntax

ChangeEnable("button-id", "button-macro")

Parameters

Argument	Definition
button-id	Identifier assigned to the button in the CreateButton macro or, for a standard Help button, one of the following predefined button identifiers: btn_contents (Contents), btn_search (Search), btn_back (Back), btn_history (History), btn_previous (<<), or btn_next (>>). The button identifier must appear in quotation marks.
button-macro	Help macro run when the user chooses the button. The macro must appear in quotation marks.

Example

The following macro assigns the JumpID macro to the Contents button and enables it:

ChangeButtonBinding("btn_contents", "JumpId(`dict.hlp', `dict_contents')"):

Comments

This macro is equivalent to calling both ChangeButtonBinding and EnableButton.

ChangeItemBinding (CIB)

This macro assigns a Help macro to an item that you add to a WinHelp menu using the AppendItem macro.

Syntax

ChangeItemBinding("item-id", "item-macro")

Parameters

Argument	Definition
item-id	Identifier assigned to the item in the AppendItem macro. The item identifier must appear in quotation marks.
item-macro	Help macro that is run when the user selects the item. The macro must appear in quotation marks.

Example

The following macro changes the menu item identified by "time_item" so that it starts the Windows Clock program:

ChangeItemBinding("time_item", "ExecProgram(`clock', 0)")

Comments

WinHelp ignores this macro if it is run in a secondary window.

CheckItem (CI)

This macro places a check mark beside a menu item added to a Help menu using the AppendItem macro.

Syntax

CheckItem("item-id")

Parameters

Argument	Definition
item-id	Identifier assigned to the item in the AppendItem macro. The item identifier must appear in quotation marks.

Example

The following macro places a checkmark beside the menu item named "syntax-item":

CheckItem("syntax-item")

Comments

Use the UncheckItem macro to clear the checkmark.
WinHelp ignores this macro if it is run in a secondary window.

CloseSecondarys (Win95 Only)

Closes all but the current secondary window.

Syntax

CloseSecondarys()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

If the BookmarkDefine macro is run from a pop-up window, the bookmark is attached to the topic that invoked the pop-up window.

CloseWindow

This macro closes the specified window, which is either the main WinHelp window or a secondary window.

Syntax

CloseWindow("window-name")

Parameters

Argument	Definition
window-name	The name of the window to close. The name "main" is reserved for the primary Help window. For secondary windows, the window name must be defined in the Help Windows Specifications form under the Options menu. This name must appear in quotation marks.

Example

The following macro closes the secondary window "keys":

CloseWindow("keys")

Comments

The secondary windows names can also be found in the [WINDOWS] section of the project (.HPJ) file. The project file is created every time the RTF file is written in the Help Magician.
If the window does not exist, WinHelp ignores the macro.

Compare (Win95 Only)

Displays a Help file in a second instance of WinHelp. The current Help file and the second Help file are displayed side-by-side. Most actions performed in one Help file (for example, clicking jumps or the Back and browse buttons) will be automatically reflected in the other file.

Syntax

Compare("helpfile")

Parameters

None

Example

The following macro opens the file "machines.hlp" and displays it side by side with the currently opened help file:

Compare("machines.hlp")

Comments

This macro is useful for comparing original and translated versions of the same Help file.

You can run this macro from the Jump dialog box by typing the following in the Enter Topic Identifier text box:

!compare("filename.hlp")

Contents

This macro displays the contents topic in the current Help file. The contents topic can be set in the Compiler Options form under the Options menu.

Syntax

Contents()

Parameters

None

Comments

The contents topic is defaulted to the first topic in the help file. If multiple files are used, the contents topic will come from the main help file.

ControlPanel (Win95 Only)

Opens a control panel applet with a specific tab on top.

Syntax

ControlPanel("CPL_name", "panel-name", tabnum)

Parameters

Argument	Definition
CPL_name	Specifies the name of the program that contains the control panel applet.
panel_name	Specifies the name of the control panel applet. This must be identical to the text that appears under the control panel applet's icon.
tabnum	Specifies the number of the tab to display on top. The first tab is number 0, the second tab is number 1, and so on.

Comments

Help Magician converts the ControlPanel macro into the ExecFile macro, which is in turn converted to EF. EF is the only form of this macro that WinHelp can use.

Not all control panel applets recognize the panel_name and tabnum parameters.

CopyDialog

This macro displays the Copy dialog box (same as the Copy command on the Edit menu).

Syntax

CopyDialog()

Parameters

None

Comments

This macro is obsolete for Windows 95. Please use the CopyTopic macro instead.

Use of this macro in secondary windows is discouraged.

CopyTopic (CT)

This macro copies all the text in the currently displayed topic onto the Windows Clipboard.

Syntax

CopyTopic()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

This macro copies text only; it does not copy bitmaps or any other images in the Help topic.

If the CopyTopic macro is run from a pop-up window, only the text of the topic that invoked the pop-up window is placed on the Clipboard.

A copyright notice can be set in the Appearance form under the Options menu.

CreateButton (CB)

This macro adds a new button to the WinHelp toolbar.

Syntax

CreateButton("button-id", "caption", "macro")

Parameters

Argument	Definition
button-id	Name that WinHelp uses internally to identify the button. This name must appear in quotation marks. Use this name in the DisableButton or DestroyButton macro if you want to remove or disable the button, or in the ChangeButtonBinding macro if you want to change the Help macro that the button runs in certain topics.
caption	The text that appears on the button. This caption must appear in quotation marks. To designate a letter as a keyboard access key for this button, place an ampersand (&) before a letter in this text. The button caption is case-sensitive and can contain up to 29 characters, beyond which the caption is clipped.
macro	Help macro or macro string that is run when the user chooses the button. The macro must appear in quotation marks. Multiple macros in a macro string must be separated by semicolons (;).

Example

The following macro creates a new button labeled "Order Info" that jumps to a topic with the context string "orderform" in the BUYPROG.HLP file when the button is chosen:

CreateButton("btn_order", "&Order Info", "JumpId(`buyprog.hlp', `orderform')")

Comments

WinHelp ignores this macro if it is run in a secondary window.

WinHelp allows a maximum of 16 authored buttons. It allows a total of 22 buttons, including the standard browse buttons, on the toolbar.

If the BrowseButtons macro is used with one or more CreateButton macros in the Project Macro Definition, the order of the browse buttons on the WinHelp toolbar is determined by where the BrowseButtons macro is listed in relation to the other macros in the Project Macro Definition. The button order may be changed in the Macro Editor form available from the Define Macros menu item under the Macros menu.

DeleteItem

This macro removes a menu item that was added using the AppendItem macro.

Syntax

DeleteItem("item-id")

Parameters

Argument	Definition
item-id	The item identifier string used in the AppendItem macro. The item identifier must appear in quotation marks.

Example

The following macro removes the menu item "Tools" appended in the example for the AppendItem macro:

DeleteItem("mnu_tools")

Comments

WinHelp ignores this macro if it is run in a secondary window.

DeleteMark

This macro removes a text marker added with the SaveMark macro.

