1 Introduction

The file format describes the document file for the program "HelpReader". The file extension is called ".tdhr" (Two Dimension Help Reader).

2 Value types

Type	Description	Area
INT8	8Bit with sign	-128 to 127
INT16	16Bit with sign	-32.768 to 32.767
INT32	32Bit with sign	-2.147.483.648 to 2.147.483.647
INT64	64Bit with sign	-9.223.372.036.854.775.808 to 9.223.372.036.854.775.807
BYTE	8Bit unsigned	0 to 255
UINT16	16Bit unsigned	0 to 65.535
UINT32	32Bit unsigned	0 to 4.294.967.295
UINT64	64Bit unsigned	0 to 18.446.744.073.709.551.615
CHAR	8Bit character	0 to 255
WCHAR	16Bit character	0 to 65.535
FLOAT	32Bit floating point	± 1.5e-45 to ± 3.4e38
DOUBLE	64Bit floating point	± 5.0e-324 to ± 1.7e308
MEMORY	Memory in bytes	
...[]	Array	see section 2.1
-> {	Start of the loop	see section 2.2
} <-	End of the loop	see section 2.2
...	Next table	see section 2.3
...!	Table with condition	see section 2.4
!	Dependence	see section 2.5

Table 2: Value types

2.1 Array

The set consists of a specific value type. The count of the set is detailed in the information and is usually the previous format value.

Example:

```
A Array INT16[] contains a certain count of INT16 values { INT16, INT16, INT16,
INT16, ... }.

INT16[], BYTE[], UINT32[], WCHAR[], usw.
```

2.2 The loop

In a loop, the format is repeatedly run through. The count of run through is specified in detail in the information and is usually the previous value.

2.3 Next table

The file format is displayed further in the section and the table specified.

2.4 Next table with condition

The file format continues in the section and the table if the condition is met.

2.5 Dependence

The value exists only in the file when a specific condition is met.

3 Description

3.1 File format

Type	Name	Description	Info
UINT32	IDNumber	The file must have the ID number (0x52484454).	4.1
INT32	FileCount	The number of document files.	4.2
INT64	FileSize	The size of a single file in bytes.	4.3
INT64	FileMaxSize	The size of the entire document in bytes.	4.4
BYTE	Version	The version number is 1 in this file format.	4.5
UINT16	Flags	The bit values for the file format.	4.6
...!	Flags = 0x0001	3.2 Preview format, Table 3.2	
...!	Flags = 0x0002	3.3 Symbol format, Table 3.3	
...!	Flags = 0x0004	3.4 Information format, Table 3.4	
UINT32	SettingBackColor	The background color for the document.	4.7
UINT32	SettingWindowColor	The window color for the document.	4.8
INT32[]	SettingOutline	The distances to the outline of structure.	4.9
INT32[]	SettingPage	The distances to the page content.	4.10
INT32	LanguageCount	The number of languages used.	4.11
-> {	Language		
BYTE	LanguageLetterLength	The number of characters of the language abbreviation.	4.12
WCHAR[]	LanguageLetter	The abbreviation of the language with two letters.	4.13
} <-	Language		
...		3.5 Text format, Table 3.5	
...		3.8 Field image format, Table 3.8	
...		3.9 Display image format, Table 3.9	
...		3.10 Play file format, Table 3.10	
...		3.11 Extended image format, Table 3.11	
...		3.12 Outline format, Table 3.12	

Table 3.1: File format

3.2 Preview format

Type	Name	Description	Info
BYTE	PreviewImageMode	The color content for the preview image.	5.1
INT32	PreviewImageWidth	The width of the preview image.	5.2
INT32	PreviewImageHeight	The height of the preview image.	5.3
INT32	PreviewImageSize	The size of the memory with the preview image in bytes.	5.4
MEMORY	PreviewImage	The memory with the preview image file.	5.5
...		3.1 File format, Table 3.1	

Table 3.2: Preview format

3.3 Symbol format

Type	Name	Description	Info
BYTE	IconImageMode	The color content of the document symbol.	6.1
INT32	IconImageWidth	The width of the symbol.	6.2
INT32	IconImageHeight	The height of the symbol.	6.3
INT32	IconImageSize	The memory size of the symbol file.	6.4
MEMORY	IconImage	The symbol file for the document.	6.5
...		3.1 File format, Table 3.1	

Table 3.3: Symbol format

3.4 Information format

Type	Name	Description	Info
INT32	InfoProducerLength	The number of characters of the producer name.	7.1
WCHAR[]	InfoProducer	The name of the producer or a company.	7.2
INT32	InfoInternetLength	The number of characters for an internet address.	7.3
WCHAR[]	InfoInternet	The name of the internet address.	7.4
INT32	InfoMailLength	The number of characters for an e-mail address.	7.5
WCHAR[]	InfoMail	The name of the e-mail address.	7.6
...		3.1 File format, Table 3.1	

Table 3.4: Information format

3.5 Text format

Type	Name	Description	Info
INT32	LetterImageSize	The memory size of the image file.	8.1
MEMORY	LetterImage	The image file for the characters.	8.2
BYTE	LetterUseInfo	The use of extra information. Value = 0	8.3
...		3.6 Font format, Table 3.6	
...		3.1 File format, Table 3.1	

Table 3.5: Text format

3.6 Font format

Type	Name	Description	Info
INT32	FontCount	The number of all used fonts.	9.1
-> {	Font		
INT16	FontHeight	The absolute height of the characters.	9.2
INT16	FontAscent	The base height of the characters.	9.3
...		3.7 Letter format, Table 3.7	
} <-	Font		
...		3.5 Text format, Table 3.5	

Table 3.6: Font format

3.7 Letter format

Type	Name	Description	Info
INT32	LetterCount	The number of all used characters of a font.	10.1
-> {	Letter		
INT16[]	LetterWidth	The widths to represent a character.	10.2
INT32	LetterImagePosition	The memory location for the drawing picture.	10.3
INT16!	LetterImageWidth	The width of the character image.	10.4
INT16!	LetterImageHeight	The height of the character image.	10.5
<- {	Letter		
...		3.6 Font format, Table 3.6	

Table 3.7: Letter format

3.8 Field image format

Type	Name	Description	Info
INT32	FieldImageCount	The number of field images.	11.1
-> {	FieldImage		
BYTE	FieldImageMode	The color content of the images.	11.2
INT32	FieldImageWidth	The width of the image.	11.3
INT32	FieldImageHeight	The height of the image.	11.4
INT32	FieldImageSize	The size of the image file.	11.5
MEMORY	FieldImage	The image file for the page field.	11.6
<- {	FieldImage		
...		3.1 File format, Table 3.1	

Table 3.8: Field image format

3.9 Display image format

Type	Name	Description	Info
INT32	DisplayImageCount	The number of display images for the field content.	11.1
-> {	DisplayImage		
BYTE	DisplayImageMode	The color content of the images.	11.2
INT32	DisplayImageWidth	The width of the image.	11.3
INT32	DisplayImageHeight	The height of the image.	11.4
INT32	DisplayImageSize	The size of the image file.	11.5
MEMORY	DisplayImage	The display image file within the page field.	11.6
<- {	DisplayImage		
...		3.1 File format, Table 3.1	

Table 3.9: Display image format

3.10 Play file format

Type	Name	Description	Info
INT32	PlayerFileCount	The number of playable files.	12.1
-> {	PlayerFile		
INT32	PlayerFileType	The file type for playing.	12.2
INT64	PlayerDuration	The playing time of the medium.	12.3
INT32	PlayerDisplayWidth	The presentation width of the medium.	12.4
INT32	PlayerDisplayHeight	The presentation height of the medium.	12.5
INT32	PlayerFileSize	The memory size for the play file.	12.6
MEMORY	PlayerFileMemory	The memory with the play file.	12.7
<- {	PlayerFile		
...		3.1 File format, Table 3.1	

Table 3.10: Play file format

3.11 Extended image format

Type	Name	Description	Info
INT32	ExpandedImageCount	The number of images for the outline extension.	11.1
-> {	ExpandedImage		
BYTE	ExpandedImageMode	The color content of the images.	11.2
INT32	ExpandedImageWidth	The width of the image.	11.3
INT32	ExpandedImageHeight	The height of the image.	11.4
INT32	ExpandedImageSize	The size of the image file.	11.5
MEMORY	ExpandedImage	The memory with the image file for the outline.	11.6
<- {	ExpandedImage		
...		3.1 File format, Table 3.1	

Table 3.11: Extended image format

3.12 Outline format

Type	Name	Description	Info
INT32	OutlineCount	The number of outlines.	13.1
-> {	Outline		
UINT16	OutlineFlags	The bit values for the outline settings.	13.2
INT32[]	OutlineDistance	The distances to the representable outline.	13.3
INT32[]	OutlineFrame	The distances from the display area to the text.	13.4
INT32[]	OutlineExpandedXY	The X and Y values for the extended image.	13.5
INT32	OutlineExpandedCount	The number of partial images in the extended image.	13.6
INT32	OutlineExpandedIndex	The zero-based index for the extended image.	13.7
INT32[]!	OutlineCurveXY	The rounding of the outline.	13.8
UINT32[]!	OutlineColor	The background colors of the outline.	13.9
INT32[]!	OutlineShadowXY	The X and Y values for the outline shadow.	13.10
UINT32!	OutlineShadowColor	The color for the outline shadow.	13.11
UINT32!	OutlineBorderColor	The color for the outline border.	13.12
...		3.13 Outline text format, Table 3.13	
...!	OutlineFlags = 0x0040	3.14 Page format, Table 3.14	
INT32	OutlineChildCount	The number of child outlines.	13.13
} <-	Outline		
...		3.1 File format, Table 3.1	

