Any text that appears before the first heading is not included in .chm file. So for example, if this Word document was converted to a chm file, all this text would not be included. This allows a spot to include a table of contents in the Word document.

Table of Contents

[About 1](#_Toc396164143)

[Default Functionality 1](#_Toc396164144)

[Headings 1](#_Toc396164145)

[Filenames 1](#_Toc396164146)

[About CHM Tags 1](#_Toc396164147)

[InvariantName Tag [CHM InvariantName=”InvarNameTag”] 1](#_Toc396164148)

[ImageNumber [CHM InvariantName=”ImageNumber” ImageNumber=”10”] 1](#_Toc396164149)

[Tooltips [CHM InvariantName=”Tooltips”] 1](#_Toc396164150)

[Hyperlinks [CHM InvariantName=”Hyperlinks”] 1](#_Toc396164151)

[Linking to External Files [CHM InvariantName=”ExternalLinks”] 1](#_Toc396164152)

[Linking Across Documents [CHM InvariantName=”AcrossLinks”] 1](#_Toc396164153)

[Indexing [CHM InvariantName=”Indexing”] 1](#_Toc396164154)

[Curly Brackets [CHM InvariantName=”IndexingCurlyBrackets”] 1](#_Toc396164155)

[The Bookmark Attribute [CHM InvariantName=”IndexingBookmark”] 1](#_Toc396164156)

[The Categories Attribute [CHM InvariantName=”IndexingCategories”] 1](#_Toc396164157)

[Category Displayed Page [CHM InvariantName=”IndexingCategoryPage”] 1](#_Toc396164158)

[Sub Categories [CHM InvariantName=”IndexingSubCategories”] 1](#_Toc396164159)

[The Text Attribute [CHM InvariantName=”IndexingTextAttribute”] 1](#_Toc396164160)

[The SeeAlso Attribute [CHM InvariantName=”IndexingSeeAlso”] 1](#_Toc396164161)

[Disabling Indexing [CHM InvariantName=”IndexingDisable”] 1](#_Toc396164162)

[International Characters [CHM InvariantName=”Globalization”] 1](#_Toc396164163)

Also, a possible future enhancement may allow to define a path to the setting.ini file to use when doing the compiling. It might look like:

IniSettings=”<path to ini\somefile.ini>”

Where path to ini can be a relative path or an absolute path. For now, these options are set in the user interface.

# About

This document describes how to format a Word document so that it can be easily converted to a help file (.chm).

{Heading styles} are used to organize the content of a Word document into a hierarchical tree structure. Each heading becomes a link in the help file’s {Table of Contents} and clicking on that link displays the content.

Any text that appears before the first heading is not included{Text=”Hiding Text”} in the .chm help file. For example, this Word document contains a preamble, but it does not show up in the chm help file. This is by design.

# Default Functionality

The {default functionality} is to include the following:

|  |  |
| --- | --- |
|  | As you can see, the buttons are Back, Forward, Home, Font, Print and Options. There is a menu bar. The tabs are Contents, Index, Search and Favorites. The Search tab provides the advanced search features to make for a powerful way to search the help file.  It’s easy to insert images and tables into the chm file. When inserting an screenshot image, to make sure the quality is high, save it as a png file, and then insert the png image file into the Word doc. |

# Headings

Only {Headings 1 to 6} impact the layout of the {TOC Text=”Table of Contents”}. When converting a Word doc to a chm file, the first step is to convert it to one big html file. Then that big html file is broken down into smaller html files based on the headings. Word automatically replaces headings 1 to 6 with respective header tags <h1> to <h6>. However, anything above 6 Word uses a different textual representation, e.g. “<p class=MsoHeading7 ...” The {parser Text=”Headings” Categories=”Parser”} only supports the header tags. Other styles can be used freely, it’s just that the parser doesn’t treat them special.

When a small heading appears after a large heading, it appears as a sub-topic of the previous heading. When two headings have the same heading size, they appear at the same level. When a large heading follows a small heading, the large heading becomes a sub-topic of the first previous heading found that has a heading size larger than its own. If no such heading is found, e.g. <h1>, then the section appears at the top level in the help file’s table of contents. For example, this section “Headings” and the previous section “About” both appear at the top level in the TOC.

The content of a page is going to be everything from the beginning of the heading to the start of the next heading.

## Filenames

The “Filenames” heading is defined using “Heading 2” style. Thus, it is nested inside the “Headings” section, which uses “Heading 1” style.

The [Headings](#_Headings_1) section mentions that the big file is broken into smaller files, but it doesn’t say how the filenames are chosen. The {filenames} are basically the heading text. For example, this section is converted into a file called “Filenames.html”.

