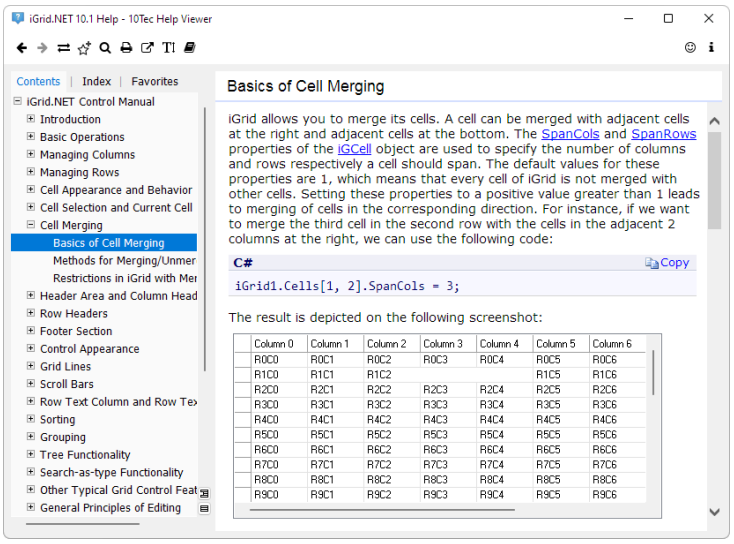
10Tec Help Viewer
Version 2.1
Manual



CONTENTS

CONTENTS **1**

GENERAL INFORMATION **2**

HELP LIBRARY CONCEPT **3**

INTERFACE **4**

MOUSE AND KEYBOARD COMMANDS 4
SPECIAL FUNCTIONS OF THE ESC KEY 4
FAVORITES TAB 5
INCREMENTAL SEARCH IN TOPIC LISTS 5
BULK COLLAPSE/EXPAND OPERATIONS IN CONTENTS 6
FILTER BOX FOR INDEX 7

CONFIGURATION FILE **8**

APPLICATION SETTINGS 8
USER SETTINGS 21

COMMAND LINE PARAMETERS **24**

GENERAL INFORMATION 24
<HELPFILENAME> PARAMETER 24
MAIN SWITCHES 24
SERVICE SWITCHES 25

COMMAND FILES **27**

GENERAL INFORMATION 27
COMMAND FILE SECTIONS 28
USE OF COMMAND FILE SECTIONS 29
GUIDE ON COMMAND FILES FOR C# DEVELOPERS 29

USER INTERFACE LOCALIZATION **32**

MISCELANEOUS **34**

COMMON TEMPLATE FIELDS 34
SUPPORTED HELP FILE FEATURES 34
LIMITATIONS OF THE FREE VERSION 35

ADDENDUM **36**

TYPICAL CONTENTS OF CONFIGURATION FILE 36
F1 CONTEXT HELP COMMAND FILE EXAMPLE 38
C# SAMPLE OF HELP VIEWER CALL 39

GENERAL INFORMATION

10Tec Help Viewer is an application for browsing help files in the Microsoft Help Container format (MSHC). This is the latest format of help files developed by Microsoft. Files of this format are specific zip archives with the .mshc extension. They contain help topics in the HTML format with added tags, such as keywords and the position in the table of contents. MSHC help files are designated mainly for integration into the Microsoft Help Viewer application distributed with Microsoft Visual Studio.

Developers can create help files for end-user applications in this modern format. Numerous help authoring tools automate this process. However, there is one significant problem – viewing of MSHC files. If you want to browse an MSHC help file the traditional way, you must install Microsoft Visual Studio with Microsoft Help Viewer and discard the ability to use Microsoft's online documentation by switching the help preference from online content to local help. There is an alternative H3Viewer tool for browsing MSHC files developed by the Helpware Group, but it also requires Microsoft Visual Studio to be installed on the pc. All this makes hardly possible using of the MSHC format for providing help systems in the real world.

