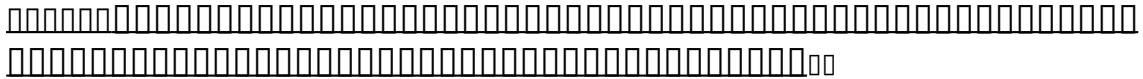


QDHELP  
Product described in this manual whenever without notice.

I have made every effort to be sure that QDHELP is an effective tool for generating help files for MS Windows. With that said, the following mandatory legal mumbo jumbo is necessary.

QDHELP is distributed "as is" without any warranty of any kind.  
IN NO EVENT SHALL PHIL ALLEN BE LIABLE OR RESPONSIBLE FOR ANY LOSS, DAMAGE, OR OTHER PROBLEM CAUSED BY USING QDHELP. THE ENTIRE RISK OF USING QDHELP IS YOURS.



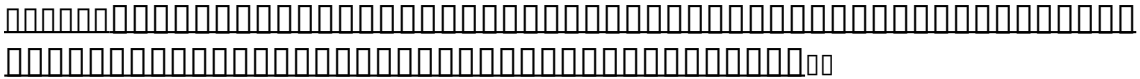
## 1. Intro

Welcome to QDHELP, the Quick and Dirty Help writer's assistant for MS Windows. QDHELP has a noble purpose, to save you money! Currently, you can get the Microsoft help compiler with a number of software development products. It comes standard equipment with the MS Visual C++, Borland C++ for Windows, and Visual Basic Professional. The problem is that the help compiler expects its input to be in rich text format (RTF). One of the few ways to generate RTF is by using Microsoft Word. Coincidence? You can be the judge of that. Since QDHELP costs \$49.00 (you do plan to register, don't you?) and Microsoft Word retails for \$495.00 you can save yourself a cool \$456.00 by using QDHELP to generate your help files. Of course if you already own Microsoft Word you save nothing except time. QDHELP makes it simple to generate a help file. Using any ASCII editor you can have MS Windows help with topics, standard links, popup links, bitmap links, browse sequences and keyword searches in no time!

QDHELP has many features that make your task of writing help files easier than ever before. This document presents detailed information on all of them. Try QDHELP and I'm sure you will be more productive as you generate MS Windows help files the Quick and Dirty way!







winhelp C:\MYHELP\TEST.HLP

This will cause a program icon to be put into your current Windows group which should be a question mark (Like the WinHelp Icon) but with your input filename as the name under the icon. If you double click the icon it will activate winhelp and read in your help file all in one shot. Pretty neat, eh!

### 3.3 Hints and Tips

Read the WHATSNEW.DOC file in your distribution. This file contains the changes between this and the previous versions of QDHELP. You will definitely want to know about these things.

Read the README file in your distribution. This file contains the latest information that may have not made it into this manual.

QDHELP **is** case sensitive. This means you must enter the commands in lower case except where noted in this manual.

Do **not** put tab characters into your QDHELP text. You can tab to where the text will start for indentation purposes. If you want to put tabs into your text use the RTF command `\tab`. This will insert a tab into the help file.

QDHELP comes with extensive on-line documentation in the file QDHELP.HLP. You can load this file into the MS Windows help for viewing. See Section 3.3 Testing Your Help Files. The QDHELP file that generated this help is included in your distribution with the name QDHELP.QDH. This is a good place to start looking to see how to lay out a QDHELP file. To generate QDHELP.HLP from source do the following:

```
QDHELP QDHELP.QDH
```

This should output a file named QDHELP.RTF.

```
HC QDHELP.HPJ
```

This runs the help compiler and generates the file QDHELP.HLP. This, of course, is the file that you can load and run under MS Windows.



## 4. The HYPE

QDHELP has several features built in to allow you to generate help files for MS Windows 3.x quickly. The following list, while not complete, will give you an idea of the benefits of using QDHELP.

QDHELP lets you . . .