Syntax

DeleteMark("marker-text")

Parameters

Argument	Definition
marker-text	Marker text specified in the SaveMark macro. The marker text must appear in quotation marks.

Example

The following macro removes the marker "Saving Money" from the Financial Help file:

DeleteMark("Saving Money")

Comments

If the marker does not exist when the DeleteMark macro is run, WinHelp displays a "Topic not found" error message.

DestroyButton

This macro removes a button added with the CreateButton macro.

Syntax

DestroyButton("button-id")

Parameters

Argument	Definition
button-id	Identifier assigned to the button in the CreateButton macro. The button identifier must appear in quotation marks. **The button identifier cannot duplicate an identifier used for one of the standard Help buttons.** (See the ChangeButtonBinding macro for a list of these identifiers.)

Comments

WinHelp ignores this macro if it is run in a secondary window.

DisableButton (DB)

This macro disables (rays out) a button added with the CreateButton macro.

Syntax

DisableButton("button-id")

Parameters

Argument	Definition
button-id	Identifier assigned to the button in the CreateButton macro. The button identifier appears in quotation marks.

Comments

WinHelp ignores this macro if it is run in a secondary window.

A button disabled by the DisableButton macro cannot be used in the topic until an EnableButton macro is run.

DisableItem (DI)

This macro disables and dims a menu item added with the AppendItem macro.

Syntax

DisableItem("item-id")

Parameters

Argument	Definition
item-id	Identifier assigned to the menu item in the AppendItem macro. The item identifier must appear in quotation marks.

Comments

WinHelp ignores this macro if it is run in a secondary window.

A menu item disabled by the DisableItem macro cannot be used in the topic until an EnableItem macro is run.

EnableButton (EB)

This macro re-enables a button disabled with the DisableButton macro.

Syntax

EnableButton("button-id")

Parameters

Argument	Definition
button-id	Identifier assigned to the button in the CreateButton macro. The button identifier must appear in quotation marks.

Comments

WinHelp ignores this macro if it is run in a secondary window.

EnableItem (EI)

This macro re-enables a menu item disabled with the DisableItem macro.

Syntax

EnableItem("item-id")

Parameters

Argument	Definition
item-id	Identifier assigned to the menu item in the AppendItem macro. The item identifier must appear in quotation marks.

Comments

WinHelp ignores this macro if it is run in a secondary window.

EndMPrint

Dismisses the printing message box and terminates the printing of multiple topics.

Syntax

EndMPrint()

Parameters

None

ExecFile (Win95 Only)

Runs a program or the program associated with a file.

Syntax

ExecFile("program", "arguments", display-state, "topic-ID")

Parameters

program	Specifies the name of the program to be run or the name of a file. If a file is specified, the program associated with that file type is started.
arguments	Specifies the command-line arguments to send to the program.
display-state	Specifies a value indicating how the program's window is to be shown. If this parameter is not specified, SW_SHOW is used (activates the window and shows it in its default size and position).
topic-ID	Specifies the ID of the topic to display if the specified file or program cannot be started.

Comments

If a path is specified, WinHelp first searches for the file in the specified path. If the file is not found or no path was specified, WinHelp searches the same locations as it does when it searches for Help files:

- The folder of the current Help file.
- The current folder.
- The System subfolder in the Windows folder.
- The Windows folder.
- The folders listed in the PATH environment.
- The location specified in the Winhelp.ini file.
- The Help portion of the registry.

Example

The following example opens the WIN.INI file in the program associated with .ini files:

ExecFile(win.ini)

ExecProgram (EP)

This macro runs a Windows-based application.

Syntax

ExecProgram("command-line", display-state)

Parameters

Argument	Definition
command-line	Command line for the application to be executed. The command line must appear in quotation marks. WinHelp searches for this application in the current directory, followed by the Windows directory, the user's path, and the directory of the currently displayed Help file.
display-state	A value indicating how the application is shown when executed. A value of 0 indicates normal, 1 indicates minimized, and 2 indicates maximized. For a complete list of display states, select the "Display" hints button.

Example

The following macro runs the Windows Clock program in its normal window size:

ExecProgram("clock.exe", 0)

Comments:

This macro is obsolete for Windows 95. Please use the ExecFile macro instead.

Exit

This macro exits the WinHelp application (same as the Exit command on the File menu).

Syntax

Exit()

Parameters

None

ExtAbleItem

This macro enables or disables an existing menu item. It is functionally equivalent to the combination of the EnableItem and DisableItem macros.

Syntax

ExtAbleItem("item-id", enabled-state)

Parameters

Argument	Definition
item-id	Identifier assigned to the menu item in the AppendItem or ExtInsertItem macros. The item identifier must appear in quotation marks.
enabled-state	An integer specifying the initial state of the menu item. Use a '0' to enable the item and '1' to disable the item.

Example

The following macro disables the menu item "mnu_tools":

ExtAbleItem("mnu_tools", 1)

Comments

WinHelp ignores this macro if it is run in a secondary window.

ExtInsertItem

This macro functions identically to the InsertItem macro with one exception: it has an extra argument to enable or disable the item on startup.

Syntax

ExtInsertItem("menu-id", "item-id", "item-name", "macro", position, enabled-state)

Parameters

Argument	Definition
menu-id	Either a standard WinHelp menu name or the name used in the InsertMenu macro to create the menu. Standard menu names are mnu_file (File menu), mnu_edit (Edit menu), mnu_bookmark (Bookmark menu), and mnu_helpon (Help menu). The menu identifier must appear in quotation marks. The new item is inserted into this menu.
item-id	Name that WinHelp uses internally to identify the menu item. The item identifier must appear in quotation marks. Use this name in DisableItem, EnableItem, DeleteItem, ExtAbleItem, ChangeItemBinding macros if you want to disable/enable or remove the item, or change the operations that the item performs in certain topics.
item-name	Name that WinHelp displays on the menu for the item. This name is case-sensitive and must appear in quotation marks. Within the quotation marks, place an ampersand (&) before the character used for the item's keyboard access key.
macro	Help macro or macro string that is run when the user chooses the menu item. The macro must appear in quotation marks. Multiple macros in a string must be separated by semicolons (;).
position	An integer specifying the position in the menu where the new item will appear. Position 0 is the first or topmost position in the menu.
enabled-state	An integer specifying the initial state of the menu item. Use a '0' to enable the item and '1' to disable the item.

Example

The following macro inserts a menu item labeled "Tools" as the third item (0,1,2) on a menu that has an identifier "mnu_books":

ExtInsertItem("mnu_books", "mnu_tools", "&Tools", "JI(`tools.hlp', `first_topic')", 3, 0)

Selecting the menu item causes a jump to a topic with the context string "first_topic" in the TOOLS.HLP file.

Comments

WinHelp ignores this macro if it is run in a secondary window.

Make sure that the keyboard access keys you assign to menu items are unique. If you assign a key that conflicts with other menu access keys, WinHelp displays the error message Unable to add item and ignores the macro.

Use "mnu_main" for the menu-id to place an item directly on the WinHelp main menu bar.

