
Help RTF Statements

Help RTF statements are an extended subset of tokens defined by the Microsoft Rich Text Format (RTF) standard. The RTF statements specify character and paragraph properties, such as font, color, spacing, and alignment, for text and graphics in the Help file. This appendix describes the subset of RTF statements recognized by the Windows Help compiler. You should use this appendix if you plan to edit RTF files directly or create an RTF reader/writer that converts files to RTF format. By observing the rules in this appendix, you ensure that your RTF files will compile successfully.

The following sections outline the basic elements used in RTF files and explain the rules for their use. A sample RTF file describes in detail the different components that make up the file. The appendix also includes a comprehensive reference of all RTF statements supported by Windows Help 3.1.

Rich-Text

The rich-text format (RTF) standard is a method of encoding formatted text and graphics for easy transfer between different applications and different operations. Generally, it is used by all Microsoft Word applications—Word for Windows, Word for the Macintosh, and Word for MS-DOS—in order to move word-processing documents between different platforms without having to rely on special translation software or conversion utilities. Because the RTF standard provides a format for text and graphics interchange that can be used with different output devices and operating systems, Windows Help also supports this standard. That means you can use any text editor that generates RTF output, including your own custom RTF editor, to create the source files that are built into Help files.

Software that takes a formatted file and turns it into an RTF file is referred to in this appendix as an RTF *writer*. Software that translates an RTF file into a formatted file is referred to as an RTF *reader*. An RTF writer separates the application's control information from the actual text and writes a new file containing the text and RTF groups associated with that text. An RTF reader does the converse of this operation.

Elements of

Help RTF statements are presented to the Microsoft Help compiler in topic files, which are specified in the [FILES] section of a Help project file. To the Help compiler a topic file consists of the following elements:

- RTF statements
- Groups
- Unformatted text

RTF statements, control symbols, and braces constitute control information. Text grouping is used to define the format and placement of text and graphics in the Help file. All other characters in RTF text constitute plain text.

RTF Statements

An *RTF statement* is a formatted tag that specifies a particular kind of information in the RTF file. An RTF statement consists of a backslash (\) followed by an RTF statement name and delimiter:

`\statement-name<delimiter>`

RTF statements always begin with a backslash (\). The *statement-name* identifies the RTF statement and specifies what it does. No spaces or other characters may separate the backslash and the statement-name.

An RTF statement may optionally have a *number* parameter immediately after the name. This number must be in the range of a signed integer (-32767 through 32767). No spaces or other characters may separate the name and the numeric parameter (if any).

Certain statements control properties (such as bold and italic) that have only two states. When the RTF statement has no parameter or has a non-zero parameter, the statement is used to turn the property on. When the RTF statement has a 0 (zero) parameter, the statement is used to turn the property off. For example, `\b` turns on bold, whereas `\b0` turns off bold.

Delimiters

RTF statements must be separated from subsequent text or statement parameters by a delimiter. A delimiter can be one of the following:

- A space. (In this case, the space is considered part of the statement.)
- Any character other than a letter or digit. (In this case, the delimiting

character terminates the statement but is not considered part of the statement. A “letter” is an upper- or lowercase ASCII letter.)

When a space is used as a delimiter, the Help compiler discards it. If any other character is used, the compiler processes it as text or the start of another RTF statement. For example, if a backslash is used as a delimiter, the compiler interprets it as the beginning of the next RTF statement.

For example, the following line demonstrates usage of the `\tab` statement and a space delimiter:

`text in paragraph\tab more text`

Control Symbols

A *control symbol* consists of a backslash (\) followed by a single non-letter. They require no further delimiting. The following control symbols are supported:

Control Symbol	Meaning
<code>\ </code>	Formula character
<code>\~</code>	Nonbreaking space
<code>\-</code>	Optional hyphen
<code>_</code>	Nonbreaking hyphen
<code>\:</code>	Subentry in an index entry

Groups

A *group* consists of Help RTF statements and text enclosed in braces (**{ }**). The opening brace **{** indicates the start of the group, and the closing brace **}** indicates the end of the group, as shown in this example:

```
{   group start, and  
}   group end.
```

Different groups perform different functions in the RTF file. Some groups specify such information as the fonts, styles, colors, summary information, and document-format attributes used in the file. Other groups simply include text text and the RTF statements that define attributes for that text. Still other groups specify picture data, footnotes, bookmarks, annotations, and section- and paragraph-formatting attributes for the file.

Formatting specified within a group affects only the text within that group. Text within a group inherits any formatting of the text preceding the group.

Destinations

Some statements, referred to as *destinations*, mark the beginning of a collection of related text. A destination is marked by a control word that signals the beginning of the group. An example of a destination is the **\footnote** group, where the Help-specific text follows the statement. Destination statements and their following text must be enclosed in braces, as in this example:

```
{\footnote main_contents}
```

Unformatted Text

Unformatted text consists of any combination of 7-bit ASCII characters. Although characters whose values are greater than 127 are not permitted in topic files, the **** statement can be used to insert them in the final Help file. The Help compiler treats spaces as part of the text, but it discards carriage return and linefeed characters.

Because the backslash character (\) and braces ({ }) have specific meanings in RTF, they must be preceded with a backslash if you want to use them as plain text, as shown by the control symbols \b, \i, and \j.

When reading a stream of RTF text, an RTF reader must accomplish the following tasks:

Semantics

1. Separate RTF control information from plain text.
2. Act on control information.

This is designed to be a relatively simple process, as described in the next section.

Some control information just contributes special characters to the plain text stream. Other information changes the *program state*, which includes document properties as a whole and a stack of *group states* that apply to parts of the document. The group state is saved by the opening { brace and is restored by the closing } brace.

The current group state specifies:

- ⁿ The *destination* or part of the document that the plain text is building up.
- ⁿ The section formatting properties.
- ⁿ The paragraph formatting properties.
- ⁿ The character formatting properties.

3. Collect and properly dispose of any remaining plain text as directed by the current group state.

Acting on Control Information

When parsing the RTF stream, the RTF reader should follow this process:

1. Read the next character.
2. If the character is an opening brace (char == '{'), store the current state on the stack.

Or if the current state does not change, then continue.

3. If the character is a closing brace (`char == '}'`), retrieve the current state from the stack.

Generally, this will change the state.

4. If the character is a backslash (`char == '\\'`), collect the RTF statement or control symbol and its parameter.

a. If there is one, look up the statement or symbol in the symbol table (a constant table) and act according to the description found there.

The parameter is left available for use by the action.

b. Leave a read pointer before or after the delimiter, as appropriate.

c. After completing the action, continue.

The different actions are listed in the next section.

5. Otherwise, if the character is anything other than `{`, `}`, or `\`, write the plain text character to the current destination using the current formatting properties.

6. Read the next character.

Acting on Symbol Table Entries

When looking up RTF statements and symbols in the symbol table, the RTF reader can take any of the following actions:

▮ Change the destination to the destination described in the table entry.

Most destination changes are legal only immediately after an opening brace. Other restrictions may also apply (for example, footnotes may not be nested).

▮ Change formatting property.

The symbol table entry will describe the property and whether the parameter is required.

▮ Insert a special character.

The symbol table entry will describe the character code.

▮ End of paragraph.

This can also be viewed as just a special character.

ⁿ End of section.

This can also be viewed as just a special character. B-7

ⁿ Ignore the character.

The RTF File

This section describes the different groups that make up an RTF file. It first presents a sample RTF file, which is used in examples throughout the section. It then describes the required and optional components of an RTF file.

Sample RTF File

The following is an example of a simple, but complete (one topic), RTF file that can be compiled by the Help compiler:

```
{\rtf1\ansi\deff0\deflang1024

{\fonttbl
{\f0\froman Times New Roman;}
{\f1\froman Symbol;}
{\f2\swiss Arial;}
{\f3\froman MS Serif;}
{\f4\swiss MS Sans Serif;}
}

{\colortbl;
\red0\green0\blue0;
\red0\green0\blue255;
\red0\green255\blue0;
\red255\green0\blue0;
\red255\green255\blue255;

#{\footnote graphics_cont}
${\footnote Graphics}

\pard\plain\li120\sb340\sa120\sl-320\fs28 Graphics

\par \pard\plain\li120\fs20 Use graphic images to illustrate concepts and present
information visually. Graphics include line art, icons, screen shots of the interface,
and graphics with hot spots ("hypergraphics").
\par
}
```

As noted earlier, the entire RTF file is one group, so the file begins and ends with braces ({}). Other groups nested within the group containing the file define the header and text information.

Required Entries

The following entries must appear in the order shown immediately after the opening brace of the RTF file:

- The RTF statement **\rtfn**. This RTF statement identifies the version (given by the number *n*) of the RTF standard used in the file. For the Help compiler, this RTF statement must be **\rtf1**.
- An RTF statement identifying the character set used in the file. The RTF character set described in this appendix corresponds to the **\ansi** character set statement.

System default values, such as font number (**\deffn** statement), follows the character set statement. The **\deffn** statement specifies which font defined in the font table (see the next section) is the default font for the file.

The required entries appear as follows in the sample RTF file:

```
{\rtf1\ansi\deff0
```

Font Table

The first group listed after the opening statements is the font table group. All of the fonts that the RTF file will use must be declared in the font table, which begins with the **\fonttbl** statement. These fonts should correspond to the fonts that will be present on the end user's machine.

The font table group has the following form:

```
{\fonttbl
{\font-number\font-family font-name;}
{\font-number\font-family font-name;}
.
.
.
}
```

The actual font table is the group beginning with the RTF statement **\fonttbl**. Each font is defined as a separate group within the font-table group. Each font definition has three components: the *font-number*, the *font-family*, and the *font-name*.

The *font-number* is an integer that identifies the font.

The *font-family* is one of the following standard font families defined by the Windows operating system:

Control Word	Font Family	Examples
fnil	Default or unknown	
froman	Roman	MS Serif and Palatino
fswiss	Swiss	Helvetica, MS Sans Serif
fmodern	Modern	Courier, Elite, Pica
fscript	Script	Cursive, Script, Zapf Chancery
fdcor	Decorative	Old English, Zapf Dingbats
ftch	Technical	Symbol

Each font definition must end with a semicolon (;) immediately before the closing brace.

If the RTF file uses a default font, the RTF statement `\def\font-number` appears before the font-table group. The *font-number* is an integer indicating which font in the table is used for the default

For example, the following group defines nine fonts:

```
{\fonttbl
{\f0\froman Times New Roman;}
{\f1\froman Symbol;}
{\f2\swiss Arial;}
{\f3\froman MS Serif;}
{\f4\swiss MS Sans Serif;}
{\f5\modern Courier;}
{\f6\decor ZapfDingbats;}
{\f7\swiss HelveticaCondensed;}
{\f8\swiss Helvetica;}
}
```

This example defines the following fonts:

Font Number	Font Family	Font
0	Roman	Times New Roman
1	Technical	Symbol
2	Swiss	Arial
3	Roman	MS Serif
4	Swiss	MS Sans Serif
5	Modern	Courier

6	Decorative	Zaph Dingbats
7	Swiss	Helvetica Condensed
8	Swiss	Helvetica

Color Table

If the Help file uses color, the RTF file must also define a color table group to specify the individual colors. The color table group has the following form:

```
{\colortbl;
\rednumber\greennumber\bluenumber;
\rednumber\greennumber\bluenumber;
.
.
.
}
```

Each definition specifies a value in the range 0 through 255 for the red, green, and blue components of the color. Unlike font entries in the the font table, color entries in the color table do not specify a color number. Instead, the Help compiler assumes that the first color defined is color zero, the second color is color one, and so on.

The following example defines the standard 16 colors in the Windows palette:

```
{\colortbl;
\red0\green0\blue0;
\red0\green0\blue255;
\red0\green255\blue255;
\red0\green255\blue0;
\red255\green0\blue255;
```

```
\red255\green0\blue0;  
\red255\green255\blue0;  
\red255\green255\blue255;  
\red0\green0\blue127;  
\red0\green127\blue127;  
\red0\green127\blue0;  
\red127\green0\blue127;  
\red127\green0\blue0;  
\red127\green127\blue0;  
\red127\green127\blue127;  
\red192\green192\blue192;  
}
```

Style Sheet

In the standard RTF specification, the style sheet group defines the different styles defined for a document. In Word for Windows, a style is a named combination of formats (including those for characters, paragraphs, tabs, borders, and frames) that authors can apply to text using a defined sequence of keystrokes. Authors can add styles to a document template.

The Help compiler ignores the style sheet group in an RTF file. If you plan to use the RTF file only to create a Help file, you can omit the style-sheet group from the file to save space. However, if you also plan to print documents from the RTF file, you can include a style sheet group to define styles for the document. Then, when you open the RTF file in Word for Windows, you can apply the styles that you have defined to text.

A style sheet group has the following format:

```
{stylesheet  
  {\style-number\style-specs style-name;}  
  {\style-number\style-specs style-name;}  
  .  
  .  
  .  
}
```

Each style definition appears in a group within the style sheet group. The style definition is identified by a unique style number.

The exact style specifications follow the style number. The style specifications can include RTF statements for any type of character, text, tab, border, or frame formatting. (RTF statements for these different formats are defined later in this appendix.)