- o Use your favorite programming (ASCII) editor to generate your help files.  
(Why should you have to bother learning a word processor?).
- o Automatically generate the .HPJ file needed by the help compiler.
- o Take advantage of all the new Windows 3.1 WinHelp features, such as running macros from hyperlinks.
- o Automatically generate an .HHH file which contains the values needed to do context sensitive help.
- o Automatically generate a glossary topic in your help file. You define which topics you want included in the glossary by using the /glossarytopic keyword to define those topics.
- o Include RTF commands directly in your help.
- o Use templates to get the same look and feel for all your help topics easily.
- o Automatically number your browse sequences
- o Include multiple files in your input, allowing you to better segment your help file writing.
- o Generate help files with a syntax that make sense and is readable.
- o Get done with those help files fast! (So you can get back to what you really want to do, cut some code!)

```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```

## 5. QDHELP Language Commands

Commands must be the first non white space characters on an input line. In QDHELP white space is a combination of space or tab characters.

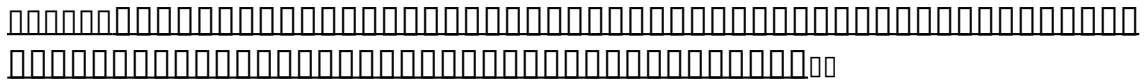
This section will discuss the commands that the QDHELP language understands. Don't be intimidated, there are not that many. We will cover each command in some detail. The following is a list of all the commands that QDHELP understands. They are sorted in order of likely usage in a help file. The detailed explanations are in the same order.

Command	Description
/hpjoptions	pass data into the OPTIONS section of the .HPJ file
/hpjfiles	pass data into the FILES section of the .HPJ file
/hpjbuildtags	pass data into the BUILDTAGS section of the .HPJ file
/hpjconfig	pass data into the CONFIG section of the .HPJ file
/hpjbitmaps	pass data into the BITMAPS section of the .HPJ file
/hpjmap	pass data into the MAP section of the .HPJ file
/hpjalias	pass data into the ALIAS section of the .HPJ file
/hpjwindows	pass data into the WINDOWS section of the .HPJ file
/hpjbaggage	pass data into the BAGGAGE section of the .HPJ file
/include	includes another file in the current help file
/pragma	set option to influence the operation of the QDHELP
parser	
/defformat	set the default format for the entire document or a single
topic	
/topic	start the definition of a help topic
/glossarytopic	start the definition of a help topic and put the topic into the
glossary	
/endtopic	end the definition of a help topic
/title	set the title of a help topic
/keywords	set the search keywords for a help topic
/browse	set the browse category and position of a help topic
/defformat	set the default format for the document or topic
/para	start a paragraph
/endpara	end a paragraph
/topicmacro	assign a macro to execute when a topic is selected
/helpid	assign the context sensitive help value to this topic
/text	format text in a special way (i.e., bold, underline etc.)
/link	make a hypertext link
/popuplink	make a popup hypertext link
/bitmaplink	make a bitmap picture hypertext link
/macrolink	make a macro execution hypertext link

□□

```
#####  
#####  
/bitmap          add a bitmap to the current help topic  
//              comment, ignore rest of line
```





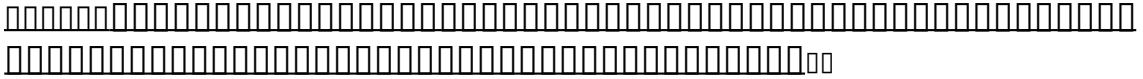
## 5.1 /hpjoption command

Syntax:

### **/hpjoptions OPTION, value**

The /hpjoptions command is used to set one of the option variables available in the .hpj file used by HC. The valid options are the following:

BMROOT	Root directory for finding bitmaps.
BUILD	Define build criteria.
COMPRESS	Select type of Compression used.
CONTENTS	Select context of contents screen.
COPYRIGHT	Add copyright string to About dialog box.
ERRORLOG	File to write HC compilation messages to.
FORCEFONT	Force use of specific fonts.
ICON	Specify help minimized icon.
LANGUAGE	Sort order for Scandinavian language.
MAPFONTSIZE	Map fonts to different sizes.
MULTIKEY	Select alternate keyword mapping for topics.
OLDKEYPHRASE	Use old keyphrase table.
OPTCDROM	Optimize for CDROM use.
REPORT	Select display of build messages.
ROOT	Root directory to find topic and data files.
TITLE	Specify help window title bar text.
WARNING	Select level of warning messages.



## 5.2 /hpjfiles Command

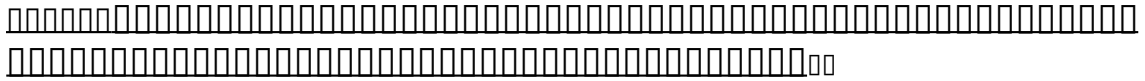
Syntax:

**/hpjfiles filename**

The /hpjfiles command is used to include files in the .hpj file used by HC. Normally QDHELP will insert the file name into the .HPJ file for you. However, if you have other .RTF files that you want included in a compile with a file that is being generated by QDHELP just include a line like the following:

```
/hpjfiles myfile.rtf
```

and that should do the trick.



### 5.3 /hpjbuildtags Command

Syntax:

**/hpjbuildtags BUILDTAG**

The /hpjbuildtags command is used to specify valid build tags for the current help file. Normally QDHELP will insert all the build tags that it finds in your .QDH file into this section for you. This command is here so that if you have other build tags that you need defined (lets say of other .RTF files you are including in this help file which were not generated by QDHELP) you can do so.



#### 5.4 /hpjconfig Command (Win 3.1)

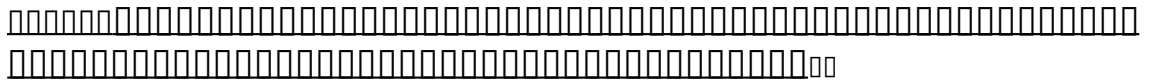
Syntax:

#### **/hpjconfig CONFIG DATA**

The /hpjconfig command is used to set data in the [CONFIG] section of the .hpl file. The config section is used to hold Macros that will be executed when the help file is first started. For example, if you want to have left and right browse buttons in your help you would add the following command to your .QDH file:

```
/hpjconfig BrowseButtons()
```

This will cause the macro BrowseButtons() to be run on start up and your help will have browse buttons on the button bar. This is also the place where you register other Dynamic Link Library (DLL) routines that you wish to call from within your help file.



## 5.5 /hpjbitmaps Command

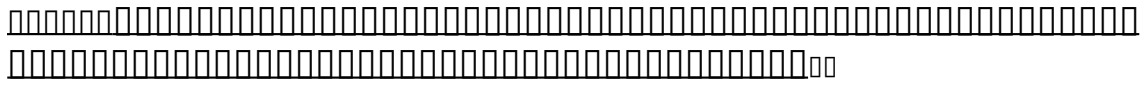
Syntax:

**hpjbitmaps c:\path\file.bmp**

This command specifies bitmap files to be included in the build. Bitmap files need only be specified if they can not be found in the directories in BMROOT or ROOT.







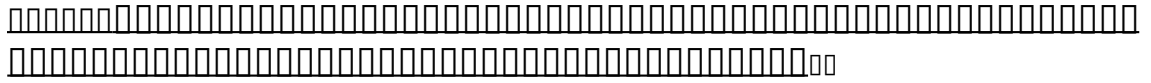
## 5.8 /hpjwindows Command (Win 3.1)

Syntax:

**/hpjwindows win\_name="caption",(h\_pos, v\_pos, width,height),sizing,(RGB client ) , ( RGB nonscroll)**

This command tells the help compiler the size, location, and colors for a secondary help window. If the win\_name = "main" these properties are applied to the main help window.





## 5.9 /hpjbaggage Command (Win 3.1)

Syntax:

### **/hpjbaggage file**

The /hpjbaggage command allows you to tell the help compiler to store files in its internal file system, not on the DOS file system. This allows better access time for multimedia data.

```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```

## 5.10 /include Command

### **Syntax:**

**/include [ FILENAME | FILENAME.EXT ]**

This command inserts the text of another file into the input file at that point. The only parameter to the /include command is a file name. If the file name does not have an extension and the file cannot be found in the current directory, then the extension .QDH is added to the file name and an attempt to open the new file name is made. If the file cannot be found an error is displayed.

### **Placement:**

The /include command is valid anywhere inside a QDHELP input file.

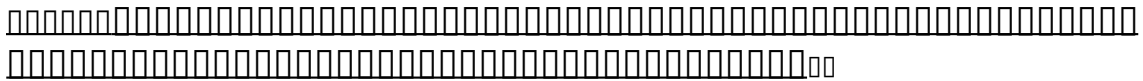
### **Limitations.**

The include files can only be nested six (6) deep. This means you cannot have any more than six layers of include files including other include files. If you need more than six levels of include files find an easier subject for which to write a help file!

### **Example:**

```
/include FILE2.QDH
```

This command will read in the file named FILE2.QDH. QDHELP will process the file just as if the text had been in the original file.



## 5.11 /defformat Command

Syntax:

### **/defformat format commands**

The /defformat command allows you to set the default format for the entire document. The format commands can be text formatting commands. This format will be applied to all text in the document. If the /defformat command is used inside of a /topic then the format will only be used for that topic. If there is a document wide format a topic format will override the document format. You will notice that this topic is a different color from the rest of the topics. This is caused by using a /defformat command in this topic to change the text color. The actual command is given below.