Use "mnu_floating" to place an item in the floating menu. Remember to execute the FloatingMenu() macro to enable the floating menu option.

ExtInsertMenu

This macro functions similarily to the InsertMenu macro with a couple of enhancements:

1. It has an extra argument to allow placing the menu under other menus as a sub-menu.
2. It has an extra argument to enable or disable the item on startup.

Syntax

ExtInsertMenu("parentmenu-id", "submenu-id", "submenu-name", submenu-position, enabled-state)

Parameters

Argument Definition

parentmenu-id Either a standard WinHelp menu name or the name used in the InsertMenu macro to create the menu. Standard menu names are . The menu identifier must appear in quotation marks. The new item is inserted into this menu. If "mnu_main" is used, the menu will be inserted into WinHelp's main menu bar.

submenu-id Name that WinHelp uses internally to identify the submenu item. The item identifier must appear in quotation marks. Use this name in DisableItem, EnableItem, DeleteItem, ExtAbleItem, ChangeItemBinding macros if you want to disable/enable or remove the item, or change the operations that the item performs in certain topics.

submenu-name Name that WinHelp displays on the submenu for the item. This name is case-sensitive and must appear in quotation marks. Within the quotation marks, place an ampersand (&) before the character used for the submenu's keyboard access key.

submenu-position An integer specifying the position in the parent menu where the new submenu will appear. Position 0 is the first or topmost position in the parent menu.

enabled-state An integer specifying the initial state of the menu item. Use a '0' to enable the item and '1' to disable the item.

Example #1

The following macro inserts a submenu item labeled "Tools" as the third item (0, 1, 2) on a parent menu that has an identifier "mnu_books":

ExtInsertMenu("mnu_books", "mnu_tools", "&Tools", 2, 0)

Example #2

The following macro inserts a menu labeled "Charts" as the second item (0, 1) on WinHelp's main menu:

ExtInsertMenu("mnu_main", "mnu_chts", "&Charts", 1, 0)

Example #3

The following macros insert a submenu labeled "File" as the first item on WinHelp's floating menu and adds four menu items to it:

```
ExtInsertMenu("mnu_floating", "mnu_ffile", "&File", 0, 0)
AppendItem("mnu_ffile", "itm_file1", "&Open", "FileOpen()")
AppendItem("mnu_ffile", "itm_file2", "&Print Topic", "Print()")
AppendItem("mnu_ffile", "itm_file3", "P&rint Setup...", "PrinterSetup()")
AppendItem("mnu_ffile", "itm_file4", "&Exit", "Exit()")
```

To see this example function, click the right mouse button to display the floating menu.

Comments

WinHelp ignores this macro if it is run in a secondary window.

Make sure that the keyboard access keys you assign to menu items are unique. If you assign a key that conflicts with other menu access keys, WinHelp displays the error message Unable to add menu and ignores the macro.

Use "mnu_main" for the menu-id to place a menu directly on the WinHelp main menu bar.

Use "mnu_floating" to place a sub-menu in the floating menu. Remember to execute the FloatingMenu() macro to enable the floating menu option.

FileExist (Win95 Only)

Checks to see whether the specified file or program exists.

Syntax

FileExist("filename")

Parameters

filename	Specifies the name of the file.

Comments

This macro can be used in conjunction with macros such as IfThenElse, which use the result of a Boolean macro to determine what action to take.

If a path is specified, WinHelp first searches for the file in the specified path. If the file is not found, or no path was specified, WinHelp searches the same locations it does to find a Help file:

- The folder of the current Help file.
- The current folder.
- The System subfolder in the Windows folder.
- The Windows folder.
- The folders listed in the PATH environment.
- The location specified in the Winhelp.ini file.
- The Help portion of the registry.

Example

The following macro checks to see if "Myapp.exe" has been installed. If the file is present, WinHelp runs it. If the file is not present, WinHelp displays a topic:

IfThenElse(FileExist(myapp.exe), ExecFile(myapp), JumpId(install_my_app))

FileOpen

This macro displays the Open dialog box (same as the Open command on the File menu).

Syntax

FileOpen()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

Find (Win95 only)

Displays the Find tab in the Help Topics dialog box.

Syntax

Find()

Parameters

None

Finder (Win95 only)

Displays the Help Topics dialog box.

Syntax

Finder()

Parameters

None

FloatingMenu

This macro displays the floating menu, providing items have been added to it.

Syntax

FloatingMenu()

Parameters

None

Comments

Menu items should be added to the floating menu before calling this macro.

Typically the user would click the right mouse button to display the floating menu, but this macro provides the help author (you) a way to display it from a button or a hotspot, for example.

You can add as many as 20 menu items to the context menu.

Example

The following macros insert a submenu labeled "File" as the first item on WinHelp's floating menu and adds four menu items to it:

```
ExtInsertMenu("mnu_floating", "mnu_ffile", "&File", 0, 0)
AppendItem("mnu_ffile", "itm_file1", "&Open", "FileOpen()")
AppendItem("mnu_ffile", "itm_file2", "&Print Topic", "Print()")
AppendItem("mnu_ffile", "itm_file3", "P&rint Setup...", "PrinterSetup()")
AppendItem("mnu_ffile", "itm_file4", "&Exit", "Exit()")
Floatingmenu()
```

Click the right mouse button to see an example of the floating menu.

Flush (Win95 only)

Causes WinHelp to process any pending messages, including previously called macros.

Syntax

Flush()

Parameters

None

FocusWindow

This macro changes the focus to the specified window, which is either the main WinHelp window or a secondary window.

Syntax

FocusWindow("window-name")

Parameters

Argument	Definition
window-name	The name of the window to have the focus. The name "main" is reserved for the primary Help window. For secondary windows, the window name must be defined in the Help Windows Specifications form under the Options menu. This name must appear in quotation marks.

Example

The following macro changes the focus to the secondary window "glossary":

FocusWindow("glossary")

Comments

If the window does not exist, WinHelp ignores the macro.

Generate (Win95 only)

Posts a message to the currently active Help window.

Syntax

Generate("message", "wParam", "iParam")

Parameters

message	Specifies a message to send to the currently active Help window.
wParam	Specifies the first argument of the message.
lParam	Specifies the second argument of the message.

GoToMark

This macro jumps to a marker set with the SaveMark macro.

Syntax

GoToMark("marker-text")

Parameters

Argument	Definition
marker-text	Marker text specified in the SaveMark macro. The marker text must appear in quotation marks.

Example

The following macro jumps to the marker "Saving Money" in the Financial Help file:

GoToMark("Saving Money")

HAnimateCommand

This is currently a non-supported Viewer macro.
Its inclusion at this point is strictly for future version compatibility.

HAudioCommand

This is currently a non-supported Viewer macro.
Its inclusion at this point is strictly for future version compatibility.

HAudioDialog

This is currently a non-supported Viewer macro.
Its inclusion at this point is strictly for future version compatibility.

HDisplayBitmap

This is currently a non-supported Viewer macro.
Its inclusion at this point is strictly for future version compatibility.

HelpOn

This macro displays the Using Help file for the WinHelp application (same as the Using Help command on the Help menu).