The last part of the style definition is the style name. This is the name that appears in the Word for Windows drop-down list boxes that list the document's styles. A single space must separate the style specifications from the style name.

Each style definition must end with a semicolon (;) immediately before the closing brace.

In Word for Windows, styles may be based on previously defined styles. In this case, the style inherits the attributes of the previously defined style. A style may also define a style to be applied to the next paragraph in the document. The following RTF statements may appear in the style sheet to activate these features:

Control Word	Meaning
<code>\sbasedonstyle-number</code>	The style defined in this group inherits the attributes of the style with the given style number. If this style does not inherit attributes from any style, <i>style-number</i> is 0.
<code>\snextstyle-number</code>	The style with the given style number is applied to the next paragraph in the document. If no style should be applied to the next paragraph in the document, <i>style-number</i> is 0.

For example, here is a portion of a sample style sheet:

```
{\stylesheet
{\s229\li2160\ri720\tldot\tx8280\tqr\tx8640 \f5\fs20\lang1033 \sbasedon0\snext0 toc 4;}
.
.
.
{\s17\qc\sb240\sl240\keepn \b\fs28\lang1033 \sbasedon0\snext17 head;}
{\s18\qj\sb240\sl240 \b\fs20\lang1033 \sbasedon0\snext18 Company;}
{\s19\qj\sb120\sl240\keepn \i\fs20\lang1033 \sbasedon0\snext19 Product;}
}
```

This portion of the style sheet defines four styles numbered 229 (named “toc 4”), 17 (named “head”), 18 (named “Company”), and 19 (named “Product”). Style sheets 17 through 19 assume that the same style will be applied to the next paragraph as to the paragraph to which the style is currently assigned. None of these styles inherits the attributes of a previously defined style.

Topic Information

After setting up the required RTF entries for the Help file, you define the topic information. Topics consist of plain text and graphics that the user sees, Help-specific information which is defined in topic footnotes, character and paragraph formatting information, and page break delimiters.

Font and Formatting Information

Before any text is placed in the RTF file, the font name and font size must be specified. The `\fn` statement specifies the font name (*n* matches the font number defined in the font table). The `\fsn` statement specifies the font size (*n* is given in half-points). For example, using the sample font table just described, the following example defines the text as 10-point MS Sans Serif:

```
\f4\fs20
```

If you want the text to have any special formatting characteristics, you must also define those before you write out the plain text. The Help compiler supports a number of character and paragraph formatting attributes that you can use to change the appearance and placement of text and graphics.

The following example defines the topic title text as 14-point MS Serif with 17 points of space before, 6 points space after, and 16 points of leading. The topic title paragraph is also indented 6 points from the left margin:

```
\pard\plain \li120\sb340\sa120\sl-320 \f3\fs28
```

Help-Specific Information

Help-specific information is specified in footnote groups and formatting information embedded within plain text.

Help Features in Footnote Groups

The following types of Help information are specified in **\footnote** groups:

- ⁿ Context strings
- ⁿ Topic titles
- ⁿ Browse sequences
- ⁿ Keyword-index entries

ⁿ Help macros executed on topic entry

ⁿ Build tags

Appendix B B-15

ⁿ Comments

Context Strings

Topics within a Help file are usually identified by a unique context string. The following group specifies a context string for a topic:

#{\footnote *context string*}

A context string is any string of up to 255 characters. Valid characters are the alphabetic characters A—Z, the numeric characters 0—9, and the period (.) or underscore (_) character.

For example, this RTF entry defines the context string “main_contents”:

#{\footnote main_contents}

Topic Titles

Typically, a topic will also contain a topic title, which identifies the topic in Help dialog boxes. The following group specifies a topic title:

\${\footnote *title-text*}

A title string is any string of up to 50 characters. Any printable ASCII character may appear in a topic title. However, ASCII characters such as braces ({ }), brackets ([]), and backslashes (\) that are used as special characters in RTF must be prefixed by backslashes.

This example defines the title “Saving a Document”:

\${\footnote Saving a Document}

Browse Sequences

A topic may belong to a single *browse sequence*, which is a group of topics the user can view in forward or backward sequence.

The following group assigns a topic to a browse sequence and to one or more topic groups:

+{\footnote [sequence-name][:sequence-number]}

Microsoft Windows Help Authoring Guide The *sequence-name* is the name of the browse sequence to which the topic is assigned. It may be omitted if the Help file has only one browse sequence.

The *sequence-name* may be followed by a *sequence-number*, which indicates where in the sequence the topic appears. In browse sequences, topics appear in order of sequence numbers. If no sequence number appears, topics in the sequence appear in their physical order within the file. If the [FILES] section of the project file lists more than one RTF file, topics from the first RTF file appear first in the browse sequence, then topics from the second RTF file, and so on.

A colon must immediately precede the sequence number, if it appears. If a sequence name also appears, a colon must separate the sequence name and sequence number.

Keyword Entries

Keywords are used for searches with the Search button. The Search dialog box displays a list of the keywords defined for topics in the Help file. From the Search dialog box, the user can jump to any topic in which a keyword has been defined.

The following group assigns one or more keywords to a topic:

K{\footnote keyword;[keyword;]...}

Each *keyword* is a word assigned to the keyword index. Keywords must be separated by semicolons (;).

Help Macros Executed on Topic Entry

Help can execute commands automatically when the user jumps to a topic (from a hot spot, browse button, or search list).

The following group specifies macros to be executed on topic entry:

!{\footnote macro-string}

The *macro-string* can be any Help macro documented in Chapter 15 of this authoring guide or registered in the [CONFIG] section of the project file.

Build Tags

Build tags are labels used to exclude certain topics from a build. Topics without build tags are always included in the build. Topics with build tags are included or excluded using a build expression in the Help project file.

The following group assigns one or more build tags to a topic:

***{\footnote *tag-name*;*[tag-name];...*}**

A *tag-name* is a label assigned to the topic and referenced in the [BUILD] section of the Help project file. Build tags are case-insensitive and can contain up to 32 alphanumeric characters, except spaces. A build tag must be the first footnote in a topic. Multiple build tags must be separated by semicolons (;).

Comments

Comments are any text that you want to include with a topic for your own purposes. Comments are ignored by the Help compiler.

The following group assigns a comment to a topic:

@{\footnote *comment-text*}

Because comments are ignored, the *comment-text* can have as many standard characters as you want, including accented characters and spaces.

Help Features Embedded in Text

Some Help features are implemented using normal RTF commands or special codes embedded within text. The following types of Help features can be embedded in text:

- ▮ Layout features like nonscrolling regions and nonwrapping text
- ▮ Hot spots for activating jumps, displaying pop-up windows, and executing Help macros
- ▮ Bitmaps
- ▮ Embedded panes for custom DLL objects

Layout Features

The following standard RTF commands are used to specify layout features in Help files:

Control Word	Meaning
<code>\keepn</code>	Makes affected text and pictures part of the nonscrolling region
<code>\keep</code>	Makes affected text and pictures nonwrapping so they are truncated instead of wrapped if the window is resized

Hot Spots

A *hot spot* is a region within a topic that performs an action when the user selects the region with the mouse. This action may be jumping to another topic, displaying a topic in a pop-up window, or executing a Help macro.

The following group is used to create pop-up hot spots in a topic:

`{\ul hot-spot text}{\v [%]*context-string}`

The *hot-spot* text is the topic text that performs the action when the user selects it. This text may be one of the standard `\{bm\}` RTF commands if the hot spot is a bitmap. (See the section titled “Bitmaps” later in this appendix for information about `\{bm\}` commands.)

The percent sign (%) or asterisk (*) is used if you want to make the hot spot invisible (no green color, no underline) or just underlined (no green color). See Chapter 8, “Creating Links and Hot Spots,” for more information.

The *context-string* identifies the topic to be displayed in the pop-up window.

The following groups are used to create jump or macro hot spots in a topic:

`{\uldb hot-spot text}{\v [%]*hot-spot action}`

Or, you can use this format:

`{\strike hot-spot text}{\v [%|*]hot-spot action}`

The *hot-spot* text is the topic text that performs the action when the user selects it. This text may be one of the standard `\{bm\}` RTF commands if the hot spot is a bitmap. (See the section titled “Bitmaps” later in this appendix for information about `\{bm\}` commands.)

The percent sign (%) or asterisk (*) is used if you want to make the hot spot invisible (no green color, no underline) or just underlined (no green color). See Chapter 8, “Creating Links and Hot Spots,” for more information.

The *hot-spot action* may be one of the following:

A jump destination. This action has the following form:

context-string[*@HLP-filename*][>*window-name*]

The *context-string* identifies the topic the hot spot jumps to. *@.HLP-filename*, if present, identifies a different Help file in which the jump destination appears. If the jump destination should appear in a secondary window, >*window-type* identifies the name of window in which the topic should appear. The *window-type* must be defined in the [WINDOWS] section of the project file. (See Chapter 17 for more information about the [WINDOWS] section.)

A Help macro. This action has the following form:

!macro-string

The *macro-string* identifies the action Help performs when the user selects the hot spot. It can be any standard Help macro listed in Chapter 15 or any external DLL function registered in the [CONFIG] section of the project file.

If the hot-spot text is invisible in the title, a percent sign (%) immediately precedes the *hot-spot action*.

Bitmaps

Bitmaps with up to 16-colors can be inserted into a topic with a command that tells Help the name of the bitmap file and how to position it. This command has the following form:

`\{bm[c, l, r[wd] bitmapfile\}`

The `\{bmx\}` commands specify how the bitmap is aligned in the topic, as shown in the following table:

Control Word Meaning

<code>\{bmc\}</code>	Positions the bitmap using character alignment. The bitmap is handled as if it were another character and it appears at the position in the line indicated by the reference. Text follows the bitmap, positioned at the base of the bitmap.
<code>\{bml\}</code>	Positions the bitmap along the left margin, with text wrapping automatically along the right edge of the image.
<code>\{bmr\}</code>	Positions the bitmap along the right margin, with text wrapping automatically along the left edge of the image.

The bitmap data is stored separately from the topic text, and a single copy of the bitmap is used for all the `\{bmc\}`, `\{bml\}`, and `\{bmr\}` commands displaying that bitmap.

The `\{bmcwd\}`, `\{bmlwd\}`, and `\{bmrwd\}` have the same effects as the corresponding commands without “**wd**,” except that the bitmap data is stored in-line with the topic text, in the same location as the reference. Bitmaps displayed by these references cannot be larger than 64K.

The *bitmapfile* is the filename for the 16-color bitmap displayed in the topic. Either this bitmap must be in a path given by the **ROOT** or **BMROOT** option in the project file, or it must be listed in the [BITMAPS] section of the project file. (See Chapter 17 for more information about these options.)

Embedded Windows

Embedded windows display text, pictures, or other objects in a window embedded within a Help topic. Authors can write their own custom DLLs to render elements in an embedded window. (See Chapter 20, “Writing DLLs for the Windows Help,” for more information.)

An RTF entry for an embedded window has the following syntax:

`\ew[c, l, r]` *DLL-name, window-class, author-data* **B-21**

The `\ewx\` commands specify how the window is aligned in the topic, as shown in the following table:

Control Word Meaning

<code>\ewc\</code>	Positions the window using character alignment. The window is handled as if it were another character and it appears at the position in the line indicated by the reference. Text follows the window, positioned at the base of the window.
<code>\ewl\</code>	Positions the window along the left margin, with text wrapping automatically along the right edge of the window.
<code>\ewr\</code>	Positions the window along the right margin, with text wrapping automatically along the left edge of the window.

The *DLL-name* is the filename of the DLL that renders the object in the window. This name should not have an extension, but it can have a relative path.

The *window-class* is the window-class name for the embedded pane as defined in the C source file for the DLL.

The *author-data* is a programmer-defined string passed to the DLL (in the WM_CREATE message) when Help creates the window.

Topic End

When there is more than one topic in an RTF file, each topic should end with a **`\page`** statement:

`\page`

Ending the last topic in a file with a **`\page`** statement is optional.

Sample Topic

Microsoft Windows Help Authoring Guide In the sample RTF file, the following entries make up the single topic the file defines:

`#{\footnote graphics_cont}
${\footnote Graphics}`

`\pard\plain \li120\sb340\sa120\sl-320 \f3\fs28 Graphics`

`\par \pard\plain \li120 \f4\fs20 Use graphic images to illustrate concepts and present information visually. Graphics include line art, icons, screen shots of the interface, and graphics with hot spots ("hypergraphics").`

`\par`

Overview of Help RTF Statement

Although the Help compiler supports many RTF statements, it does not support them all. The following tables present the RTF statements by group, and briefly describe the statements that Help

does and does not support in this release of Windows Help. The following tables are not intended to provide detailed information about the RTF statements. For complete information, see the "Help RTF Statement Reference," later in this Appendix.

Statement descriptions follow these conventions.