Example:

```
/defformat \cf9
```

Once a /defformat is in place all text in the document will have the specified formatting attributes applied to it. You can, however, override the /defformat commands. For example, if you set the defformat to the following:

```
/defformat \fs20\cf5
```

the font size is set to 20 and the foreground color to color number 5 for the whole document. Now if inside a topic you use another /defformat command

```
/topic
```

```
/defformat \fs40
```

```
/endtopic
```

You will override the document font size of 20 with a font size of 40, but because you did not change the foreground color it will remain color 5. Using this information it is possible to lay your /defformat out so as to make your help writing as easy as possible. If most of your paragraphs in a topic need a space of 100 after them but some need as space of 0, set your /defformat in the topic as follows:

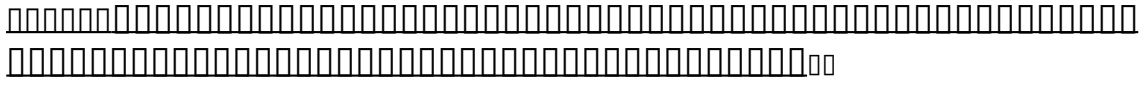
```
/defformat \sa100
```

Now on the paragraphs that need 0 space do the following:

```
/para \sa000  
/endpara
```

This causes the paragraph to have no spacing after it., which is just what we wanted.

□□









### 5.13 /pragma Command

Syntax:

#### **/pragma option**

The /pragma command is used to set internal QDHELP processor options. The following options are supported:

- debug
- noheader
- nohpj

The options cause the following actions:

debug

This option causes QDHELP to put special RTF commands into the resultant help file. The commands put a new button on the WinHelp button bar named "Source". This button will bring up the file which contains the source for the help page which you are currently viewing. The default editor used is Windows Notepad.

If you do not wish to use Notepad you must set the environment variable QDHELP\_DEBUG\_EDITOR in your DOS environment prior to running QDHELP:

```
set QDHELP_DEBUG_EDITOR=b.exe
```

NOTE: you must be in the directory with the source files for the debug source browse to work. If you run WinHelp by calling it with a full path to the help file the source browse will not be able to find the source files.

noheader

This option suppresses the output of the .HHH file. The QDHELP default is to generate the .HHH file.

nohpj

This option suppresses the output of the .HPJ file. The QDHELP default is to generate the .HPJ file. If you suppress the .HPJ file you must place the correct information in the .HPJ file for the /debug option to work. Look at a .HPJ file generated by QDHELP with debug on and off to see the special things QDHELP adds to the .HPJ file for debug.



## 5.14 /title Command

**Syntax:**

**/title search title text**

Use the title command to designate the search title for the topic. In the above example the string "search title text" would become the search title for this topic. The search title is displayed when a search is done on a keyword defined for the topic.

**Placement:**

The title command is only valid within a topic definition before any paragraphs have been defined.

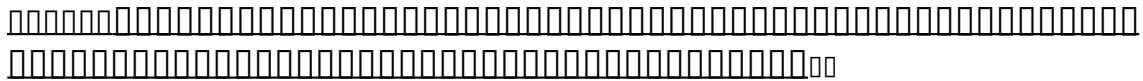
**Limitations:**

None.

**Example:**

See the example for the /keyword command.





## 5.15 /keywords Command

### Syntax:

**/keywords word1[;word2...]**

The keyword command is used to set the keywords that will find a particular topic. The keywords are used by the Microsoft Windows 3.0 help engine. These are the words you will find in the Search For window of the dialog box that appears when you press the search button while using help.

### Placement:

The keyword command is only valid within a topic definition before any paragraphs have been defined.

### Limitations:

None.

### Example:

```
/topic KEYWORD_EXAMPLE
```

```
    /title Copying and Pasting Text
```

```
    /keywords cut;document;Document menu commands;double space;edit
```

```
    /para
```

```
    This is just some text in an example
```

```
    /endpara
```

```
/endtopic
```

In the above example five (5) keywords are defined for this topic. The keywords are cut, document, Document menu commands, double space, and edit. In the search command of the MS Windows help engine these three words would be available in the Search For list in the Search window (Figure 1). Choosing one and performing the search would cause the title string "Copying and Pasting Text" to be shown in the Topics Found list in the Search window (Figure 2). Also note that the keyword cut must have also been defined in the topics with titles "Edit Menu Commands" and "Pasting,Copying, and Cutting Pictures".

Search For:

```
-----
|cut                               |
-----
-----
|cut                               |
|document                          |
|Document menu commands            |
|double space                      |
-----
```

□□

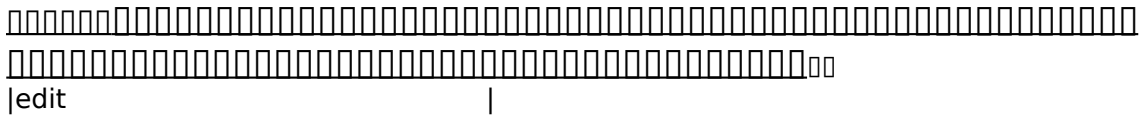


Figure 1.