Syntax

HelpOn()

Parameters

None

HelpOnTop

This macro forces the Help window to always stay on top of other applications so that it is always visible.

Syntax

HelpOnTop()

Parameters

None

Comments

Use of this macro with the main window is discouraged. To set the "on top" feature for the main window, check the "Always On Top" menu item off the Help menu in WinHelp. Windows Help dows not provide a way to monitor the state of the "on top" feature so it is up to the user to track the status of it.

History

This macro displays the Windows Help History window, which shows the last 40 topics the user has viewed since opening a Help file in WinHelp. It has the same effect as choosing the History button on the WinHelp toolbar.

Syntax

History()

Parameters

None

Comments

WinHelp ignores this macro if it is run in a secondary window.

The number of topics kept in the history list can be changed by using the BACKTRACK switch in the Win.ini file.

IfThen

This macro runs a Help macro if a given marker exists. It uses the IsMark macro to make the test.

Syntax

IfThen(IsMark("marker-text"), "macro")

Parameters

Argument	Definition
marker-text	Marker text tested by the IsMark macro. The marker text must appear in quotation marks.
macro	The Help macro or macro string that is run if the marker exists. The macro must appear in quotation marks. Multiple macros in a macro string must be separated by semicolons (;).

Example

The following macro jumps to the topic with the context string "bal_chk" if a marker named "Balance Checkbook" has been set by the SaveMark macro:

IfThen(IsMark("Balance Checkbook"), "JI(`finance.hlp', `bal_chk')")

IfThenElse

This macro runs one of two Help macros, provided a marker exists. It uses the IsMark macro to make the test.

Syntax

IfThenElse(IsMark("marker-text"), "macro1", "macro2")

Parameters

Argument	Definition
marker-text	Marker text tested by the IsMark macro. The marker text must appear in quotation marks.
macro1, macro2	WinHelp runs macro1 if the marker exists and macro2 if it does not. Both macros must appear in quotation marks. Multiple macros in either macro string must be separated by semicolons (;).

Example

The following macro jumps to the topic with the context string "bal_chk" if a marker named "Balance Checkbook" has been set by the SaveMark macro. If the marker does not exist, it jumps to the contents screen for the FINANCE.HLP file:

IfThenElse(IsMark("Balance Checkbook"), "JI(`finance.hlp', `bal_chk')","JumpContents(`FINANCE.HLP')")

InitMPrint (Win95 only)

Initializes WinHelp in preparation for printing multiple topics.

Syntax

InitMPrint()

Parameters

None

Comments

This macro displays the Printer Setup dialog box. If a user clicks OK, the InitMPrint macro returns TRUE. Otherwise, it returns FALSE. This macro should be used as the first parameter in an IfThen macro to ensure that printing multiple topics occurs only if a user clicks OK.

InsertItem

This macro inserts a menu item at a given position on an existing menu. The menu can be either one you create with the InsertMenu macro or one of the standard WinHelp menus.

Syntax

InsertItem("menu-id", "item-id", "item-name", "macro", position)

Parameters

Argument	Definition
menu-id	Either a standard WinHelp menu name or the name used in the InsertMenu macro to create the menu. Standard menu names are mnu_file (File menu), mnu_edit (Edit menu), mnu_bookmark (Bookmark menu), and mnu_helpon (Help menu). The menu identifier must appear in quotation marks. The new item is inserted into this menu.
item-id	Name that WinHelp uses internally to identify the menu item. The item identifier must appear in quotation marks. Use this name in DisableItem, EnableItem, DeleteItem, ExtAbleItem, ChangeItemBinding macros if you want to disable/enable or remove the item, or change the operations that the item performs in certain topics.
item-name	Name that WinHelp displays on the menu for the item. This name is case-sensitive and must appear in quotation marks. Within the quotation marks, place an ampersand (&) before the character used for the item's keyboard access key.
macro	Help macro or macro string that is run when the user chooses the menu item. The macro must appear in quotation marks. Multiple macros in a string must be separated by semicolons (;).
position	An integer specifying the position in the menu where the new item will appear. Position 0 is the first or topmost position in the menu.

Example

The following macro inserts a menu item labeled "Tools" as the third item on a menu that has an identifier "mnu_books":

InsertItem("mnu_books", "mnu_tools", "&Tools", "JI(`tools.hlp', `first_topic')", 3)

Selecting the menu item causes a jump to a topic with the context string "first_topic" in the TOOLS.HLP file.

Comments

WinHelp ignores this macro if it is run in a secondary window.

Make sure that the keyboard access keys you assign to menu items are unique. If you assign a key that conflicts with other menu access keys, WinHelp displays the error message Unable to add item and ignores the macro.

InsertMenu

This macro adds a new menu to the WinHelp menu bar.

Syntax

InsertMenu("menu-id", "menu-name", menu-position)

Parameters

Argument	Definition
menu-id	Name that WinHelp uses internally to identify the menu. The menu identifier must appear in quotation marks. Use this identifier in the AppendItem macro to add commands to the menu.
menu-name	Name for the menu that WinHelp displays on the menu bar. This name is case-sensitive and must appear in quotation marks. Within the quotation marks, place an ampersand (&) before the character used for the menu's keyboard access key.
menu-position	Number telling WinHelp which position on the menu bar the new menu name will have. Positions are numbered from left to right, with position 0 being the leftmost menu.

Example

The following macro adds a menu named "Utilities" to WinHelp:

InsertMenu("menu_util", "&Utilities", 3)

The label "Utilities" appears as the fourth menu on the WinHelp menu bar. The user presses ALT+U to display the menu and its commands.

Comments

WinHelp ignores this macro if it is run in a secondary window.

Make sure that the keyboard access keys you assign to menus are unique. If you assign a key that conflicts with other menu access keys, WinHelp displays the error message "Unable to add menu" and ignores the macro.

IsBook (Win95 only)

Determines whether WinHelp is running as a standalone system (a double-clicked book icon), or if it is being run from a program. This macro can be used as the first parameter of a IfThen or IfThenElse macro to take specific action depending on whether the current Help file is being run as a double-clicked book icon.

Syntax

IsBook()

Parameters

None

Comments

The IsBook macro returns a value of TRUE if WinHelp is being run as a book; otherwise, the macro returns FALSE.

IsMark

This macro determines whether a marker set by the SaveMark macro exists. It is used as a parameter to the conditional macros IfThen and IfThenElse.

Syntax

IsMark("marker-text")

Parameters

Argument	Definition
marker-text	Marker text tested by the IsMark macro. The IsMark macro returns a True value if the mark exists and a False value if it does not. The marker text must appear in quotation marks.

Example

The following macro jumps to the topic with the context string "bal_chk" if a marker named "Balance Checkbook" has been set by the SaveMark macro:

```
IfThen(IsMark("Balance Checkbook"), "JI(`finance.hlp', `bal_chk')")
```

Comments

The Not macro can be used to reverse the results of the IsMark macro.

IsNotMark (Win95 only)

Tests whether or not a marker that was set by the SaveMark macro exists.