10Tec Help Viewer was developed as a viable solution to view MSHC files. It is a lightweight portable application that does not require installation and does not depend on Microsoft Visual Studio. The only system requirement is the .NET Framework of the version 4.7.2 or 4.8. Fortunately, all latest versions of Windows starting from Windows 7 already have one of these frameworks preinstalled, so there will be no need to install something else to launch the application in the vast majority of cases.

The first versions of 10Tec Help Viewer were created solely for the help files for iGrid.NET, a WinForms grid control developed by 10Tec Company. But the application has evolved, and now it can be used to browse a wide reach of MSHC help files. The enhanced application infrastructure makes it even possible viewing of the contents of existing CHM files after their conversion to the MSHC format, which can be done with the mshcMigrate utility developed by the aforementioned Helpware Group.

It is noteworthy that the application makes extensive use of iGrid itself in its interface: for the hierarchical table of contents on the Contents tab, for the filterable index on the Index tab, and for the multi-book list of favorite topics on the Favorites tab. As such, 10Tec Help Viewer is a good demonstration of the iGrid.NET features.

The interface of 10Tec Help Viewer is modelled on the interface of Microsoft Help Viewer. It is a one-window application with the toolbar at the top, the navigation pane at the left, and the topic browser pane in the remaining space. The main keyboard combinations are also the same in both applications.

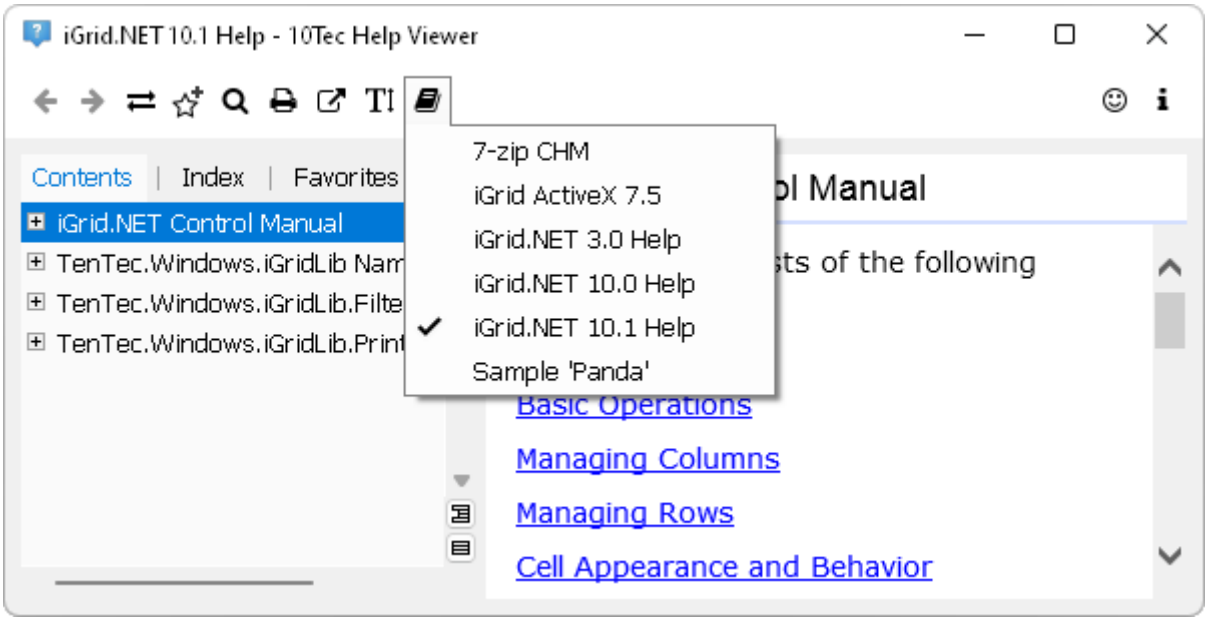
The application can be configured with numerous settings. It supports automation through the command-line interface to open a given help file and a given topic in it.