### 3 Topics Found



Figure 2.



```
#####  
#####
```

```
/topic6  
/browse category2,AUTO  
/endtopic
```

Using the above defined topics, while topic1 is displayed by the help engine the browse backward key will be off and the browse forward key will be on. Pressing the browse forward key would bring you to topic2. At this point the browse backward key would be on as well as the browse forward key. Hitting the browse forward key again brings us to topic3. At this point the browse backward key is on and the browse forward key is off because this is the last topic in category1.

For category2 we are using the AUTO browse numbering feature of QDHELP. This means that topic4 will be the first in the browse sequence for category2 (it came first in the input file) and that topic5 will be second and topic6 will be last. In all ways they will act just like topics 1,2 and 3. The benefit of using AUTO is that to add a new topic you do not have to know what the last position value used was. All you have to do is figure out where in the browse sequence you want the new topic to fall and add the text at that point in the input file.



## 5.17 /para /endpara Commands

### Syntax:

**/para [paragraph format commands]**

...

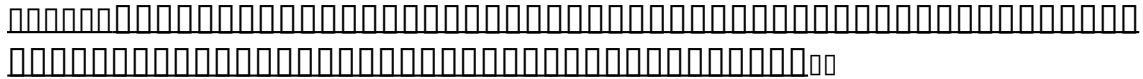
**/endpara**

The /para /endpara commands mark the start and end of a paragraph, respectively. There are several commands that are only valid inside a paragraph definition. They are /text, /link, /popuLink, /bitmap and /bitmaplink. For more information see section 6.1, Command Placement.

The paragraph format commands are RTF commands that are passed through to the help compiler unchanged. Below is a list of the most useful RTF commands for paragraph formatting and their meanings. Many commands deal with the unit called twips. A twip is 1/1440 of an inch (that is not very much!).

NOTE: that RTF commands begin with a backslash (\) whereas QDHELP commands begin with a slash (/).

RTF Command	Category	Description
\ql	Justification	Paragraph left justified.
\qr	Justification	Paragraph right justified.
\qj	Justification	Paragraph plain justified.
\qc	Justification	Paragraph centered.
\fiXXX	Indentation	First line indented XXX twips
\liXXX	Indentation	Left margin all lines indented XXX twips
\riXXX	Indentation	Right margin all lines indented XXX twips
\saXXX	Spacing	XXX twips after the last line of the paragraph
\sbXXX	Spacing	XXX twips before the first line of the paragraph.
\slXXX	Spacing	XXX twips space between lines of the paragraph
\brdrt	Border Placement	Border on top of the paragraph
\brdrb	Border Placement	Border on bottom of the paragraph
\brdrl	Border Placement	Border on left edge of the paragraph
\brdrr	Border Placement	Border on right edge of the paragraph
\box	Border Placement	Border on all sides of the paragraph
\brdrs	Border Style	Single line border
\brdrth	Border Style	Single thick line border
\brdrsh	Border Style	Shadow border (try it, you'll like it!)
\brdrdb	Border Style	Double line border



<code>\brdrdot</code>	Border Style	Dotted line border
<code>\keep</code>	Word wrap	Turn off word wrapping.
<code>\keepn</code>	Scrolling Region	Create a non -scrolling region

The paragraph format can contain any combination of these. Of course, if you give it two (2) different border styles for one paragraph the results will be unpredictable.

**Placement:**

The para command is only valid in a topic after the definition of the title, keyword, and browse sequences. A topic does not need to have a title, keyword, or browse command in it, but if it does the para must come after any that are included.

**Limitations:**

None.

**Example:**

```
/topic PARA_FORMAT_EXAMPLE  
  
  /title Paragraph Format Example  
  
  /para \sa200 \box  
  This text would have a border drawn around every side of it and would have  
200 twips of  
  blank space placed after it.  
  /endpara  
  
  /para \qc  
  This paragraph would be centered. It would appear 200 twips after the last  
line of the  
  previous paragraph. This is due to the previous paragraph's \sa200  
command.  
  /endpara  
  
/endtopic
```





## 5.19 /helpid

Syntax:

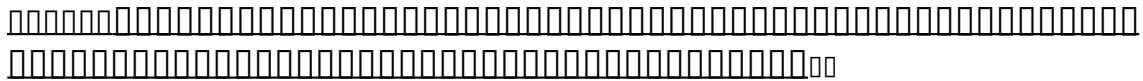
**/helpid value**

The /helpid allows you to assign a specific value for the context id of a topic. If no helpid is give QDHELP will automatically generate a value for a topic. These are the values that are written into the .HHH and .VB header files. If you are writing help in a situation where the program has already defined the values of the context sensitive help ids then this command will allow you to tie that value to the topic.

