MiniHelp Plus v3.1 Help File

Table of Contents

Introduction to MiniHelp Plus v3.1

Welcome to MiniHelp Plus v3.1, the improved version of MiniHelp Plus v1.0, and MiniHelp, that appeared in the January, 1994 issue of COMPUTE magazine. For details on what is new in this version, click on the **What's New** link in the "Learning About MiniHelp Plus v3.1" section.

To define the scope of this Help file, it will be assumed that you already have some knowledge of Help file writing. Yet even if you are a beginner to writing your own Help files, MiniHelp Plus v3.1 should be very easy for you to learn, because there are only **twenty-four** (yup, only 24) *topic commands* to remember and **nine** (only nine) *project commands* to remember. That adds 28 new commands to the features of the original MiniHelp.

What you do not need is to fork out hundreds of dollars for a word processor like Microsoft Word for Windows that is capable of RTF file output because MiniHelp Plus v3.1 takes your ordinary ASCII text file (with the inclusion of the 31 simple MiniHelp Plus v3.1 commands) and turns it into an RTF file. And what's even better, it is **only $15.00** (add $2.50 Shipping & Handling if you want me to send you a copy on disk). You are on the honor system, though. MiniHelp Plus v3.1 has not been disabled in any way. All that I ask is that you register your copy if you find MiniHelp Plus v3.1 useful. There are no nag screens. Registered users will be entitled to one free upgrade, when one becomes available.

What you will need is the Windows Help Compiler. HCP is the only Help Compiler supported by MiniHelp Plus v3.1. It will create the final Help file for Windows 3.1 or better. HCP is available in free distribution and can be found in the download area of America On-Line, Compuserve and on many local bulletin boards around the country. HCP was chosen because it is the only version of the Help Compiler that utilizes the Extended Memory on your system. HC31, I find, works well with smaller Help files or Help files with very few graphics. However, throw in a Hot Spot graphic and HC31 returns an "out of memory" message, since it is only capable of utilizing 640K of memory. This problem does not occur with HCP, and I've done Help files with lots of graphics and Hot Spot graphics. I've yet to receive an "out of memory" message.

You will also need an ASCII text editor you are comfortable with. I find that Windows Notepad works well for me, despite its shortcomings. Whatever you choose, just make sure it is capable of pure ASCII text file output. You may use Windows Write, but please take care NOT to include any special formatting commands--headers, footers, bold, italic, underline, etc--and make sure you save the file as an ASCII .TXT file.

Now, with all that out of the way, lets get down to the business of how to use MiniHelp Plus v3.1. Simply *"Click"* on the words highlighted in green to learn about the MiniHelp Plus v3.1 commands and syntax. Alright, let's write some Help!

Click here to learn more about MiniHelp Plus v3.1

Click Here

The $400 or better word processor for Windows from the company owned by Computing Guru Bill Gates. You could, if you want, buy it and contribute to his retirement fund. Or you could use MiniHelp Plus v3.1 and save the money for your own retirement! Anyway, MiniHelp Plus v3.1 is much, much easier to use, as well as being "infinitely faster," according to one user who has attempted to create Help files with RTF capable word processors.

Rich Text Format, the file type output by many high $$$ word processing programs. It is also the format or file type used by the Windows Help Compiler to build the Help file.

Learning About MiniHelp Plus v3.1

Writing Help files with MiniHelp Plus v3.1 is **very** easy. In fact, this Help file was written with MiniHelp Plus v3.1. In this section, we will deal with What's New and the commands and their syntax. ***It is very important that you follow the syntax and rules very explicitly, or your resultant RTF file will be corrupt.*** (Sorry, but there is no error checking for the RTF file in MiniHelp Plus v3.1.) Also, the default file extension used by MiniHelp Plus v3.1 is **".src".** MiniHelp Plus v3.1 requires that the ".src" file extension is used (since that is how MiniHelp Plus v3.1 saves the file in its own private ".INI" file). Also, by using the ".src" extension, you will be able to distinguish your MiniHelp Plus v3.1 source files from any other text files that may be on your system.

What's New in MiniHelp Plus v3.1

Topic Commands & Syntax

NOTE: Items with two stars (**) next to them have new features!

.link

.pop **

.topic

.title

.keyword **

.browse

.bmlink

.bitmap

.rbitmap

.lbitmap

.macrolink **

.tmacro

.bold

.italic

.boldital

.ul

.ulbold

.ulitalic

.ulboldital

.cfore

.fontsize

.include

Sample MiniHelp Source File (.src)

Project Commands & Syntax

NOTE: MiniHelp Plus v3.1 automatically generates a HPJ file for you that is ready to go and has all the basics of the HPJ file, so all of these commands are optional.

.hpjopt **

.hpjfiles

.hpjmap

.hpjconfig

.hpjbitmap

.hpjbtag

.hpjalias

.hpjwin

.hpjbaggage

Learning About MiniHelp Plus v3.1

What's New in MiniHelp Plus v3.1

Version History

MiniHelp Plus v3.1 adds many new, significant features to MiniHelp Plus v1.0, v2.0, and v2.3, and signifies a quantum leap in improvement over the original MiniHelp. The original, while adequate for simple Help files and novices, leaves out a lot of the commands us "seasoned" Help file writers use a lot. MiniHelp Plus v1.0 took care of that with the addition of 15 (fifteen) new commands. In addition, you can embed native RTF commands in the text (was available in MiniHelp Plus v1.0) That will be addressed in a separate section of the file, called RTF Commands.

The original commands were:

.topic, .title, .keyword, .link (undocumented in the original), and **.fontsize.**

The new commands in MiniHelp Plus v1.0 were:

.bmlink, .bitmap, .browse, .macrolink, .tmacro, .bold, .italic, .boldital, .cfore, and **.include.**

MiniHelp Plus v2.3 added these commands:

.rbitmap, lbitmap, .ul, .ulbold, .ulitalic, .ulboldital, .hpjopt, .hpjfiles, hpjbitmap, .hpjconfig, .hpjmap, .hpjbaggage, .hpjbtag, .hpjwin, and .hpjalias.

Here's what MiniHelp Plus v3.1 adds (and the list is long!):

- **Finally,** MiniHelp Plus v3.1 has an easy-to-use Windows® front-end shell/user interface. Now, running MiniHelp Plus v3.1 is as simple as pointing at a button and clicking the mouse button! Now, you no longer have to go into File Manager (File Mangler as some affectionately refer to it) to run MiniHelp Plus v3.1 via the "drag-n-drop" method or run MiniHelp Plus v3.1 from the command line. In fact, you can no longer use these methods with MiniHelp Plus v3.1 since it accepts no command line arguements. The original MiniHelp Plus program code has been intricately married and has become a part of the front end shell. I had originally intended to write either a Notepad add in or another (geez...there are so many of these!) text editor (yawn) dedicated to processing MiniHelp Plus v3.1 source files. But I thought that was TOO restrictive. That would mean that everyone who used MiniHelp would have to become accustomed to using yet another text editor. Then I thought, what about a simple "ToolPad" from which to launch MiniHelp Plus vx.x! (Lightbulb goes on!) Now this way, everyone can use the text editor of his/her choosing and that he/she has become accustomed to. So you can run MiniHelp Plus v3.1 as an icon on the desktop, and when you are done typing in your source file, you can open up the shell, click a couple of buttons and be on your way quickly to finishing your Help file project!

- MiniHelp Plus v3.1 uses **its own** private ".INI" file (mhplus31.ini) to store its configuration information in. You need only configure MiniHelp Plus v3.1 once, at the beginning of each ***different*** Help file writing project. And what's best, nothing is ever added to your WIN.INI file (and we all know how some programmers like to bloat your already over-bloated WIN.INI file!). All the information MiniHelp Plus v3.1 needs is stored right in the mhplus.ini file, which is placed in your

Windows directory. But be sure to **first** configure MiniHelp Plus v3.1 before running it, or you will be alerted with a message telling you to do so first.

• The Project Error File option (errorlog=filename.err) is ***automatically*** inserted into the OPTIONS section of the HPJ file created by MiniHelp Plus v3.1 so that you can view the errors generated by the Help Compiler during the compilation of your Help file. Simply click on the **Error File** button and this file will open in Windows® Notepad for you to view at your leisure.

• New additions to the **.pop** and **.macrolink** commands allow you to define either text or a bitmapped image to use as the hot spot that performs the link. Briefly, simply issue the command as you normally would (for those of you lacking experience with previous versions of MiniHelp Plus, refer to these commands in this Help file), except you specify a bitmapped image's filename in the quotes with **.bmp** as the last four (4) characters of the text. That's all there really is to it! MiniHelp Plus v3.1 automatically scans what you have typed in and redirects the RTF file generation appropriately. Pretty nifty, eh?!

• You can view the Help file you have created with MiniHelp Plus v3.1 and the Help Compiler simply by clicking on the **View Help** button.

• MiniHelp Plus v3.1 allows you to launch your favorite text editor with a click of the **Text Editor** button. You may specify your default ASCII text editor when you configure MiniHelp Plus v3.1. If none is specified, MiniHelp Plus v3.1 will launch Windows® Notepad by default.

For a complete description of how to use the MiniHelp Plus v3.1 front end shell/user interface, be sure to view the accompanying Help file, "ToolPad Help."