10Tec Help Viewer is a DPI-aware application. Its interface elements are drawn with the best quality on high-resolution screens. You can also adjust interface elements for traditional displays with the 96dpi resolution – for example, increase the image size in the toolbar and specify another font for the application window. The interface of 10Tec Help Viewer can be localized as well.

HELP LIBRARY CONCEPT

10Tec Help Viewer, or simply Help Viewer for short, is based on the concept of help library. Help library is a collection of help files or books you can open and view. In the case of Help Viewer, the help library is a set of MSHC files in a folder. You can add a new book simply by adding its MSHC file to the folder and remove a book by removing its MSHC file from the folder. Help Viewer uses its own folder as the help library folder by default, but you can this override this with command line switches.

When you launch the application, it searches for MSHC files in the help library folder. All found MSHC files are added to the list of available books. This list drops down when you click the toolbar button with a book image. You can select the file to view from this list:



This list of available MSHC files contains file titles that are differ from the filenames in the general case. When Help Viewer finds an MSHC file to add to the list, first it checks whether the corresponding help manifest file exists in the same folder. Help manifest files have the .msha extension and the same file name before the dot. If such a file exists, Help Viewer retrieves the file title from the manifest. If the manifest file does not exist, Help Viewer replaces the underscore characters in the file name with spaces and cut off the .mshc extension to get the file title to display in the choice list.

The list of help files is sorted using natural sorting: strings are placed in alphabetical order, except that multi-digit numbers are treated as if they were a single character. The picture above demonstrates the help files for iGrid.NET 3.0, 10.0 and 10.1 placed in the order expected by a human due to natural sorting.

When the application is launched for the first time, the first found help file is opened in the application automatically. When you launch the application again, it will try to open the last viewed help file if it still exists.

Help Viewer creates a special index file for an MSHC file when the file is opened for the first time. This index is used to speed up the population of the list on the Index tab when the MSHC file is opened again. Help Viewer's index files are stored in files with the .hvidx extension.

INTERFACE

Mouse and Keyboard Commands

All operations in the interface can be performed with the mouse. Help Viewer also provides keyboard shortcuts for the main operations. They can be used to speed up interaction with the application, especially when you need to search the required bits of information using text search. The table below lists available keyboard commands:

ACTION	SHORTCUT	NOTES
Switch to the Contents tab	Alt+C	
Switch to the Index tab	Alt+I	
Switch to the Favorites tab	Alt+F	
Previous topic	Alt+Left	Also the Back mouse button.
Next topic	Alt+Right	Also the Forward mouse button.
Show topic in Contents	Ctrl+S	Sync the currently viewed topic with the Contents tree.
Add topic to Favorites	Crl+D	
Find text in topic	Ctrl+F	
Print topic	Ctrl+P	
Open topic in external browser	Ctrl+E	Opens the current topic in the default Internet browser.
Use “BeginsWith” filter in Index	F7	
Use “Contains” filter in Index	F8	
Toggle between the “BeginsWith” and “Contains” filter in Index	Ctrl+K	