Syntax

IsNotMark("marker-text")

Parameters

Argument	Definition
marker-text	Marker text tested by the IsNotMark macro. The IsNotMark macro returns FALSE if the mark exists or TRUE if it does not. The marker text must appear in quotation marks.

Comments

The IsNotMark macro is used as a parameter in the conditional macros IfThen or IfThenElse. This result is the exact opposite of the implicit IsMark macro that is usually used for the IfThen and IfThenElse macros.

JumpContents

This macro jumps to the contents topic of a specified Help file. The contents topic can be set in the Compiler Options form under the Options menu.

Syntax

JumpContents("filename")

Parameters

Argument	Definition
filename	The name of the destination file for the jump. The file name must appear in quotation marks. If WinHelp cannot find this file, it displays an error message and does not perform the jump.

Example

The following macro jumps to the contents topic of the PROGMAN.HLP file:

JumpContents("PROGMAN.HLP")

Comments

If the CONTENTS option is not specified, WinHelp jumps to the first topic in the Help file.
WinHelp ignores this macro if it is run in a secondary window.
Use the word "THISFILE" to refer to the current help file.

JumpContext (JC)

This macro jumps to a topic identified by a context number. The context number must be entered in the "Context #" field in the Help Magician.

Syntax

JumpContext("filename", context-number)

Parameters

Argument	Definition
filename	The name of the destination file for the jump. The file name must appear in quotation marks. If WinHelp cannot find this file, it displays an error message and does not perform the jump.
context-number	Context number of the topic in the destination file. The context number must assigned for the topic to be recognized. If the context number does not exist or cannot be found, WinHelp jumps to the contents topic or the first topic in the file instead, and displays an error message.

Example

The following macro jumps to the topic mapped to the context number 1501 in the PROGMAN.HLP file:

JumpContext("PROGMAN.HLP", 1501)

Comments

Use the word "THISFILE" to refer to the current help file.

JumpHash (Win95 only)

Jumps to a topic identified by a hash code.

Syntax

JumpHash("filename>window", hash-code)

Parameters

Argument	Definition
filename	Specifies the name of the Help file containing the hash number. This optional parameter is used when jumping to a topic that is not in the current Help file.
window-name	Specifies the window type in which to display the topic.
hash-code	Specifies the hash number of the topic in the destination file.

Comments

You can use the Report command in Help Magician to list the hash numbers of all the topics in a Help file. This enables you to display information about a Help file for which you have no topic, project, or contents files.

JumpHelpOn

This macro jumps to the contents topic of the "Using Help" file. The "Using Help" file is either the default WINHELP.HLP file or the Help file designated by the SetHelpOnFile macro.

Syntax

JumpHelpOn()

Parameters

None

Example

The following macro jumps to the contents topic of the designated Using Help file:

JumpHelpOn()

Comments

If WinHelp cannot find the specified Help file, it displays an error message and does not perform the jump.

JumpId (JI)

This macro jumps to the topic with the specified context string in the Help file.

Syntax

JumpId("filename", "context-string")

Parameters

Argument	Definition
filename	Name of the Help file (.HLP) containing the context string. The file name must appear in quotation marks. If WinHelp does not find this file, it displays an error message and does not perform the jump.
context-string	Context string of the topic in the destination file. The context string must appear in quotation marks. If the context string does not exist, WinHelp jumps to the contents topic for that file instead.

Example

The following macro jumps to a topic with "second_topic" as its context string in the Help file SECOND.HLP:

JumpId("second.hlp", "second_topic")

Comments

Use the word "THISFILE" to refer to the current help file.

JumpKeyword (JK)

This macro opens the indicated Help file (.HLP), searches through the keyword table, and displays the first topic containing the keyword specified in the macro.

Syntax

JumpKeyword("filename", "keyword")

Parameters

Argument	Definition
filename	The name of the .HLP file that contains the desired keyword table. The file name must appear in quotation marks. If this file does not exist, WinHelp displays an error message and does not perform the jump.
keyword	The keyword that the macro searches for. The keyword must appear in quotation marks. If WinHelp finds more than one match, it displays the first matched topic. If it does not find any matches, it displays a "Not a keyword" message and the contents topic of the destination file.

Example

The following macro opens the first topic that has "hands" as an index keyword in the Help file CLOCK.HLP:

JumpKeyword("clock.hlp", "hands")

Comments

Use the word "THISFILE" to refer to the current help file.
WinHelp searches through the 'K' keyword table, the default table that the Help Magician generates. Multiple keyword tables are ignored.

This macro is functionally equivalent to the KLink macro.

KLink (Win95 only)

Searches for keywords specified by K-footnotes.

Syntax

KLink("keyword", type, "topicID", window)

Parameters

Argument	Definition
keyword	Specifies one or more keywords to search for. Separate multiple keywords using a semicolon. If any keyword contains a comma, enclose the entire keyword string in quotation marks.
type	Specifies the action to perform if one or more keywords is found. If this parameter is not specified or is zero, the default action is to display the Topics Found dialog box containing the topic title. This parameter can be one or more of the following values, separated by spaces.

Value	Meaning
JUMP (1)	Specifies that if only one topic matches any of the keywords, WinHelp should jump directly to that topic.
TITLE (2)	Specifies that if a keyword is found in more than one Help file, WinHelp should display the title of the Help file (as specified in the contents [.cnt] file) beside the topic title in the Topics Found dialog box.
TEST (4)	Specifies that the macro should return a value indicating whether or not there is at least one match. The TestKLink macro is converted by Help Workshop into an KLink macro with this parameter.

topic-ID	Specifies the topic to display in a pop-up window if no matches are found. If this parameter is not specified, WinHelp displays a message box with the text “No additional information is available”. To specify a topic in a different Help file, the topic ID should end with an ‘@’ character and the name of the Help file.
window	Specifies the window in which to display the topic. If this parameter is not specified, the window that is specified for a topic (if one is defined) is used, or the default or current window is used. If this macro results in an interfile jump, the window must be defined in the project file for the Help file that is being jumped to.

Examples

The following example generates a list of topics that contain the "network printing" or "local printing" keywords in K-footnotes:

KLink(network printing; local printing)

The following example generates a list of topics that contain the keyword "how_to" in K-footnotes and displays those topics in the Steps window:

KLink(how_to, , , Steps)

Comments

The KLink macro searches for K-keyword matches in the current Help file. If the Help file is associated with a contents file, WinHelp searches all Help files specified in the contents file (by the :Index and :Link commands) for matching K-keywords.

The KLink macro is identical to the ALink macro except that it searches for K-keywords instead of A-keywords.

mciExecute

Basic Commands

The following list summarizes the basic commands. The use of these messages by a device is optional.

Command	Description
load	Recalls data from a disk file.
pause	Pauses playing or recording.
play	Starts transmitting output data.
record	Starts recording input data.
resume	Resumes playing or recording on a paused device.
save	Saves data to a disk file.
seek	Seeks forward or backward.
set	Sets the operating state of the device.
status	Obtains status information about the device. The status command is also listed in the group of required commands. In the basic group, options are added for devices that use linear media with identifiable positions.
stop	Stops playing or recording.