Additionally, MiniHelp Plus v3.1, as did MiniHelp Plus v2.3, *automatically* generates a basic Help Project File (HPJ) at the same time it is converting your ASCII text file into an RTF file (this feature is the reason for nine (9) of the MiniHelp Plus v3.1 commands). The commands are there to allow you to manipulate and enhance the HPJ file that is generated. MiniHelp Plus v3.1 also *automatically* generates an optional C-type header file (file extension is **.hhh)** for generating **context-sensitive** Help, just as did previous versions. MiniHelp Plus v3.1 generates this whether you want it or not. Whether you utilize it or not, is up to you, but it is produced anyway. If you choose not to use it, simply delete the header file. However, once you gain more understanding of Windows Help and the Help macros, I think you will want to retain and use it in your Help file writing endeavors.

Let me explain the large jump in version numbers. There was a MiniHelp Plus v2.5 that I chose **not** to release. First of all, it didn't seem to be that significant of an improvement (to me anyway). Second of all, I wanted to make the next release available with a front end shell/user interface. While I was working on that shell, I discovered after reviewing my source code that it in fact would not be as difficult as I had originally thought to make this into one complete package. In fact, the shell was completed and worked admirably when I made the decision to combine as much of the shell and MiniHelp Plus v2.5 into one (not two, as the MiniHelp Plus v2.5 shell dictated) complete program. I felt this did the job without sacrificing any of MiniHelp Plus v2.5's functionality and usefulnes, plus it yielded several additional benefits that I felt enhanced the "product" (i.e., the automatic generation of the project error file, the ability to configure MiniHelp Plus to run from only one click of the mouse button, and of course, a friendly, easy to use front end shell/user interface.) Therefore, the version numbering left the realm of MiniHelp Plus *v2.something* because of its obvious benefits and increased features.

Why have MiniHelp Plus v3.1 generate your HPJ file instead of writing it out yourself? Well, first of all, you are dealing with only one file. Second of all, if you have changes to make in it, you can do it right here instead of switching between two different documents. Thirdly, it guarantees you consistent results. That is, every time you compile the source data from MiniHelp Plus v3.1, the results the 100th time will be the same as the first time you compiled it. Fourth, it's easy and convenient. Why not use it: it's generated automatically. Just let the program work for you. I know this is a feature I would have really

appreciated when I was first learning how to write Help files. It would have freed me from having to learn a lot about how the Help Compiler (HC) uses the HPJ file at the beginning of my Help file writing learning curve.

See the list of commands under **Commands & Syntax** for an detailed explanation of each command. As you can see, MiniHelp Plus v3.1, just as in previous versions, relies exclusively on the use of "dot" commands, that is the command preceeded by a period. When entering "dot" commands, enter them on their own line (e.g., if you are typing along and want a word in boldface, hit enter without a space preceeding it, type the *.bold* command followed by a space and the text you want in boldface. Hit enter again (without a space), and continue typing your text).

Note that all the "dot" commands must have spaces between the command and the text/parameter it is defining.

Learning About MiniHelp Plus v3.1

.link

Use the command **.link** to create a hot spot in your text that jumps you to another topic. The syntax is as follows:

.link CONTENTS "Table of Contents"

This example would link you with the topic, "CONTENTS" and display the text, "Table of Contents" in green on your Help screen. Thus, when you clicked on the words displayed in green, you would be wisked away to the topic named "CONTENTS." Although it is not necessary to enter the topic name in all caps, it is recommended for readability. They tend to grab your attention when you are reading your source file. **Make sure you enter a space at the end of the previous line before issuing this command so that a space appears before your highlighted text.** Otherwise, the highlighted text will run on with the previous word in the sentence without a space preceding it.

Changes

Learning About MiniHelp Plus v3.1

.pop

The **.pop** command creates what is known as a Popup Link. First of all, Popup Links are distinguished from regular Links by their appearance on the screen. Popup Links are underlined with a dashed line, while Links have a solid underline. Secondly, a Popup Link doesn't jump you to another portion of your Help document. Instead, it's information is displayed in a little window of its own. Popup Links are very good for displaying definitions and offering an extended explanation of a subject without taking the reader away from the current topic. For an example of this, click either on the phrase Microsoft Word for Windows or RTF.

The proper syntax for **.pop** is exactly the same as for the **.link** command. It is as follows:

.pop MICROSOFT "Microsoft Word for Windows"

This would link the Popup Link with the topic MICROSOFT and display the text "Microsoft Word for Windows" in green on the screen with a dashed underline. This is exactly how the Popup Link in the Introduction section is defined. **Make sure you enter a space at the end of the previous line before issuing this command so that a space appears before your highlighted text.** Otherwise, the highlighted text will run on with the previous word in the sentence without a space preceding it.

Remember, all "dot" commands must be on a line of their own. If this still isn't clear, take a look at the sample source file included with MiniHelp Plus v3.1.

BUT WAIT A MINUTE!!

There is more! The previous definition was for MiniHelp Plus v2.3, v2.0, and v1.0. This version, MiniHelp Plus v3.1, adds more functionality to the **.pop** command. NOW, you can define *either* text or a bitmapped image as a popup link. Here is how and why: when you enter the text to display on the screen enclosed in quotes, MiniHelp Plus v3.1 checks to see whether or not the last four characters inside the quotes are **.bmp.** If they are, MiniHelp Plus v3.1 redefines the output to the RTF file and reformats it so a bitmapped image is displayed. If it does not find **.bmp** as the last four characters of the text enclosed in quotes, it then formats the text in the usual way, to display text on the screen in green with a dashed green underline.

For example:

.pop IDEA1 "key1.bmp"	produces a popup link displaying the bitmap key1.bmp.
.pop IDEA1 "my idea"	produces a popup link displaying "my idea" as text, in green, with a green dashed underline.

Be sure, when defining a bitmapped image as a hot spot for a popup link, **do not** include any path information; that should be defined in the HPJ file under [OPTIONS] with the BMROOT options. You really should be keeping your house in good order by keeping all your bitmapped images in one directory (at least the ones you are using in your Help files, anyway.)

Learning About MiniHelp Plus v3.1

.topic

All topics in a Help file must be defined. This is the phrase the Help Compiler, and subsequently, the Help Engine, use to "map" its way around. If a topic had no name, it would have no idea of where to take you when you wanted to make a jump by clicking on a Hot Spot. Your topic name ***must*** be typed exactly as you have it defined in your link. No spaces are allowed in your topic name; only alphanumerics (a-z, A-Z, 0-9, and the underline) are allowed. Stick with those, and you'll never get into trouble. The proper syntax is as follows:

.topic MICROSOFT

Although it is not required, I strongly recommend making all your topic names all CAPITAL letters. That way, they stand out in your source file so you can spot them easily and quickly. Believe it or not, this can be a very big help!

Learning About MiniHelp Plus v3.1

.title

.title is the command that gives a title to your topic. This command is optional but the Help Compiler will issue you a warning if it comes across a topic without a title; it expects them all to be titled. The title is used in the Search dialog box to assist with searching for specific content related to certain word(s) or phrases and is displayed in the History dialog box when the History button is pressed on the button bar. The syntax, as follows, is exactly the same as for **.topic** except that spaces are allowed:

.title Microsoft Word for Windows

Changes

Learning About MiniHelp Plus v3.1

.keyword

The **.keyword** command lists additional phrases and word to place in the "Search" dialog box. This is very helpful to your Help file reader since it allows him or her to search for a topic using several different "keywords." This command, although optional, is strongly recommended for this reason. The syntax is:

.keyword Bill Gates

RESTRICTIONS

With this command, you are restricted to 255 characters per line and the keywords must be separated with semicolons (;) and no spaces before or after the semicolon. Caution: Be sure not to add an 's' to the command or it will not work! Thus, if you want to provide three or four keywords for the user to use in the "Search" dialog box, you would have to make three or four separate entries, or separate the words/phrases with semicolons. For example, with the MICROSOFT topic cited earlier in the **.pop** command, the keyword phrases are:

.keyword Microsoft
.keyword Bill Gates or

.keyword Microsoft;Bill Gates

Thus, the words "Microsoft" and "Bill Gates" would be listed in the "Search" dialog box, and would direct the Help file reader to the topic MICROSOFT, if selected.

Learning About MiniHelp Plus v3.1

.browse

The **.browse** command allows you to specify the Browse sequence through a Help file. In case you are unfamiliar about what I mean, some Help files have the browse buttons **(<< or >>)** included in the button bar. When you press one of these, it jumps you to the next topic in the sequence. This command allows you to tell the Help Compiler which topic is to be shown next if one of these buttons is pushed. To enable the browse buttons in your Help file, see the the HPJ section of this Help file.

This command is optional, and need not be included if you do not plan to include the browse buttons in your Help file.

Syntax

.browse 001
.browse LEARN:007

As you can see from the syntax, you decide how functional you want the browse sequencing to be. In the first example, you simply use a number. Note the leading zeros. These are necessary because the Help Compiler uses an ASCII sorting techniqe, not a numeric sort. You may use any number of zeros but make sure you are consistent throughout your source file, using the same number of digits each time. Otherwise, a higher sequence number could appear before a lower sequence number. For example, 100 is numerically higher than 99, but 100 will appear long before 99 does, because Help is comparing the first two digits in the number string. To keep things in their proper order, you must make 99 to a three digit number, 099. In the first example, you would assign each topic a browse sequence with its own unique number, incrementing that number for each browse sequence specified.