Table 3.12: Outline format

3.13 Outline text format

Type	Name	Description	Info
-> {	OutlineText	Dependencies: OutlineFlags, LanguageCount	14.1
UINT32	OutlineTextColor	The color used for the outline text.	14.2
INT32	OutlineTextWidth	The width of the text representation.	14.3
INT32	OutlineTextHeight	The height of the text representation.	14.4
INT32	OutlineTextLength	The number of characters in the outline text.	14.5
INT32	OutlineFontIndex	The zero-based index for the outline font.	14.6
INT32	OutlineLetterIndexCount	The number of character numbers used.	14.7
INT32[]	OutlineLetterIndex	The memory with the used character numbers.	14.8
INT32	OutlineInfoTypeSize	Is not used. Value = -1	14.9
INT32	OutlineInfoIndexCount	The reference value for the index memory.	14.10
INT32!	OutlineInfoIndexSize	The memory size of the compressed index memory.	14.11
BYTE[]!	OutlineInfoIndex	The index memory with all character indices.	14.12
} <-	OutlineText		
...		3.12 Outline format, Table 3.12	

Table 3.13: Outline text format

3.14 Page format

Type	Name	Description	Info
INT32	PageLineCount	The number of lines in a page.	15.1
-> {	PageLine		
INT32[]	PageLineHeight	The height of a line. Dependency: LanguageCount	15.2
...		3.15 Page field format, Table 3.15	
} <-	PageLine		
...		3.12 Outline format, Table 3.12	

Table 3.14: Page format

3.15 Page field format

Type	Name	Description	Info
INT32	PageFieldCount	The number of page fields in a row.	16.1
-> {	PageField		
BYTE	PageFieldType	The field type determines the further file format.	16.2
...!	PageFieldType = 0	3.16 Empty field format, Table 3.16	
...!	PageFieldType = 1	3.17 Text field format, Table 3.17	
...!	PageFieldType = 2	3.18 Image field Format, Table 3.18	
...!	PageFieldType = 3	3.19 Player field format, Table 3.19	
...		3.20 Field setting format	
} <-	PageField		
...		3.14 Page format, Table 3.14	

Table 3.15: Page field format

3.16 Empty field format

Type	Name	Description	Info
INT32	FieldEmptyWidth	The width of the field for the display content.	17.1
INT32	FieldEmptyHeight	The height of the field for the display content.	17.2
...		3.15 Page field Format, Table 3.15	

Table 3.16: Empty field format

3.17 Text field format

Type	Name	Description	Info
-> {	FieldText	Dependencies: FieldTextFlags, LanguageCount	18.1
UINT16	FieldTextFlags	The value for the settings of the text.	18.2
INT32	FieldTextWidth	The width of the field for the display content.	18.3
INT32	FieldTextHeight	The height of the field for the display content.	18.4
INT32	FieldTextColorCount	The number of colors used in the text.	18.5
UINT32[]	FieldTextColor	The colors used in the text.	18.6
INT32	FieldTextFontCount	The number of fonts used in the text.	18.7
INT32[]	FieldTextFont	The fonts used in the text.	18.8
-> {	FieldTextLetter	Dependency: FieldTextFontCount	18.9
INT32	FieldTextLetterCount	The number of character numbers used.	18.10
INT32[]	FieldTextLetter	The characters for the corresponding font.	18.11
} <-	FieldTextLetter		
INT32	FieldTextLength	The number of characters in the text.	18.12
INT32	FieldTextInfoTypeSize	The memory size for the character type.	18.13
BYTE[]	FieldTextInfoType	The memory with the types of all characters.	18.14
INT32!	FieldTextInfoFontSize	The memory size for the font index.	18.15
BYTE[]!	FieldTextInfoFont	The memory with font indices of all characters.	18.16
INT32!	FieldTextInfoColorSize	The memory size for the color index.	18.17
BYTE[]!	FieldTextInfoColor	The memory with color indices of all characters.	18.18
INT32	FieldTextIndexCount	The reference value for the index memory.	18.19
INT32!	FieldTextIndexSize	The memory size for the character index.	18.20
BYTE[]!	FieldTextIndex	The memory with all character indices.	18.21
INT32[]!	FieldTextSeparatorIndex	The separator numbers of all fonts.	18.22
} <-	FieldText		
...		3.15 Page field Format, Table 3.15	

Table 3.17: Text field format

3.18 Image field format

Type	Name	Description	Info
-> {	FieldImage	Dependencies: FieldImageFlags, LanguageCount	19.1
UINT16	FieldImageFlags	The bit values for the settings of the image.	19.2
INT32	FieldImageWidth	The width of the field for the display content.	19.3
INT32	FieldImageHeight	The height of the field for the display content.	19.4
INT32!	FieldImageUrlLength	The number of characters of the Internet address.	19.5
WCHAR[]!	FieldImageUrl	The Internet address for downloading the image.	19.6
INT32[]!	FieldImageOriginalSize	The width and height of the original image.	19.7
INT32!	FieldImageDisplayIndex	The zero-based index for the display image.	19.8
<- {	FieldImage		
...		3.15 Page field Format, Table 3.15	

Table 3.18: Image field format

3.19 Player field format

Type	Name	Description	Info
BYTE	FieldPlayerUseImage	Determines if a preview image is included.	20.1
...!	FieldPlayerUseImage = 1	3.18 Image field format, Table 3.18	
-> {	FieldPlayer	Dependencies: FieldPlayerFlags, LanguageCount	20.2
UINT16	FieldPlayerFlags	The bit values for the settings to play.	20.3
INT32	FieldPlayerWidth	The width of the field for the display content.	20.4
INT32	FieldPlayerHeight	The height of the field for the display content.	20.5
UINT32	FieldPlayerColor	The presentation color before playing.	20.6
INT32!	FieldPlayerUrlLength	The number of characters of the Internet address.	20.7
WCHAR[]!	FieldPlayerUrl	The Internet address for downloading the file.	20.8
BYTE!	FieldPlayerType	The play format of the original file.	20.9
INT32!	FieldPlayerDisplayWidth	The display width of the original file.	20.10
INT32!	FieldPlayerDisplayHeight	The display height of the original file.	20.11
INT64!	FieldPlayerDuration	The playing time of the original file.	20.12
INT32!	FieldPlayerIndex	The zero-based index of the play file.	20.13
<- {	FieldPlayer		
...		3.15 Page field Format, Table 3.15	

Table 3.19: Player field format

3.20 Field setting format

Type	Name	Description	Info
-> {	FieldSetting	Dependencies: Flags (text, ...), LanguageCount	21.1
UINT16	FieldFlags	The bit values for the page field settings.	21.2
INT32[]	FieldDistance	The distances to the display area of the field.	21.3
INT32	FieldWidth	The width of the entire field.	21.4
INT32[]!	FieldCurveXY	The rounding of the displayable field.	21.5
INT32[]!	FieldFrame	The distances to the content of displayable field.	21.6
INT32[]!	FieldShadowXY	The position of the field shadow.	21.7
UINT32!	FieldShadowColor	The color of the field shadow.	21.8
UINT32!	FieldBorderColor	The border color of the displayable field.	21.9
UINT32!	FieldBackColor	The background color of the displayable field.	21.10
INT32[]!	FieldImageXY	The position of the field image.	21.11
INT32!	FieldImageCount	The number of partial images.	21.12
INT32!	FieldImageIndex	The zero-based index of the field image.	21.13
INT32!	FieldImageStartTime	The start time of the animation.	21.14
INT32!	FieldImageDisplayTime	The display time of a partial image of animation.	21.15
INT32[]!	FieldImageShadowXY	The position of the shadow of a field image.	21.16
UINT32!	FieldImageShadowColor	The color of the shadow of a field image.	21.17
} <-	FieldSetting		
INT32!	FieldIDLength	The number of characters of identification.	21.18
WCHAR[]!	FieldID	The characters of identification for a page field.	21.19
...!	FieldFlags = 0x0400	3.21 Various field format, Table 3.21	
...		3.15 Page field format, Table 3.15	

Table 3.20: Field setting format

3.21 Various field format

Type	Name	Description	Info
UINT16	FieldVariousFlags	The bit values for the actions of the page field.	22.1
INT8[]!	FieldVariousHoverXY	The X and Y values for highlighting the field.	22.2
INT32!	FieldVariousIDLength	The number of identification to call.	22.3
WCHAR[]!	FieldVariousID	The characters to call the identification.	22.4
-> {	FieldVariousTip	Dependencies: FieldVariousFlags, LanguageCount	22.5
INT32!	FieldVariousTipLength	The number of characters of the speech bubble.	22.6
WCHAR[]!	FieldVariousTip	The characters of speech bubble above page field.	22.7
} <-	FieldVariousTip		
-> {	FieldVariousLink	Dependencies: FieldVariousFlags, LanguageCount	22.8
INT32!	FieldVariousLinkLength	The number of characters of the link.	22.9
WCHAR[]!	FieldVariousLink	The characters of the link to call.	22.10
} <-	FieldVariousLink		
...		3.20 Field setting format, Table 3.20	

Table 3.21: Various field format

4 Information about the field format

4.1 Identification number

The identification number identifies the file format. The number can also be displayed with 4 letters (TDHR: Two Dimension Help Reader).