Comments

If a driver supports a basic command, it must also support a standard set of options for the command.

This is not a standard WinHelp macro and needs to be registered using the RegisterRoutine macro should you be creating the project file manually.

Menu

This macro displays the Define dialog box (same as the Define command on the Bookmark menu).

Syntax

BookmarkDefine()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

If the BookmarkDefine macro is run from a pop-up window, the bookmark is attached to the topic that invoked the pop-up window.

MPrintHash (Win95 only)

Prints a topic identified by a hash code. This macro must be used in conjunction with the InitMPrint and EndMPrint macros.

Syntax

MPrintHash(hash-code)

Parameters

Argument	Definition
hash-code	Specifies the hash number of the topic to be printed.

Comments

You can use the Report command in Help Magician to list the hash codes of all the topics in a Help file. This enables you to display information about a Help file for which you have no topic, project, or contents files.

MPrintID (Win95 only)

Prints a topic. This macro must be used in conjunction with the InitMPrint and EndMPrint macros.

Syntax

MPrintID("topicID")

Parameters

Argument	Definition
topicID	The topic ID (context string) of the topic to be printed.

Next

This macro displays the next topic in the browse sequence for the Help file. It has the same effect as choosing the Browse Forward (>>) button.

Syntax

Next()

Parameters

None

Comments

If the current topic is the last of a browse sequence, this macro does nothing.

WinHelp ignores this macro if it is run in a secondary window.

NoShow (Win 95 only)

Prevents a Help window from being displayed if it has not already been displayed.

Syntax

NoShow()

Parameters

None

Not

This macro reverses the True or False result returned by the IsMark macro. It is used with the IsMark macro as a parameter to the conditional macros IfThen and IfThenElse.

Syntax

Not(IsMark("marker-text")

Parameters

Argument	Definition
marker-text	Marker text tested by the IsMark macro. The Not macro returns a False value if the mark exists or a True value if it does not. The marker text must appear in quotation marks.

Example

The following macro jumps to the topic with context string "Settle Accounts" if a marker named "Balance Checkbook" has not been set by the SaveMark macro:

IfThen(Not(IsMark("Balance Checkbook")), "JI(`finance.hlp', `Settle Accounts')")

PopupContext (PC)

This macro displays a topic identified by a context number. The context numbers are entered via the Help Magician's "Context #" field.

Syntax

PopupContext("filename", context-number)

Parameters

Argument	Definition
filename	The name of the file that contains the topic to be displayed in the pop-up window. The file name must appear in quotation marks. If WinHelp cannot find this file, it displays an error message.
context-number	Context number of the topic to be displayed in the pop-up window. The context number must entered in the "Context #" entry field in the Help Magician. If the context number does not exist or cannot be found, WinHelp displays the contents topic or the first topic in the file instead.

Example

The following macro displays in a pop-up window the topic mapped to the context number 1501 in the file PROGMAN.HLP:

PopupContext("progman.hlp", 1501)

Comments

Use the word "THISFILE" to refer to the current help file.

PopupHash (Win95 only)

Displays in a pop-up window the topic identified by a hash code.

Syntax

PopupHash("filename", hash-code)

Parameters

Argument	Definition
filename	Specifies the name of the destination Help file for the jump. Use this optional parameter if the pop-up is not in the current Help file.
hash-code	Specifies the hash code of the topic to be printed.

Comments

You can use the Report command in Help Magician to list the hash numbers of all the topics in a Help file. This enables you to display information about a Help file for which you have no topic, project, or contents files.

PopupId (PI)

This macro displays a topic from a specified file in a pop-up window.

Syntax

PopupId("filename", "context-string")

Parameters

Argument	Definition
filename	The name of the file that contains the pop-up window topic. The file name must appear in quotation marks. If this file does not exist, WinHelp displays an error message.
context-string	Context string of the topic in the destination file. The context string must appear in quotation marks. If the requested context string does not exist, WinHelp displays the contents topic or the first topic in the file in the pop-up window. Context strings can be entered in the "Context $" entry field in the Help Magician.

Example

The following macro displays in a pop-up window a topic identified by the context string "second_topic" in the file SECOND.HLP:

PopupId("second.hlp", "second_topic")

Comments

Use the word "THISFILE" to refer to the current help file.

PositionWindow (PW)

This macro sets the size and position of the main Help window or an existing secondary window.

Syntax

PositionWindow(x, y, width, height, state, "window-name")

Parameters

Argument	Definition
x, y	X and Y coordinates of the upper-left window corner. Positions are defined in terms of WinHelp's 1024 x 1024 coordinate system.
width, height	Gives the default width and height of the window. Window sizes, like positions, are defined in terms of WinHelp's coordinate system.
window-state	Specifies how the window is sized. This parameter is 0 for normal size and 1 for maximized. If the parameter is 1, WinHelp ignores the x, y, width, and height parameters.
window-name	The name of the window to position. The name "main" is reserved for the primary Help window. Secondary windows must be defined in the Help Windows Specifications form under the Options menu in the Help Magician. This name must appear in quotation marks.

Example

The following macro positions the secondary window "Samples" in the upper-left corner (100, 100) with a width and height of 500 (in WinHelp coordinates):

PositionWindow(100, 100, 500, 500, 0, "Samples")

Comments

If the window to be positioned does not exist, WinHelp ignores the macro.

Prev

This macro displays the previous topic in the browse sequence for the Help file. It has the same effect as choosing the Browse Back (<<) button.

Syntax

Prev()

Parameters

None

Comments

If the currently displayed topic is the first topic of a browse sequence, this macro does nothing. WinHelp ignores this macro if it is run in a secondary window.

Print

This macro sends the currently displayed topic to the printer.

Syntax

Print()

Parameters

None

Comments

This macro should be used only to print topics in windows other than the main Help window. For example, it can be used to print topics displayed in secondary windows, provided the user doesn't have a dialog box open at the time of printing.

Use of this macro in secondary windows is discouraged.

PrinterSetup

This macro displays the Print Setup dialog box (same as the Print Setup command on the File menu).

Syntax

PrinterSetup()

Parameters

None

Comments

Use of this macro in secondary windows is discouraged.

This macro is obsolete for Windows 95. WinHelp always displays the Print Setup dialog box before printing.

RegisterRoutine (RR)

This macro registers a function within a DLL as a Help macro. Registered functions can be used in macro hot spots, topic macros, or project macros, just as standard Help macros are used. (It should be noted that the Help Magician performs all necessary macro registrations for you automatically when the RTF file is written.)

Syntax

RegisterRoutine("DLL-name", "function-name", "format-spec")

Parameters

Argument	Definition
DLL-name	The file name of the DLL being called. The file name must appear in quotation marks. If WinHelp cannot find the DLL, it displays an error message and does not perform the call.
function-name	The name of the function to be executed in the designated DLL. The function name must appear in quotation marks.
format-spec	A string specifying the formats of parameters passed to the function. The format string must appear in quotation marks. Characters in the string represent C parameter types: "u" for unsigned short, "U" for unsigned long, "i" for short int, "I" for long int, "s" for string (near char *), "S" for string (far char *), or "v" for void. WinHelp automatically makes sure these formats match the parameter types specified in the function prototype.