In the second example, a topic name is specified (LEARN), followed by a colon, then the number. This way, this particular topic has its browse sequence linked with another specified topic. Thus, unless you are within the topic specified in the browse sequence, you can not browse through those topics with the browse buttons. This is very effective for grouping related topics together. No spaces are allowed. The second example, by the way, is the exact browse sequence given to this topic, browse.

RESTRICTIONS

- Whichever way you decide to number your sequences (example 1 or example 2), you must be consistent throughout your source file. Otherwise, you risk generating an error.

- Only **one** browse sequence per topic. Otherwise, you will generate an error.

- BE CONSISTENT WITH YOUR NUMBERING! If you assign the same number to two different topics, you will either generate an error or the compiler will drop one of them from the sequence. And don't forget the leading zeros.

Learning About MiniHelp Plus v3.1

.bmlink

.bmlink creates a Hot Spot out of a bitmap picture. It allows you to insert a bitmap as a character in the text and link it with a topic. When the Help file reader clicks on the picture, they will be taken to the topic specified in the link.For an example, see below (click on the bitmap graphic below).

Syntax

The syntax for this command is exactly the same as for the *.link* command, with one exception: you place the name of the bitmap within the quotes, instead of text to appear on the screen.

EXAMPLE

.bmlink BMLINK "chereup.bmp"

Will simply link you with the top of this topic.

Do not include any path information; that will be specified in your Help Project File. Also, it is easiest to do if all your files (MiniHelp Plus v3.1, the source file, the Help Project File, your RTF files, your bitmap files, and the Help Compiler) are in one directory. That way, path information is not necessary. Also, since the bitmap is treated just like text, all paragraph and text formating RTF commands can be applied (see the section RTF Commands for more information concerning applicable RTF Commands).

Learning About MiniHelp Plus v3.1

.bitmap

The **.bitmap** command inserts a bitmap into your text and treats it as a character in the line of text. Therefore, just as with the bmlink command, all paragraph and text formatting commands can be applied (See the section "RTF Commands"). Unlike the bmlink command, this command simply inserts the bitmap *without* creating a link to another topic. Some possible uses include adding bullets to your printed text (see the browse command for an example), adding a decorative picture to the topic pages, or adding emphasis to some of your text. At the bottom of this topic, you can see some examples.

Syntax

.bitmap ***filename.bmp***

Note that there are no quotes around the filename. Simply type the command, space, filename. It's that easy. Make sure you specify your bitmap in the BITMAP section of the Help Project File.

EXAMPLES

.bitmap recorder.bmp
This bitmap appears at the left of the text.

produces this result:

 This bitmap appears at the left of the text.

This bitmap
.bitmap recorder.bmp
appears in the middle of the text.

produces this result:

This bitmap  appears in the middle of the text.

This bitmap appears at the right of the text.
.bitmap recorder.bmp

produces this result:

This bitmap appears at the right of the text. 

See how easy that is! Simply place the bitmap in the spot you want it to show up in in the text.

Changes

Learning About MiniHelp Plus v3.1

.macrolink

.macrolink creates a Hot Spot in the text and links it with one of Help's many macro commands. When the highlighted text is clicked on, the macro is executed. It is formatted exactly like the link command and the bmlink command.

Additionally, the macrolink command works just like the pop command: it, too will accept and define a bitmapped image that, when clicked on, will execute a Help macro. It automatically checks to see if the characters **".bmp"** are at the end of the text in quotes. If it is, the specified bitmapped image is inserted as a hot spot. If ".bmp" is not found, then it is processed as to display text on the screen in green with a solid green underline.

Syntax

.macrolink ***Name of Macro*** **"Text to display on screen"**

.macrolink ***Name of Macro*** **"filename.bmp"**

Below is a partial list of some of the macros you can include and what they do. Unless otherwise noted, the parenthesis stay empty. There are many more macros (52 Help macros total) but there is not space here to list them. For example, three other macros too complex to describe here were used to create the additional menu you see on the menu bar and to insert the extra menu items in the "File" menu. Refer to either Microsoft Windows 3.1: Programmer's Reference, Volume 4, Resources (Chapter 15 of this book covers all the macros in detail.) or get a copy of the invaluable Microsoft Help Writers Authoring Guide (available in free distribution on many BBS's around the country).

About() : Displays the "About Help..." dialog box.

BrowseButtons() : Enables the browse buttons on the button bar.

Contents() : Jumps the Help file reader back to the topic identified as the contents in the Help Project File. (If it is not defined, then Help defaults to the first topic it encounters.

CopyTopic() : Copies the current topic of the main help window to the clipboard.

CreateButton(parameters **) :** Typically, this is one you include in your Help Project File (See the "HPJ" section). You can however, include it in your Help file text. For example, to make it easy for the Help file reader to print out a particular topic, you might include a button for that purpose when you enter that topic. And this macro can have other macros nested in it as parameters.

CreateButton("btn_print","&Print","Print()")

where "btn_print" is the button-id, "&Print" is the text displayed on the button (the "&" signifies the letter to enable its use from the keyboard, in this case, the 'P'; always place the ampersand in front of the letter you want to use in this way), and the macro (in this case, the Print() macro).

The above definition of "CreateButton" will create a "Print" button on the button bar, that, when clicked on by the user, will print out the current topic.

DestroyButton("button-id") : Removes a button from the button bar.

DisableButton("button-id") : Disables (grays out the print) on a button, instead of removing it from the button bar.

EnableButton("button-id") : Enables a disabled button on the button bar.

Exit() : Exits Help when called.

Print() : Prints the current topic when called.

EXAMPLES

.macrolink About() "About Help"	About Help
.macrolink CopyTopic() "Copy this topic"	Copy this topic
.macrolink Print() "Print this topic"	Print this topic

Learning About MiniHelp Plus v3.1

.tmacro

The **.tmacro** command causes a macro to be executed whenever a topic is entered. The command must appear within a topic (after you have defined the topic), just like when you entered this topic and the "About Help" dialog box was displayed. It ***MUST*** be the first command right after **.topic.**

For a partial listing of macros, see the macrolink section.

Syntax

.tmacro ***Macro Name***

Learning About MiniHelp Plus v3.1

Typeface Commands

.bold, .italic, .boldital, .ul,

.ulbold, .ulitalic, .ulboldital

The typeface commands, **.bold, .italic, .boldital, .ul, .ulbold, .ulitalic,** and **.ulboldital,** control how the text in your Help file appears on the user's screen. As you can probably already tell, the different commands relate directly to how you want the text to appear in your Help file.

For example, if you want some of your text to appear as **boldface** type, simply type the **.bold** command on a line of its own, followed by the text you want in boldface. To my knowledge, there is no limit to how much text can appear after one of the typeface commands. Below, is the syntax and some examples of how to use the commands.

Syntax

.typefacecommand *text to display*

EXAMPLES

This will print out
.bold text
in boldface.

Will print out as:

This will print out **text** in boldface.

.ul This line of text should be underlined.

Will print out as:

This line of text should be underlined.

.ulitalic This text will be italic and underlined.

Will print out as:

This text will be italic and underlined.

This will print
.boldital bolditalic
print on the screen.

Will display as:

This will print ***bolditalic*** print on the screen.

.ulbold This line is in boldfaced type and should be underlined.

Will print out as:

<u>This line is in boldfaced type and should be underlined.</u>

.ulboldital This line will be underlined and in bold italics.

Will print out as:

<u>This line will be underlined and in bold italics.</u>

Be sure **not** to overuse the underlining command, **.ul,** since it can quickly become an overwhelming and confusing situation for your Help file reader to see green and black underlining all over the screen. Used judiciously, however, the **.ul** and other typeface commands can effectively add the emphasis you need to the point you are trying to make.

Learning About MiniHelp Plus v3.1

.include

The **.include** command is a special command designed to help overcome the file size limitations of many of the text editors (i.e. Windows Notepad and others). Notepad, for example, is limited to a space of approximately 23 to 27 KB with WordWrap turned on, and about 38 KB with it turned off. If you find that you are running out of space when typing your source file, simply type this command followed by the *filename.extension.* ***Remember, the file extension used by MiniHelp Plus is*** **.SRC.** This is important, or MiniHelp Plus v3.1 will not run. What MiniHelp Plus does when it encounters this command, is open up the second file you specified and compile it all into an RTF file that has the name of the very first file. Thus, if you were typing along in a file called "myhelp.src" and issued the command **.include myhelp2.src,** MiniHelp Plus would go right along and compile "myhelp.src" and "myhelp2.src" into one RTF file called "myhelp.rtf."

RESTRICTION

The **.include** command **MUST, MUST, MUST** be the very last command in your file. If you issue it before then, any text following the command will NOT be compiled.

You may, however, link several files together, just as long as the **.include** command is the last command listed in each file.

Twips

There are 1440 twips in an inch (thus, one twip equals 1/1440 of an inch). To compute out inch equivalencies, multiply the inches by 1440. Therefore, 1/4 inch is 360 twips, 1/2 inch is 720 twips, 1 inch is 1440 twips, 1.5 inches is 2160 twips, and so on. This allows for ABSOLUTE screen positioning.