4.2 Number of files

The value indicates the number of files that the entire document contains. The value can not be less than 1. The program "FileViewer" can not read documents with multiple files.

Example: `Test.dthw, Test.dthw.001, Test.dthw.002, usw.`

4.3 File size

The value indicates the size in bytes in which the document is divided. The value can not be less than 1 megabyte (1,048,576 bytes). The document is divided into files of this size. The last file contains the rest of the document with an independent file size. In the example, a document with the size of 22MByte is divided into 10MByte.

Expamle: `Test.dthw = 10MByte, Test.dthw.001 = 10MByte, Test.dthw.002 = 2MByte`

4.4 Document size

The value specifies the document size in bytes. The value can not be smaller than the total size of all project files (see 4.2 and 4.3).

4.5 File version

The file version is always 1 for this description.

4.6 Format bits

The bit values determine the further content of the file format. The bit value "UseTwoLetterLanguage" must always be set for this file format.

Name	Bit	Description
UsePreview	0x0001	The file format contains a preview image. Section 5
UseIcon	0x0002	The file format contains a document icon. Section 6
UseInfo	0x0004	The file format contains additional information. Section 7
UseTwoLetterLanguage	0x0008	The abbreviation of the country names consist of two letters.

Table 4.6: Format bits

4.7 Background color

The value specifies the background color for the document. The value is an ARGB color. It can not be less than 0xFF000000. If the value is 0, no background color is used.

See the document "HelpWriter.tdhr": Settings, Background

4.8 Window color

The value indicates the window color for the document. The value is an ARGB color. The color can only be used if it matches the background color. If the value is 0, no window color is used.

4.9 Distances to the outline structure

The distances to the outline structure consist of 4 values. The left, top, right, and bottom spaces are used to represent the outline structure. The values can not be smaller than 0 or larger than 1000 pixels.

See the document "HelpWriter.tdhr": Settings, Outline border

4.10 Distance to the page content

The distances to the page content consist of 4 values. The left, top, right, and bottom spaces are used to display the page content. The values can not be smaller than 0 or larger than 1000 pixels.

4.11 Number of languages

The value indicates the number of languages used. The document is language neutral if the value is 0. The count is used for the loop "Language" (see Table 3.1).

4.12 Number of characters for the language abbreviation

The number of characters for the abbreviation of language. The value for this file format is always 2.

4.13 Abbreviation of the language

The memory contains the characters of the abbreviation of the language. The length of the abbreviation (see section 4.12) determines the number of characters. Table 4.13 lists all languages and their abbreviations according to ISO 639-1.

	Language		Language		Language
af	Afrikaans	hi	Hindi	pa	Punjabi
ar	Arabic	hr	Croatian	pl	Polish
az	Azeri	hu	Hungarian	pt	Portuguese
be	Belarusian	hy	Armenian	ro	Romanian
ca	Catalan	id	Indonesian	ru	Russian
cs	Czech	is	Icelandic	sa	Sanskrit
da	Danish	it	Italian	sk	Slovak
de	German	ja	Japanese	sl	Slovenian
dv	Divehi	ka	Georgian	sq	Albanian
el	Greek	kk	Kazakh	sr	Serbian
en	English	kn	Kannada	sv	Swedish
es	Spanish	ko	Korean	sw	Kiswahili
et	Estonian	ky	Kyrgyz	ta	Tamil
eu	Basque	lt	Lithuanian	te	Telugu
fa	Persian	lv	Latvian	th	Thai
fi	Finnish	mk	Macedonian	tr	Turkish
fo	Faroese	mn	Mongolian	tt	Tatar
fr	French	mr	Marathi	uk	Ukrainian
gl	Galician	ms	Malay	ur	Urdu
gu	Gujarati	nl	Dutch	uz	Uzbek
he	Hebrew	no	Norwegian	vi	Vietnamese

Table 4.13: Languages according to ISO 639-1

5 Information about the preview image format

The document contains a preview image if the bit value "UsePreview" (see 4.6) is set. The preview image is created in the program "HelpWriter". By default, a PNG compressed image file is used.

See the document "HelpWriter.tdhr": Settings, Thumbnail

5.1 Color content of the preview image

The value determines the transparent color content of the image. Each pixel consists of a 32-bit value, the so-called ARGB color. The alpha value (A) indicates the permeability of the color.

Name	Value	Description
Normal	0	All alpha values are equal to 0xFF.
Transparent	1	The alpha value can be 0x00 or 0xFF.
Alpha	2	The alpha value can contain all numbers (0x00 to 0xFF).

Table 5.1: Color mode

5.2 Image width

The width of the preview image in pixels. The value can not be less than 1 or more than 16,000.

5.3 Image height

The height of the preview image in pixels. The value can not be less than 1 or more than 16,000.

5.4 Memory size

The value specifies the file size of the preview image in bytes. The value can not be less than 1.

5.5 Image memory

The memory contains the preview file that was created for the document. For the size see 5.4.

6 Information about the Symbol format

The document contains an icon if the bit value "UseIcon" (see 4.6) is set. The image is displayed in the window frame on the top left. The maximum dimension is 120 x 29 pixels (pixels).

See the document "HelpWriter.tdhr": Settings, Producer

6.1 Color content of the symbol image

The value determines the transparent color content of the image. Each pixel consists of a 32-bit value, the so-called ARGB color. The alpha value (A) indicates the permeability of the color. See table 5.1 Color mode.

6.2 Image width

The width of the symbol image in pixels. The value can not be less than 1 or more than 16,000. The window frame displays a maximum of only 120 pixels.

6.3 Image height

The height of the symbol image in pixels. The value can not be less than 1 or more than 16,000. The window frame displays a maximum of only 29 pixels.

6.4 Memory size

The value indicates the file size of the symbol image in bytes. The value can not be less than 1.

6.5 Image memory

The memory contains the image file that was created for the document. For the size see 6.4.

7 Discription about the information format

The document contains additional information if the bit value "UseInfo" (see 4.6) is set.

See the document "HelpWriter.tdhr": Settings, Producer

7.1 Length of the producer name

The value indicates the number of characters for the manufacturer name. The name is not included if the number is -1 or 0. The value can not be greater than 1000.

```
if(FieldImageUrlLength == -1) return null; //Use the value -1
if(FieldImageUrlLength == 0) return String.Empty;
```

7.2 Producer name

The name consists of a certain number of characters (letters). The name length (see 7.1) determines the number of characters. A character is a 16 bit unsigned value. The memory size for the name results from the name length times 2 bytes.

Memory size: `InfoProducerLength * 2Bytes (16Bit)`

7.3 Length of the internet address

The value indicates the number of characters for the internet address. The address is not included if the number is -1 or 0. The value can not be greater than 1000.

7.4 Internet address

The address consists of a certain number of characters (letters). The length (see 7.3) determines the number of characters.

7.5 Length of the e-mail address

The value indicates the number of characters for the e-mail address. The address is not included if the number is -1 or 0. The value can not be greater than 1000.

7.6 E-mail address

The address consists of a certain number of characters (letters). The length (see 7.5) determines the number of characters.

8 Information about the text format

All characters are present in an image file. The image is PNG compressed and has 32Bit ARGB color values. The alpha value will not be used here. The illustrated letters (characters) are black on a white background.

8.1 Memory size

The value indicates the memory size for the text image in bytes. The value can not be less than 0.

8.2 Image memory

The memory contains the image file with the characters used. The size is given in section 8.1. If the size is 0 no image memory is used.

8.3 Extra information

There are no extra information. The value is always 0.

9 Information about the font format

9.1 Number of fonts

The value indicates the number of fonts used. It can not be less than 0 or greater than 10,000. The number is used for the loop "Font" (see Table 3.6).

9.2 Font height

The value indicates the line height in pixels. It can not be smaller than 1 or larger than 16,000 (pixels). Figure 9.2 shows the difference between font height and size.

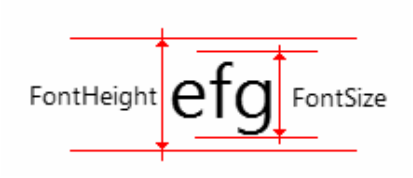


Figure 9.2: Font height

9.3 Base height

The value indicates the centerline of the font. It can not be smaller than the font size or greater than the font height (see 9.2). The base height is used to adjust different font heights in a text line (see 9.4).

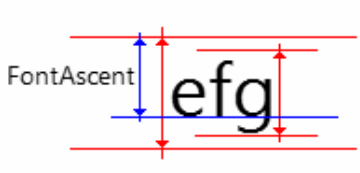


Figure 9.3: Base height

9.4 Line of text

If different font heights are used in a text line, the line must be aligned to the largest base height (3) and font height (1). The base height (2) of the smaller font is subtracted from the height (3), and the result (5) is added as the starting position (Y) in the drawing. Thus, all font heights can be aligned with the center line (4).