Convention	Meaning
------------	---------

Bold Statement	Statement IS supported by the Help compiler.
-----------------------	---

Plain Statement	Statement is IGNORED by the Help compiler.
-----------------	---

Parentheses	Indicates the default value for the statement.
-------------	--

Overloaded	Indicates that the RTF statement has a specific meaning in Help that is different from its print-based usage.
------------	---

Overloaded Statements

The compiler interprets some RTF statements differently from their normal print-based usage. For example, standard RTF specifies that the **\uldb** statement indicates a double underline, but the Help compiler uses this statement to indicate a hot spot. The following table lists the RTF statements that are overloaded for the Help compiler.

Statement Print-based usage		Help compiler usage
\footnote	Footnote	Special topic commands
\keep	Keeps paragraph intact	Makes text nonwrapping
\keepn	Keeps paragraph with next	Creates a nonscrolling region at the top of the topic
\page	Creates page break	Ends the current topic
\strike	Creates strikethrough	Indicates a hot spot
\trqc	Centers table row with respect to its containing column	Uses relative column widths

<code>\ul</code>	Creates continuous underline	Indicates a link to a pop-up topic
------------------	------------------------------	------------------------------------

Microsoft Windows Help Authoring Guide

<code>\uldb</code>	Creates double underline	Indicates a hot spot
--------------------	--------------------------	----------------------

<code>\w</code>	Creates hidden text	Indicates the context string to jump to
-----------------	---------------------	---

Character Set

Microsoft Help supports all RTF character sets.

Statement	Character set	Action
<code>\ansi</code>	ANSI character set	Supported.
<code>\windows</code>	Windows (default)	Supported.
<code>\mac</code>	Apple Macintosh	Supported.
<code>\pc</code>	OEM code page 437	Supported.
<code>\pca</code>	International English code page 850	Supported.

Special Characters

Special characters are defined as they exist in Microsoft Word for the Macintosh. Other characters may be added for interchanging files with other applications. If the Help compiler does not recognize a character, it is ignored.

For simplicity, ASCII 9 is treated the same as **\tab** and ASCII 10 is treated the same as **\par**. ASCII 13 is ignored. The control code \<10> is also ignored, even though it may be used to indicate a soft carriage return.

Statement	Meaning	Action
*	Custom RTF destination.	Ignored.
\-	Optional hyphen.	Ignored.
\:	Subentry in index entry.	Ignored.
_	Nonbreaking hyphen.	Ignored.
\	Formula character.	Ignored.
\~	Nonbreaking space.	Ignored.

Microsoft Windows Help Authoring Guide	Hexadecimal value of specified character.	Supported.
\bullet	Bullet.	Ignored.
\cell	End of table cell.	Supported.
\chatn	Annotation reference.	Ignored.
\chdate	Current date.	Ignored.
\chftn	Auto-numbered footnote reference.	Ignored.
\chftnsep	Anchoring character for footnote separator.	Ignored.
\chftnsepc	Anchoring character for footnote continuation.	Ignored.
\chpgn	Current page number.	Ignored.
\chpict	Placeholder character for picture.	Ignored.

<code>\chtime</code>	Current time.	Ignored.
<code>\column</code>	Required column break.	Ignored.
<code>\emdash</code>	Em-dash.	Ignored.
<code>\endash</code>	En-dash.	Ignored.
<code>\ldbquote</code>	Left double quotation mark.	Ignored.
<code>\line</code>	Required line break; same as SHIFT+ENTER.	Supported.
<code>\lquote</code>	Left single quotation mark.	Ignored.
<code>\page</code>	End of current topic.	Overloaded.
<code>\par</code>	End of paragraph.	Supported.

--	--	--

<code>\rdbquote</code>	Right double quotation mark.	Ignored.
------------------------	------------------------------	----------

Microsoft Windows Help Authoring Guide

<code>\row</code>	End of table row.	Supported.
-------------------	-------------------	------------

<code>\rquote</code>	Right single quotation mark.	Ignored.
----------------------	------------------------------	----------

<code>\sect</code>	End of section and paragraph.	Supported.
--------------------	-------------------------------	------------

<code>\tab</code>	Tab character.	Supported.
-------------------	----------------	------------

--	--	--

Destinations

A destination change resets all properties to their default values. Changes are legal only at the beginning of a group (statement and text enclosed in braces).

Statement	Meaning	Action
-----------	---------	--------

<code>\colortbl</code>	Color table. See RTF Reference.	Supported.
------------------------	---------------------------------	------------

<code>\comment</code>	Comment text.	Ignored.
-----------------------	---------------	----------

<code>\fonttbl</code>	Font table. See RTF Reference.	Supported.
-----------------------	--------------------------------	------------

<code>\footer</code>	Footer for current section.	Ignored.
<code>\footnote</code>	Topic information. See RTF Reference.	Overloaded.
<code>\header</code>	Header for the current section.	Ignored.
<code>\info</code>	Document information block.	Ignored.
<code>\nextfile</code>	Next file to print or index.	Ignored.
<code>\pict</code>	Picture.	Supported.
<code>\rtf</code>	Version of RTF standard used.	Supported.
<code>\stylesheet</code>	Document stylesheet.	Ignored.
<code>\template</code>	Document template.	Ignored.

Document Formatting

Windows Help version 3.1 does not support any document formatting statements.

Statement	Meaning	Action
<code>\defformat</code>	Saves document in RTF format.	Ignored.
<code>\deftabn</code>	Default tab width.	Ignored.
<code>\enddoc</code>	Footnotes at end of document.	Ignored.
<code>\endnotes</code>	Footnotes at end of section.	Ignored.
<code>\facingp</code>	Facing pages.	Ignored.
<code>\fracwidth</code>	Fractional character widths.	Ignored.
<code>\ftnbj</code>	Footnotes at bottom of page.	Ignored.
<code>\ftncn</code>	Footnote separator for continued footnote notice.	Ignored.

Appendix B			
B-31			
<code>\ftnrestart</code>	Restart footnote numbers each page.		Ignored.
<code>\ftnsep</code>	Footnote separator.		Ignored.
<code>\ftnsepc</code>	Footnote separator for continued footnotes.		Ignored.
<code>\ftnstartn</code>	Starting footnote number.		Ignored.
<code>\ftntj</code>	Footnotes beneath text.		Ignored.
<code>\guttern</code>	Gutter width.		Ignored.
<code>\hyphhotz</code>	Hyphenation hot zone.		Ignored.
<code>\landscape</code>	Print in landscape format.		Ignored.
<code>\linestartn</code>	Starting line number.		Ignored.
<code>\makebackup</code>	Backup document.		Ignored.

Microsoft Windows Help Authoring Guide

<code>\margbn</code>	Bottom margin.	Ignored.
<code>\margln</code>	Left margin.	Ignored.
<code>\margmirror</code>	Switches margin definitions.	Ignored.
<code>\margrn</code>	Right margin.	Ignored.
<code>\margtn</code>	Top margin.	Ignored.
<code>\paperhn</code>	Paper height.	Ignored.
<code>\paperwn</code>	Paper width.	Ignored.
<code>\pgnstartn</code>	Starting page number.	Ignored.
<code>\psover</code>	Print PostScript over text.	Ignored.

--	--	--

<code>\revbarn</code>	Vertical revision marks.	Ignored.
-----------------------	--------------------------	----------

<code>\revisions</code>	Turns on revision marking.	Ignored.
-------------------------	----------------------------	----------

<code>\revpropn</code>	Revision properties.	Ignored.
------------------------	----------------------	----------

<code>\widowctrl</code>	Enable widow control.	Ignored.
-------------------------	-----------------------	----------

--	--	--

Section Formatting

Windows Help version 3.1 does not support any section formatting statements.

Statement	Meaning	Action
<code>\colbreak</code>	Break code.	Ignored.
<code>\colsn</code>	Number of columns.	Ignored.
<code>\colsn</code>	Space between columns.	Ignored.
<code>\endnhere</code>	Include endnotes in this section.	Ignored.

Microsoft Windows Help Authoring Guide		
Evenbreak	Break code.	Ignored.
\footeryn	Footer y position from bottom of page.	Ignored.
\headeryn	Header y position from top of page.	Ignored.
\linebetcol	Line between columns.	Ignored.
\linecont	Line number continued from previous section.	Ignored.
\linemodn	Line number modulus.	Ignored.
\lineppage	Line number restart on each page.	Ignored.
\linerestart	Line number restart at 1.	Ignored.
\linestartsn	Beginning line number. (1)	Ignored.
\linexn	Line number, text distance.	Ignored.

<code>\nobreak</code>	Break code.	Ignored.
<code>\oddbreak</code>	Break code.	Ignored.
<code>\pagebreak</code>	Break code.	Ignored.
<code>\pgncont</code>	Continuous page numbering. (default)	Ignored.
<code>\pgndec</code>	Page number format, decimal.	Ignored.
<code>\pgncltr</code>	Page number format, lowercase letter.	Ignored.
<code>\pgnlcrm</code>	Page number format, lower case roman.	Ignored.
<code>\pgnrestart</code>	Restart page numbers at 1.	Ignored.
<code>\pgnstartsn</code>	Beginning page number. (1)	Ignored.

--	--	--	--

\pgnucltr	Page number format, uppercase letter.	Ignored.
-----------	---------------------------------------	----------

Microsoft Windows Help Authoring Guide

\pgnucrm	Page number format, uppercase roman.	Ignored.
----------	--------------------------------------	----------

\pgnxn	Auto page number, x pos.	Ignored.
--------	--------------------------	----------

\pgnyn	Auto page number, y pos.	Ignored.
--------	--------------------------	----------

\sbkcol	Section break starts new column.	Ignored.
---------	----------------------------------	----------

\sbkeven	Section break starts at even page.	Ignored.
----------	------------------------------------	----------

\sbknone	No section break.	Ignored.
----------	-------------------	----------

\sbkodd	Section break starts at odd page.	Ignored.
---------	-----------------------------------	----------

\sbkpage	Section break starts new page.	Ignored.
----------	--------------------------------	----------

\sectd	Reset to default section properties.	Ignored.
--------	--------------------------------------	----------

<code>\titlepg</code>	Title page is special.	Appendix B	B-37	Ignored.
-----------------------	------------------------	-------------------	-------------	----------

<code>\vertal</code>	Bottom-aligned text.	Ignored.
----------------------	----------------------	----------

<code>\vertalc</code>	Vertically centered text.	Ignored.
-----------------------	---------------------------	----------

<code>\vertalj</code>	Vertically justified text.	Ignored.
-----------------------	----------------------------	----------

<code>\vertalt</code>	Top-aligned text. (default)	Ignored.
-----------------------	-----------------------------	----------

Paragraph Formatting

The following statements specify paragraph formatting properties.

Statement	Meaning	Action
<code>\box</code>	Boxed paragraph.	Supported.
<code>\brdrb</code>	Bottom border.	Supported.

\brdrbar	Ouside border.	Supported.
<hr/>		
Microsoft Windows Help Authoring Guide		
\brdrbtw	Border between paragraphs.	Ignored.
\brdrdb	Double border.	Supported.
\brdrdot	Dotted border.	Supported.
\brdrhair	Hairline border.	Ignored.
\brdr l	Left border.	Supported.
\brdr r	Right border.	Supported.
\brdrs	Single-thickness border.	Supported.
\brdrsh	Shadow border.	Supported.
\brdr t	Top border.	Supported.

\brdrth	Thick border.	Appendix B	B-39	Supported.
\brspn	Space between border and object.			Ignored.
\fin	First-line indent. (0)			Supported.
\intbl	Table paragraph.			Supported.
\keep	Nonwrapping text.			Overloaded.
\keepn	Nonscrolling region.			Overloaded.
\lin	Left indent. (0)			Supported.
\noline	No line numbering.			Ignored.
\pagebb	Page break before.			Ignored.
\pard	Default paragraph properties.			Supported.

Microsoft Windows Help Authoring Guide

<code>\qc</code>	Centered.	Supported.
<code>\qj</code>	Justified.	Supported.
<code>\ql</code>	Left-aligned (default).	Supported.
<code>\qr</code>	Right-aligned.	Supported.
<code>\rin</code>	Right indent. (0)	Supported.
<code>\san</code>	Space after. (0)	Supported.
<code>\sbn</code>	Space before. (0)	Supported.
<code>\sbys</code>	Side-by-side paragraphs.	Ignored in Help 3.1. Supported in Help 3.0.
<code>\sln</code>	Line spacing or leading.	Supported.

<code>\sn</code>	Style.	Appendix B	B-41	Ignored.
<code>\tbn</code>	Bar tab.			Supported.
<code>\tldot</code>	Leader dots.			Ignored.
<code>\tlhyph</code>	Leader hyphens.			Ignored.
<code>\tlth</code>	Leader thick line.			Ignored.
<code>\tlul</code>	Leader underscore.			Ignored.
<code>\tqc</code>	Centered tab.			Supported.
<code>\tqdec</code>	Decimal-aligned tab.			Ignored.
<code>\tqr</code>	Flush-right tab.			Supported.
<code>\txn</code>	Custom tab position.			Supported.

Character Formatting

The following statements specify character formatting properties.

Statement	Meaning	Action
<code>\b</code>	Bold	Supported.
<code>\caps</code>	All capitals.	Ignored.
<code>\dnn</code>	Subscript position. (6 pt.)	Ignored.
<code>\expandn</code>	Expansion. (0)	Ignored.
<code>\fn</code>	Font number	Supported.
<code>\fsn</code>	Font size (24 pt.)	Supported.
<code>\i</code>	Italic	Supported.