The Help Project File - HPJ

The Help Project File (hereinafter referred to as HPJ) is nothing more than an ASCII text file giving the Help Compiler instructions about compiling your Help file. The simplest, most basic HPJ file would be as follows:

[FILES]
myhelp.rtf

telling the Help Compiler the file to include in the "build" (the compilation of your RTF file into a bona fide Help file).

However, there are several more HPJ Commands. They can add functionality, direct the Help Compiler to files stored elsewhere, tell the Help Compiler what bitmap images to include in the build, print a copyright on your Help file, specify the icon that is displayed when your Help file is minimized, log a file of the errors encountered during the build, determine the level of warnings, report on the screen the progress of the build and any errors found, specify the contents screen, or compress the resulting Help file so it takes up the least amount of space on your system.

That is just some of what a good, properly constructed HPJ file can do. What you decide to include is up to you, just as long as you have the very basic needs of the Help Compiler met with the minimal HPJ file illustrated above.

There are nine (9) sections to a HPJ file, but I have included only the five (5) most frequently used sections. For information on the others, I strongly urge you to obtain a copy of the Microsoft Help Writer's Authoring Guide, available in free distribution on many online services and BBS's around the country. It has invaluable information that you'll refer back to on a regular basis, as I do. Only what is in the above example is required. Below are the sections, and inside those sections, are the variables each section permits.

The Help Project File - HPJ

[OPTIONS] Section

This section has the most options available, and is as the name says, optional. You can use some, all, or none of the options available. That is up to you. You may list them in either all upper or all lower case. Here they are, in alphabetical order:

Key: HAG is Microsoft Help Writer's Authoring Guide.

BMROOT = *drive:\directory name* Specifies the root directory for finding Bitmap files.

BUILD = *criteria* Define build criteria. Most folks define it as 'ALL', if they use it at all. See the HAG for more information.

CITATION = *Citation text*
Allows a lengthier copyright notice (2K of text) that is appended to the end of whatever the end-user copies to the clipboard. For example:

CITATION = Copyright (c) 1994, Domajigy Software, Inc. Information in this document is subject to change without notice and does not represent a commitment on the part of Domajigy Software, Inc. The software, which includes information contained in any databases, described in this document is furnished under a license agreement or nondisclosure agreement and may be used or copied only in accordance with the terms of that agreement. It is against the law to copy the software except as specifically allowed in the license or nondisclosure agreement. No part of this document may be reproduced in any form or by any means, electronic or mechanical, for any purpose without the express written permission of Domajigy Software, Inc.

COMPRESS = compression-level
Parameter
OFF No Compression.
MEDIUM Medium level of compression (block compression). (approx. 40%)
HIGH High compression (block and key-phrase compression). (approx. 50%)

With HIGH compression, the Help compiler creates a phrase-table file with a .PH extension if it doesnt find one in the project root directory. If the Help compiler finds a file with a .PH extension, it uses the file for the current build. Because the .PH file speeds up the compression process when little text has changed since the last build, you might want to keep the phrase file around if you compile the same Help file several times with compression. However, you get maximum compression if you delete the .PH file before starting each build, or set the OLDKEYPHRASE parameter (below) to NO.

CONTENTS Specifies the context string of the main Contents screen.

COPYRIGHT = *Your name (c) 1994* Add copyright string to the About Help dialog box. 50 characters maximum allowed.

ERRORLOG = *filename.err* File to write the Help Compiler messages and errors to. MiniHelp Plus v3.1 ***automatically*** inserts this into the [OPTIONS] section of the HPJ file, along with the appropriate filename.

FORCEFONT = *fontname* See the HAG for more information; seldom used.

ICON = *filename.ico* Specify the icon displayed when your Help file is minimized.

LANGUAGE = *language name* Sets the sort order for keywords in the Search dialog box.

The default sorting order is the English-language order.Microsoft Windows Help version 3.1 supports only English and Scandinavian sorting.

MAPFONTSIZE = *fontsize 1[-fontzize 2]:fontsize desired*

You probably won't need to use this; I've written several Help files and have never had a need to use it. For more information, refer to the HAG.

MULTIKEY = *character to use for alternate keyword table.*

Seldom used; see the HAG for more information.

OLDKEYPHRASE = *yes or no* Specifies whether or not to use the old keyphrase table generated with high compression. Set to yes, it uses the old keyphrase table. Set to no, it generates a new keyphrase table with each build.

OPTCDROM = *yes or no* Specifies if Help file should be optimized to be read off of a CD-ROM. See the HAG for more information.

REPORT = *on or off* Displays messages on the screen during the build. These messages indicate when the compiler is performing the different phases of the build, including scanning the file for compression, compiling the file, verifying context strings, and resolving jumps, keywords, and browse sequences.

Comments: Unlike the ERRORLOG option, messages displayed by using the REPORT option are not written to a file.

ROOT = *drive:\directory* Specifies the directory to find the topic and data files.

TITLE = *name of Help file* Specifies the name to display on the title bar of your Help file (ya know, the little bar across the very top of the window that has the minimize and maximize buttons on it).

WARNING = *1, 2, or 3* Specifies the level of warnings in a report.

1 Report only the most severe errors.
2 Report an intermediate number of errors.
3 Report all fatal errors and warnings. (Preferred)

Specifying a warning level of 3 will help you catch more errors during the build.

The Help Project File - HPJ

[FILES] Section

This is where you declare all the RTF files to include in the build of your Help file. The [FILES] section lists all RTF topic files used to build the Help file. A Help project file **must** have a [FILES] section.

Specify the full or partial path of a topic file. If a partial path is given, the help compiler uses the directories specified by the ROOT option to construct a full path. If a path is not indicated in either the ROOT parameter of the [OPTIONS] section or in the information supplied in the [FILES] section, the Help Compiler looks in the directory where the HPJ file resides. If a file is not on the defined path and cannot be found, the Help Compiler reports an error.

The Help Project File - HPJ

[CONFIG] Section

This section is used to hold macros that will be executed when the Help file is first started. One of the major uses is to create buttons that will appear on the button bar. For an example, see the HPJ file under Sample HPJ File.

The Help Project File - HPJ

[BITMAP] Section

This is where you specify the bitmap images to include in the build of your Help file. You can enter either the entire path, or just a partial path (the filename.bmp) if the path is specified in the BMROOT parameter of the [OPTIONS] section or if they are in the same directory as the HPJ file.

The Help Project File - HPJ

[WINDOWS] Section

This section is optional; if nothing is specified, Windows Help applys the default settings automatically. This is where you specify certain attributes about the main Help window and define any secondary windows you might want to include (for information on secondary windows, see the HAG for complete and full details). Note that any parameters not used, must still be delimited by the commas (that means you still need to include the commas, even if you let Windows Help set them to the defaults). This does not apply at the end of the command string. In the following example, if I only wanted to specify the window state, I need not type anything at all after the first zero in the line but must still include the two commas before it.

Here is a sample line, from the HPJ file used to build the Help file for MiniHelp Plus v1.0:

main = , , 0, , (192, 192, 192), 0

NOTE: If you want to make your Help screen maximized at startup, change the first "0" to a "1".

Syntax

window_name = "caption", (x-coord, y-coord, width, height), window-state, (scrolling-RGB), (nonscrolling-RGB), ontop-state.

* **window_name** Specifies the name of the window that uses the defined attributes. If the name is **main,** then it is applied to the main Help Window. For a secondary window, the name may be any unique name (other than main), up to 8 characters.

* **caption** Specifies the text that should appear in the title bar of the window. Although the main window name is usually declared in the TITLE parameter of the [OPTIONS] section, if a different name is declared here, the name that appears in the [WINDOWS] section determines the title displayed in the title bar.

* **x-coor, y-coord** Specifies the x and y coordinates of the upper left corner of the window in "Help units." Windows Help assumes the screen is 1024 units wide and tall, regardless of resolution. So, to compute the coordinates as they relate to your screen resolution (e.g. 640 x 480 for VGA), multiply the pixel of your screen where you want the appropriate corner to be displayed by the result of (1024/640 for the x coordinate, 1024/480 for the y coordinate). This will give you the screen position to place the upper left corner. Thus, if you want the upper left corner to appear at 320 pixels from the left side of the screen and 120 pixels down from the top, your result would be something like:

x = (320*(1024/640) = 512
y = (120*(1024/480) = 256

so these would be the values you plug into the x and y coordinates.

* **width** Specifies the width of the window, in Help units (see above).

* **height** Specifies the height of the window, in Help units (see above).

* **window-state** Specifies whether or not the Window is maximized or set to the size determined in the sizing parameter above.

0 = Normal State (default)

1 = Maximized State

* **scrolling-RGB** Specifies the backfround color for the windows scrolling region. Colors are given as standard RGB valuse, where RRR, GGG, and BBB are three digit numbers in the range 0 to 255, representing the red, green, and blue components of the color. If this parameter is not given, Help uses the default Windows color system specified by the end user in the Control Panel.

* **nonscrolling-RGB** Same as above, but specifies the background color for the windows nonscrolling region, if one is defined.