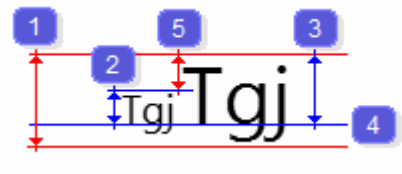


Figure 9.4: Line of text

10 Information about the letter format

10.1 Number of characters

The value indicates the number of used characters (letters) for the corresponding font. It can not be less than 1 or greater 65536.

10.2 Character widths

The set consists of three 16-bit values. These widths are labeled a, b and c and are indicated in pixels. The values a and c can be negative or 0.

Type	Name	Description
INT16	a	The overhang at the beginning of the character.
INT16	b	The width of the displayed character.
INT16	c	The overhang at the end of the character.

Table 10.2: Character widths

The example explains how the position (X) of the letter "b" (italic) can be determined from the character widths.

Figure 10.2: Character widths

The letter "b" starts with a distance (a) to the previous character "A". The width (b) of the letter passes over the following character "c". At the end of the letter "b" the negative distance (c) for the overhang is subtract. The character "c" is displayed at the current position.

10.3 Memory position

The memory position is specified in bytes and refers to the character image (see 8.2). The value can not be less than -1. If the value is -1, the drawing image is not used and the image width (see 10.4) and image height (see 10.5) are not available in the character format. Only the two control characters "Carriage Return" (0x000D) and "New Line" (0x000A) have no image memory. To determine the pixels of the character memory, see program examples 10.6.

10.4 Image width

The value specifies the width of a character in pixels. It can not be less than 1. The image width is not available in the format if the memory position (see 10.3) contains the value -1.

10.5 Image height

The value indicates the height of a character in pixels. It can not be less than 1. The image height does not exist in the format if the memory position (see 10.3) contains the value -1.

10.6 Program examples

In program 10.6.1, the pixels for a character (letter) are copied from the character image (see 8.2). The character image is passed as "Image", the memory position (10.3) as "Pos", the image width (10.4) as "Width" and the height (10.5) as "Height" of the function.

```
public Byte[] LetterMemory(Bitmap Image, Int32 Pos, Int32 Width, Int32 Height) {
  //System.Drawing.Bitmap, System.Drawing.Imaging.BitmapData
  //System.Runtime.InteropServices.Marshal

  Int32 Size = Width * Height * 4;
  Byte[] Memory = new Byte[Size];

  BitmapData Data = Image.LockBits(new Rectangle(0, 0, Image.Width, Image.Height),
                          ImageLockMode.ReadOnly, PixelFormat.Format32bppArgb);

  Marshal.Copy(new IntPtr(Data.Scan0.ToInt64() + Pos), Memory, 0, Size);

  Image.UnlockBits(Data);
  return Memory;
}
```

Program 10.6.1: Character memory

Program 10.6.2 shows how to combine a black letter on a white background with a text color. The function is given the color of the background as "BackImage", the color of the character image (8.2) as "Letter" and the new text color as "Color".

```
public Color LetterColor(Color BackImage, Color Letter, Color Color) {
  //System.Drawing.Color

  if(Letter.R == 0 && Letter.G == 0 && Letter.B == 0)
    return Color;

  if(Letter.R == 255 && Letter.G == 255 && Letter.B == 255)
    return BackImage;

  Int32 Red = (Letter.R * BackImage.R + (255 - Letter.R) * Color.R) / 255;
  Int32 Green = (Letter.G * BackImage.G + (255 - Letter.G) * Color.G) / 255;
  Int32 Blue = (Letter.B * BackImage.B + (255 - Letter.B) * Color.B) / 255;
  return Color.FromArgb(Red, Green, Blue);
}
```

Program 10.6.2: Text color

11 Information about the field image format

The field image (see 3.8) is also used as a foreground or background in a page field. It can also be displayed animated.

See the document "HelpWriter.tdhr": Page, Settings, Picture

The display image (see 3.9) is drawn as content in a page field.

See the document "HelpWriter.tdhr": Page, Image

The expanded image (see 3.11) is displayed if an outline has child elements (outlines).

See the document "HelpWriter.tdhr": Outline, Expanded

11.1 Number of images

The value indicates the number of existing images. It can not be less than 0. The loops "FieldImage" (3.8), "DisplayImage" (3.9) and "ExpandedImage" (3.11) use the value as the number of repetitions. If the value is 0, the corresponding image format is not available. In the following formats, the images are assigned via a zero-based index.

11.2 Color content

The value (see Table 5.1) determines the transparent color content of the image. Each pixel consists of a 32-bit value, the so-called ARGB color. The alpha value (A) indicates the permeability of the color.

11.3 Image width

The value specifies the width of the image in pixels. It can not be less than 1 or more than 16,000.

11.4 Image height

The value specifies the height of the image in pixels. It can not be less than 1 or more than 16,000.

11.5 Memory size

The value is specified in bytes and determines the size of the image file (see 11.6). It can not be less than 1.

11.6 Image file

The memory contains the image file. For the file size see section 11.5.

12 Information about the play file format

The play file is displayed inside a page field. It can be a video, song or animation.

See the document "HelpWriter.tdhr": Page, Player

12.1 Number of files

The value indicates the number of existing files. It can not be less than 0. The loop "PlayerFile" uses the value as the number of repetitions. If the value is 0, the next file format does not exist. In the play field format (see section 20), the files are mapped using a zero-based index (see 20.13).

12.2 File type

The value determines the file type (see 12.7).

Name	Value	Description
Sound	0	The file is an audio file.
SoundImage	1	The file is an audio file with a picture that can be displayed.
Video	2	The file is a video without sound.
VideoAudio	3	The file is a video with sound.
Gif	4	The file is a GIF animation.
TDAnimation	5	The file is an animation (with sound). Created: "PicturePaint"

Table 12.2: File type

12.3 Playing time

The value indicates the playing time for the file. It can not be less than 1.

Playing time: `PlayerDuration * 100ns (Nanosekunden)`

12.4 Display width

The value determines the original display width in pixels. It can not be less than 0 or greater than 16,000. For the file type "Sound" (see 12.2) the value is 0.

12.5 Display height

The value determines the original display height in pixels. It can not be less than 0 or greater than 16,000. For the file type "Sound" (see 12.2) the value is 0.

12.6 Memory size

The value is specified in bytes and determines the size of the file (see 12.7). It can not be less than 1.

12.7 File memory

The memory contains the playable file. For the file size see 12.6.

13 Information about the outline format

13.1 Number of outlines

The value indicates the number of existing structures. It can not be less than 1. The loop "Outline" uses the value as the number of repetitions. If the number of child outlines (see 13.13) is greater than 0, this value can be added.

13.2 Setting bits

The setting bits are used in the program "HelpWriter". Only the value "UseLanguage" affects the file format. In the outline text format (see section 3.13) the number of the loop "OutlineText" is determined on the basis of this bit.

Name	Bit	Description
UseRound	0x0001	The outline field is drawn rounded.
UseShadow	0x0002	The outline field has a shadow.
UseBorder	0x0004	The outline field is displayed with a border color.
UseBackground	0x0008	The outline field has a background color.
UseBackgroundStyle	0x0010	The outline field is drawn with a color gradient.
UseLanguage	0x0020	The outline text can be displayed in multiple languages.
UsePage	0x0040	The outline has one page.

Table 13.2: Setting bits

13.3 Distances

The set consists of 4 values. The left, top, right and bottom distances determine the position of the outline field. The values can not be less than 0 or greater than 10,000 pixels.

See the document "HelpWriter.tdhr": Outline, Distance

13.4 Frame

The set consists of 4 values. The left, top, right, and bottom margins of the outline field for text representation. The values can not be smaller than 0 or larger than 1000 pixels.

See the document "HelpWriter.tdhr": Outline, Frame

13.5 Position of the expanded image

The set consists of 2 values. The X and Y coordinates indicate the position of the expanded image. The values can not be less than 0 or greater than 10,000 pixels.

See the document "HelpWriter.tdhr": Outline, Expanded

13.6 Number of partial images

The value determines the number of partial images in the expanded image (see 3.11). It can not be less than 2 or greater than 999. The width of a partial image is determined by this number. The height is equal to the height of the expanded image. The first partial image is displayed when the outline is closed, the last when opened. The presentation time of a partial image in an animation (number > 2) is 15ms.

13.7 Index of the expanded image

The value indicates the zero-based index for the expanded image (see 3.11). It can not be smaller than 0 or larger and equal to the number of existing images (see 11.1).

13.8 Curve

The set consists of 2 values. The X and Y sizes indicate the curves in the corners of the outline field. The values can not be smaller than 0 or larger than 100 pixels. The values are read if the setting bit "UseRound" (see 13.2) is present. The method "DrawEllipseWidth" (see 13.14) creates the rounding for a rounded rectangle.

```
Int32 PosX = OutlineDistanceTop;
Int32 PosY = OutlineDistanceLeft;
Int32 Width = OutlineFrameLeft + OutlineFrameRigth + OutlineTextWidth;
Int32 Height = OutlineFrameTop + OutlineFrameBottom + OutlineTextHeight;
UInt32 Color = OutlineBorderColor;

DrawEllipseWidth(PosX, PosY, Width, Height, Int32 RoundX, Int32 RoundY, UINT Color)
```

See the document "HelpWriter.tdhr": Outline, Curve

13.9 Background color

The set consists of 3 values. The middle, start and end colors are ARGB colors. They can not be less than 0xFF000000. Only the middle color is used as the background color. It is read if the setting bit "UseBackground" (see 13.2) is present. If the setting bit "UseBackgroundStyle" is set, the start and end colors are also read to represent a color gradient. The method "CreateRoundColor" (see program 13.15.1) creates all the colors for the gradient.

```
Int32 Count = OutlineFrameTop + OutlineFrameBottom + OutlineTextHeight;

CreateRoundColor(Count, 1.0, Color Begin, Color Middle, Color End)
```

See the document "HelpWriter.tdhr": Outline, Background

13.10 Shadow position

The set consists of 2 values. The X and Y coordinates indicate the position of the shadow. The values can not be smaller than 0 or larger than 1000 pixels. The values are read if the setting bit "UseShadow" (see 13.2) is present.