	<code>\outl</code>	Outline.	Appendix B B-43	Ignored.
	<code>\plain</code>	Resets application's default character formatting properties.		Supported.
	<code>\revised</code>	Text added after revision marking turned on.		Ignored.
	<code>\scaps</code>	Small capitals		Supported.
	<code>\shad</code>	Shadow.		Ignored.
	<code>\strike</code>	Jump or macro hot spot; identical to <code>\uldb</code>		Overloaded.
	<code>\ul</code>	Pop-up hot spot		Overloaded.
	<code>\uld</code>	Dotted underline.		Ignored.
	<code>\uldb</code>	Jump or macro hot spot; identical to <code>\strike</code>		Overloaded.
	<code>\ulnone</code>	Stop all underlining.		Ignored.

Microsoft Windows Help Authoring Guide

<code>\ulw</code>	Word underline.	Ignored.
<code>\upn</code>	Superscript position. (6 pt.)	Ignored.
<code>\v</code>	Context string or macro	Overloaded.

Tables

The following statements specify table formatting properties.

Statement	Meaning	Action
<code>\cellxn</code>	Set absolute position of a table cell's right edge.	Supported.
<code>\clbrdrb</code>	Bottom table cell border.	Ignored.
<code>\clbrdrl</code>	Left table cell border.	Ignored.
<code>\clbrdr</code>	Right table cell border.	Ignored.

<code>\clbrdrt</code>	Top table cell border.	Appendix B	B-45	Ignored.
<code>\clmgf</code>	Mark first cell in a range of cells to be merged.			Supported.
<code>\clmrg</code>	Merge current cell with preceding cell.			Supported.
<code>\trgaphn</code>	Space between text in adjacent cells.			Supported.
<code>\trleftn</code>	Set the position of left margin for the first cell in table row.			Supported.
<code>\trowd</code>	Set table row defaults.			Supported.
<code>\trqr</code>	Right aligns text in each cell of table row.			Ignored.
<code>\trqc</code>	Relative column widths.			Overloaded.
<code>\trql</code>	Left align text in each cell of table row. (default)			Supported.
<code>\trrhcn</code>	Height of table row.			Ignored.

Pictures

The following statements specify picture formatting.

Statement	Meaning	Action
<code>\binn</code>	Binary picture data.	Supported.
<code>\box</code>	Boxed picture.	Supported.
<code>\brdrb</code>	Bottom picture border.	Supported.
<code>\brdrbar</code>	Outside picture border.	Supported.
<code>\brdrdb</code>	Double picture border.	Supported.
<code>\brdrdot</code>	Dotted picture border.	Supported.
<code>\brdrhair</code>	Hairline picture border.	Ignored.

<code>\brdl</code>	Left picture border.	Appendix B	B-47	Supported.
<code>\brdr</code>	Right picture border.			Supported.
<code>\brdrs</code>	Single-thickness picture border.			Supported.
<code>\brdrsh</code>	Shadow picture border.			Supported.
<code>\brdrt</code>	Top picture border.			Supported.
<code>\brdrth</code>	Thick picture border.			Supported.
<code>\dibitmap</code>	Device independent bitmap.			Supported.
<code>\macpict</code>	QuickDraw picture.			Ignored.
<code>\piccropbn</code>	Bottom cropping value.			Ignored.
<code>\piccropln</code>	Left cropping value.			Ignored.

Microsoft Windows Help Authoring Guide

<code>\piccroprn</code>	Right cropping value.	Ignored.
<code>\piccroptn</code>	Top cropping value.	Ignored.
<code>\pichgoaln</code>	Desired picture height.	Supported.
<code>\pichn</code>	Picture height.	Supported.
<code>\picscaled</code>	Scale picture to fit in frame.	Ignored.
<code>\picscalexn</code>	Horizontal scaling value. (100)	Supported.
<code>\picscaleyn</code>	Vertical scaling value. (100)	Supported.
<code>\picwgoaln</code>	Desired picture width.	Supported.
<code>\picwn</code>	Picture width.	Supported.

<code>\wbitmap</code> <i>n</i>	Picture type (Default is Windows bitmap).	Supported.
Appendix B B-49		
<code>\wbmbitspixel</code> <i>n</i>	Bits per pixel. (1)	Supported.
<code>\wbmplanes</code> <i>n</i>	Number of bitmap color planes. (1)	Supported.
<code>\wbmwidthbytes</code> <i>n</i>	Bitmap width, in bytes.	Supported.
<code>\wmetafile</code> <i>n</i>	Windows metafile.	Supported.

Absolute-Positioned Objects

Windows Help version 3.1 does not support any absolute-positioned-object statements.

Statement	Meaning	Action
<code>\abswn</code>	Absolute width of paragraph text.	Ignored.
	Horizontal distance from text in main text flow.	Ignored.

`\dxfrtextn`

Microsoft Windows Help Authoring Guide

	Positions horizontally relative to column	Ignored.
	.	

`\phcol`

	Positions horizontally relative to margin.	Ignored.
--	--	----------

`\phmrg`

	Positions horizontally relative to page.	Ignored.
--	--	----------

`\phpg`

	Centers horizontally within reference frame	Ignored.
--	---	----------

`\posxc`

	Positions horizontally inside reference frame.	Ignored.
--	--	----------

`\posxi`

	Positions to left within reference frame.	Ignored.
--	---	----------

`\posxl`

Appendix B			B-51
\	Positions from left edge of reference frame.		Ignored.
posxn			
	Positions horizontally outside reference frame.		Ignored.
\posxo			
	Positions to right within reference frame.		Ignored.
\posxr			
	Positions at bottom of reference frame.		Ignored.
\posyb			
	Centers vertically within reference frame.		Ignored.
\posyc			
	Positions vertically to be in-line.		Ignored.
\posyil			

`\posyn`

Positions at top of reference frame.

Ignored.

`\posyt`

Positions vertically relative to margin.

Ignored.

`\pvmrg`

Positions vertically relative to page.

Ignored.

`\pvpg`

Annotations

Windows Help version 3.1 does not support any annotation statements.

Statement	Meaning	Action
<code>\annotation</code>	Annotation group reference.	Ignored.

\antid	Annotation author's identification text.	Ignored.
--------	--	----------

\chatn	Annotation reference character.	Ignored.
--------	---------------------------------	----------

Headers and Footers

Windows Help version 3.1 does not support any header and footer statements.

Statement	Meaning	Action
\footerf	Footer for first page.	Ignored.
\footerl	Footer for left-hand pages.	Ignored.
\footerr	Footer for right-hand pages.	Ignored.
\headerf	Header for first page.	Ignored.
\headerl	Header for left-hand pages.	Ignored.

\headerr	Header for right-hand pages.	Ignored.
----------	------------------------------	----------

Microsoft Windows Help Authoring Guide

--	--	--

Document Information

Windows Help version 3.1 does not support any information statements.

Statement	Meaning	Action
\author	Document's author.	Ignored.
\buptim	Backup time.	Ignored.
\comment	Comment text.	Ignored.
\creatim	Creation time.	Ignored.
\doccomm	Comments in Edit Summary Info dialog box.	Ignored.
\dyn	Day.	Ignored.
\edmins <i>n</i>	Total editing time.	Ignored.

<code>\hrn</code>	Hour.	Appendix B	B-55	Ignored.
<code>\idn</code>	Internal identification number.			Ignored.
<code>\keywords</code>	Selected document keywords.			Ignored.
<code>\minn</code>	Minutes.			Ignored.
<code>\mon</code>	Month.			Ignored.
<code>\nextfile</code>	Next file.			Ignored.
<code>\nofcharsn</code>	Number of characters.			Ignored.
<code>\nofpagesn</code>	Number of pages.			Ignored.
<code>\nofwordsn</code>	Number of words.			Ignored.
<code>\operator</code>	Last person to make changes.			Ignored.

Microsoft Windows Help Authoring Guide

\printtim	Last print time.	Ignored.
\revtim	Revision time.	Ignored.
\secn	Seconds.	Ignored.
\subject	Subject matter.	Ignored.
\title	Document title.	Ignored.
\vern <i>n</i>	Internal version number.	Ignored.
\version <i>n</i>	Document version number.	Ignored.
\yr <i>n</i>	Year.	Ignored.

Fields

Windows Help version 3.1 supports only one field statement.

Statement	Meaning	Action
\flddirty	Change made since last update.	Ignored.
\fldedit	Text edited since last update.	Ignored.
\fldinst	Field instructions.	Ignored.
\fldlock	Field is locked.	Ignored.
\fldpriv	Result is in unknown format.	Ignored.
\fldrslt	Most recent calculated result of the field.	Supported.

Index Entries

Windows Help version 3.1 does not support any index statements.

Statement	Meaning	Action
\bxe	Bold cross-reference or page number.	Ignored.
\ixe	Italic cross-reference or page number.	Ignored.
\rx	Generate page numbers for range of text specified by bookmark name.	Ignored.
\tx	Use text instead of page number.	Ignored.

Table of Contents Entries

Windows Help version 3.1 does not support any table of contents statements.

Statement	Meaning	Action
\tcf <i>n</i>	Type of table of contents.	Ignored.
\tcl <i>n</i>	Level number. (1)	Ignored.

Bookmarks

Appendix B B-59

Windows Help version 3.1 does not support any bookmark statements.

Statement	Meaning	Action
<code>\bkmkstartn</code>	Start of specified bookmark.	Ignored.
<code>\bkmkend</code>	End of specified bookmark.	Ignored.

Help RTF Statement Reference

This section lists the Windows Help RTF statements in alphabetic order.

The Help RTF Statement Reference lists in alphabetic order all RTF

statements supported by the Microsoft Help compiler version 3.1. Statement descriptions provide the following information.

Heading	Information
Statement	RTF statement supported by the Help compiler.
Comments	Notes about using the RTF statement, including any restrictions.
Example	Example of the RTF statement.

See Also Cross-references to similar Help RTF statements.

\ansi
\ansi

The **\ansi** statement sets the American National Standards Institute (ANSI) character set. The Windows character set is essentially equivalent to the ANSI character set.

See Also **\windows**

\b
\b

The **\b** statement starts bold text. The statement applies to all subsequent text up to the next **\plain** or **\b0** statement.

Comments No **\plain** or **\b0** statement is required if the **\b** statement and subsequent text are enclosed in braces. Braces limit the scope of a character property statement to just the enclosed text.

The **\b0** statement was first supported in the Microsoft Help compiler version 3.1.

Example The following example sets Note to bold:
{\b Note} Setting the Auto option frees novice users from determining their system configurations.

See Also **\i, \plain, \scaps**

\bin
\bin**n**

The **\bin** statement indicates the start of binary picture data. The Help compiler interprets subsequent bytes in the file as binary data. This statement is used in conjunction with the **\pict** statement.

Parameter	<i>n</i> Specifies the number of bytes of binary data following the statement.
Comments	<p>A single space character must separate the \bin statement from subsequent bytes. The Microsoft Help compiler assumes that all subsequent bytes, including linefeed and carriage return characters, are binary data. These bytes can have any value in the range 0 through 255. For this reason, the \bin statement is typically used in program-generated files only.</p> <p>If the \bin statement is not given with a \pict statement, the default picture data format is hexadecimal.</p>
See Also	\pict

\bmc
\{bmc *filename*\}

The **bmc** statement displays a specified bitmap or metafile in the current line of text. The statement positions the bitmap or metafile as if it were the next character in the line, aligning it on the base line and applying the current paragraph properties.

Parameter	<i>filename</i> Specifies the name of a file containing either a Windows bitmap, a placeable Windows metafile, a multiresolution bitmap, or a segmented graphics bitmap.
------------------	---

Comments	<p>Since the bmc statement is not a standard RTF statement, the Microsoft Help Compiler relies on the opening and closing braces, including the backslashes (\), to distinguish the statement from regular text.</p> <p>If a file containing a metafile is specified, the file must contain a placeable Windows metafile; the Help compiler will not accept standard Windows metafiles. Furthermore, Windows Help sets the MM_ANISOTROPIC mode prior to displaying the metafile, so the placeable Windows metafile must either set the window origin and extents or set some other mapping mode.</p>
-----------------	---

Example	<p>The following example inserts a bitmap representing a keyboard key in a paragraph:</p> <pre> \par Press the \{bmc escape.bmp\} key to return to the main window. \par </pre>
----------------	---

See Also

bml, bmr, \wbitmap

Microsoft Windows Help Authoring Guide

\bml

\{bml *filename*\}

The **bml** statement displays a specified bitmap or metafile at the left margin of the Help window. The first line of subsequent text aligns with the upper-right corner of the image and subsequent lines wrap along the right edge of the image.

Parameters

filename

Specifies the name of a file containing either a Windows bitmap, a placeable Windows metafile, a multiresolution bitmap, or a segmented graphics bitmap.

Comments

Since the **bml** statement is not a standard RTF statement, the Help compiler relies on the opening and closing braces, including the backslashes (\), to distinguish the statement from regular text.