* **ontop-state** Specifies whether a secondary window type stays on top of other windows. The main Help window cannnot be set as the topmost window. This parameter can be one of the following parameters:

0 Display the window normally, not on top of other windows. Default.
1 Display the window on top of other windows. If this value is given, the user cannot change the window behavior using the Always On Top command in Help.

The Help Project File - HPJ

Sample HPJ File

Note that if you want to include comments in your HPJ file, start the line with a semicolon (;).

```
[OPTIONS]
TITLE=Help For MiniHelp
CONTENTS=TOC
COMPRESS=high
COPYRIGHT=(c) 1994, Paul Arnote
OLDKEYPHRASE=no
WARNING=3
REPORT=1

[FILES]
mhp.rtf

[CONFIG]
BrowseButtons ()
CreateButton ("btn_exit","E&xit","Exit()")
CreateButton ("btn_print","&Print","Print()")
CreateButton ("btn_notepad","&Notepad","ExecProgram(`notepad.exe', 0)")

[BITMAPS]
disk.bmp
smbullet.bmp
recorder.bmp

[WINDOWS]
main = , , 0, , (192, 192, 192), 0
```

Learning About MiniHelp Plus v3.1

Sample MiniHelp Source File (.src)

NOTE: Feel free to copy this listing to the clipboard, then into NotePad. (Hint: You can copy any topic in Help to the clipboard simply by holding down the CONTROL key and hitting INSERT.) Save it and attempt to compile it, using what you have learned here, into a bona fide Help file. Before doing so, however, you will need to write and save a HPJ file that the Help Compiler can use to build the Help file.

```
.topic contents
.title Table of Contents
.keyword Table of Contents
.browse 0001
.fontsize 24
\keepn
\sa300
.bold Table of Contents
\par
\pard

.fontsize 10
Here is a test of how
.bold MiniHelp
performs when called upon to perform various functions.


.link %MH " MiniHelp"
performs admirably, great for writing small or large Help files (this  indicated a non-visible
hotspot in the first occurrence of the word
.bold MiniHelp
in this paragraph).


The command
.pop POPUP "pop"
allows you to create popuplinks.


.fontsize 15
How do you make a bitmaplink?


.fontsize 12
Click on the disk
.bmlink DISK "disk.bmp"
  and jump to another topic!


.fontsize 10
MiniHelp also does support the use of
.link *BROWSE "Browse indexing"
```

directly with the use of the command,
.bold .browse.

\sb300
.macrolink Exit() "Goodbye"

\sb300
.bitmap btn_up.bmp
    This is a bitmap that does nothing but look good!

\sb300
\qr
    This is a bitmap that does nothing but look good!
.bitmap btn_up.bmp

\sb300
\qc
.bitmap btn_up.bmp
    This is a bitmap that does nothing but look good!
\par
\pard

.topic MH
.title MiniHelp
.keyword MiniHelp
.keyword Help Authoring aid
.browse 0002
.tmacro About()

.fontsize 14
This file was prepared with MiniHelp, the easy and inexpensive
approach to Help Authoring.

.topic POPUP
.title Popuplinks
.keyword popuplinks
.keyword .pop
.browse 0003

.fontsize 10
That's right, provisions were made for popuplinks, something the
original MiniHelp did not have. To do one, simply issue the command
.bold .pop,
followed by the name of the topic to jump to, followed by the text to display on the screen set off in quotes.

.topic DISK
.title Bitmap Links
.keyword Bitmap Links
.browse 0004

.fontsize 10

With the
.bold .bmlink
command, creating Hot Spots with bitmaps is very easy!

.topic BROWSE
.title Browse Sequencing
.keyword Browse
.keyword Browse Sequencing
.browse 0005

.fontsize 10
You can use Browse Sequencing (which allows you to use the Browse Buttons) by simply using the command
.bold .browse,
followed by the appropriate parameters! See the MiniHelp Plus v1.0 online Help File for documentation.

MiniHelp Plus v3.1

Hints & Tips

- You can control HOW Help displays Hot Spots. If you want it to be hidden, that is the same color as the surrounding text with NO underlining, put a percent (%) sign in front of the topic name in the link command. For example, **.link %MH "MiniHelp"** will print the word "MiniHelp" the same color as the rest of the text on the screen WITHOUT underlining it. Yet it's a valid Hot Spot that jumps you to another topic.

If you want to make your Hot Spot the same color as the surrounding text but be underlined, place an asterick (*) in front of the topic name in the link command, just as you did above with the percent sign. For example, **.link *MH "MiniHelp"** will print the word "MiniHelp" the same color as the rest of the text on the screen WITH an underline.

- Sample your output periodically. That is, go ahead and compile your uncompleted file, from time to time, and run the Help Compiler on it to see what problems you may or may not have in your formatting or such. It only takes a few minutes, during which you can relax a bit. Also, it is much easier to make notes on a few things that need changed than trying to remember a whole google of errors.

- Get into the habit of cancelling out your formatting commands when they are no longer needed by either issuing the \par \pard commands or \fi000 \li000 commands. This will help greatly in preserving the format you desire and cut down on the number of errors you will have to go back and fix. Also, the \ pard command does not affect the fontsize command. If your resultant Help file is not what you had expected, you may have forgotten to change the fontsize at a place in the source file you had planned.

- Also, if you are getting unexpected results in your formatting, go back and check your RTF commands. If you have limited them to only one or two to a line, this will be easy. Make sure they are declared with the \ and not the /. The former is the backslash. The latter is the slash, found on the "?" key. All RTF Commands **must** use the backslash character.

- To print out a backslash in your Help file, you **must** enter it twice in a row (i.e. xx but not x).

- Go back and review the sample files I have provided. In the Sample HPJ file, for example, I have introduced another common macro that I did not talk about earlier (Hint: in the last CreateButton command, the macro executed is ExecProgram, which causes a program to be ran from Help). By the way, this **is** the HPJ file I used to make the Help file for MiniHelp Plus v1.0.

- If you want to insert a comment into your source MiniHelp Plus v3.1 file, simply start a new line with a semi-colon (;). When MiniHelp Plus v3.1 encounters a line that begins with a semi-colon, it ignores it and goes on to the next line.

- Be sure that the two Help files, MiniHelp Plus v3.1 Help (MHPLUS31.HLP) and MiniHelp Plus v3.1 Shell Help (MHPSHELL.HLP) are either in the same directory as MiniHelp Plus v3.1 or in your Windows directory. Same thing for the Help Compiler (HCP.EXE) and its error code file (HCP.ERR). Also, make

sure all your source files are in the same directory as MiniHelp Plus v3.1. Otherwise, you will not be able to access these files from the MiniHelp Plus v3.1 ToolPad (a.k.a., the front end shell/user interface).

- If you are unable to view the Help file you generated by clicking on the **View Help** button, chances are you had been accessing or in another directory when you configured MiniHelp Plus v3.1, which retrieves your current working directory and stores and uses that path in defining the path to the created Help file in its .INI file. Go back to the main directory you have MiniHelp Plus v3.1 in and re-configure (simply clicking OK in the Configure dialog box should work) so the proper path information is saved in the MiniHelp Plus v3.1 .INI file. Or, you can go to the Windows® directory, open up the MiniHelp Plus v3.1 .INI file (mhplus31.ini) and manually edit the entry so it points to the proper path your newly-created Help file is in.

You can use **Recorder** that comes with Windows and create Recorder Macros that can be executed with just one keystroke (i.e., any function key like F2; avoid using F1 since many programs use it for its Help file). Recorder Macros can greatly simplify many mouse and/or keyboard intensive tasks. I strongly urge you to learn how to use the Recorder Macros. You will find they can save you time and keystrokes/mouse movements.

MiniHelp Plus v3.1

Known Bugs, Errata, & Acknowledgments

Known Bugs

As far as bugs go, I know of only two, and this may be just the way Windows seeks data. This entire Help file was written with MiniHelp Plus v3.1. As I went along, any other encountered bugs were fixed and I believe the program to now be relatively bug free. MiniHelp Plus v3.1 was created with Borland's Turbo C++.

One bug is when trying to run MiniHelp Plus v3.1 on source files in another directory. **Do not** attempt to run MiniHelp Plus v3.1 on files in another directory other than the one you have MiniHelp Plus v3.1 installed in. This means that the files you are attempting to convert (especially ones that link files together with the **.include** command) **MUST** be in the same directory as the MiniHelp Plus v3.1 program file.

The other bug is with the ".macrolink" command. If the macro you are defining must have spaces as part of the macro string, the ".macrolink" command will not parse it correctly and will attempt to define the text that appears after the space as the text to display on the screen as a hot spot for the link.

While I'm at it, let me clue you in on an apparent bug with the ExecProgram macro. Contrary to the documentation on this macro, you **cannot** run a program or file unless it is in your Windows directory. Thus, if you want to run "Notepad," there is no problem. But if you attempt to run a program that *is not* in your Windows directory, Windows Help will inform you that it is unable to run, open, or locate the specified file, **even when your path information is correct!**

However, if you should find any others, please let me know so they can be fixed in upcoming versions (work on future versions of MiniHelp Plus v?.? is already underway). You may contact me on America OnLine. My screen name there is **AerosolMan.** Or, you may write to me at the following address:

Paul Arnote
1208 Randolph
Leavenworth, KS 66048

Also, please drop me a note telling me what you thought about the program, comments (positive or negative), and any ideas you think will make it better.