Example

The following DLL call registers a routine "RetString" in the DLL named HELPLIB.DLL. RetString takes arguments of types far char *, short int, and unsigned long.

RegisterRoutine("HELPLIB", "RetString", "S=iU")

Comments

This is an advanced level macro and its use is discouraged for novices.

RemoveAccelerator

This macro removes a Help macro from a shortcut key that had been previously assigned.

Syntax

RemoveAccelerator(key, shift-state)

Parameters

Argument	Definition
key	The Windows virtual-key value. For a list of these keys, select the **Acc Keys** hints button.
shift-state	A number specifying the combination of ALT, SHIFT, and CTRL keys used with the shortcut key: 0 (none), 1 (SHIFT), 2 (CTRL), 3 (SHIFT+CTRL), 4 (ALT), 5 (ALT+SHIFT), 6 (ALT+CTRL), or 7 (ALT+SHIFT+CTRL).

Example

The following macro removes the macro assigned to the key sequence Ctrl-P:

RemoveAccelerator(0x50, 2")

Comments

WinHelp does not display an error message if you try to remove an unassigned accelerator key.
Use the AddAccelerator macro to assign a macro to a key combination.

ResetMenu

This macro returns the WinHelp menus to the default configuration.

Syntax

ResetMenu()

Parameters

None

SaveMark

This macro saves the location of the currently displayed topic and file and associates a text marker with that location. The GoToMark macro can then be used to jump to this location.

Syntax

SaveMark("marker-text")

Parameters

Argument	Definition
marker-text	Text used to identify the topic location. The marker text must appear in quotation marks, and it must be unique. If the same text is used for more than one marker, WinHelp recognizes only the most recently entered marker.

Example

The following macro saves the marker "Balance Checkbook" in the current topic in the Financial Help file:

SaveMark("Balance Checkbook")

Comments

In addition to GoToMark, WinHelp offers the following other macros for use with text markers:

DeleteMark removes any defined marker.

IsMark tests whether a given marker has been set in the Help file. Not negates the result of this test.

IfThen and IfThenElse run one or more Help macros if a given marker has been set. These use the IsMark (and optional Not) macro to test whether the marker is set.

Text markers are not saved if the user exits and then restarts WinHelp.

Search

This macro displays the dialog box for the Search button, which allows users to search for topics using keywords defined in the Help Magician. It has the same effect as choosing the Search button.

Syntax

Search()

Parameters

None

Comments

WinHelp ignores this macro if it is run in a secondary window.

SetContents

This macro designates a specific topic as the contents topic within the Help file.

Syntax

SetContents("filename", context-number)

Parameters

Argument	Definition
filename	The name of the Help file that contains the desired contents topic. The file name must appear in quotation marks. If WinHelp cannot find the file, it displays an error message and does not perform the jump.
context-number	Context number of the topic in the specified file. The context number must be assigned in the destination Help file. If the context number does not exist or cannot be found in the destination help file, WinHelp displays an error message.

Example

The following macro sets the topic mapped to the context number 1501 in the PROGMAN.HLP file as the contents topic:

SetContents("PROGMAN.HLP", 1501)

After running this macro, pressing the Contents button causes a jump to the specified topic.

Comments

Use the word "THISFILE" to refer to the current help file.

SetHelpOnFile

This macro designates the specific Help file that replaces WINHELP.HLP, the default Using Help file in the Windows environment.

Syntax

SetHelpOnFile("filename")

Parameters

Argument	Definition
filename	The name of the replacement Using Help file. The file name must appear in quotation marks. If WinHelp cannot find this file, it displays an error message.

Example

The following macro sets the Using Help file as MYHELP.HLP:

SetHelpOnFile("myhelp.hlp")

Comments

If this macro appears within a topic in the Help file, the replacement file is set after execution of the macro. If this macro appears in the Project Macro Definition, the replacement file is set when the Help file is opened.

Comments

Use the word "THISFILE" to refer to the current help file.

This macro is obsolete for Windows 95. WinHelp version 4.0 will ignore this macro.

SetPopupColor (Win95 only)

Sets the background color for all subsequent pop-up windows.

Syntax

SetPopupColor(r, g, b)

Parameters

Argument	Definition
r	Specifies the red component of the color. This value is an integer in the range 0 to 255.
g	Specifies the green component of the color. This value is an integer in the range 0 to 255.
b	Specifies the blue component of the color. This value is an integer in the range 0 to 255.

Comments

After this macro is run, the set color applies to all pop-up topics.

Example

The following macro sets the background color for pop-up windows to blue:

SetPopupColor(0, 0, 255)

ShellExecute (Win95 only)

Opens or prints the specified file.).

Syntax

ShellExecute("filename", "options", show-flag, "operation", "path", "topic-id")

Parameters

Argument	Definition
filename	Specifies the name of the file to open.
options	Specifies parameters passed to the program when the filename parameter specifies an executable (.exe) file. If the filename parameter specifies a document file, this parameter is empty.
show-flag	Specifies how the program is shown when it is opened. If the filename parameter specifies a document file, this parameter should be zero.

Value	Meaning
HIDE (0)	Hides the window.
MAXIMIZE (3)	Activates the window and maximizes it.
MINIMIZE_ACTIVATE (6)	Activates the window and minimizes it.
MINIMIZE_NOACTIVE (7)	Minimizes the window, but WinHelp keeps the focus.
NOACTIVATE (4)	Displays a window in its current state, but WinHelp keeps the focus. (If the window was minimized before this call, it stays minimized.)
NORMAL (5)	Activates the window and displays it in its current size and position.
RESTORE (9)	Restores the window to its original size and position if the window is minimized or maximized.

operation	Specifies the operation to perform. This parameter can be "open" or "opencpl" or "print". If this parameter is not specified, "open" is the default value.
path	Specifies the default folder. Use “” if you do not need to specify a default folder.
topic-id	Specifies the ID of the topic to display if this macro fails.

Comments

The filename can be a document file or an executable file. If it is a document file, this function opens or prints it, depending on the value of the operation parameter. If it is an executable file, this function opens it, even if the operation parameter specifies print.

If no path is specified with the filename, WinHelp searches the following locations in the order listed:

- The folder in which the Winhelp.exe program is located

- The folder that contains the current Help file
- The System subfolder in the Windows folder
- The Windows folder
- The PATH environment
- The registry

The ShellExecute macro calls the Windows ShellExecute function.

ShortCut (Win95 only)

Runs the specified program if it is not already running. If the specified program is running, WinHelp activates it. If the wParam parameter is specified, a WM_COMMAND message with the specified wParam and lParam values are sent to the program.