If a file containing a metafile is specified, the file must contain a placeable Windows metafile; the Help compiler will not accept standard Windows metafiles. Furthermore, Windows Help sets the MM_ANISOTROPIC mode prior to displaying the metafile, so the placeable Windows metafile must either set the window origin and extents or set some other mapping mode.

Example

The following example places a bitmap at the left margin. The subsequent paragraph wraps around the bitmap:

```
\par
\{bml roadmap.bmp\}
The map at the left shows the easiest route to the school.
Although many people use Highway 125, there are fewer stops
and less traffic if you use Ames Road.
```

See Also

bmc, bmr, \wbitmap

\bmr

\{bmr *filename*\}

The **bmr** statement displays a specified bitmap or metafile at the right margin of the Help window. The first line of subsequent text aligns with the upper-left corner of the image and subsequent lines wrap along the left edge of the image.

<hr/>	
Parameter	<i>filename</i> Specifies the name of a file containing either a Windows bitmap, a placeable Windows metafile, a multiresolution bitmap, or a segmented graphics bitmap.
Comments	<p>Since the bmr statement is not a standard RTF statement, the Help compiler relies on the opening and closing braces, including the backslashes (\), to distinguish the statement from regular text.</p> <p>If a file containing a metafile is specified, the file must contain a placeable Windows metafile; the Help compiler will not accept standard Windows metafiles. Furthermore, Windows Help sets the MM_ANISOTROPIC mode prior to displaying the metafile, so the placeable Windows metafile must either set the window origin and extents or set some other mapping mode.</p>
Example	<p>The following example places a bitmap at the right margin. The subsequent paragraph wraps around the bitmap:</p> <pre>\par \{bmr roadmap.bmp\} The map at the right shows the easiest route to the school. Although many people use Highway 125, there are fewer stops and less traffic if you use Ames Road.</pre>
See Also	bmc, bml, \wbimap

\box \box	
	<p>The \box statement draws a box around the current paragraph or picture. The statement applies to all subsequent paragraphs or pictures up to the next \pard statement.</p>
Comments	<p>For paragraphs, Windows Help uses the height of the paragraph, excluding space before or after the paragraph, as the height of the box. For pictures (as defined by \pict statements), Windows Help uses the specified height of the picture as the height of the box. For both paragraphs and pictures, the width of the box is equal to the space between the left and right indents.</p> <p>Windows Help draws the box using the current border style.</p>
Example	<p>The following example draws a box around the paragraph:</p> <pre>\par \box {\b Note} Setting the Auto option frees novice users from</pre>

determining their system configurations.
`\par \pard`

Microsoft Windows Help Authoring Guide
`\brdrb`, `\brdrl`, `\brdrr`, `\brdrt`, `\pard`

`\brdrb`
`\brdrb`

The **\brdrb** statement draws a border below the current paragraph or picture. The statement applies to all subsequent paragraphs or pictures up to the next **\pard** statement.

Comments Windows Help draws the border using the current border style.

See Also **\box**, **\brdrbar**, **\brdrl**, **\brdrr**, **\brdrt**, **\pard**

`\brdrbar`
`\brdrbar`

The **\brdrbar** statement draws a vertical bar to the left of the current paragraph or picture. The statement applies to all subsequent paragraphs or pictures up to the next **\pard** statement.

Comments Windows Help draws the border using the current border style.

In a print-based document, the **\brdrbar** statement draws the bar on the right side of paragraphs on odd-numbered pages, but on the left side of paragraphs on even-numbered pages.

See Also **\box**, **\brdrb**, **\brdrl**, **\brdrr**, **\brdrt**, **\pard**

`\brdrdb`
`\brdrdb`

The **\brdrdb** statement selects a double line for drawing borders. The selection applies to all subsequent paragraphs or pictures up to the next **\pard** statement.

See Also **\brdrdot**, **\brdrs**, **\brdrsh**, **\brdrth**, **\pard**

\brdrdot

\brdrdot

The **\brdrdot** statement selects a dotted line for drawing borders. The selection applies to all subsequent paragraphs or pictures up to the next **\pard** statement.

See Also

\brdrdb, \brdrs, \brdrsh, \brdrth, \pard

\brdrl

\brdrl

The **\brdrl** statement draws a border to the left of the current paragraph or picture. The statement applies to all subsequent paragraphs or pictures up to the next **\pard** statement.

Comments

Windows Help draws the border using the current border style.

See Also

\box, \brdrb, \brdrbar, \brdrr, \brdrt, \pard

\brdrr

\brdrr

The **\brdrr** statement draws a border to the right of the current paragraph or picture. The statement applies to all subsequent paragraphs or pictures up to the next **\pard** statement.

Comments

Windows Help draws the border using the current border style.

See Also

\box, \brdrb, \brdrbar, \brdrl, \brdrt, \pard

\brdrs

\brdrs

The **\brdrs** statement selects a standard-width line for drawing borders. The selection applies to all subsequent paragraphs or pictures up to the next **\pard** statement.

See Also

\brdrdb, \brdrdot, \brdrsh, \brdrth, \pard

\bdrsh
\bdrsh

The **\bdrsh** statement selects a shadow outline for drawing borders. The selection applies to all subsequent paragraphs or pictures up to the next **\pard** statement.

See Also **\bdrddb, \bdrdot, \bdrs, \bdrth, \pard**

\brdrt
\brdrt

The **\brdrt** statement draws a border above the current paragraph or picture. The statement applies to all subsequent paragraphs or pictures up to the next **\pard** statement.

Comments Windows Help draws the border using the current border style.

See Also **\box, \bdrb, \bdrbar, \brdl, \bdr, \pard**

\bdrth
\bdrth

The **\bdrth** statement selects a thick line for drawing borders. The selection applies to all subsequent paragraphs or pictures up to the next **\pard** statement.

See Also **\bdrdb, \bdrdot, \bdrs, \bdrsh, \pard**

\cell
\cell

The **\cell** statement marks the end of a cell in a table. A cell consists of all paragraphs from a preceding **\intbl** or **\cell** statement to the ending **\cell** statement. Windows Help formats and displays these paragraphs using the left and right margins of the cell and any current paragraph properties.

Comments This statement was first supported in the Microsoft Help compiler version 3.1.

<hr/>	
Example	<p>The following example creates a two-column table. The second column contains three separate paragraphs, each having different paragraph properties:</p> <pre> \cellx2880\cellx5760 \intbl Alignment\cell \ql Left-aligned \par \qc Centered \par \qr Right-aligned\cell \row \pard </pre>
See Also	<code>\cellx</code> , <code>\intbl</code> , <code>\row</code> , <code>\trgaph</code> , <code>\trleft</code> , <code>\trowd</code>
<hr/>	

`\cellx`
`\cellxn`

The `\cellx` statement sets the absolute position of the right edge of a table cell. One `\cellx` statement must be given for each cell in the table. The first `\cellx` statement applies to the leftmost cell, the last to the rightmost cell. For each `\cellx` statement, the specified position applies to the corresponding cell in each subsequent row of the table up to the next `\trowd` statement.

Parameter *n*

Specifies the position of the cell’s right edge, in twips. The position is relative to the left edge of the Help window. It is not affected by the current indents.

Comments

A table consists of a grid of cells in columns and rows. Each cell has an explicitly defined right edge; the position of a cell’s left edge is the same as the position of the right edge of the adjacent cell. For the left-most cell in a row, the left edge position is equal to the Help window’s left margin position. Each cell has a left and right margin between which Windows Help aligns and wraps text. By default, the margin positions are equal to the left and right edges. The `\trgaph` and `\trleft` statements can be used to set different margins for all cells in a row.

This statement was first supported in the Microsoft Help compiler version 3.1.

Example

The following example creates a three-column table having two rows. The positions of the right edges of the three cells are 2, 4, and 6 inches, respectively:

```

\cellx2880\cellx5760\cellx8640
\intbl

```

Row 1 Cell 1\cell
Row 1 Cell 2\cell
Row 1 Cell 3\cell
Row 1 Cell 4\cell

Microsoft Windows Help Authoring Guide

\intbl
Row 2 Cell 1\cell
Row 2 Cell 2\cell
Row 2 Cell 3\cell
Row 2 Cell 4\cell
\row \pard

See Also \cell, \intbl, \row, \trgaph, \trleft, \trowd

\cb
\cbn

The **\cb** statement sets the background color. The new color applies to all subsequent text up to the next **\plain** or **\cb** statement.

Parameter

n
Specifies the color number to set as background. The number must be an integer number in the range 1 to the maximum number of colors specified in the color table for the Help file. If an invalid color number is specified, Windows Help uses the default background color.

Comments

No **\plain** or **\cb** statement is required if the **\cb** statement and subsequent text are enclosed in braces. Braces limit the scope of a character property statement to the enclosed text only.

If the **\cb** statement is not given, the default background color is the text color set by Control Panel.

Example

The following example creates a light grey background color:

```
{\colortbl;\red128\green128\blue128;}  
{\cb1 This background is light grey.}
```

See Also

\cf, **\colortbl**

\cf
\cfn

The **\cf** statement sets the foreground color. The new color applies to all

subsequent text up to the next **\plain** or **\cf** statement.

Parameter	<i>n</i>	Appendix B	B-69
		Specifies the color number to set as foreground. The number must be an integer number in the range 1 to the maximum number of colors specified in the color table for the Help file. If an invalid color number is specified, Windows Help uses the default foreground color.	
Comments		No \plain or \cf statement is required if the \cf statement and subsequent text are enclosed in braces. Braces limit the scope of a character property statement to the enclosed text only.	
		If the \cf statement is not given, the default foreground color is the text color set by Control Panel.	
Example		The following example displays green text: <code>{\colortbl;\red0\green255\blue0;} {\cf1 This text is green.}</code>	
See Also		\cb , \colortbl	

\chftn
\chftn*n*

The **\chftn** statement sets the footnote reference character for subsequent **\footnote** statements.

The Microsoft Help compiler ignores this statement.

Parameters	<i>n</i>	Specifies the footnote reference character.
See Also		\footnote

\clmgf
\clmgf

The **\clmgf** statement specifies the first cell in a range of cells to be merged.

The Microsoft Help compiler ignores this statement.

Microsoft Windows Help Authoring Guide All cells between the **\clmgf** statement and a subsequent **\clmrg** statement are combined into a single cell. The left edge of the new cell is the same as that of the leftmost cell to be merged; the rightedge is the same as that of the rightmost cell.

This statement was first supported in the Microsoft Help compiler version 3.1.

See Also **\clmrg**

\clmrg
\clmrg

The **\clmrg** statement merges the current cell with the preceding cell.

The Microsoft Help compiler ignores this statement.

Comments All cells between the **\clmgf** statement and a subsequent **\clmrg** statement are combined into a single cell. The left edge of the new cell is the same as that of the leftmost cell to be merged; the rightedge is the same as that of the rightmost cell.

This statement was first supported in the Microsoft Help compiler version 3.1.

See Also **\clmgf**

\colortbl
{**\colortbl**
\red**\green****\blue**;
.
.
.
}

The **\colortbl** statement creates a color table for the Help file. The color table consists of one or more color definitions. Each color definition consists of one **\red**, **\green**, and **\blue** statement specifying the amount of primary color to use to generate the final color. Each color definition must end with a semicolon (;).

Parameters *r*
Specifies the intensity of red in the color. It must be an integer in the

range 0 through 255.

g

Specifies the intensity of green in the color. It must be an integer in the range 0 through 255.

b

Specifies the intensity of blue in the color. It must be an integer in the range 0 through 255.

Comments

Color definitions are implicitly numbered starting at zero. A color definition's implicit number can be used in the **\cf** statement to set the foreground color.

The default colors are the window text and window background colors set by Control Panel. To override the default colors, a **\colortbl** statement and **\cf** and **\cb** statements must be given.

Example

The following example creates a color table containing two color definitions. The first color definition is empty (only the semicolon is given), so color number 0 always represents the default color. The second definition specifies green; color number 1 can be used to display green text:

```
{\colortbl;\red0\green255\blue0;}
```

See Also

\cb, **\cf**

\deff
\deff*n*

The **\deff** statement sets the default font number. Windows Help uses the number to set the default font whenever a **\plain** statement is given or an invalid font number is given in a **\f** statement.

Parameters

n

Specifies the number of the font to be used as the default font. This parameter must be a valid font number as specified by the **\fonttbl** statement for the Help file.

Comments

If the **\deff** statement is not given, the default font number is zero.

See Also

\f, **\fonttbl**, **\plain**

`\ewc`
`\{ewc DLL-name, window-class, author-data\}`

The **ewc** statement displays information (bitmap, multimedia element, and so on.) under the control of a dynamic-link library in the current line of text. The statement positions the object as if it were the next character in the line, aligning it on the base line and applying the current paragraph properties.

Parameters

DLL-name

Specifies the name of the DLL that controls the embedded window. The file name should not include an extension or be fully qualified, but it can include a relative path. Windows Help assumes .DLL or .EXE to be the default extension.

window-class

Specifies the name of the embedded window class as defined in the source code for the DLL.

author-data

Specifies an arbitrary string, which Windows Help passes to the embedded window when it creates the window. This string can be one or more substrings separated by any punctuation mark except a comma. The DLL is responsible for parsing this string.