ERRATA

Curly braces are a problem. You cannot use them from the keyboard without corrupting your resultant RTF file. I have had troubles with them.

While we are at it, here are the key sequences to print special characters in Windows Help, including curly braces. Whenever you want to use any of these characters in your Help file, simply substitute the following key sequences, preceded and followed by a space:

Format: **\'** *Hexidecimal ASCII number* (no spaces between any of these!)

Left curly brace	\'7B	{
Right curly brace	\'7D	}
Bullet	\'95	•
Trademark	\'99	™
Copyright	\'A9	©
Register	\'AE	®
Plus/minus	\'B1	±
1/4	\'BC	¼
1/2	\'BD	½
3/4	\'BE	¾

ACKNOWLEDGMENTS

The original program, MiniHelp, was written by Tom Campbell (c) 1994, and appeared in the January 1994 issue of Compute magazine. I feel this program was meant to grow.

MiniHelp Plus v1.0 first appeared in early February 1994 on America Online.

MiniHelp Plus v2.0 first appeared in late February/early March 1994 on America Online.

MiniHelp Plus v2.3 first appeard in mid to late March 1994 on America Online.

MiniHelp Plus v3.1 (which you obviously have) first appeared in June 1994 on America Online.

Many thanks to my wife, Barbie, who put up with me spending many endless hours sitting in front of the computer working on MiniHelp Plus v1.0 thru MiniHelp Plus v3.1 without complaining much about the lack of attention.

MiniHelp Plus v3.1

Help File Writing Offer

I know that writing Help files can be a long, arduous task, especially if you do not like doing it.

That is why I am offering to write your Help file for you. If you are a programmer who needs a Help file for their application, or just someone who wants to put some frequently referenced material into the interactive, hypertext Help environment, contact me at the following address:

Paul Arnote
1208 Randolph
Leavenworth, KS 66048

Writing Help files is something I really enjoy doing, and as you can see from this, it's something I DO know about. Prices are inexpensive and I will consider trades in lieu of payment. Plus, it frees you up so that you can get back to cutting some more code or doing whatever it is you do best.

Remember, Windows Help is not only a great vehicle for assisting end users with using their programs, but Windows Help can also be used very effectively to present **Employee Manuals & Handbooks, Company Policy & Proceedure Manuals, Teaching Aids** (presenting material in a very organized and efficient manner for learning), and anything else that might ask for interaction from the reader. Anything printed is a candidate for the interactive hypertext environment of Windows Help.

MiniHelp Plus v3.1

Plans for Future Enhancements

Of course, and as always, I want to hear from you (registered user or not) concerning any ideas you might have that might make MiniHelp Plus v?.? easier and better to use.

Originally, I had a short list of enhancements for a future release of MiniHelp Plus v3.x. But I got to tinkering with the program source code and added those features in a matter of just a few hours, so now you have the benefits of some of those features in this current release, instead of having to wait for yet another release. Those additions were the ability to launch your favorite ASCII text editor from MiniHelp Plus v3.1 with the default being Windows® Notepad, the ability to view your Project Error File from MiniHelp Plus v3.1, and the ability to view your newly-compiled Help file from MiniHelp Plus v3.1. I had also considered adding the ability to configure MiniHelp Plus v3.1 to run different versions of the Windows® Help Compiler (namely HC31.EXE), but chose not to do so. Why, you ask? Because of the obvious benefits of using HCP.EXE (discussed in the Introduction section of this Help file) and because MiniHelp Plus v3.1 is packaged and shipped with the latest known version of the extended Help Compiler (3.10.505).

As always, your ideas will be taken into consideration for future releases.

RTF Commands

Paragraph Formatting Commands

There are quite a few paragraph formatting commands available as RTF Commands. Whatever you command you issue, to "unissue" it (make it not effective any longer), in most cases all you have to do is issue the command again, but this time, use three zeros after the command. Exceptions will be noted as we go along. Paragraph formatting commands use a measurement called twips. Some of these commands you will use a lot; others not so often, but you are always glad they're there when you need them. The commands actually are easy to use. For clarity and to prevent problems from arising when you convert your MiniHelp Plus source file to an RTF file, type one command per line.

Syntax

\keep Prevents the text from wrapping to fit the Help window. Applies to current paragraph and all subsequent paragraphs until the **\par** and **\pard commands are issued.**

\lixxx Indents the left margin xxx twips. To cancel, issue the command \li000.

\par Marks the end of the current paragraph. MiniHelp Plus v3.1 inserts these automatically whenever it sees a blank line. You need to issue it manually proir to issuing the \pard command to end certain formatting and border commands.

\pard Restores all paragraph properties to default values. Must be used after the text following the \keep, \keepn, \box, \brdrt, \brdrb, \brdrr, \brdrl, \brdrbar commands to terminate these commands actions.

\qc Centers all text in paragraph until \pard, \ql, or \qr commands issued.

\ql Aligns all text in the paragraph at the left margin until \pard, \qc, or \qr commands issued.

\qr Aligns all text in the paragraph at the right margin until \pard, \qc, or \ql commands issued.

\qj Windows Help ignores this command, meant to format justified text. This command is of very little use since the variable sizing of the Help window makes the results of this command unpredictable.

\rixxx Indents the right side of the paragraph xxx twips until \par and \pard commands issued or \ri000 command issued.

\saxxx Specifies the amount of blank space after a paragraph in twips until \par and \pard commands issued or \sa000 command issued.

\sbxxx Specifies the amount of blank space before a paragraph in twips until \par and \pard commands issued or \sb000 command issued.

RTF Commands

Text Formatting Commands

MiniHelp Plus v3.1 has four built in text formatting commands, the bold, italic, bolital, and ul commands. The RTF Commands allow further text formatting with ease.

Syntax

\fx specifies what font to use for your text. In MiniHelp Plus v3.1, there are three fonts defined in the font table of the output RTF topic file. They are:

\f0 Times New Roman
\f1 Courier New
\f2 Arial (the default font in MiniHelp Plus v3.1)

These three fonts were chosen to provide maximum compatibility with all systems. All three are TrueType (R) fonts that are shipped with Windows, so everyone has them.

To change the font, issue the command \f and the number of the font you want to use. Be sure there are NO spaces between the \f and the font number. Also, this RTF command must appear on a line of all its own. All subsequent text that follows it will be displayed in the font you specified in the command until you change it back.

\line Although actually a paragraph formatting command, we include it here since it is a command that can be embedded in your text. To use it, preceed the command with a blank space, type \line, follow with another blank space, and continue typing your text. This command breaks the current line without ending the paragraph. Subsequent text starts on the next line and is aligned and indented according to the formatting of the current paragraph.

\tab Inserts a tab into your text. Default tab setting is 1/2 inch. Default setting can be changed with the command, \tx, below. Use it just like the \line command above.

\tb Advances to the next tab stop. Use it just like the \line command above.

\tqc Advances to the next tab stop and centers the text. Use it just like the \line command above.

\tqr Advances to the next tab stop and aligns the text to the right. Use it just like the \line command above.

\txn Specifies the tab stop position, in twips (n), in relation to the left margin. This is one command that really should go on a line of its own.

RTF Commands

Creating a Non-Scrolling Region

A non-scrolling region (like at the top of the screen in gray that says "RTF Commands" "Creating a Non-Scrolling Region") is of keen interest to many Help file writers because it keeps the Help file reader oriented to the topic he or she is reading by keeping the topic name always in sight. In MiniHelp Plus, it truly isn't hard to do.

Syntax & Restrictions

\keepn

Must appear on a line by itself.

Must not have ANY blank lines before it.

MUST be terminated with the \par and \pard commands before going any further. Otherwise, the compiler will return an error that says the non-scrolling region exceeds the page length and creates NO non-scrolling region at all.

EXAMPLE

```
.topic NOSCROLL
.title Defining a Non-Scrolling Region
.keyword Defining a Non-Scrolling Region
.fontsize 12
\fi000
\li000
\keepn
.bold RTF Commands
.fontsize 18

\fi400
.bold Creating a Non-Scrolling Region
\par
\pard
```

is the exact entry from the MiniHelp Plus source code that created the non-srolling region you see at the top of this screen.

Learning About MiniHelp Plus v3.1

.rbitmap

The **.rbitmap** command inserts a bitmap into your Help file and *places it at the right margin.* Unlike the bmlink command, this command simply inserts the bitmap *without* creating a link to another topic. This command also differs drastically from the **.bitmap** command in that instead of inserting a bitmap *as a character in your text,* it places the bitmap on the right margin of your Help screen and wraps the text that appears after the command around the left edge of the bitmap. Some possible uses include adding a picture to the topic pages, or adding emphasis to some of your text. At the bottom of this topic, you can see an example.

Syntax

.rbitmap ***filename.bmp***

Note that there are no quotes around the filename. Simply type the command, space, filename. It's that easy. Make sure you specify your bitmap in the BITMAP section of the Help Project File (see the HPJ section).