**Placement :**

This command must be placed between the /topic /endtopic pair .





## 5.20 /text Command

### Syntax:

**/text format commands , text string**  
**or**  
**{ format command text string }**

The /text command allows special inline formatting of text in your paragraphs. The format commands are applied to the text string which follows.

The text format commands are RTF commands that are passed through to the help compiler unchanged. Below is a list of the most useful RTF commands for text formatting and their meanings. Some commands deal with the unit called twips. A twip is 1/1440 of an inch (that is not very much!).

NOTE: RTF commands begin with a backslash (\) whereas QDHELP commands begin with a slash (/).

RTF Command	Category	Description
\b	Text Formatting	Bold Text
\i	Text Formatting	Italic Text
\strike	Text Formatting	Strike Thru Text
\ul	Text Formatting	Underline Text
\cfXXX where XXX is	Text Formatting	Change color of text to color XXX, 0 thru 15.
\fsXXX	Text Formatting	Change font size to XXX twips
\fXXX Rmn,	Text Formatting	Change font where XXX is 0 = Tms 2 = Helv and 4 = Courier.

### Placement:

This command can only appear inside a /para /endpara group. If found outside an error will be generated.

### Limitations:

None.

### Example:

```
/topic TEXT_FORMAT_EXAMPLE  
  
  /title Text Format Example  
  
  /para  
  This is some nonaffected text  
  /text \b,This text would be printed bold
```

□□

XX  
XX

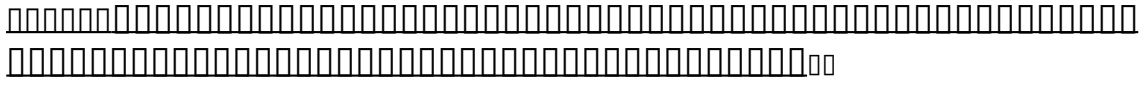
/endpara

/para  
This is some nonaffected text {\b This text would be printed bold} Indeed.  
/endpara

/endtopic

Note: there is no comma (,) between the format commands and the text string when using the { format string } syntax.





```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```

## 5.22 /popuplink Command

### Syntax:

#### **/popuplink TOPIC\_NAME, link text [,line]**

The popuplink command is the way that you see a help topic without actually moving to it. This command will cause a link from the current topic to the TOPIC\_NAME topic. The link will be activated by pressing the mouse button while over the "link text" that will be dashed underlined and printed in green (standard look for help popup hyperlink in MS Windows). See the /link command for a description of the ,line option.

When activated the topic linked to will be displayed in a popup window for as long as the left mouse button is held down.

### Placement:

This command can only appear inside a /para /endpara group. If found outside an error will be generated.

### Limitations:

None.

### Example:

```
/topic POPUP_LINK_EXAMPLE
```

```
    /title Popup Link Example
```

```
    /para
    This is some more text that is meaningless.
    /popuplink POPUP_TEXT, Show popup text
    /endpara
```

```
/endtopic
```

```
/topic POPUP_TEXT
```

```
    /title Popup Text
```

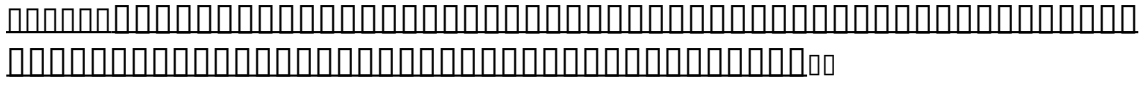
```
    /para
    This text will pop up in a window while the user holds down the left mouse
button
    on the "Show popup text" link in the POPUP_LINK_EXAMPLE topic.
    /endpara
```

```
/endtopic.
```

The popup links work just like the regular links except when chosen instead of moving to the new topic the topic is shown in a popup window.