Comments

Since the **ewc** statement is not a standard RTF statement, the Microsoft Help compiler relies on the opening and closing braces, including the backslashes (\), to distinguish the statement from regular text.

Because the embedded window is treated as text, paragraph formatting properties assigned to the paragraph also apply to the window. Text coming before or after the window does not wrap around the window. Help adjusts the height of the line containing the embedded window to allow enough space for the embedded window.

If you use the **\sl** statement in conjunction with an **ewc** statement, don't specify a negative value. If you do, the embedded window might appear on top of the succeeding paragraph when Windows Help displays the topic.

Example

The following example displays a 256-color bitmap in an embedded window within a paragraph:

```
\par  
preceding text \{ewc MVBMP, ViewerBMP, ..\art\ocean.bmp\} following text...
```


`\ewl`
`\{ewl DLL-name, window-class, author-data\}`

The **ewl** statement displays information (bitmap, multimedia element, and so on.) under the control of a dynamic-link library at the left margin of the Help window. The first line of subsequent text aligns with the upper-right corner of the embedded window and subsequent lines wrap along the right edge of the window.

Parameters

DLL-name

Specifies the name of the DLL that controls the embedded window. The filename should not include an extension or be fully qualified, but it can include a relative path. Windows Help assumes .DLL or .EXE to be the default extension.

window-class

Specifies the name of the embedded window class as defined in the source code for the DLL.

author-data

Specifies an arbitrary string, which Windows Help passes to the embedded window when it creates the window. This string can be one or more substrings separated by any punctuation mark except a comma. The DLL is responsible for parsing this string.

Comments

Since the **ewl** statement is not a standard RTF statement, the Microsoft Help compiler relies on the opening and closing braces, including the backslashes (\), to distinguish the statement from regular text.

Do not put any space characters between the **ewl** statement and the paragraph text unless you want the first line of text indented from the rest of the text that wraps along the right edge of the window. To be sure that text wraps correctly around an embedded window, insert a soft carriage return at the end of each line of text.

Example

The following example places an embedded window at the left margin. The subsequent paragraph wraps around the embedded window:

```
\par
\{ewl FADE, AmfWnd, clipbrd.amf}
The map at the left shows the easiest route to the school.
```

Although many people use Highway 125, there are fewer stops
and less traffic if you use Ames Road.

Microsoft Windows Help Authoring Guide
See Also **ewc**, **ewr**

`\ewr`

`\{ewr DLL-name, window-class, author-data\}`

The **ewr** statement displays information (bitmap, multimedia element, and so on.) under the control of a dynamic-link library at the right margin of the Help window. The first line of subsequent text aligns with the upper-left corner of the embedded window and subsequent lines wrap along the left edge of the window.

Parameters

DLL-name

Specifies the name of the DLL that controls the embedded window. The filename should not include an extension or be fully qualified, but it can include a relative path. Windows Help assumes .DLL or .EXE to be the default extension.

window-class

Specifies the name of the embedded window class as defined in the source code for the DLL.

author-data

Specifies an arbitrary string, which Windows Help passes to the embedded window when it creates the window. This string can be one or more substrings separated by any punctuation mark except a comma. The DLL is responsible for parsing this string.

Comments

Since the **ewr** statement is not a standard RTF statement, the Microsoft Help compiler relies on the opening and closing braces, including the backslashes (\), to distinguish the statement from regular text.

Do not put any space characters between the **ewr** statement and the paragraph text unless you want the first line of text indented from the rest of the text that wraps from the left topic margin. To be sure that text wraps correctly around an embedded window, insert a soft carriage return at the end of each line of text.

Example

The following example places an embedded window at the right margin. The subsequent paragraph wraps around the embedded window:

```
\par
\ewl FADE, AmfWnd, clipbrd.amf\
```

The map at the right shows the easiest route to the school.
Although many people use Highway 125, there are fewer stops
and less traffic if you use Ames Road.

Appendix B B-75

See Also ewc, ewl

\f
\fn

The \f statement sets the font. The new font applies to all subsequent text up to the next \plain or \f statement.

Parameters n
 Specifies the font number. This parameter must be one of the integer font numbers defined in the font table for the Help file.

Comments The \f statement does not set the point size of the font; use the \fs statement instead.

No \plain or \f statement is required if the \f statement and subsequent text are enclosed in braces. Braces limit the scope of a character property statement to just the enclosed text.

If the \f statement is not given, the default font is defined by the \deff statement (or is zero if no \deff statement is given).

Example The following example uses the Arial font to display text:

```
{\fonttbl {\f0\fswiss Arial;}}  
\par  
{\f0  
This text illustrates the Arial font.}  
\par
```

See Also \deff, \fonttbl, \fs, \plain

\fi
\fin

The \fi statement sets the first-line indent for the paragraph. The new indent applies to the first line of each subsequent paragraph up to the next \pard statement or \fi statement. The first-line indent is always relative to the current left indent.

Parameters

n

Specifies the indent, in twips. This parameter can be either a positive or negative number.

Microsoft Windows Help Authoring Guide

Comments

If the **\fi** statement is not given, the first-line indent is zero by default.

Example

The following example uses the first-line indent and a tab stop to make a numbered list:

```
\tx360\li360\fi-360
1
\tab
Insert the disk in drive A.
\par
2
\tab
Type a:setup and press the ENTER key.
\par
3
\tab
Follow the instructions on the screen.
\par \pard
```

See Also

\li, **\pard**

\field

\field

The **\field** statement defines a field.

Comments

The Microsoft Help compiler ignores this statement and all related field statements except the **\fldrslt** statement.

See Also

\fldrslt

\fldrslt

\fldrslt

The **\fldrslt** statement specifies the most recently calculated result of a field. The Microsoft Help compiler interprets the result as text and formats it using the current character and paragraph properties.

Comments

The Help compiler ignores all field statements except the **\fldrslt** statement. Any text associated with other field statements is ignored.

```
\fonttbl
\fonttbl {
\fn $\textit{family}$   $\textit{font-name}$ ;
.
.
.
}
```

The **\fonttbl** statement creates a font table for the Help file. The font table consists of one or more font definitions. Each definition consists of a font number, a font family, and a font name.

Parameters

- n*
Specifies the font number. This parameter must be an integer. This number can be used in subsequent **\f** statements to set the current font to the specified font. In the font table, font numbers should start at zero and increase by one for each new font definition.
- family*
Specifies the font family. This parameter must be one of the following.

Value	Meaning
fnil	Unknown or default fonts (default)
froman	Roman, proportionally spaced serif fonts (for example, MS Serif and Palatino)
fswiss	Swiss, proportionally spaced sans serif fonts (for example, Swiss)

fmodern Fixed-pitch serif and sans serif fonts (for example, Courier, Elite, and Pica)

fscript Script fonts (for example, Cursive)

fdecor Decorative fonts (for example, Old English and ITC Zapf Chancery)

ftech Technical, symbol, and mathematical fonts (for example, Symbol)

font-name

Specifies the name of the font. This parameter should specify an available Windows font.

Comments

If a font with the specified name is not available, Windows Help chooses a font from the specified family. If no font from the given family exists, Windows Help chooses a font having the same character set as specified for the Help file.

The **\deff** statement sets the default font number for the Help file. The default font is set whenever the **\pard** statement is given.

See Also

\deff, \f, \fs, \pard

\footnote

n{\footnote *text*}

The **\footnote** statement defines topic-specific information, such as the topic's build tags, context string, title, browse number, keywords, and execution macros. Every topic must, at least, have a context string to give the user access to the topic through links.

Specifies the footnote character. It can be one of the following.

Value	Meaning
-------	---------

*	Specifies a build tag. The Microsoft Help compiler uses build tags to determine whether it should include the topic in the Help file. The <i>text</i> parameter can be any combination of characters but must not contain spaces. Uppercase and lowercase characters are treated as equivalent characters (case insensitive). If a topic has build-tag statements, they must be the first statements in the topic. The Microsoft Help compiler checks a topic for build tags if the project file specifies a build expression using the BUILD option.
#	Specifies a context string. The <i>text</i> parameter can be any combination of letters and digits but must not contain spaces. Uppercase and lowercase characters are treated as equivalent characters (case insensitive). The context string can be used with the \v statement in other topics to create links to this topic.
\$	Specifies a topic title. Windows Help uses the topic title to identify the topic in the Search and History dialog boxes. The <i>text</i> parameter can be any combination of characters including spaces.
+	Specifies the browse-sequence identifier. Windows Help adds topics having an identifier to the browse sequence and allows users to view the topics by using the browse buttons. The <i>text</i> parameter can be a combination of letters and digits. Windows Help determines the order of topics in the browse sequence by sorting the identifier alphabetically. If two topics have the same identifier, Windows Help assumes that the topic that was compiled first is to be displayed first. Windows Help uses the browse sequence identifier only if the browse buttons have been enabled by using the BrowseButtons macro.
K	Specifies a keyword. Windows Help displays all keywords in the Help file in the Search dialog box and allows a user to choose a topic to view by choosing a keyword. The <i>text</i> parameter can be any combination of characters including spaces. If the first character is the letter K; it must be preceded with an extra space or a semicolon. More than one keyword can be given by separating the keywords with semicolons (;). A topic cannot contain keywords unless it also has a

topic title.

Microsoft Windows Help Authoring Guide

! Specifies a Help macro. Windows Help executes the macro when the topic is displayed. The *text* parameter can be any Help macro.

@ Specifies an author-defined comment about the Help topic. This footnote is provided for authoring purposes only. The Help compiler ignores this footnote when building the Help file. The *text* parameter can be any combination of letters and digits, including accented characters and spaces.

If *n* is any letter (other than K), the footnote specifies an alternative keyword. Windows applications can search for topics having alternative keywords by using the **HELP_MULTIKY** command with the **WinHelp** function.

text

Specifies the build tag, context string, topic title, browse-sequence number, keyword, or macro associated with the footnote. This parameter depends on the footnote type as specified by the *n* parameter.

Comments

A topic can have more than one build-tag, context-string, keyword, and help-macro statement, but must not have more than one topic-title or browse-sequence-number statement.

In print-based documents, the **\footnote** statement creates a footnote and the footnote is anchored to the character immediately preceding the **\footnote** statement.

Example

The following example defines a topic titled “Short Topic.” The context string “topic1” can be used to create links to the topic. The keywords “example topic” and “short topic” appear in the Search dialog box and can be used to choose the topic for viewing. Or the user can find the topic by browsing since the topic is included in the short browse sequence. The topic also includes an entry macro that starts the Clock application when Help displays the topic:

```
 ${\footnote Short Topic}
#{\footnote topic1}
K{\footnote example topic;short topic}
+{\footnote short:015}
!{\footnote ExecProgram('clock.exe', 0)}
This topic has a title, context string, and two keywords.
```

	<code>\par</code> <code>\page</code>	
See Also	<code>\chftn</code> , <code>\v</code>	Appendix B B-81

<code>\fs</code> <code>\fsn</code>	<p>The <code>\fs</code> statement sets the size of the font. The new font size applies to all subsequent text up to the next <code>\plain</code> or <code>\fs</code> statement.</p>	
Parameters	<i>n</i>	Specifies the size of the font, in half points.
Comments	<p>The <code>\fs</code> statement does not set the font face; use the <code>\f</code> statement instead.</p> <p>No <code>\plain</code> or <code>\fs</code> statement is required if the <code>\fs</code> statement and subsequent text are enclosed in braces. Braces limit the scope of a character property statement to just the enclosed text.</p> <p>If the <code>\fs</code> statement is not given, the default font size is 24.</p>	
Example	<p>The following example sets the size of the font to 10 points:</p> <pre>{\fs20 This line is in 10 point type.} \par</pre>	
See Also	<code>\f</code> , <code>\plain</code>	

<code>\'</code> <code>\'hh</code>	<p>The <code>\'</code> statement converts the specified hexadecimal number into a character value and inserts the value into the Help file. The appearance of the character when displayed depends on the character set specified for the Help file.</p>	
Parameters	<i>hh</i>	Specifies a two-digit hexadecimal value.
Comments	<p>Since the Microsoft Help compiler does not accept character values greater than 127, the <code>\'</code> statement is the only method to insert such character values into the Help file.</p>	

Example	The following example inserts a trademark in a Help file that uses the \windows statement to set the character set:
Microsoft Windows Help Authoring Guide	ABC\99 is a trademark of the ABC Product Corporation.

See Also	\ansi, \pc, \pca, \windows
----------	-----------------------------------

\i	
\i	
	The \i statement starts italic text. The statement applies to all subsequent text up to the next \plain or \i0 statement.
Comments	No \plain or \i0 statement is required if the \i statement and subsequent text are enclosed in braces. Braces limit the scope of a character property statement to just the enclosed text.
Example	The following example sets “not” to italic: You must {i not} save the file without first setting the Auto option.
See Also	\b, \plain, \scaps

\intbl	
\intbl	
	The \intbl statement marks subsequent paragraphs as part of a table. The statement applies to all subsequent paragraphs up to the next \row statement.
Comments	This statement was first supported in Microsoft Help compiler version 3.1.
Example	The following example creates a three-column table having two rows: \cellx1440\cellx2880\cellx4320 \intbl Row 1 Column 1\cell Row 1 Column 2\cell Row 1 Column 3\cell \row \intbl Row 2 Column 1\cell Row 2 Column 2\cell Row 2 Column 3\cell \row \pard
See Also	\cell, \cellx, \row, \trgraph, \trleft, \trowd

`\keep`

`\keep`

The **\keep** statement prevents Windows Help from wrapping text to fit the Help window. The statement applies to all subsequent paragraphs up to the next **\pard** statement.