EXAMPLE

.rbitmap recorder.bmp
This bitmap appears at the right side of the Help screen and wraps the text around the left edge of the bitmap.

produces this result:

This bitmap appears at the right side of the Help screen and wraps the text around the left edge of the bitmap.

Learning About MiniHelp Plus v3.1

.lbitmap

The **.lbitmap** command inserts a bitmap into your Help file and *places it at the left margin.* Unlike the bmlink command, this command simply inserts the bitmap *without* creating a link to another topic. This command also differs drastically from the **.bitmap** command in that instead of inserting a bitmap *as a character in your text,* it places the bitmap on the left margin of your Help screen and wraps the text that appears after the command around the right edge of the bitmap. Some possible uses include adding a picture to the topic pages, or adding emphasis to some of your text. At the bottom of this topic, you can see an example.

Syntax

.lbitmap ***filename.bmp***

Note that there are no quotes around the filename. Simply type the command, space, filename. It's that easy. Make sure you specify your bitmap in the BITMAP section of the Help Project File (see the HPJ section).

EXAMPLE

.lbitmap recorder.bmp
This bitmap appears at the left side of the Help screen and wraps the text around the right edge of the bitmap.

produces this result:

This bitmap appears at the left side of the Help screen and wraps the text around the right edge of the bitmap.

Learning About MiniHelp Plus v3.1

.hpjopt

This command creates the [OPTIONS] section of the Help Project File, and is as the name says, optional. However, there are many useful and desired options to include here. For a more complete discussion of what each option does, see the HPJ Section of this Help file. Briefly, here is an alphabetical list of what they are:

BMROOT	Specifies the root directory for bitmaps.
BUILD	Define the build criteria of the Help file.
CITATION	Allows you to specify a long copyright disclaimer. 2K of text is maximum allowed.
COMPRESS	Select the level of compression of the Help file.
CONTENTS	Selects which topic is displayed as the Contents screen.
COPYRIGHT	Specify a short copyright disclaimer for the About dialog box. 50 characters maximum allowed.
ERRORLOG	Specifies the file to write the HC messages to for later review.
FORCEFONT	Forces the use of the specified font.
ICON	Specifies the icon to use when the Help file is minimized.
LANGUAGE	Specifies the keyword sort order (English is default).
MAPFONTSIZE	Change the fonts displayed to different sizes.
MULTIKEY	Specifies alternate keyword table to use for topics.
OLDKEYPHRASE	Specifies use of old key phrase table (Yes or No).
REPORT	Setting to on (1) displays build messages to screen.
ROOT	Specifies the root directory for topic files.
TITLE	Specifies the text to display on the title bar of your Help file.
WARNING	Specifies the level of warning messages from the HC.

As you can see, there are several options to choose from. I always include the COMPRESS, REPORT, TITLE, WARNING, ERRORLOG, CONTENTS, OLDKEYPHRASE, COPYRIGHT, and CITATION options in all the Help files I write.

Syntax

.hpjopt Copyright = 1994, John Doe
.hpjopt Title = Help for ME!
.hpjopt Oldkeyphrase = no

and so on. Each option must be declared with the **.hpjopt** command. Spaces are allowed in all .hpj project commands, except those specifying directories or files.

Learning About MiniHelp Plus v3.1

.hpjfiles

The **.hpjfiles** command is used to insert other files not generated by MiniHelp Plus v3.1 into the HPJ. When MiniHelp Plus v3.1 generates your Help Project File, it **automatically** inserts the filename and path of the RTF file generated by MiniHelp Plus v3.1 into the [FILES] section of your HPJ. You may use this command to include other RTF files in your Help file that were not created with MiniHelp Plus v3.1.

Syntax

.hpjfiles somefile.rtf

See the HPJ Section of this Help file for a more complete description of the HPJ [FILES] section.

Learning About MiniHelp Plus v3.1

.hpjmap

This command inserts the information into the [MAP] section of the HPJ file that specifies the definitions for context-sensitive Help. This may be done either by entering it directly (a long, arduous task) or by simply using the C-type Header file, like the one that MiniHelp Plus v3.1 generates. The syntax area below shows both ways. As you become more familiar with writing Help files and the Help macros (some of which can *only* be used with defined context-numbers), you will come to appreciate this feature of Help.

Just as in the [FILES] section previously mentioned, MiniHelp Plus v3.1 **automatically** inserts the header file (and its path information) into the [MAP] section of your HPJ file. You may use this command if you have header files not generated with MiniHelp Plus v3.1 or if you want to define context-sensitive Help for the topics of an RTF topic file not generated by MiniHelp Plus v3.1 (since all RTF topic files generated by MiniHelp Plus v3.1 also have an C-type header file generated automatically).

Syntax

hpjmap #include<myhelp.hhh>

to use a header file other than that generated by MiniHelp Plus v3.1 to define context-sensitive Help.

.hpjmap TOPIC1 1
.hpjmap TOPIC2 2
.hpjmap TOPIC5 6
.hpjmap TOPIC10 3

to define the context numbers yourself.

The order is not significant--only that each topic defined has a unique id number.

Learning About MiniHelp Plus v3.1

.hpjconfig

The **.hpjconfig** command defines items (Help macros) to place in the [CONFIG] section of the HPJ file. This is the section where you create, define, enable, disable, and "destroy" buttons that appear on the button bar at the top of the Help window, where you manipulate the menus of the Help window, and whatever other macros you might want to include in your Help file. There are a total of 56 Help macros available. I strongly recommend obtaining a copy of the Microsoft Help Writers Authoring Guide (a Help file, itself) to learn how to use these powerhouse Help tools. For a discussion of 10 of the most commonly used macros, see the .macrolink topic of this Help file. Below is a brief example and the syntax for this command.

```
.hpjconfig CreateButton("btn_print", "&Print", "Print()")
.hpjconfig BrowseButtons()
.hpjconfig JumpContents("progman.hlp")
```

Learning About MiniHelp Plus v3.1

.hpjbitmap

This command specifies the bitmaps to include in your Help file. You can specify path information here, if you like (I recommend simply using filename.bmp and specifying the directory where your bitmaps can be found in the BMROOT option in the [OPTIONS] section). Whenever you issue either the **.bitmap, .lbitmap, .rbitmap,** or **.bmlink** commands, I recommend you issue the .hpjbitmap command immediately before or after you issue one of the aforementioned commands. That way, you will not forget to add the bitmap information to your HPJ file.

Syntax

.hpjbitmap racecar.bmp

Learning About MiniHelp Plus v3.1

.hpjbtag

.hpjbtag defines the builtags in the [BUILDTAGS] section of the HPJ file to be used when building (compiling) your Help file. A detailed explanation of buildtags and their usage is beyond the scope of this Help file. I refer you, instead, to the Microsoft Help Writers Authoring Guide.

Syntax

.hpjbtag *BuildTag data*

Learning About MiniHelp Plus v3.1

.hpjalias

.hpjalias defines a second topic name to another topic. Refer to the Microsoft Help Writers Authoring Guide for more information.

Syntax

.hpjalias *TOPICNAME* = *ALIASNAME*

Learning About MiniHelp Plus v3.1

.hpjwin

This is the command you can use to define attributes of your Help window (background color of the scrolling and nonscrolling regions, the size of the Help window, whether or not the Help window is to topmost window on the screen, whether it is minimized, maximized, or normal on startup, and the Help window caption (not necessary for the main Help window if you use the TITLE option in the [OPTIONS] section)). Refer to the HPJ Section of this Help file for a detailed explanation. Refer, also, to the Microsoft Help Writers Authoring Guide for an even more detailed explanation.

Syntax

.hpjwin main= , , 1, , (192, 192, 192), 0

Learning About MiniHelp Plus v3.1

.hpjbaggage

tells the Help Compiler (HC) to store files in its internal file system, and not on the DOS file system. This helps allow better access times for multimedia data. Refer to the Microsoft Help Writers Authoring Guide for complete details.

Syntax

.hpjbaggage *filename*

License Agreement

MiniHelp Plus v3.1

License Agreement

INSTALLATION OF **MINIHELP PLUS V3.1** ON YOUR COMPUTER SYSTEM IMPLIES AGREEMENT WITH THE TERMS AND CONDITIONS BELOW.

DISTRIBUTION OF **MINIHELP PLUS V3.1,** ITS ACCOMPANY PROGRAMS AND DOCUMENTATION IS CONSIDERED AS IS. THE AUTHOR OFFERS NO WARRANTIES OF ANY KIND, EXPRESSED OR IMPLIED. THIS INCLUDES, BUT IS IN NO WAY LIMITED TO, WARRANTIES OF **MINIHELP PLUS V3.1'S** MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. UNDER NO CIRCUMSTANCES WILL THE AUTHOR BE LIABLE FOR ANY DAMAGES WHICH RESULT FROM THE USE OF THIS PROGRAM OR THE INABILITY TO USE IT. EXCLUSION FROM LIABILITY INCLUDES, BUT IS NOT LIMITED TO, LOST PROFITS, LOST SAVINGS, OR ANY OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES.

MiniHelp Plus v3.1 is distributed as Shareware. It is not free, freeware, or in the public domain. You may use .bold MiniHelp Plus v3.1for a trial period of thirty days, at no cost to you, to determine if it fits your needs. If you decide to use **MiniHelp Plus v3.1** regularly, you are expected to register it and pay the applicable registration fee. Individual copies of the unregistered version of **MiniHelp Plus v3.1** may be given to your friends and associates for the same thirty day free trial period. You may also upload **MiniHelp Plus v3.1** to the public section of a public BBS.