See the document "HelpWriter.tdhr": Outline, Shadow

13.11 Shadow color

The value specifies the ARGB color for the shadow. It can not be less than 0xFF000000. The color is read if the setting bit "UseShadow" (see 13.2) is present.

13.12 Border color

The value indicates the ARGB color for the border. It can not be less than 0xFF000000. The color is read if the setting bit "UseBorder" (see 13.2) is present.

See the document "HelpWriter.tdhr": Outline, Border

13.13 Number of child outlines

The value indicates the number of child outlines. It can not be less than 0. The value can be added to the number (see 13.1) of the loop "Outline".

13.14 Rounding

The C ++ method shows an incomplete program excerpt for creating a rounding. The method is used when the value "RoundX" is greater than or equal to "RoundY".

```
void DrawEllipseWidth(__int32 PosX, __int32 PosY, __int32 Width, __int32 Height,
                                          __int32 RoundX, __int32 RoundY, UINT Color)
{
  double dA = 1.0 / (RoundX * RoundX);
  double dB = 1.0 / (RoundY * RoundY);

  __int32 x = RoundX, y = 0, Xd, Yd;

  __int32 Xp = PosX + Width - RoundX - 1; Xn = PosX + RoundX;
  __int32 Yp = PosY + Height - RoundY - 1; Yn = PosY + RoundY;

  double dY = (y + 1.0) * (y + 1.0) * dB;
  double dX = (x - 0.5) * (x - 0.5) * dA;
  double vY, vX;

  while(dX + dY <= 1.0) {
    y++;
    dY = y * y * dB;

    if(dX + dY > 1.0) {
      x--;
      dX = (x - 0.5) * (x - 0.5) * dA;
    }

    Yd = Yp + y; Xd = Xp + x;
    pMem[Xd + Yd * MemWidth] = Color;

    Xd = Xn - x;
    pMem[Xd + Yd * MemWidth] = Color;

    Yd = Yn - y; Xd = Xp + x;
    pMem[Xd + Yd * MemWidth] = Color;

    Xd = Xn - x;
    pMem[Xd + Yd * MemWidth] = Color;
  }

  dY = (y + 0.5) * (y + 0.5) * dB;

  while(x > 1) {
    x--;
    dX = x * x * dA;

    if(dX + dY < 1.0) {
      y++;
      dY = (y + 0.5) * (y + 0.5) * dB;
    }

    Yd = Yp + y; Xd = Xp + x;
    pMem[Xd + Yd * MemWidth] = Color;

    Xd = Xn - x;
    pMem[Xd + Yd * MemWidth] = Color;

    Yd = Yn - y; Xd = Xp + x;
    pMem[Xd + Yd * MemWidth] = Color;

    Xd = Xn - x;
    pMem[Xd + Yd * MemWidth] = Color;
  }
}
```

Program 13.14: Rounding

13.15 Color gradient

The method creates all colors for a round gradient.

```
public UInt32[] CreateRoundColor(Int32 Count, Double Factor, Color Begin,
                                                 Color Middle, Color End) {
  if(Count < 1) return new UInt32[0];

  UInt32[] ColorArray = new UInt32[Count];

  if(Count < 4) {
    if(Count == 1) {
      ColorArray[0] = (UInt32) Middle.ToArgb();
      return ColorArray;
    }

    if(Count == 2) {
      ColorArray[0] = (UInt32) Middle.ToArgb();
      ColorArray[1] = (UInt32) End.ToArgb();
      return ColorArray;
    }

    ColorArray[0] = (UInt32) Begin.ToArgb();
    ColorArray[1] = (UInt32) Middle.ToArgb();
    ColorArray[2] = (UInt32) End.ToArgb();
    return ColorArray;
  }

  Int32 SizeA = Count / 2;
  Int32 SizeB = Count - SizeA;

  this.CreateSineColor(ColorArray, 0, SizeA, Factor, Begin, Middle);
  this.CreateCosineColor(ColorArray, SizeA, SizeB, Factor, Middle, End);
  return ColorArray;
}
```

Program 13.15.1: Color gradient

The method creates all colors for a round gradient from begin to middle.

```
public void CreateSineColor(UInt32[] ColorArray, Int32 Index, Int32 Count,
                                    Double Factor, Color Begin, Color End) {
  if(ColorArray.Length == 1) {
    ColorArray[0] = (UInt32) Begin.ToArgb();
    return;
  }

  double A = End.A - Begin.A;
  double R = End.R - Begin.R;
  double G = End.G - Begin.G;
  double B = End.B - Begin.B;

  double dCount = (Count - 1) * (Count - 1);

  for(Int32 i = 0; i < Count; i++) {
    double Value = Math.Pow(1.0 - i * i / dCount, Factor);

    ColorArray[Index + Count - 1 - i] =
         (UInt32) Color.FromArgb(Convert.ToInt32(Begin.A + Value * A),
                                 Convert.ToInt32(Begin.R + Value * R),
                                 Convert.ToInt32(Begin.G + Value * G),
                                 Convert.ToInt32(Begin.B + Value * B)).ToArgb();
  }
}
```

Program 13.15.2: Sine colors

The method creates all colors for a round gradient from middle to end.

```
public void CreateCosineColor(UInt32[] ColorArray, Int32 Index, Int32 Count,
                                    Double Factor, Color Begin, Color End) {
  if(ColorArray.Length == 1) {
    ColorArray[0] = (UInt32) Begin.ToArgb();
    return;
  }

  double A = End.A - Begin.A;
  double R = End.R - Begin.R;
  double G = End.G - Begin.G;
  double B = End.B - Begin.B;

  double dCount = (Count - 1) * (Count - 1);

  for(Int32 i = 0; i < Count; i++) {
    double Value = Math.Pow(1.0 - i * i / dCount, Factor);

    ColorArray[Index + i] = (UInt32) Color.FromArgb(
                                Convert.ToInt32(End.A - Value * A),
                                Convert.ToInt32(End.R - Value * R),
                                Convert.ToInt32(End.G - Value * G),
                                Convert.ToInt32(End.B - Value * B)).ToArgb();
  }
}
```

Program 13.15.3: Cosine colors

14 Information about the outline text format

14.1 Number of repetitions

The number of repetitions of the loop "OutlineText" depends on the setting bit "UseLanguage" (see 13.2). If the bit is set, the outline text format (see 3.13) is repeated for each installed language (see 4.11). If the document is language-neutral, the format is read only once.

14.2 Text color

The value indicates the ARGB text color. It can not be less than 0xFF000000.

14.3 Text width

The value indicates the entire width of the outline text. It can not be smaller than 1 or larger than 16,000 pixels.

14.4 Text height

The value indicates the total height of the outline text. It can not be smaller than 1 or larger than 16,000 pixels. The value is identical to the corresponding font height (see 9.2).

14.5 Text length

The value determines the number of characters for the outline text. It can not be less than 1 or greater than 1000.

14.6 Font index

The zero-based index of the font, which was read in format 3.6. The value can not be less than 0 or greater and equal to the font count (see 9.1).

14.7 Number of characters

The value specifies the number of zero-based indexes for the characters read in format 3.7. The value can not be smaller than 1 or greater than the number of characters (see 10.1).

14.8 Character indices

The set contains all indices for the characters used in the outline text. An index can not be less than 0 or greater and equal to the number of characters (see 10.1). It can only happen once in the set. The corresponding character set is determined by the font index (see 14.6).

14.9 Memory size for the character types

The memory for the character types is not used in this format description. The value is always -1.

14.10 Reference value for the character indices

The value is used as the reference for the compressed index memory (see 14.12). It can not be less than 1. It is identical to the number of character indices (see 14.7).

14.11 Memory size for the character indices

The value indicates the size of the index memory in bytes. The value will not be used if the reference value (see 14.10) is equal to 1. Then the number of character indices (see 14.7) is also 1 and only one character is used in the outline text.