In the example given above, the header text was all English alphabet characters and didn’t contain any spaces or special characters. In the real world, header names will contain international characters and spaces and special characters. In order to create a URL friendly filename, a sequence of conversions must be done. The help file viewer cannot handle filenames with special characters.

In windows, there are 9 characters which are not allowed to be part of a filename. They are: '/' '\\' ':' '\*' '?' '"' '<' '>' '|'. In Addition to those, CHM does not like filenames that contain the space character. All those characters, plus the space character are converted to the dash character: '-'

The next step is to convert any non-English alphabet characters to their equivalent, if one exists. For example, ü -> u. However, characters that do not have an equivalent cannot be converted, for example, 汉语ü 🡪 汉语u. Another conversion is done that replaces consecutive dashes with a single dash. After that, PunyCode is used to convert the rest of the filenames (which handles the Chinese characters). If after all the conversions the filename is blank, then the name “File” is used. Subsequent duplicate filenames will have a -k appended, where k is an integer, e.g. File-2.html, File-3.html, etc.

To summarize, the conversion process is:

1. Start with the header text.
2. Replace Window’s special characters with a dash.
3. Replace international characters with the English alphabet equivalent if possible.
4. Replace consecutive dashes in a row with a single dash.
5. Convert any remaining international characters to PunyCode.
6. Check if filename is blank, if so, then choose “File”.

For example: 汉语ü 🡪 xn--nswy88du

If you want to explicitly specify the filename, you can do that by using a CHM tag and the [InvariantName attribute](#_InvariantName_Tag_[CHM_1). Invariant names are subject to the same conversion process from step #2 onwards.

## About CHM Tags

The {[CHM] tag} is a way to encode {meta information} in the Word doc that the conversion process will use when converting to a chm file. Currently, the CHM tag only has impact if it appears in the text of a heading style 1 to 6. During the {conversion process}, the [CHM] tag is automatically removed before the chm file is compiled, thus the end users never see the CHM tag.

Currently there are two attributes associated with a CHM tag, [InvariantName](#_InvariantName_Tag_[CHM_1) and [ImageNumber](#_Image_Number_[CHM). These are explained in the following sections.

Here is an example of how a CHM tag would look in a Word document:

Starting a Business [CHM InvariantName=”StartingBusiness” ImageNumber=”10”]

### InvariantName Tag [CHM InvariantName=”InvarNameTag”]

**Note:** The InvariantName tag is only for programmers who want to link specific screens in their application to specific topics in the help file. If you are working with a programmer then you may want to ask them if they want to use this feature.

Up until now, CHM {InvariantName} hasn’t been used. However, it’s recommended to always use it if you are planning on making your software available to an {international market Categories=”International”}.

One of the goals of the NüHelp project was that when a user was using our software, the user could press the {“F1” key} (or click the help button), and the help file would open to the topic explaining about the screen the user was currently. This is very convenient for the user since they don’t have to search.

In order to do that, when the help file is opened, it has to be passed the name of the page to display. Recall that we use the heading text as the filename (unless a CHM tag is defined).

Now imagine that this document was converted to another language. That means that all the text in the headings would change. Thus, our program wouldn’t be able to find the correct help section since all the filenames are different. The help file would open, but the content would appear as a page not found error. This is where the InvariantName property comes in.

The InvariantName specifies that the name of the file to save. Once defined, it should never be changed, hence it’s “invariant”. For example, even though this section is called “InvariantName Tag”, the filename created will be “InvarNameTag.html”. The “.html” extension is automatically added so you do not have to add it yourself.

**Important**: The InvariantName cannot contain spaces. The Microsoft help file {compiler} cannot handle filenames that contain spaces, they will show up a page not found error. Also, invariant names must be unique, otherwise the software wouldn’t know which page to open. **If the InvariantName cannot be used as-is, then it will be slightly modified to make it useable, and a warning message is displayed. It is up to the user to resolve the underlying issue.**

If the InvariantName tag isn’t used, and the header text contains spaces, the filename replaces spaces with a dash character, otherwise the links would be broken.

Another attribute that can be defined in the CHM tag is the [ImageNumber](#_Image_Number_[CHM), which is defined in the next section.

### ImageNumber [CHM InvariantName=”ImageNumber” ImageNumber=”10”]

Image number is optional. If no {ImageNumber} is defined, then the book “1” and text “11” are used by default.

In the help file’s table of contents, you can specify which icon appears beside the text based on a number. The closed book is 1, the open book is 2, etc.



These are the default icons

**Very Important:** You cannot use a book or folder for a leaf node (no children). Sections that contain other sections must use an image number 1 to 8 inclusive, otherwise the chm file won’t generate correctly. Similarly, {leaf nodes} must use an image number 9 or higher.