You may not modify or disassemble **MiniHelp Plus v3.1,** nor distribute any modified or disassembled versions of **MiniHelp Plus v3.1. MiniHelp Plus v3.1** may not be included with any other product without written permission from its author, Paul Arnote.

Registered copies of **MiniHelp Plus v3.1** can be used on more than one computer at a time as long as no more than one of these computers is running **MiniHelp Plus v3.1** at the same time. You may make backup copies of **MiniHelp Plus v3.1** as necessary for archival purposes only.

U.S. Government RESTRICTED RIGHTS: Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subdivision (b)(3)(ii) of the Rights in Technical Data and Computer Software

clause at 252.227-7013.

Back To What You Were Doing

MiniHelp Plus v3.1

Project Commands

Here is a quick reference for the new Project Commands to use in tailoring your Help Project File.

.hpjopt

.hpjfiles

.hpjmap

.hpjconfig

.hpjbitmap

.hpjbtag

.hpjalias

.hpjwin

.hpjbaggage

MiniHelp Plus v3.1

Topic Commands

Here is a quick reference to all of the MiniHelp Plus v3.1 Topic Commands.

.link

.pop

.topic

.title

.keyword

.browse

.bmlink

.bitmap

.rbitmap

.lbitmap

.macrolink

.tmacro

.bold

.italic

.boldital

.ul

.ulbold

.ulitalic

.ulboldital

.cfore

.fontsize

.include

Learning About MiniHelp Plus v3.1

.fontsize

The **.fontsize** command allows you to control the size of the fonts you display on the Help screen. Issue the command, followed by the size of the font you want to display, in points. For example:

.fontsize 6
This is some 6 pt text to display.

.fontsize 8
This is some 8 pt text to display.

.fontsize 10
This is some 10 pt text to display.

.fontsize 15
This is some 15 pt text to display.

.fontsize 20
This is some 20 pt text to display.

.fontsize 24
This is some 24 pt text to display.

Will display as:

This is some 6 pt text to display.
This is some 8 pt text to display.
This is some 10 pt text to display.
This is some 15 pt text to display.
This is some 20 pt text to display.
This is some 24 pt text to display.

Try to keep the maximum size limit of your fonts to 24 pts or less. You may use any number between 6 and 24 to specify the font size.

Learning About MiniHelp Plus v3.1

.cfore

First of all, note that this **does not** change the color of the Hot Spot text. If you wish to do that, you must change the settings in your WIN.INI file. USE EXTREME CAUTION DOING SO! I recommend using "sysedit.exe", found in either your Windows directory or your Windows System directory. At least then, you will have a backup copy...just in case something gets really screwed up. The backup copy's file extension will be **.SYD.** Refer to the file WININI.WRI in the Windows Resource Kit for details on changing the jump colors.

.cfore allows you to change the color of your normal text to any of the 16 colors supported by Windows Help. Use a two digit number (i.e., 01, 07, 14) to specify the color.

Syntax

.cfore (any two digit number representing the color you want)

EXAMPLE
(taken from this Help file's MiniHelp Plus source file)

```
  link in the
.cfore 14
  "Learning About MiniHelp Plus V3.1"
.cfore 01
  section.
```

will print out as:

link in the "Learning About MiniHelp Plus v3.1" section.

You should include a leading and trailing space on the text to be effected to permit proper word spacing (sorry, this command does not insert them automatically like the typeface commands).

Here is the color table, the best I can tell (I only have a monochrome monitor but am able to derive the colors from my long experience as a photographer and from the chance I had to work with a color monitor on loan to me while my monochrome monitor was in the shop).

Black = 1
Dk Gray = 2
Lt Gray = 3
Lt Red = 4
Dk Red = 5
Lt Green = 6
Dk Green = 7
Lt Blue = 8
Dk Blue = 9
Yellow = 10
Dk Yellow = 11
Lt Cyan = 12
Dk Cyan = 13

Magenta = 14
Purple = 15
White = 16

RTF Commands

Besides the 33 commands of MiniHelp Plus v3.1, you may also use certain RTF commands to help format text and paragraphs. There are some mighty powerful commands here that, depending on how they are used, can make your Help file look like a disaster area or make it look like the greatest thing ever published. Exercise care when using these commands and you should have no problems.

Using RTF Commands with MiniHelp Plus v3.1

Typically, when using RTF Commands with MiniHelp Plus v3.1, you should place them on a line of all their own. Also, to avoid problems converting your source file to RTF and to improve readability of your source file, limit your RTF commands to only one or two per line (one would be best). This is also a great aid in debugging your Help file, should something end up not being formatted properly. Only certain RTF Commands can be embedded in your text and those will be pointed out as we go along.

RTF Commands

Borders

Placing borders around your text is very easy. For an example of bordered text, see the "Introduction" section of this Help File. There, you will find that the first paragraph is bordered with a double line. Follow these commands explicitly and everything will work out well for you. The border commands are actually paragraph formatting commands, but are presented here separately, due to their uniquity.You **MUST** make sure that after you are done typing the text you want bordered, that you type the **\par** and **\pard** commands (on separate lines is best), otherwise Help will draw the borders throughout your whole topic. For example, a \box command without these commands, will draw a box around your entire topic.

Syntax

\box Places a box around the current paragraph or picture and all subsequent paragraphs or pictures until the \pard command is issued. Defaults to the thin line border style if no border style specified. May be used by itself if the thin border style is desired.

\brdrsh Places a shadow style border around your text. Must use with \box command, if you wish to use this border sytle.

\brdrdb Places a double line style border around your text. Must use with \box command, if you wish to use this border style.

\brdrth Places a thick line style border around your text. Must use with \box command, if you wish to use this border style.

\brdrs Places a thin line border style around your text. This is the default if no border style is specified. May use with \box command, if you wish to specify the border type.

\brdrbar Places a border to the left of the current paragraph or picture and all subsequent paragraphs until the \pard command is issued.

\brdrl Same effect as \brdrbar.

\brdrr Places a border to the right of the current paragraph or picture and all subsequent paragraphs until the \pard command is issued.

\brdrb Places a border below the current paragraph or picture and all subsequent paragraphs until the \pard command is issued.

\brdrt Places a border above the current paragraph or picture and all subsequent paragraphs until the \pard command is issued.

EXAMPLE

```
\brdrdb
\box
Welcome to MiniHelp Plus v3.1, the improved version of MiniHelp that appeared in the January,
1994 issue of COMPUTE magazine.  For details on what is new in this version, click on the
.bold What's New
link in the
.cfore 07
```

"Learning About MiniHelp Plus v3.1"
.cfore 01
section.
\par
\pard

will print out as:

Welcome to MiniHelp Plus v3.1, the improved version of MiniHelp Plus v1.0, and of MiniHelp that appeared in the January, 1994 issue of COMPUTE magazine. For details on what is new in this version, click on the **What's New** link in the "Learning About MiniHelp Plus v3.1" section.

Enhancements to **.pop** and **.macrolink**

Please note that these two commands are now enhanced in MiniHelp Plus v3.1 to allow you to define a bitmapped image as the hot spot. All you need to do is specify the bitmap filename in the text enclosed in quotes. If the text in quotes ends in **.bmp** then a bitmapped image will be used as the hot spot. Note that the last 4 letters of the text in quotes **must** end in **.bmp** and it must be a valid bitmapped image, either in your current path or in the path specified with the BMROOT option in the HPJ file. Otherwise, the text in quotes is what will appear on the screen.

Note that the **"Changes"** button you clicked on in the non-scrolling region of the screen was created with the new, enhanced **.pop** command. The format is:

.pop ENHANCE "changes.bmp"

.keyword Command Format Change

You may now chain together several keyword phrases or words with one **.keyword** command, just as long as you separate the different phrases/words with a semicolon **(;)** and just as long as the total length of the different phrases or words does not exceed 255 characters, including the semicolons. If you need more, you may, however, issue the **.keyword** command again with the additional keyword phrases or words. Furthermore, if you issue the **.keyword** command from the middle of a topic's text, only the text that follows will be displayed when the user/reader selects the word or phrase from the **Search** dialog box.

This Help file was created with

MiniHelp Plus v3.1

Written by Paul Arnote

Derived from the original MiniHelp that appeared in the January, 1994 issue of Compute Magazine.

Written by Tom Campbell

If you would like to use MiniHelp Plus v3.1 to create your Help files, you may obtain your SHAREWARE copy by either downloading it off of America Online or a local bulletin board (depending if it has made it there yet).

Or you may write to:

Paul Arnote
1208 Randolph
Leavenworth, Kansas 66048

Send $15.00, plus $2.50 to cover shipping and handling, the disk, and duplication costs. Be sure to specify disk size, density, and program name, as well as your name & address. Cash, check or money order. ***Please make checks/money orders out to Paul Arnote.*** Registered users will receive one free upgrade when it becomes available. This version is not crippled in any way. Please register your copy if you find it useful and support the shareware concept, as well as helping to keep me interested enough to keep improving on this program.

Back To What You Were Doing