14.12 Memory width the character indices

The size of the index memory is given in section 14.11. The memory is not present if the reference value (see 14.10) is 1. The memory is bit compressed and contains the zero based indices for the used character numbers (see 14.8). An index refers to the set of character numbers (indices). In program example 14.13, the index memory is decompressed if the reference value is less than or equal to 256. If more than 256 characters are used, the result memory "ResultArray" must be defined with 16-bit values.

```
Int32 LetterLength = OutlineTextLength;
Int32 ValueMax = OutlineInfoIndexCount;

Byte[] ValueArray = OutlineInfoIndex;
```

14.13 Sample program

```
Int32 LetterLength;
Int32 ValueMax;

Byte[] ValueArray;  //Input memory
Byte[] ResultArray = new Byte[LetterLength]; //Output memory

Int32 BitCountMax = (Int32) Math.Ceiling(Math.Log(ValueMax, 2)); //Bits
Int32 BitCount = 0;
Int32 Index = 0;

for(Int32 i = 0; i < LetterLength; i++) {
  Byte Value = ValueArray[Index];

  Value <<= BitCount;
  BitCount += BitCountMax;

  if(BitCount >= 8) {
    if(BitCount == 8) {
      BitCount = 0;

      Value >>= 8 - BitCountMax;

      Index++;
    } else {
      BitCount -= 8;

      Byte ValueNext = ValueArray[++Index];
      ValueNext >>= 8 - BitCount;

      Value >>= 8 - BitCountMax;
      Value |= ValueNext;
    }
  } else {
    Value >>= 8 - BitCountMax;
  }

  ResultArray [i] = Value;
}
```

Program 14.13: Decompression

15 Information about the page format

The outline has one page if the setting bit "UsePage" (see 13.2) is present.

15.1 Number of lines

The value indicates the number of lines in a page. It can not be less than 1 or greater than 1000. The number is used to repeat the loop "PageLine".

15.2 Line height

The number of values in the set is determined by the installed languages. If the document is language-neutral, only one line height is specified. A height can not be smaller than 1 or larger than 1,000,000 pixels.

16 Information about the page field format

16.1 Number of fields

The value indicates the number of fields in a line. It can not be less than 0 or greater than 1000. The number is used to repeat the loop "PageField".

16.2 Field type

The field type determines the further file format.

Name	Value	Description
Empty	0	The field is displayed empty. See section 16
Text	1	The field is displayed with a text. See section 17
Image	2	The field is displayed with an image. See section 18
Player	3	The field can play a video, music or animation file. See section 19

Table 16.2: Field type

17 Information about the empty field format

17.1 Field width

The value indicates the width of the field. It can not be smaller than 1 or larger than 800,000 pixels.

17.2 Field height

The value indicates the height of the field. It can not be smaller than 1 or larger than 800,000 pixels.

18 Information about the text field format

18.1 Number of repetitions

The number of repetitions of the loop "FieldText" depends on the setting bit "UseLanguage" (see 18.2). If the bit is set, the text field format (see 3.17) is repeated for each installed language (see 4.11). If the document is language-neutral, the format is read only once.

18.2 Settings

The settings are used in the program "HelpWriter". Only the values "UseLanguage" and "Use-Separator" affect the file format. In the text field format (see section 3.17) the number of the loop "FieldText" is determined by the bit "UseLanguage".

Name	Value	Description
Alignment_Left	0x0001	The text is displayed on the left side.
Alignment_Right	0x0002	The text is displayed on the right side.
Alignment_Center	0x0003	The text is displayed in the middle.
Alignment_Justified	0x0004	The text is displayed in justification.
	Bit	
UseLanguage	0x0010	The field text can be displayed in multiple languages.
UseSizeable	0x0020	The page field can change its size.
UseSeparator	0x0040	Separators can be used in the field text.

Table 18.2: Settings

18.3 Field width

The value indicates the width of the field contents. It can not be smaller than 1 or larger than 800,000 pixels. The absolute field width is given by the width (see 21.4).

18.4 Field height

The value indicates the height of the field content. It can not be smaller than 1 or larger than 800,000 pixels. The absolute field height is specified by the line height (see 15.2).

18.5 Number of colors

The value indicates the number of colors used. It can not be less than 1 or greater than 256.

18.6 Text colors

The set contains all the used ARGB colors of the text. The number of the set is determined by the number of colors (see 18.5). A color value can not be less than 0xFF000000. See the example in section 18.23.

18.7 Number of fonts

The value specifies the number of zero-based indices for the fonts that were read in the 3.6 format. The value can not be less than 1 or greater than 256.

18.8 Font indices

The set contains all indices for the fonts used in the field text. An index can not be less than 0 or greater and equal to the font count (see 9.1). It can only happen once in the set. See the example in section 18.23.

18.9 Number of repetitions

The number of fonts (see 18.7) determines the repetitions of the loop "FieldTextLetter". For each font used, a set of character numbers (indices) is given.

18.10 Number of characters

The value specifies the number of zero-based indices for the characters read in format 3.7. The value can not be smaller than 1 or greater than the number of characters (see 10.1).

18.11 Character indices

The set contains all indices for the characters used in the field text. An index can not be less than 0 or greater and equal to the number of characters (see 10.1). It can only happen once in the set. The corresponding amount of characters is determined by the font index (see 18.8). See the example in section 18.23.

18.12 Text length

The value determines the number of characters for the outline text. It can not be less than 1.

18.13 Memory size of character types

The value indicates the size of the memory for the character types. It can not be less than 1.

18.14 Memory with the character types

The information memory contains a 3-bit value for each character. The additional information can only have the maximum value 0x07 (3 bits) if the value "Control" and the bit "NewLine" are contained. For conversion to byte memory see program example 14.13. See the example in section 18.23.

```
Int32 LetterLength = FieldTextLength;
Int32 ValueMax = 8;

Byte[] ValueArray = FieldTextInfoType; //Input memory
```

The memory contains additional information about the character used. A word that can be split contains the value 0x01 (Seperator) at the separation point. In addition, after the letter, a separator (see 18.22) is displayed when a word is separated. For a space, the value 0x02 (Space) is specified. A line break consists of two special characters. At the end the character "Carriage Return" with the value 0x000D is inserted. The character at the beginning of the line has the value 0x000A and is called "New Line". Both special characters have the information (Control) with the value 0x03. The beginning, the new line is additionally marked with the bit 0x04 (NewLine), the result is the information with the value 0x07. If a specific text width is used (see 18.2), the bit (NewLine) may also have been set at the first letter of a word.

Name	Value	Description
Letter	0x00	A character without additional information.
Seperator	0x01	The word can be shared with a seperator.
Space	0x02	The character is a space.
Control	0x03	The special character is specified for a line break.
	Bit	
NewLine	0x04	The bit is set when a new line starts.

Table 18.14: Character information

18.15 Memory size for the font indices

The value indicates the size of the memory for the font indices. It can not be less than 1. If only one font is used in the text, this value does not exist in the format.

18.16 Memory with the font indices

The memory contains a bit-oriented font index for each character. The index is derived from the read order of the font indices (see 18.8). If only one font is used in the text, this memory is not present in the format. For decompression in byte values see program example 14.13. See also the example in section 18.23.

```
Int32 LetterLength = FieldTextLength;
Int32 ValueMax = FieldTextFontCount;

Byte[] ValueArray = FieldTextInfoFont; //Input memory
```

18.17 Memory size for the color indices

The value indicates the size of the memory for the color indices. It can not be less than 1. If only one color is used in the text, this value does not exist in the format.

18.18 Memory with the color indices

The memory contains a bit-oriented color index for each character. The index results from the read sequence of the text colors used (see 18.6). If only one color is used in the text, this memory does not exist in the format. For decompression in byte values see program example 14.13. See also the example in section 18.23.

```
Int32 LetterLength = FieldTextLength;
Int32 ValueMax = FieldTextColorCount;

Byte[] ValueArray = FieldTextInfoColor; //Input memory
```

18.19 Reference value for the chararcter indices

The value is used as the reference for the compressed index memory (see 18.21). It can not be less than 1.

18.20 Memory size for the chararcter indices

The value indicates the size of the memory for the character indices. It can not be less than 1. The value does not exist in the format if the reference value (see 18.19) is not greater than 1.

18.21 memory with the chararcter indices

The memory contains a bit-oriented index for each character. The index results from the read order of the used character indices (see 18.11) for the corresponding font. If only one character per font is used in the text, that memory does not exist in the format. Accordingly, the reference value (18.19) is equal to 1. For decompression in byte values see program example 14.13. If more than 256 characters are used for a font, the result memory "ResultArray" must be defined with 16-bit values. See also the example in section 18.23.

```
Int32 LetterLength = FieldTextLength;
Int32 ValueMax = FieldTextIndexCount;

Byte[] ValueArray = FieldTextIndex; //Input memory
```

18.22 Seperator

If the setting bit "UseSeparator" (see 18.2) is set, an index for a delimiter (see section 10) is specified for each font used.

18.23 Example

Using the example (see 18.23) from the program "HelpWriter" to show how the different sets and memory are defined.