Comments

If the text in a paragraph exceeds the width of the Help window, Help displays a horizontal scroll bar.

In print-based documents, the **\keep** statement keeps paragraphs intact.

Example

The following example makes the paragraph beginning “Cardfile has two views–Card and List” nonwrapping text:

```
\par\pard\keep
Cardfile has two views--Card and List.
```

See Also

\line

`\keepn`

`\keepn`

The **\keepn** statement creates a nonscrolling region at the top of the Help window for the given topic. The **\keepn** statement applies to all subsequent paragraphs up to the next **\pard** statement. All paragraphs with this paragraph property are placed in the nonscrolling region.

Comments

If a **\keepn** statement is used in a topic, it must be applied to the first paragraph in the topic (and subsequent paragraphs as needed). The Help compiler displays an error message and does not create a nonscrolling region if paragraphs are given before the **\keepn** statement. Only one nonscrolling region per topic is allowed.

Windows Help formats, aligns, and wraps text in the nonscrolling region just as it does in the rest of the topic. It separates the nonscrolling region from the rest of the Help window with a horizontal bar. Windows Help sets the height of the nonscrolling region so that all paragraphs in the region can be viewed if the Help window is large enough. If the window is smaller than the nonscrolling region, the user will be unable to view the rest of the topic. For this reason, the nonscrolling region is typically reserved for a single line of text specifying the name or title of the topic.

In print-based documents, the **\keepn** statement keeps the subsequent paragraph

with the paragraph that follows it.

Microsoft Windows Help Authoring Guide

The following example places the first paragraph of the topic beginning “Cardfile has two views—Card and List” into a nonscrolling region:

```
\par\pard\keepn
Cardfile has two views--Card and List.
\par\pard
```

See Also

\page

\li
\lin

The **\li** statement sets the left indent for the paragraph. The indent applies to all subsequent paragraphs up to the next **\pard** or **\li** statement.

Parameters

n
Specifies the indent, in twips. The value can be either positive or negative.

Comments

If the **\li** statement is not given, the left indent is zero by default. Windows Help automatically provides a small left margin so that if no indent is specified the text does not start immediately at the left edge of the Help window.

Specifying a negative left indent moves the starting point for a line of text to the left of the default left margin. If the negative indent is large enough, the start of the text may be clipped by the left edge of the Help window.

Example

The following example uses the left indent and a tab stop to make a bulleted list. In this example, font number 0 is assumed to be the Symbol font:

```
Use the Auto command to:
\par
\tx360\li360\fi-360
{\f0\B7}
\tab
Save files automatically
\par
{\f0\B7}
\tab
Prevent overwriting existing files
\par
{\f0\B7}
\tab
Create automatic backup files
\par \pard
```

See Also

\fi, \pard, \ri

Appendix B B-85

\line

\line

The **\line** statement breaks the current line without ending the paragraph. Subsequent text starts on the next line and is aligned and indented according to the current paragraph properties.

See Also

\par

\mac

\mac

The **\mac** statement sets the Apple Macintosh character set.

See Also

\windows

\page

\page

The **\page** statement marks the end of a topic.

Comments

In a print-based document, the **\page** statement creates a page break.

Example

The following example shows a complete topic:

```
$(\footnote Short Topic)
#(\footnote short_topic)
Most topics in a topic file consist of topic-title and
context-string statements followed by the topic text. Every
topic ends with a {\b \page} statement.
\par
\page
```

See Also

\par

\par

<hr/>	
\par	
Microsoft Windows Help Authoring Guide	The \par statement marks the end of a paragraph. The statement ends the current line of text and moves the current position to the left margin and down by the current line-spacing and space-after-paragraph values.
Comments	The first line of text after a \par , \page , or \sect statement marks the start of a paragraph. When a paragraph starts, the current position is moved down by the current space-before-paragraph value. Subsequent text is formatted using the current text alignment, line spacing, and left, right, and first-line indents.
Example	<p>The following example has three paragraphs:</p> <pre> \ql This paragraph is left-aligned. \par \pard \qc This paragraph is centered. \par \pard \qr This paragraph is right-aligned. \par </pre>
See Also	\line, \pard, \sect

<hr/>							
\pard \pard							
The \pard statement restores all paragraph properties to default values.							
Comments	<p>If the \pard statement appears anywhere before the end of a paragraph (that is, before the \par statement), the default properties apply to the entire paragraph.</p> <p>The default paragraph properties are as follows.</p>						
<table> <tr> <th>Property</th><th>Default</th></tr> <tr> <td>Alignment</td><td>Left-aligned</td></tr> <tr> <td>First-line indent</td><td>0</td></tr> </table>		Property	Default	Alignment	Left-aligned	First-line indent	0
Property	Default						
Alignment	Left-aligned						
First-line indent	0						
<hr/>							

Right indent	0
--------------	---

Space before	0
--------------	---

Space after	0
-------------	---

Line spacing	Tallest character
--------------	-------------------

Tab stops	None
-----------	------

Borders	None
---------	------

Border style	Single-width
--------------	--------------

See Also	<code>\par</code>
----------	-------------------

`\pc`

\pc

Microsoft Windows Help Authoring Guide The \pc statement sets the OEM character set (also known as code page 437).

See Also

\windows

\pca

\pca

The \pca statement sets the International English character set (also known as code page 850).

See Also

\windows

\pich

\pichn

The \pich statement specifies the height of the picture. This statement must be used in conjunction with a \pict statement.

Parameters

n

Specifies the height of the picture, in twips or pixels, depending on the picture type. If the picture is a metafile, the width is in twips; otherwise, the width is in pixels.

See Also

\pict, \picw

\pichgoal

\pichgoaln

The \pichgoal statement specifies the desired height of a picture. If necessary, Windows Help stretches or compresses the picture to match the requested height. This statement must be used in conjunction with a \pict statement.

Parameters

n

Specifies the desired height, in twips.

See Also

\pict, \picwgoal

`\picscalex`

Appendix B B-89

`\picscalexn`

The `\picscalex` statement specifies the horizontal scaling value. This statement must be used in conjunction with a `\pict` statement.

Parameters

n

Specifies the scaling value. The parameter must be a value in the range 0 through 100, representing a percentage.

Comments

If the `\picscalex` statement is not given, the default scaling value is 100.

See Also

`\picscaley`, `\pict`

`\picscaley`

`\picscaleyn`

The `\picscaley` statement specifies the vertical scaling value. This statement must be used in conjunction with a `\pict` statement.

Parameters

n

Specifies the scaling value. The parameter must be an integer in the range 0 through 100, representing a percentage.

Comments

If the `\picscaley` statement is not given, the default scaling value is 100.

See Also

`\picscalex`, `\pict`

`\pict`

`\pict` *picture-statements picture-data*

The `\pict` statement creates a picture. A picture consists of hexadecimal or binary data representing a bitmap or metafile.

Parameters

picture-statements

Specifies one or more statements defining the type of picture, the dimensions of the picture, and the format of the picture data. It can be a combination of the following statements:

Microsoft Windows Help Authoring Guide	Statement	Description
	\wbitmap	Specifies a Windows bitmap.
	\wmetafile	Specifies a Windows metafile.
	\picw	Specifies the picture width.
	\pich	Specifies the picture height.
	\picwgoal	Specifies the desired picture width.
	\pichgoal	Specifies the desired picture height.
	\picscalex	Specifies the horizontal scaling value.
	\picscaley	Specifies the vertical scaling value.
	\wbmbitspixel	Specifies the number of bits per pixel.

\wbmwidthbytes

Specifies the bitmap width, in bytes.

\bin

Specifies binary picture data.

picture-data

Specifies hexadecimal or binary data representing the picture. The picture data follows the last picture statement.

Comments

If a data format is not specified, the default format is hexadecimal.

See Also

\bin, \pich, \pichgoal, \picscalex, \picscaley, \picw, \picwgoal, \wbimap, \wbmbitspixel, \wbmplanes, \wbmwidthbytes, \wmetafile

\picw

\picwn

The \picw statement specifies the width of the picture. This statement must be used in conjunction with a \pict statement.

Parameters

n

Specifies the width of the picture, in twips or pixels, depending on the picture type. If the picture is a metafile, the width is in twips; otherwise, the width is in pixels.

See Also

\pich, \pict

\picwgoal

\picwgoal*n*

Microsoft Windows Help Authoring Guide The **\picwgoal** statement specifies the desired width of the picture, in twips. If necessary, Windows Help stretches or compresses the picture to match the requested height. This statement must be used in conjunction with a **\pict** statement.

Parameters *n*
Specifies the desired width, in twips.

See Also **\pichgoal**, **\pict**

\plain
\plain

The **\plain** statement restores the character properties to default values.

Comments The default character properties are as follows.

Property	Default
Bold	Off
Italic	Off
Small caps	Off
Font	0
Font size	24

See Also **\b, \f, \fs, \i, \scaps**

\qc
\qc

The **\qc** statement centers text between the current left and right indents. The statement applies to subsequent paragraphs up to the next **\pard** statement or text-alignment statement.

Comments If a **\ql**, **\qr**, **\qc**, or **\qj** statement is not given, the text is left-aligned by default.

See Also **\qj, \ql, \qr, \pard**

\qj
\qj

The **\qj** statement justifies text between the current left and right indents. The statement applies to subsequent paragraphs up to the next **\pard** statement or text-alignment statement.

Comments If a **\ql**, **\qr**, **\qc**, or **\qj** statement is not given, the text is left-aligned by default.

See Also **\qc, \ql, \qr, \pard**

\ql
\ql

The **\ql** statement aligns text along the left indent. The statement applies to subsequent paragraphs up to the next **\pard** statement or text-alignment statement.

Comments If a **\ql**, **\qr**, **\qc**, or **\qj** statement is not given, the text is left-aligned by default.

See Also **\qc, \qj, \qr, \pard**

<div><div><div>\qr</div><div>\qr</div></div><div>Microsoft Windows Help Authoring Guide</div></div>	
	The \qr statement aligns text along the right indent. The statement applies to subsequent paragraphs up to the next \pard statement or text-alignment statement.
Comments	If a \ql , \qr , \qc , or \qj statement is not given, the text is left-aligned by default.
See Also	\qc , \qj , \ql , \pard

<div><div><div>\ri</div><div>\rin</div></div></div>	
	The \ri statement sets the right indent for the paragraph. The indent applies to all subsequent paragraphs up to the next \pard or \ri statement.
Parameters	<div><div><i>n</i></div><div>Specifies the right indent, in twips. It can be a positive or negative value.</div></div>
Comments	<div>If the \ri statement is not given, the right indent is zero by default. Windows Help automatically provides a small right margin so that when no right indent is specified, the text does not end abruptly at the right edge of the Help window.</div> <div>Windows Help never displays less than one word for each line in a paragraph even if the right indent is greater than the width of the window.</div>
Example	<div>In the following example, the right and left indents are set to one inch and the subsequent text is centered between the indents:</div> <div><pre>\li1440\ri1440\qc Microsoft Windows Help\line Sample File\line</pre></div>
See Also	\li , \pard

<div><div><div>\row</div><div>\row</div></div></div>	
	The \row statement marks the end of a table row. The statement ends the current row and begins a new row by moving down past the end of the longest cell in the row. The next \cell statement specifies the text of the leftmost cell in the next row.

Comments	This statement was first supported in the Microsoft Help compiler version 3.1.
----------	--

Example	The following example creates a table having four rows and two columns:
---------	---

```
\cellx2880\cellx5760
\intbl
Row 1, Column 1\cell
Row 1, Column 2\cell \row
\intbl
Row 2, Column 1\cell
Row 2, Column 2\cell \row
\intbl
Row 3, Column 1\cell
Row 3, Column 2\cell \row
\intbl
Row 4, Column 1\cell
Row 4, Column 2\cell \row
\par \pard
```

See Also	\cell, \cellx, \intbl
----------	-----------------------

\rtf
\rtfn

The **\rtf** statement identifies the file as a rich-text format (RTF) file and specifies the version of the RTF standard used.

Parameters	<i>n</i> Specifies the version of the RTF standard used. For the Microsoft Help compiler version 3.1, this parameter must be 1.
------------	--

Comments	The \rtf statement must follow the first open brace in the Help file. A statement specifying the character set for the file must also follow the \rtf statement.
----------	--

See Also	\windows
----------	----------

\sa
\san

The **\sa** statement sets the amount of vertical spacing after a paragraph. The vertical space applies to all subsequent paragraphs up to the next **\pard** or **\sa** statement.