Note: There is no point in specifying an open book or open folder as the icon. By default, it will show up as the closed version.

It is possible to create a custom {icon strip}. However, the “.bmp” file must be shipped with the .chm file. In BODY section of the .hhc file must contain the following:

<OBJECT type="text/site properties">

<param name="ImageList" value="file://%systemroot%\customicons.bmp">

<param name="Image Width" value="16">

<param name="Color Mask" value="0xff00ff">

</OBJECT>

# Tooltips [CHM InvariantName=”Tooltips”]

It’s possible to include <span title=”Tooltip example!”>tool tips</span> in the html pages. Originally I tried using a Word comment, however this did not work well when saved as html. The way I settled on was to include a span tag with the title text. Basically this is a simple way to do this in HTML.

When the Word doc is saved to html, Word replaces angle brackets with the ampersand representation. Thus, a bit of {parsing Text=”Span tag” Categories=”Parser”} is done to convert the &gt; and &lt; back to angle brackets. Currently this is the only embedded html that is supported.

When using a {tooltip}, it’s recommended to make it stand out by underlining it and possibly highlighting it. This acts a visual cue for the user to know there’s a tooltip there.

# Hyperlinks [CHM InvariantName=”Hyperlinks”]

You may have already seen a few hyperlinks in this document. Create a hyperlink to a header or bookmark in this document and the same hyperlink appears in the .chm help file. It’s also possible to {hyperlink Text=”URL” Categories=”Hyperlink”} to a URL.

[Here is a hyperlink](#CHM_Important) to the Important section of the CHM tags.

Here is a hyperlink to [google](http://www.google.ca/). Note: it’s probably not a good idea to {link to external websites} because the CHM browser may not be as secure as the browser the user normally uses.

Here is a {hyperlink Text=”Email” Categories=”Hyperlink”} to an [email](mailto:info@opulos.com?subject=Support) address.

## Linking to External Files [CHM InvariantName=”ExternalLinks”]

{Linking} to {external files} is supported. The files can be {relative paths} or {absolute paths}. Relative paths are relative to the location of the chm file. It’s recommended to use relative paths. All externally linked files must start with the prefix “file://”. In Word, press ALT+F9 to toggle to see the merge field guts of the hyperlink and to be able to edit it.

**Important Note:** Hyperlinks are just merge fields. A merge field is created by pressing CTRL+F9 and the curly brackets are not the same as regular curly brackets.

IndexingOff

Example of Relative Path: { HYPERLINK "file://output/Help2.chm" }

Example of Absolute Path: { HYPERLINK "file://C:\AbsPath\Help2.chm" }

IndexingOn

In the first example, there would have to be an “output” folder containing the Help2.chm file relative to the location of the help file that contained the link.

In the second example, the Help2.chm file would have to exist in the C:\AbsPath\ folder.

Externally linked files show up as a {Green Hyperlink} in the output CHM file.

## Linking Across Documents [CHM InvariantName=”AcrossLinks”]

This section applies when combining multiple Word doc files into a single chm file, either from the user interface or from the command line. Users may break a large document into small files to make it easier to manage. However, those smaller documents may contain hyperlinks to each other.

When a document is converted to HTML, the hyperlink may look like:

<a href="../../../../../../../temp/Test1a.docx">...</a>

or

<a href="../../../../../../../temp/Test1a.docx#bookmark">...</a>

NüHelp automatically detects these links by looking for href entries that end with .docx or .doc (excluding the #bookmark portion if it exists). NuHelp then searches those nodes (a.k.a. pages) in the project tree that have the same source filename (e.g. Test1a), and if a bookmark is specified then it looks for a node that contains that bookmark. If no bookmark is specified, then the first node is chosen.

# Indexing [CHM InvariantName=”Indexing”]

IndexingOffIndex keywords show up in the "Index" tab of the help file. It serves the same purpose as the index at the back of a book.

## Curly Brackets [CHM InvariantName=”IndexingCurlyBrackets”]

The curly brackets {} are used to create index entries. The most basic example of an <span title=”Use curly brackets to surround the word.”>{index}</span> is this. The word "index" will appear in the Index tab and it will be linked to this section of the word document.

Note: The headers should NOT contain indexing. It will be ignored.

## The Bookmark Attribute [CHM InvariantName=”IndexingBookmark”]

By default, a link is created to the document section in which the index was defined. To link to a specific place within that document, the Bookmark attribute is used. For example:

{index Bookmark=”#bookmark”}

For this to work, the document must define the “#bookmark”.