Special Functions of the Esc Key

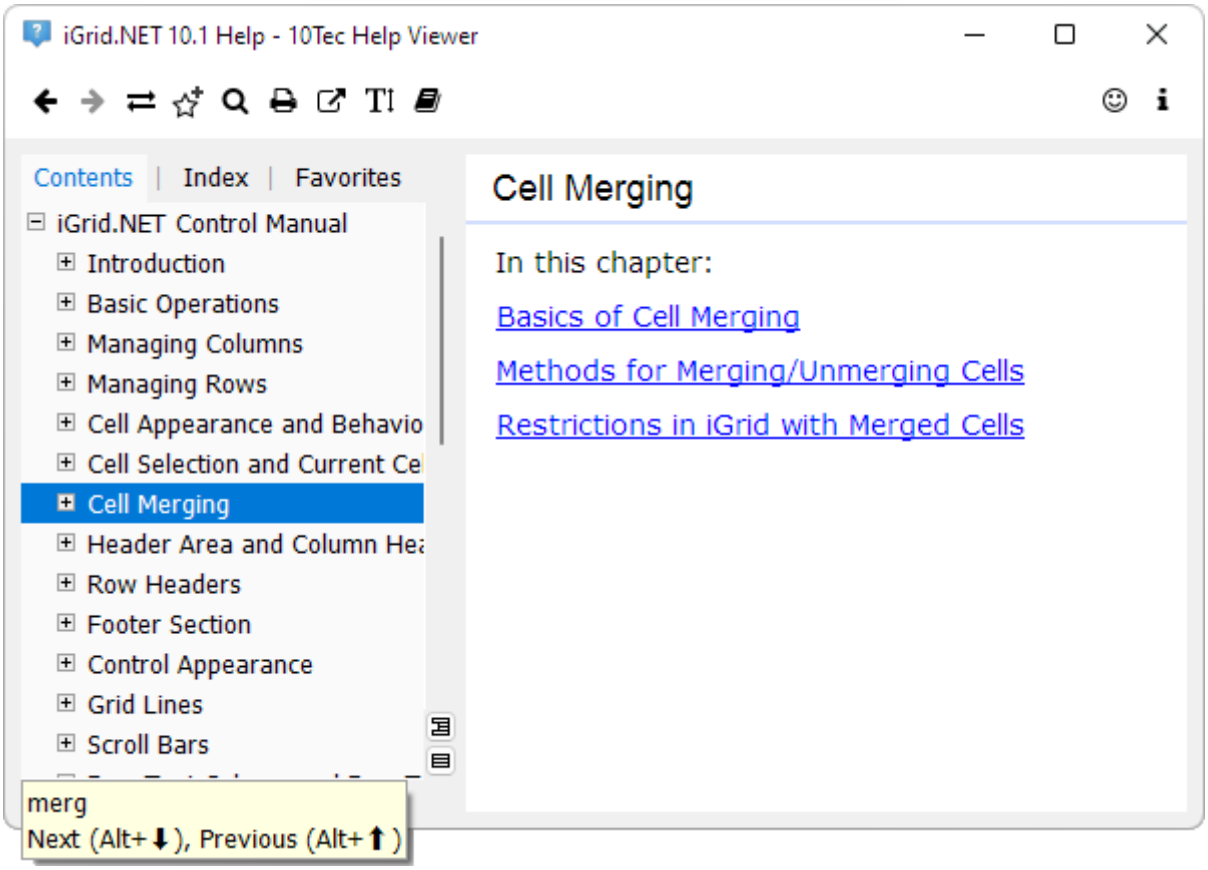
- If the input focus is in the filter box at the top of the Index tab, the Esc key clears the contents of the filter box and removes the existing filter.
- If the Find panel in the topic browser is opened (Ctrl+F), the Esc key closes this panel and cancels the text search in topic.

Favorites Tab

When the Favorites tab is active and an item in the Favorites list is selected, you can rename it (F2) or delete (Del). These actions are also available in the list's context menu.