```


```

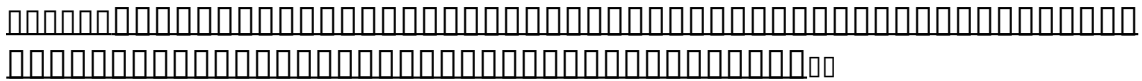












## 5.25 /bitmap Command

### Syntax:

#### **/bitmap position , FILENAME.BMP**

The /bitmap command places a bitmap into the help file at a position. The bitmap is in the file named FILENAME.BMP. The file name should **not** contain any DOS path information. The path information about where the bitmap file is located on your disk drive should be placed in the .HPJ file used by HC (MS Help Compiler). Your Help Compiler manual should answer any questions you have regarding the .HPJ file contents. The position of the bitmap is determined by the value placed into the position parameter. The valid values are shown in Table 1.

Position Value	Description
l	Left, place the bitmap at the left margin of the current text.
r	Right, place the bitmap at the right margin of the current text.
c	Character, place the bitmap at the location that the next character in the sentence would have been placed.
lwd	TBD
rwd	TBD
cwd	TBD

Table 1.

### Placement:

This command can only appear inside a /para /endpara group. If found outside an error will be generated.

### Limitations:

Text does not always align the way you would expect when using this command. You must try the command and judge the results on a case by case basis.

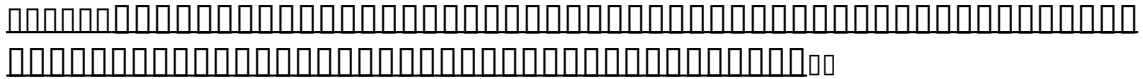
### Example:

```
/topic BITMAP_EXAMPLE

    /title Bitmap Example

    /para
    Left sided bitmap
    /bitmap l,MYBITS.BMP
    To the left!
    /endpara

    /para
    Right sided bitmap
```



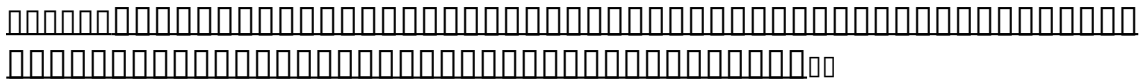
```
 /bitmap r,MYBITS.BMP  
 To the right!  
 /endpara
```

```
 /para  
 before the bitmap  
 /bitmap c,MYBITS.BMP  
 after the bitmap  
 /endpara
```

```
 /endtopic
```

In the above topic the same bitmap would be displayed three (3) times. First the bitmap would be on the left margin of the help topic, then on the right margin, and then in mid sentence. The right and left margin justified bitmaps do not always come out before the text that is after them in the paragraph. We are working to find out if this is a limitation of the help engine or QDHELP.





## 6. Language Structure

### 6.1 Command Placement Example

Some commands in QDHELP must be positioned in certain places. This section graphically describes the relationship of commands and placement in QDHELP.

The // command and the /include command can be used anywhere.

```
//
/include SOMEFILE.QDH
```

The /topic command has a specific order in which the commands should be placed in it.

```
/topic TOPIC1
```

```

-----
|           /title This is the title           | <-- These commands must be between the
| /topic and                                   | the first /para command. They can
|           /keywords one;two;three           | be in any order.
|           /browse category,1               |
|                                           |
-----

```

```

/para
Some text
/endpara

```

```

/para

```

```

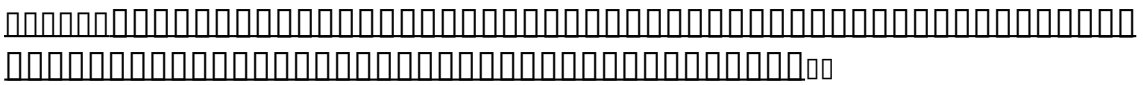
-----
|           /text \b,BOLD TEXT                 |
|           /link TOPIC2,Go To Topic 2         |
|           /popuplink TOPIC2,Popup Topic 2    | <-- These commands must be
used inside the                               | /para . . . /endpara commands.
|           /bitmaplink TOPIC2,TOPIC2.BMP     | They can
|           /bitmap c,PICTURE.BMP             | be used multiple times in any
order.                                         |
|                                           |
-----
/para

```

```

/endtopic

```



## 6.2 Command Nesting

The /topic /endtopic and /para /endpara commands must occur in matched pairs and cannot be nested.

Incorrect use of /topic /endtopic commands:

```

/topic TOPIC1
/topic TOPIC2 <--    ERROR The TOPIC1 definition must be finished before beginning
TOPIC2
/endpoint
/endpoint

```

```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```

## 7. An Annotated Example

To practice using QDHELP the following example will guide you step by step through the basics needed to generate a simple help file. As part of your distribution you will find a file named TPLATE1.QDH. Below you will find a line by line annotation of that file.