Parameters	<i>n</i> Specifies the amount of vertical spacing, in twips.
------------	---

Comments	If the \sa statement is not given, the vertical spacing after a paragraph is zero by default.
See Also	\pard, \sb

\sb \sbn	
	The \sb statement sets the amount of vertical spacing before the paragraph. The vertical space applies to all subsequent paragraphs up to the next \pard or \sb statement.
Parameters	<i>n</i> Specifies the amount of vertical spacing, in twips.
Comments	If the \sb statement is not given, the vertical spacing before the paragraph is zero by default.
See Also	\pard, \sa

\scaps \scaps	
	The \scaps statement starts small-capital text. The statement converts all subsequent lowercase letters to uppercase before displaying the text. This statement applies to all subsequent text up to the next \plain or \scaps0 statement.
Comments	No \plain or \scaps0 statement is required if the \scaps statement and subsequent text are enclosed in braces. Braces limit the scope of a character property statement to just the enclosed text. The \scaps statement does not reduce the point size of the text. To reduce point size, the \fs statement must be used.
Example	The following example displays the key name ENTER in small capitals: Press the {\scaps ENTER} key to complete the action.
See Also	\plain

\sect
\sect

The \sect statement marks the end of a section and paragraph.

See Also

\par

\sl
\sln

The \sl statement sets the amount of vertical space between lines in a paragraph. The vertical space applies to all subsequent paragraphs up to the next \pard or \sl statement.

Parameters

n

Specifies the amount of vertical spacing, in twips. If this parameter is a positive value, Windows Help uses this value if it is greater than the tallest character. Otherwise, Windows Help uses the height of the tallest character as the line spacing. If this parameter is a negative value, Windows Help uses the absolute value of the number even if the tallest character is taller.

Comments

If the \sl statement is not given or the specified amount of spacing is 1000, Windows Help automatically sets the line spacing by using the tallest character in the line.

See Also

\pard

\strike
\strike

The \strike statement creates a hot spot. The statement is used in conjunction with a \v statement to create a link to another topic. When the user chooses a hot spot, Windows Help displays the associated topic in the Help window.

The \strike statement applies to all subsequent text up to the next \plain or \strike0 statement.

Comments

No \plain or \strike0 statement is required if the \strike statement and subsequent text are enclosed in braces. Braces limit the scope of a character property

statement to just the enclosed text.

Microsoft Windows Help Authoring Guide In print-based documents, or whenever it is not followed by \v, the \strike statement creates strikeout text.

Example The following example creates a hot spot for a topic. When displayed, the hot-spot text, “Hot Spot,” is green and has a solid line under it:
`{\strike Hot Spot}{\v Topic}`

See Also \ul, \uldb, \v

\tab
\tab

The \tab statement inserts a tab character (ASCII character code 9).

Comments The tab character (ASCII character 9) has the same effect as the \tab statement.

See Also \tb, \tqc, \tqr, \tx

\tb
\tb

The \tb statement advances to the next tab stop.

See Also \tab, \tqc, \tqr, \tx

\tqc
\tqc

The \tqc statement advances to the next tab stop and centers text.

See Also \tab, \tb, \tqr, \tx

\tqr
\tqr

The **\tqr** statement advances to the next tab stop and aligns text to the right.

See Also

\tab, \tb, \tqc, \tx

Appendix B B-99

\trgaph
\trgaph*n*

The **\trgaph** statement specifies the amount of space between text in adjacent cells in a table. For each cell in the table, Windows Help uses the space to calculate the cell's left and right margins. It then uses the margins to align and wrap the text in the cell. Windows Help applies the same margin widths to each cell ensuring that paragraphs in adjacent cells have the specified space between them.

The **\trgaph** statement applies to cells in all subsequent rows of a table up to the next **\trowd** statement.

Parameters

n
Specifies the space, in twips, between text in adjacent cells. If this parameter exceeds the actual width of the cell, the left and right margins are assumed to be at the same position in the cell.

Comments

The width of the left margin in the first cell is always equal to the space specified by this statement. The **\trleft** statement is typically used to move the left margin to a position similar to the left margins in all other cells.

This statement was first supported in the Microsoft Help compiler version 3.1.

Example

The following example creates a three-column table with one-quarter inch space between the text in the columns:

```
\trgaph360 \cellx1440\cellx2880\cellx4320
\intbl
Row 1 Column 1\cell
Row 1 Column 2\cell
Row 1 Column 3\cell \row
\intbl
Row 2 Column 1\cell
Row 2 Column 2\cell
Row 2 Column 3\cell \row \pard
```

See Also

\cell, \cellx, \intbl, \row, \trleft, \trowd

\trleftn

The **\trleft** statement sets the position of the left margin for the first (leftmost) cell in a row of a table. This statement applies to the first cell in all subsequent rows of the table up to the next **\trrowd** statement.

 n

Specifies the relative position, in twips, of the left margin. This parameter can be a positive or negative number. The final position of the left margin is the sum of the current position and this value.

This statement was first supported in the Microsoft Help compiler version 3.1.

The following example creates a three-column table with one-quarter inch space between the text in the columns. The left margin in the first cell is flush with the left margin of the Help window:

```
\trigaph360\trleft-360 \cellx1440\cellx2880\cellx4320
\intbl
Row 1 Column 1\cell
Row 1 Column 2\cell
Row 1 Column 3\cell \row
\intbl
Row 2 Column 1\cell
Row 2 Column 2\cell
Row 2 Column 3\cell \row \bard
```

\cell, \cellx, \intbl, \row, \trgaph, \trowd

\trawd

The **\trrowd** statement sets default margins and cell positions for subsequent rows in a table.

This statement was first supported in the Microsoft Help compiler version 3.1.

\cell, \cellx, \intbl, \row, \trgaph, \trleft

\trqc

The **\trqc** statement directs Windows Help to dynamically adjust the width of table columns to fit in the current window.

Appendix B B-101

Comments	In a print-based document, the \trqc statement centers a table row with respect to its containing column.
	This statement was first supported in the Microsoft Help compiler version 3.1.
See Also	\trowd, \trql

\trql
\trql

The **\trql** statement aligns the text in each cell of a table row to the left.

Comments	This statement was first supported in the Microsoft Help compiler version 3.1.
See Also	\trowd, \trqc

\tx
\txn

The **\tx** statement sets the position of a tab stop. The position is relative to the left margin of the Help window. A tab stop applies to all subsequent paragraphs up to the next **\pard** statement.

Parameters	<i>n</i> Specifies the tab stop position, in twips.
Comments	If the \tx statement is not given, tab stops are set at every one-half inch by default.
See Also	\tab, \tb, \tqc, \tqr

\ul
\ul

The **\ul** statement creates a link to a pop-up topic. The statement is used in conjunction with a **\v** statement to create a link to another topic. When the user chooses the link, Windows Help displays the associated topic in a pop-up

window.

Microsoft Windows Help Authoring Guide The **\ul** statement applies to all subsequent text up to the next **\plain** or **\ul0** statement.

Comments	No \plain or \ul0 statement is required if the \ul statement and subsequent text are enclosed in braces. Braces limit the scope of a character property statement to just the enclosed text. In print-based documents, or whenever it is not followed by \v , the \ul statement creates continuous underline.
Example	The following example creates a pop-up link for a topic. When displayed, the link text, “Pop-up Link,” is green and has a dotted line under it: <code>{\ul Pop-up Link}{\v PopupTopic}</code>
See Also	\strike , \uldb , \v

\uldb
\uldb

The **\uldb** statement creates a hot spot. This statement is used in conjunction with a **\v** statement to create a link to another topic. When the user chooses a hot spot, Windows Help displays the associated topic in the Help window.

The **\uldb** statement applies to all subsequent text up to the next **\plain** or **\uldb0** statement.

Comments	No \plain or \uldb0 statement is required if the \uldb statement and subsequent text are enclosed in braces. Braces limit the scope of a character property statement to just the enclosed text. In print-based documents, or whenever it is not followed by \v , the \uldb statement creates double underline.
Example	The following example creates a hot spot for a topic. When displayed, the hot-spot text, “Hot Spot,” is green and has a solid line under it: <code>{\uldb Hot Spot}{\v Topic}</code>
See Also	\strike , \ul , \v

`\v`
`{\v context-string}`

The `\v` statement creates a link to the topic having the specified context string. The `\v` statement is used in conjunction with the `\strike`, `\ul`, and `\uldb` statements to create hot spots and links to topics.

Parameters

context-string

Specifies the context string of a topic in the Help file. The string can be any combination of characters, except spaces, and must also be specified in a context-string **\footnote** statement in some topic in the Help file.

Comments

If the context string is preceded by a percent sign (%), Windows Help displays the associated hot spot or link without applying the standard underline and color.

If the context string is preceded by an asterisk (*), Windows Help displays the associated hot spot or link with an underline but without applying the standard color.

In print-based documents, the `\v` statement creates hidden text.

For links or hot spots, the syntax of the `\v` statement is as follows:

```
[%|*]context[>secondary-window][@filename]
```

In this syntax, *secondary-window* is the name of the secondary window to jump to. When the secondary window is not specified, the jump is to the same window as the current Help topic is using. To jump to the main Help window, specify “main” for this parameter. This parameter may not be used with pop-up windows.

The *filename* parameter specifies a jump to a topic in a different Help file.

For a macro hot spot, the syntax of the `\v` statement is as follows:

```
[%|*]!macro[;macro][;...]
```

Example

The following example creates a hot spot for the topic having the context string “Topic.” Windows Help applies an underline and the color green to the text “Hot Spot” when it displays the topic:

```
{\uldb Hot Spot}{\v Topic}
```

See Also

\footnote, **\strike**, **\ul**, **\uldb**

`\wbimap` `\wbimap`*n*

The **\wbimap** statement sets the picture type to Windows bitmap. This statement must be used in conjunction with a **\pict** statement.

Parameters

n

Specifies the bitmap type. This parameter is zero for a logical bitmap.

Comments

The **\wbimap** statement is optional; if a **\wmetafile** statement is not specified, the picture is assumed to be a Windows bitmap.

Example

The following example creates a 32-by-8-pixel monochrome bitmap:

```
{  
\pict \wbimap0\wbmbitspixel1\wbmplanes1\wbmwidthbytes4\picw32\pich8  
3FFFFFFC  
F3FFFFFF  
FF3FFCFF  
FFF3CFFF  
FFFC3FFF  
FFCFF3FF  
FCFFFF3F  
CFFFFFFF3  
}
```

See Also

bmc, **bml**, **bmr**, **\pict**, **\wmetafile**

`\wbmbitspixel` `\wbmbitspixel`*n*

The **\wbmbitspixel** statement specifies the number of consecutive bits in the bitmap data that represent a single pixel. This statement must be used in conjunction with the **\pict** statement.

Parameters

n

Specifies the number of bits per pixel.

Comments

If the **\wbmbitspixel** statement is not given, the default bits per pixel value is 1.

See Also

\pict, **\wbimap**, **\wbmplanes**

`\wbmplanes`

`\wbmplanes`*n*

The **\wbmplanes** statement specifies the number of color planes in the bitmap data. This statement must be used in conjunction with a **\pict** statement.

Parameters

n

Specifies the number of bitmap planes.

Comments

If the **\wbmplanes** statement is not given, the default number of planes is 1.

See Also

\pict, **\wbitmap**, **\wbmbitspixel**

`\wbmwidthbytes`

`\wbmwidthbytes`*n*

The **\wbmwidthbytes** statement specifies the number of bytes in each scan line of the bitmap data. This statement must be used in conjunction with the **\pict** statement.

Parameters

n

Specifies the width of the bitmap, in bytes.

See Also

\pict, **\wbitmap**

`\windows`

`\windows`

The **\windows** statement sets the Windows character set.

Comments

If no **\windows**, **\pc**, or **\pca** statement is given in the Help file, the Windows character set is used by default.

See Also

\ansi, **\pc**, **\pca**

`\wmetafile`

`\wmetafile`*n*

The **\wmetafile** statement sets the picture type to a Windows metafile. This statement must be used in conjunction with the **\pict** statement.

<hr/>	
Parameters	<i>n</i>
Microsoft Windows Help Authoring Guide	Specifies the metafile type. This parameter must be 8.
Comments	Windows Help expects the hexadecimal data associated with the picture to represent a valid Windows metafile. By default, Windows Help sets the MM_ANISOTROPIC mapping mode prior to displaying the metafile. To ensure that the picture is displayed correctly, the metafile data must either set the window origin and extents by using the SetWindowOrg and SetWindowExt records or set another mapping mode by using the SetMapMode record.
Example	<p>The following example creates a picture using a metafile:</p> <pre> {{\pict\wmetafile8\picw2880\pich2880 0100090000034f0000000200090000000000 050000000b0200000000050000000c026400 6400090000001d066200ff00640064000000 000008000000fa0200000200000000000000 040000002d01000005000000140200000000 050000001302640064000500000014020000 64000500000013026400000008000000fa02 00000000000000000000040000002d010100 04000000f00100000300000000004e0dff00 8700200000500000200000000000000000} \par } </pre>
See Also	bmc , bml , bmr , \pict , \wbitemap
© 1993 Microsoft Corporation, All rights reserved	