Syntax

ShortCut("window-class", "program", "wParam", "lParam", "topic-ID")

Parameters

Argument	Definition
window-class	Specifies the class name of a top level window of the program. WinHelp uses this class name to determine if the program is already running.
program	Specifies the executable name of the program. This is the name of the program that runs if the window class name cannot be found. The .exe extension does not need to be included.
wParam	Specifies the first argument to the WM_COMMAND message that is sent to the program. If this value is not specified, the program is started, but no message is sent and the program is not activated.
lParam	Specifies the second argument to the WM_COMMAND message.
topic-ID	Specifies the topic ID of the topic to jump to if the program cannot be found. If this parameter is not specified, WinHelp displays a message box indicating that the program could not be found. If it is specified and no filename is included, the filename specified by the :Base command in the contents (.cnt) file is used. If there is no associated contents file, the current Help file is used. To specify a specific filename, the topic ID should end with an '@' character and the name of the Help file.

Comments

If you need to include a topic ID, but do not want to send the program any messages, you can either use -1 for the wParam argument and 0 for the lParam argument, or you can simply include the commas without specifying the values for the arguments as follows:

ShortCut(class_name, myapp, , , topic-ID)

Some programs, such as the Microsoft Visual C++ compiler (MSVC), cannot receive a message in the same call that runs them. In this case, you must first call the ShortCut macro without specifying wParam or lParam. You can then call this macro again and send the wParam and lParam messages. Note that the second time you call this macro, you can use an empty string ("") for the program name.

sndPlaySound

This macro plays a sound (.WAV) file.

Syntax

sndPlaySound("filename", play-flag)

Parameters

Argument	Definition
filename	The name of the sound (.WAV) file. The file name must appear in quotation marks. If WinHelp cannot find this file, it displays an error message.
play-flag	A flag that determines how long the sound plays for. Enter a '0' to terminate normally, '8' to play continuosly until the macro is called again with the '0' play-flag set.

Example

The following macro plays the "CHIMES.WAV" file:

sndPlaySound("chimes.wav", 0)

Comments

This is not a standard WinHelp macro and needs to be registered using the RegisterRoutine macro should you be creating the project file manually.

TCard (Win95 only)

Sends a message to the program that is invoking WinHelp as a training card.

Syntax

TCard("command")

Parameters

Argument	Definition
command	Specifies one of the following commands or a numeric value. If one of the following values is specified, its numeric equivalent is sent as the wParam value of the WM_TCARD message:

Value	Meaning
IDABORT (3)	The user clicked an authorable Abort button.
IDCANCEL (2)	The user clicked an authorable Cancel button.
IDCLOSE (8)	The user closed the training card.
IDHELP (9)	The user clicked an authorable Help button.
IDIGNORE (5)	The user clicked an authorable Ignore button.
IDOK (1)	The user clicked an authorable OK button.
IDNO (7)	The user clicked an authorable No button.
IDRETRY (4)	The user clicked an authorable Retry button.
IDYES (6)	The user clicked an authorable Yes button.

If a numeric value is used, the training card program is sent HELP_TCARD_DATA (16) for the wParam parameter, and the numeric value is passed as the lParam value of the WM_TCARD message.

Test (Win95 only)

Runs an internal WinHelp test.

Syntax

Test(test-num)

Parameters

Argument	Definition
test-num	Specifies the testing option. This parameter can be one of the following values:

Value	Description
1	Displays all the topics in the Help file, starting with the current topic.
2	Displays all the topics in the Help file, starting with the first topic.
3	Continuously displays all the topics in the Help file, starting with the first topic.
4	Displays all the topics in the Help file, starting with the first topic, and then quits.
5	Jumps to all topics specified in the contents file excluding macros.
6	Jumps to all topics specified in the contents file, excluding macros, and then quits.
7	Turns on Help Mode for the current instance of WinHelp only, and then launches a second instance of WinHelp on the same Help file. Both instances of WinHelp resize their windows so they can appear side-by-side and both display the first topic in the current Help file. Moving through one file (using CTRL+SHIFT commands) updates both windows. The functionality is the same as the Compare macro.

TestALink (Win95 only)

Tests whether an ALink macro has an effective link to at least one topic.

Syntax

TestALink("keyword")

Parameters

Argument	Definition
keyword	Specifies one or more A-keywords to search for. Separate multiple keywords using semicolons. If a keyword contains a comma, enclose the entire keyword string in quotation marks.

Comments

If an effective link is found, TestALink returns a value of "1".

The TestALink macro is typically used as the first parameter in an IfThen or IfThenElse macro.

Example

The following macro enables or disables a SeeAlso button, depending on whether at least one topic contains a "print" keyword:

IfThenElse(TestALink(print), ChangeEnable(btn_seealso, ALink(print)), DisableButton(btn_seealso))

TestKLink (Win95 only)

Tests whether an KLink macro has an effective link to at least one topic.

Syntax

TestKLink("keyword")

Parameters

Argument	Definition
keyword	Specifies one or more K-keywords to search for. Separate multiple keywords using semicolons. If a keyword contains a comma, enclose the entire keyword string in quotation marks.

Comments

If an effective link is found, TestKLink returns a value of "1".

The TestKLink macro is typically used as the first parameter in an IfThen or IfThenElse macro.

Example

The following macro enables or disables a SeeAlso button, depending on whether at least one topic contains a "print" keyword:

IfThenElse(TestKLink(print), ChangeEnable(btn_seealso, KLink(print)), DisableButton(btn_seealso))

UnCheckItem

This macro removes a check mark beside a menu item added to a Help menu using the AppendItem macro.

Syntax

UncheckItem("item-id")

Parameters

Argument	Definition
item-id	Identifier assigned to the item in the AppendItem macro. The item identifier must appear in quotation marks.

Example

The following macro removes a checkmark beside the menu item named "syntax-item":

UncheckItem("syntax-item")

Comments

Use the CheckItem macro to set the checkmark.
WinHelp ignores this macro if it is run in a secondary window.

UpdateWindow (Win95 only)

Jumps to the topic with the specified context string in the specified window, and then returns the focus to the window that called the macro.

This macro displays the Define dialog box (same as the Define command on the Bookmark menu).

Syntax

UpdateWindow("filename>window-state", "context-string")

Parameters

Argument	Definition
filename	Specifies the name of the Help file to jump to, if it is not the current Help file. If specified, this parameter must be followed by a greater than sign (>).
window-name	Specifies the window type to display the topic in.
topic-ID	Specifies the ID of the topic to jump to.

Comments

If the secondary window was not already created, the UpdateWindow macro is ignored.
This macro is usually used in an entry macro to update the secondary window whenever the topic is shown.

Example

This macro displays the topic "snippet" in the window "codewin" from the file "projects.hlp":

UpdateWindow("projects.hlp>codewin", "snippets")

Unknown Prototype

This prototype is not a standard WinHelp macro.

Example Syntax

SampleMacro(arg1, "arg2")

arg1 - Numerical argument. Enter any integer number in decimal or hex (ex. 3 or 0xF7).

arg2 - String argument. Enter any combination of text within the double (or single) quotes. Some characters need to be preceded with a backslash (\) such as an opening or closing parenthesis "()", single or double quotes, and the backslash.

a hash code is a unique number generated form a topic ID (context string). It is this number and not the topic ID that WinHelp uses to reference the topic.