## The Categories Attribute [CHM InvariantName=”IndexingCategories”]

Suppose you wanted “index” to appear in a category called Formatting. So do this, you would write:

{index Categories="Formatting"}

In the Index tab, it would appear as:

Formatting

Index

Notice that "Categories" is plural. It's a comma separated list of category names. A blank category name represents the root level. If you wanted a word to appear in a category and at the root, the syntax is: {someword Categories=”,CategoryName”}

Another example: {blah Categories=”A,B,C,”} would look like:

A

Blah

B

Blah

Blah

C

Blah

“Blah” would be listed in each category, as well as the root level (since there is a trailing comma after C).

### Category Displayed Page [CHM InvariantName=”IndexingCategoryPage”]

By default, when a category is double clicked, a blank page is shown. To show a specific page the syntax is:

{Categories=”Fruits=fruits.html”}

Note: you only need to define the page once. Subsequent uses of the Fruits category will automatically use the first page defined. As well, the “.html” is optional.

### Sub Categories [CHM InvariantName=”IndexingSubCategories”]

It’s also possible to specify subcategories, and it is common to see them in the index. The syntax is:

{apple Categories=”Foods|Fruits”}

The pipe character “|” is used to denote a subcategory. The output would look like:

Foods

Fruits

Apple

And here is an example of combining the displayed pages and subcategories:

{banana Categories=”Foods=foods.html|Fruits=fruits.html” }

And here is an example of combining everything:

{orange Categories=”Foods=foods.html|Fruits=fruits.html,Colors=colors.html”}

The output would be:

Colors

orange

Foods

Fruits

orange

Even though “Colors” was defined after “Foods”, it appears sorted in the index, otherwise the index wouldn’t be very useful.

## The Text Attribute [CHM InvariantName=”IndexingTextAttribute”]

You can also format the display name by using the Text attribute:

{index Text="Index Fun!"}

In fact, if you use the Text attribute, you don’t need to surround {index}. It would work equally well if you wrote {Text=”Index Fun!”} just by itself.

## The SeeAlso Attribute [CHM InvariantName=”IndexingSeeAlso”]

Another thing which you typically see in an index is:

Fruits

(see also Vegetables)

When you double click on (see also vegetables) it navigates to Vegetables in the index.

The syntax is: { SeeAlso="vegetables" Text="(see also Vegetables)" Categories="Fruits" }

If Text is omitted, it's automatically generated as "(see also \_x\_)" where \_x\_ is the value of SeeAlso.

A category isn't required though. An example might be: { SeeAlso="Money" Text="Cash" }

Then there would be an index entry called "Cash" at the root level that when double clicked navigates to the "Money" entry (also at the root level). However, it is unusual to have a See Also at the root level.

Another example: {Money seealso="Money!!!" Text="Money!!!" Categories=”,Financing”}

Money!!!

Financing

Money!!!

Double clicking “Money!!!” under financing navigates to “Money!!!” at the root level.

## Disabling Indexing [CHM InvariantName=”IndexingDisable”]

Sometimes it’s desired to turn indexing on or off. When indexing is off, curly brackets are not treated specially. To do this in the word document is simply a matter of writing

IndexingOffIndexingOffIndexingOff

{Anything in curly brackets is now treated like other text.}

IndexingOnIndexingOnIndexingOn

Now curly brackets are treated special again. \*The exception to this is if inside the curly brackets there is a list of Keywords=”…”. See the [Keywords topic](#_Keywords_[CHM_InvariantName=”Keywor).

Note: IndexingOnIndexingOn and IndexingOffIndexingOff are automatically deleted in the chm file so they won’t appear.

# Keywords [CHM InvariantName=”Keywords”]

Keywords is a quick way to create index entries, and is independent of the IndexingOffIndexingOff state.

To use keywords, open with a curly bracket, write Keywords=”followed by a list of delimited words”, followed by close curly bracket. For example:



The delimiting character can be specified in the options panel. The default-value is the semi-colon; but it can be changed by the user.

Type two {delimiting characters} in a row to skip that delimiting character. For example, IndexingOff{ “Vancouver,, Canada, Toronto,, Canada”}IndexingOn

Keywords is intended for those users who don’t want to put curly brackets around the words in the body of the text. Keywords provides a quick and easy way to type a list of keywords.

{ Keywords="Indexing; Meta; How to use Keywords; Escape;;the;;delimiter :)” }

# International Characters [CHM InvariantName=”Globalization”]

IndexingOnOther {languages} are {Text=”Globalization”}{Text=”Languages” Categories=”International”}{SeeAlso=”Globalization” Categories=”International”}supported by default. Nothing special had to be done to support this.

Chinese: 汉字/漢字

Japanese 漢字仮名交じり文

Arabic characters بْجَدِيَّة عَرَبِيَّة

Russian Здравствуйте!