ANNO: the // characters signal a comment to QDHELP. The text following the // characters are  
 ANNO: ignored completely.

```

//
// template #1 for QDHELP
//
// This is a sample template for use in building
// help files using the QDHELP program. These
// templates give you the basic building blocks
// to quickly put together a help file for your
// programs.
//
//

```

ANNO: The /topic command is the QDHELP command that starts a help topic. In help files the basic ANNO: unit of information is the topic. The name following the /topic command is the name of the given  
 ANNO: topic. Here the topic is named INDEX. This is a name usually given to the first topic in the ANNO: help file. The index is the topic that is returned to when the Index button in the help engine is ANNO: pressed.

```
/topic INDEX
```

ANNO: The /title command is the title of the topic. The title of the topic is used when keyword searches  
 ANNO: are performed in the help engine. When the search is performed a list of keywords is shown. ANNO: When a search is performed on a keyword the title of all topics under which the keyword is found ANNO: are listed.

```
/title Template #1
```

ANNO: The /para command signals the start of a new paragraph to QDHELP. The RTF commands found ANNO: after the/para command are formatting commands. In this specific example \sa 50 instructs the ANNO: Help Compiler to leave a space of 50 twips after this paragraph and to left justify the paragraph.

```
/para \sa150
```

ANNO: The /text command signals QDHELP that we want to perform some special text formatting on the ANNO: text that follows on this line. The \b\i\fs50 instruct the

```


```



Help Compiler that the text Template #1 ANNO: should be printed bold (\b) italic (\i) and with a font size of 50 twips (\fs50).

```
/text \b\i\fs50,Template #1
```

ANNO: The /endpara command signals that the current paragraph definition is finished.

```
/endpara
```

ANNO: The only thing different about this paragraph is the \fi200 paragraph format command. This ANNO: command tells the Help Compiler that the first line in this paragraph should be indented 200 twips.

```
/para \fi200 \sa150  
This is template #1 for the QDHELP system. This template  
has a large title followed by an introductory paragraph. Following  
that come bold topic headings and paragraphs for each  
topic.  
/endpara
```

ANNO: This paragraph contains no new information.

```
/para \sa150  
/text \b\fs30,Topic Heading  
/endpara
```

```
/para \fi200 \sa150  
This is the topic heading paragraph. It will contain  
information on the topic heading. Next you will find  
a series of links to the subtopics of this topic.  
/endpara
```

ANNO: This paragraph contains our first /link command

```
/para
```

ANNO: The /link command signals QDHELP to make a hypertext link between the current topic and the ANNO: topic named as the first parameter (in this case SUBTOPIC1). The second parameter is the text ANNO: that will appear for this link (in this case Sub Topic 1). This is the text that normally appears in ANNO: green with an underline in the Windows help engine.

```
/link SUBTOPIC1,Sub Topic 1  
/endpara
```

```
/para  
/link SUBTOPIC2,Sub Topic 2  
/endpara
```

□□



```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```

ANNO: The /endtopic command signals QDHELP that the definition of the current topic is finished.

/endtopic

```

//
// subtopic 1
//

```

/topic SUBTOPIC1

    /title Sub Topic 1

ANNO: The /keywords command signals QDHELP that the following keywords should be added to the ANNO: Help Compiler's list of search keywords. If the user performs a search on the one of the ANNO: keywords the title for this topic, along with any other topic that contains the keyword, will be ANNO: displayed.

    /keywords subtopic1;topic

ANNO: The /browse keyword signals QDHELP that this topic belongs to a browse category. The first ANNO: parameter after the /browse command is the browse category. The second is the position of this ANNO: topic in that browse category. In this example the category name is subtopic and this is the first ANNO: item in that category. Note: Once of the features of QDHELP is that instead of numbering each ANNO: item in a browse category you can let QDHELP do it for you. If instead of a numeric value you ANNO: place the keyword AUTO (it must be in caps) as the second parameter to the /browse command ANNO: QDHELP will number the items for the browse category in the same order in which they occur in ANNO: the input file.

    /browse subtopic,1

ANNO: You have seen this before.

```

/para \sa150
/text \b\i\fs50,Subtopic 1 Title
/endpara

```

```

/para
This will be the text for the subtopic 1 information.
/endpara

```

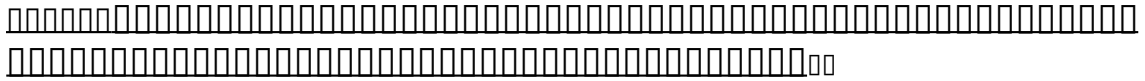
/endtopic

```

//
// subtopic 2
//

```





## **Glossary**

RTF	Rich Text Format
HC	Microsoft Help Compiler
twip	1/1440th of an inch
MS Windows	Microsoft Windows