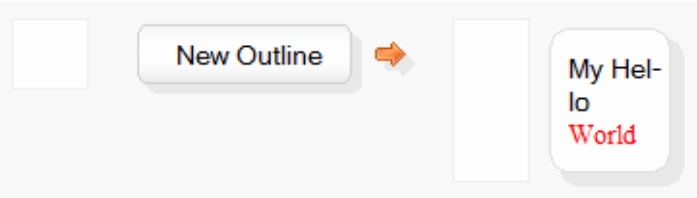


Figure 18.23: Field text

Text colors (siehe 18.6):

```
UInt32[] = {0xFF000000,0xFFFF0000} (Black, Red)
```

Font indices (siehe 18.8):

```
Int32[] = {2,5} (Microsoft Sans Serif, Times New Roman)
```

Character indices (siehe 18.11):

```
Int32[] = {10,11,0,12,15,3,5,13,16} (0x000A,0x000D, ,H,M,e,l,o,y) //M. Sans Serif
Int32[] = {15,16,17,18,19}          (W,d,l,o,r)                   //Times New Roman
```

Character types (siehe 18.14):

```
Byte[] = {0,0,2,0,0,1,0,0,3,7,0,0,0,0,0} (L,L,SP,L,L,L/SE,L,L,CT,CT/NL,L,L,L,L,L)
```

Font indices (siehe 18.16):

```
Byte[] = {0,0,0,0,0,0,0,0,0,0,1,1,1,1,1} //M. Sans Serif (0), Times New Roman (1)
```

Color indices (siehe 18.18):

```
Byte[] = {0,0,0,0,0,0,0,0,0,0,1,1,1,1,1} //Black (0), Red (1)
```

Character indices (siehe 18.21):

```
Byte[] = {4,8,2,3,5,6,6,7,1,0,0,3,4,2,1} //My Hello World
```

Seperator (siehe 18.22):

```
Int32[] = {14,22} (-,-) //M. Sans Serif (14), Times New Roman (22)
```

19 Information about the image field format

19.1 Number of repetitions

The number of repetitions of the loop "FieldImage" depends on the setting bit "UseLanguage" (see 19.2). If the bit is set, the image field format (see 3.18) is repeated for each installed language (see 4.11). If the document is language-neutral, the format is read only once.

19.2 Settings

The settings are used in the program "HelpWriter". Only the values "UseLanguage" and "UseDownload" affect the file format. In the image field format (see 3.18), the number of repetitions for the loop "FieldImage" is determined by the "UseLanguage" bit.

Name	Bit	Description
UseLanguage	0x0001	The image is displayed language-dependent.
UseFullscreen	0x0002	The image can be used in full screen mode.
UseSizeable	0x0004	The size of the image can be adjusted.
UseExpanded	0x0008	The image can be displayed enlarged.
UseEdit	0x0010	The image was created with the program "PicturePaint".
UseDownload	0x0020	The image file is downloaded from the Internet.
UseJpeg	0x0040	The image was compressed JPEG.
UsePng	0x0080	The image was compressed PNG.

Table 19.2: Settings

19.3 Field width

The value indicates the width of the field contents. It can not be smaller than 1 or larger than 800,000 pixels. The absolute field width is given by the width (see 21.4). If the setting bit "UseSizeable" (see 19.2) is not set, this width will be used for display.

19.4 Field height

The value indicates the height of the field content. It can not be smaller than 1 or larger than 800,000 pixels. The absolute field height is specified by the line height (see 15.2). If the setting bit "UseSizeable" (see 19.2) is not set, this height will be used for display.

19.5 Length of the internet address

The value indicates the number of characters for the Internet address. It can not be smaller than 1 or larger than 16000. The value is read if the setting bit "UseDownload" (see 19.2) is present.

19.6 Internet address

The address consists of a certain number of characters (letters). The length (see 19.5) determines the number of characters. A character is a 16 bit unsigned value The address is read if the setting bit "UseDownload" (see 19.2) is present. The memory size results from the length times 2 bytes.

Memory size: `FieldImageUrlLength * 2Bytes (16Bit)`

19.7 Width and height

The set contains the width and height of the original image in pixels. The values can not be less than 1 or more than 16,000. The values are read if the setting bit "UseDownload" (see 19.2) is present.

19.8 Display image

The value indicates the zero-based index for the display image (see 11.6). It can not be smaller than 0 or larger and equal to the number of display images (see 3.9). The value is read if the setting bit "UseDownload" (see 19.2) is not present.

20 Information about the player field format

20.1 Use preview image

The value indicates whether a preview image is used. If the value is 1, the image field format (see 3.18) is read out. There is no preview image if the value is 0.

20.2 Number of repetitions

The number of repetitions of the loop "FieldPlayer" depends on the setting bit "UseLanguage" (see 20.3). If the bit is set, the player field format (see 3.19) is repeated for each installed language (see 4.11). If the document is language-neutral, the format is read only once.

20.3 Settings

The settings are used in the program "HelpWriter". Only the values "UseLanguage" and "UseDownload" affect the file format. In the player field format (see section 3.19), the number of repetitions for the loop "FieldPlayer" is determined by the "UseLanguage" bit.

Name	Bit	Description
UseLanguage	0x0001	The play file is displayed language-dependent.
UseFullscreen	0x0002	The playback is performed in full screen mode.
UseSizeable	0x0004	The size of the presentation can be adjusted.
UseExpanded	0x0008	The presentation can be displayed enlarged.
UseDownload	0x0010	The play file is downloaded from the Internet.
UseRepeat	0x0020	The playback is repeated.
UsePosition	0x0040	The timeline is displayed.
UseVolumen	0x0080	The volume can be changed.

Table 20.3: Settings

20.4 Field width

The value indicates the width of the field contents. It can not be smaller than 1 or larger than 800,000 pixels. The absolute field width is given by the width (see 21.4). If the setting bit "UseSizeable" (see 20.3) is not set, this width will be used for display.

20.5 Field height

The value indicates the height of the field content. It can not be smaller than 1 or larger than 800,000 pixels. The absolute field height is specified by the line height (see 15.2). If the setting bit "UseSizeable" (see 20.3) is not set, this height will be used for display.

20.6 Background color

The value specifies the background color for the field contents before the file is played. It can not be less than 0xFF000000.

20.7 Length of the Internet address

The value indicates the number of characters for the Internet address. It can not be smaller than 1 or larger than 16000. The value is only read if the setting bit "UseDownload" (see 20.3) is set.

20.8 Internet address

The address consists of a certain number of characters (letters).The length (see 20.7) determines the number of characters. A character is a 16 bit unsigned value. The address is only read if the setting bit "UseDownload" (see 20.3) is set. The memory size results from the length times 2 bytes.

Memory size: `FieldPlayerUrlLength * 2Bytes (16Bit)`

20.9 File type

The value determines the file type (see 12.2). This value is only read if the setting bit "UseDownload" (see 20.3) is set.

20.10 Display width

The value determines the original display width in pixels. It can not be less than 0 or greater than 16,000. For the file type "Sound" (see 12.2) the value is 0. This value is only read if the setting bit "UseDownload" (see 20.3) is set.

20.11 Display height

The value determines the original display height in pixels. It can not be less than 0 or greater than 16,000. For the file type "Sound" (see 12.2) the value is 0. This value is only read if the setting bit "UseDownload" (see 20.3) is set.

20.12 Playing time

The value indicates the playing time for the file. It can not be less than 1. This value is only read if the setting bit "UseDownload" (see 20.3) is set.

Playing time: `FieldPlayerDuration * 100ns (Nanoseconds)`

20.13 Play file

The value specifies the zero-based index for the play file (see 12.7). It can not be less than 0 or greater and equal to the number of play files (see 3.10). If the setting bit "UseDownload" (see 20.3) is set, this value will not be read.

21 Information about the field setting format

21.1 Number of repetitions

The number of repetitions of the loop "FieldSetting" depends on the setting bit "UseLanguage" (see 18.2, 19.2 and 20.3) in the field contents. If the bit is set, the field setting format (see 3.20) is repeated for each installed language (see 4.11). If the document is language-neutral or the content is an empty field format (see 3.16), the format is read only once.

If the play format also has a image field format, the setting bit "UseLanguage" must be observed by both formats. If set in a format, the above condition applies to the repetitions of the loop "FieldSetting".

21.2 Settings

The setting bits affect the file format.

Name	Value	Description
AlignmentHorizontal_Left	0x0001	The field is aligned to the left.
AlignmentHorizontal_Center	0x0002	The field is centered.
AlignmentHorizontal_Right	0x0003	The field is aligned to the right.
AlignmentVertical_Top	0x0010	The field is aligned upwards.
AlignmentVertical_Center	0x0020	The field is centered.
AlignmentVertical_Right	0x0030	The field is aligned downward.
	Bit	
UseCurve	0x0004	The field is rounded off.
UseShadow	0x0008	The field has a shadow.
UseBackground	0x0040	The field is drawn with background color.
UseBorder	0x0080	The field is displayed with a border.
UseFrame	0x0100	The field has an inner frame.
UseID	0x0200	The field is assigned an ID.
UseVarious	0x0400	The field has a various field format.
UseImage	0x0800	The field has an extra field image.
UseImageAnimate	0x1000	The field image can be animated.
UseImageForeground	0x2000	The field image is drawn before the field content.
UseImageShadow	0x4000	The field image has an extra shadow.

Table 21.2: Settings

21.3 Distances

The set consists of 4 values. The left, top, right, and bottom distances to the adjacent page fields or lines are specified in pixels. The values can not be less than 0 or greater than 10,000 pixels.

See the document "HelpWriter.tdhr": Page, Settings, Distance

21.4 Field width

The value indicates the width of the page field in pixels. It can not be smaller than 0 or larger than 1,000,000 pixels.

See the document "HelpWriter.tdhr": Page, Settings, Width

21.5 Curve

The set consists of 2 values. The X and Y sizes indicate the curves in the corners of the page field. The values can not be smaller than 0 or larger than 1000 pixels. They are read if the setting bit "UseCurve" (see 21.2) is present.The method "DrawEllipseWidth" (see 13.14) creates the rounding for a rounded rectangle.

```
Int32 PosX = FieldDistanceTop;
Int32 PosY = FieldDistanceLeft;
Int32 Width = FieldFrameLeft + FieldFrameRigth + ItemWidth;
Int32 Height = FieldFrameTop + FieldFrameBottom + ItemHeight;
UInt32 Color = FieldBorderColor;

DrawEllipseWidth(PosX, PosY, Width, Height, Int32 RoundX, Int32 RoundY, UINT Color)
```

The value "ItemWidth" is determined by the field width "FieldWidth" (see 21.4) if the setting format "UseSizeable" (see 18.2, 19.2 and 20.3) has been set in the content formats. If the bit is not set, the width (see 18.3, 19.3 and 20.4) of the corresponding content is used.

The value "ItemHeight" is determined by the line height "PageLineHeight" (see 15.2), if in the content formats the setting bit "UseSizeable" (see 19.2 and 20.3) has been set. If the bit is not set, the height (see 19.3 and 20.4) of the corresponding field content is used. For the text field (see section 17) the value "ItemHeight" is given by the field height "FieldTextHeight".

See the document "HelpWriter.tdhr": Page, Settings, Curve

21.6 Frame

The set consists of 4 values. The left, top, right, and bottom frames from the page field to the field contents. The values can not be less than 0 or greater than 10,000 pixels. They are read if the setting bit "UseFrame" (see 21.2) is present.

See the document "HelpWriter.tdhr": Page, Settings, Frame

21.7 Shadow position

The set consists of 2 values. The X and Y coordinates indicate the position of the shadow. The values can not be smaller than 0 or larger than 1000 pixels. They are read if the setting bit "UseShadow" (see 21.2) is present.

See the document "HelpWriter.tdhr": Page, Settings, Shadow

21.8 Shadow color

The value specifies the ARGB color for the shadow. It can not be less than 0xFF000000. The color is read if the setting bit "UseShadow" (see 21.2) is present.

21.9 Border color

The value indicates the ARGB color for the border. It can not be less than 0xFF000000. The color is read if the setting bit "UseBorder" (see 21.2) is present.

See the document "HelpWriter.tdhr": Page, Settings, Border

21.10 Background color

The value indicates the ARGB color for the background. It can not be less than 0xFF000000. The color is read if the setting bit "UseBackground" (see 21.2) is present.

See the document "HelpWriter.tdhr": Page, Settings, Background

21.11 Position of the field image

The set consists of 2 values. The X and Y coordinates indicate the position of the field image. The values can not be less than 0 or greater than 100,000 pixels. These values are only available if the setting bit "UseImage" (see 21.2) is set.

See the document "HelpWriter.tdhr": Page, Settings, Picture

21.12 Number of partial images

The value determines the number of partial images in the field image (see 3.8). It can not be less than 1 or greater than 999. The width of a partial image is determined by this number. The height is equal to the height of the field image. If the value is greater than 1, the setting bit "UseImageAnimate" (see 21.2) is set. This value only exists if the bit "UseImage" has been set.

21.13 Index of the field image

The value indicates the zero-based index for the field image (see 3.8). It can not be smaller than 0 or larger and equal to the number of existing images (see 11.1). This value is only available if the setting bit "UseImage" (see 21.2) is set.

21.14 Start time of the animation

The value specifies the start time for an animation in milliseconds. It can not be less than 0 or greater than 3,600,000ms (1 hour). This value is only available if the setting bits "UseImage" and "UseImage-Animate" (see 21.2) are set.

21.15 Display time of the animation

The value indicates the display time for a partial image in milliseconds. It can not be less than 0 or greater than 3,600,000ms (1 hour). This value is only available if the setting bits "UseImage" and "UseImageAnimate" (see 21.2) are set.

21.16 Shadow position of the field image

The set consists of 2 values. The X and Y coordinates indicate the position of the shadow.
The values can not be smaller than 0 or larger than 1000 pixels. They are only present if the setting bits "UseImage" and "UseImageShadow" (see 21.2) are set.

See the document "HelpWriter.tdhr": Page, Settings, Picture

21.17 Shadow color of the field image

The value specifies the ARGB color for the shadow. This color uses the alpha value and can be transparent. The value is only available if the setting bits "UseImage" and "UseImageShadow" (see 21.2) are set.

21.18 Length for the ID

The value indicates the number of characters for the ID. It can not be less than 1 or greater than 100. This value is only available if the setting bit "UseID" (see 21.2) is set.

21.19 ID

The ID consists of a certain number of characters. It identifies the page field for a link in various field format (see 21.4). The length (see 21.18) determines the number of characters. A character is a 16 bit unsigned value. This value is only available if the setting bit "UseID" (see 21.2) is set. The memory size for the ID results from the length times 2 bytes.

Memory size: `FieldIDLength * 2Bytes (16Bit)`

22 Information about the various field format

The various field format (see 3.21) is present if the setting bit "UseVarious" (see 21.2) is set.

22.1 Settings

The bit values all affect the file format.

Name	Bit	Description
UseTipText	0x0001	The field displays a speech bubble.
UseTipTextLanguage	0x0002	The speech bubble is language dependent.
UseLinkPath	0x0004	The field opens a link path.
UseLinkPathLanguage	0x0008	The link path is language dependent.
UseFieldID	0x0010	Jump to a page field with the corresponding ID.
UseFieldIDHorizontal	0x0020	The field is displayed horizontally centered after the jump.
UseFieldIDVertical	0x0040	The field is displayed vertically centered after the jump.
UseHover	0x0080	The field can be highlighted.

Table 22.1: Settings

22.2 Position of the highlighted field

The set consists of 2 values. The X and Y coordinates indicate the position of the field when it is highlighted. The values can not be less than 0 or greater than 127 pixels. This value is only available if the setting bit "UseHover" (see 22.1) is set.

See the document "HelpWriter.tdhr": Page, Various

22.3 Länge für die ID

The value indicates the number of characters for the ID. It can not be less than 1 or greater than 100. This value is only available if the setting field "UseFieldID" (see 22.1) is set.

22.4 ID

The ID consists of a certain number of characters. It contains the ID of a page field (see 21.19). The length (see 22.3) determines the number of characters. A character is a 16 bit unsigned value. This value is only available if the setting bit "UseFieldID" (see 22.1) is set. The memory size for the ID results from the length times 2 bytes.

Memory size: `FieldIDLength * 2Bytes (16Bit)`

22.5 Number of repetitions

The number of repetitions of the loop "FieldVariousTip" depends on the setting bit "UseTipText-Language" (see 22.1). If the bit is set, the text of the speech bubble (see 22.6 and 22.7) is repeated for each installed language (see 4.11). If the document is language-neutral, the text is read only once.

22.6 Text length

The value indicates the number of characters for the text of the speech bubble. It can not be smaller than 1 or larger than 16000. This value is only available if the setting bit "UseTipText" (see 22.1) is set.

22.7 Text

The text of the speech bubble consists of a certain number of characters. It is displayed when the mouse cursor lingers for a few seconds over the page field. The length (see 22.6) determines the number of characters. A character is a 16 bit unsigned value. This value is only available if the setting bit "UseTipText" (see 22.1) is set.

See the document "HelpWriter.tdhr": Page, Various

22.8 Number of repetitions

The number of repetitions of the loop "FieldVariousLink" depends on the setting bit "UseLinkPath-Language" (see 22.1). If the bit is set, the link path (see 22.9 and 22.10) is repeated for each installed language (see 4.11). If the document is language-neutral, the path is read only once.

22.9 Path length

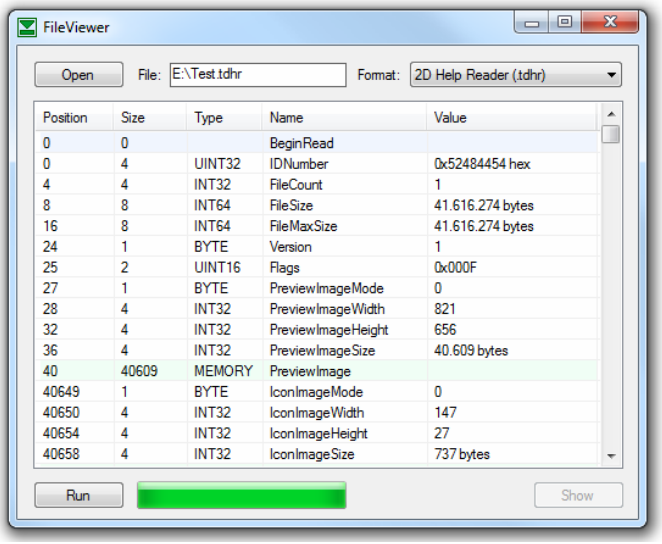
The value indicates the number of characters for the link path. It can not be smaller than 1 or larger than 16000. This value is only available if the setting bit "UseLinkPath" (see 22.1) is set.

22.10 Path

The link path consists of a certain number of characters. It is called when the mouse button on the page field is pressed. The path length (see 22.9) determines the number of characters. A character is a 16 bit unsigned value. This value is only available if the setting bit "UseLinkPath" (see 22.1) is set.

23 Program for reading the file format

On the PanotiSoft website, there is a test program under technical documents, with which the file format can be read out in a structured manner. In addition, the program code can also be downloaded. The program was written under Visual Studio 2008 with the programming language C#.



Program: FileViewerX64.zip oder FileViewerX32.zip

Project file: FileViewerCode.zip

Description: FileViewer.pdf

FileViewerCode:

Format file: FileViewerFormat.cs
Format class: FileViewerTDHelpReader