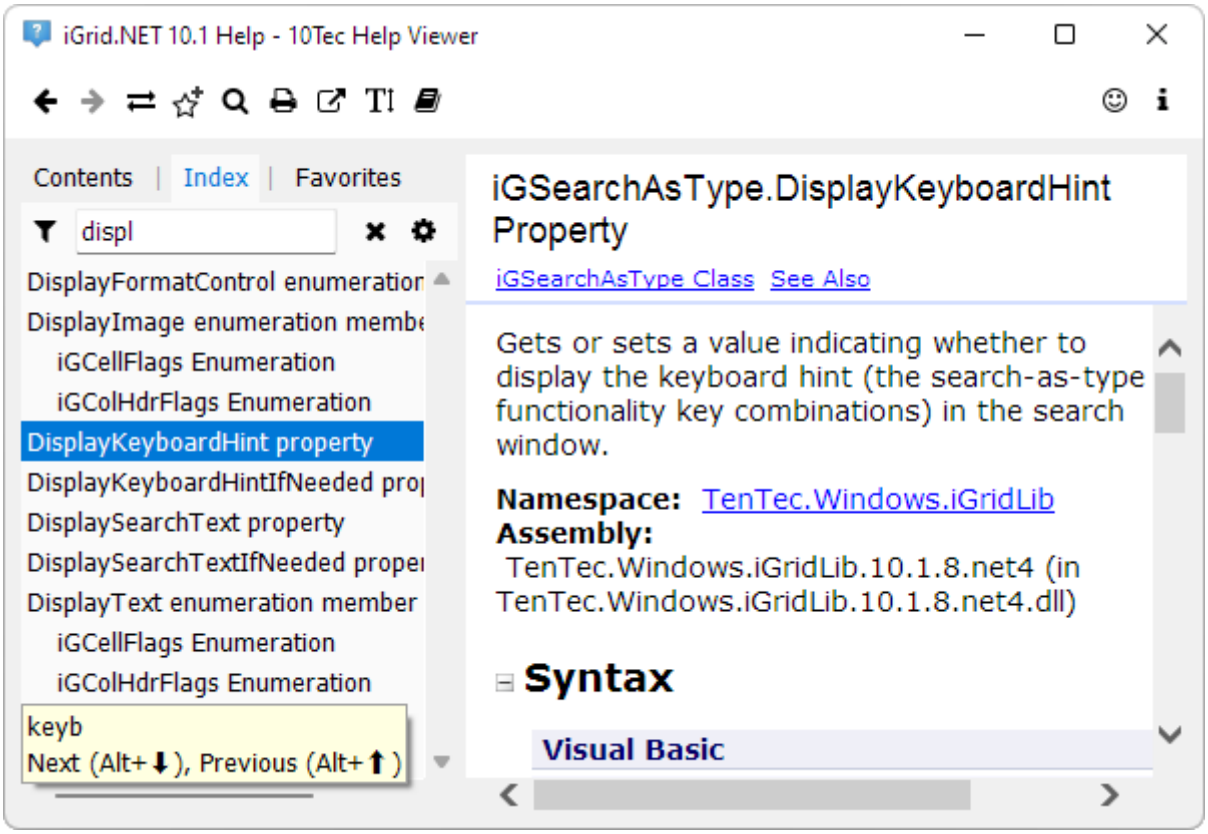
Incremental Search in Topic Lists

The topic lists on the Contents, Index, and Favorites tabs support incremental search. If a list is focused, you can start typing to find topics containing the typed characters:



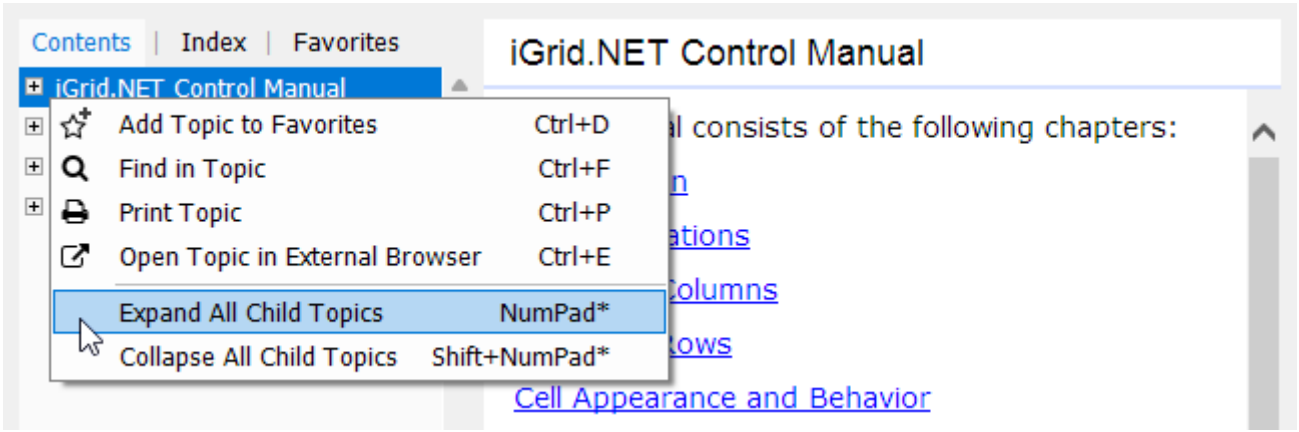
Pay attention to the following points:

- Search is performed only among currently visible topics. This helps to find topics only in a part of documentation you are currently work with. If you need to search on all topics, you can easily expand all topics with the Expand All Nodes command provided by the corresponding mini-button on the vertical scroll bar of the list.
- The list on the Index tab also supports incremental search even when the list is already filtered using its filter box above the list. This lets you apply a kind of a second filter, for example:



Bulk Collapse/Expand Operations in Contents

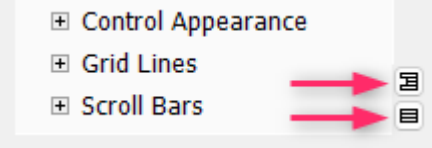
The topic list on the Contents tab provides you with some tools to quickly collapse or expand Contents node. The first of them is the topic context menu with the following commands:



The Expand All Child Topics command can be extremely useful if you want to perform incremental search in all child topics of a root topic (remember, incremental search is performed only in currently visible topics). For example, the full help file for the iGrid.NET control contains 4 root nodes: the first of them is the manual with conceptual topics, and the rest three are reference topics for the corresponding namespaces. You can open (expand) all the manual topics for incremental search using the Expand All Child Topics command.

Pay attention to the fact that the commands expanding or collapsing all child nodes of the selected node can be also performed from the numpad keyboard.

Notice also two additional mini buttons on the vertical scroll bar:



These buttons expand and collapse all nodes in the Contents list regardless of the selected node.

Filter Box for Index

The Index tab has a special text field above the topic list allowing you to filter the list. It is called 'Index filer box' or simply 'filter box' depending on the context.

When you switch to the Index tab, the filter box is automatically selected. This allows you to do the main operation with the list, i.e. filtering, immediately after switching to the Index tab. Note that you can also switch to the Index tab from the keyboard using the Alt+I key combination. This allows you to search for topics on the index tab from the keyboard without touching the mouse after the app has been launched.

Help Viewer is filtering the topic list after entering every new character. The currently selected topic (if any) is not changed until you select a new topic in the list. You can do it with the mouse or using the TAB or ENTER keys. If you press TAB, the input focus is moved to the list and the currently selected item remains unchanged. If the list did not have a selected item, the first item in the filtered list is selected automatically. If you press ENTER in the filter box, the first topic in the filtered list is selected automatically regardless of the item selected previously.

CONFIGURATION FILE

The Help Viewer settings are stored in the TenTec.HelpViewer.exe.config file in the application folder. This is a text file in the XML format. It can be viewed/edited in any text editor. You can find an example of the Help Viewer configuration file in Addendum – see Typical Contents of Configuration File.

There are two categories of settings: global application settings and user settings. They are stored as child elements of the nodes **applicationSettings** and **userSettings** of the configuration file respectively.

All available settings with their possible values are described in this chapter. Note that some settings cannot be changed in the free edition of Help Viewer. To find them, see Limitations of the Free Version.

Application Settings

AppMainFontName

Specifies the name of the font used in the interface of the application.

TYPE	String
POSSIBLE VALUES	Any font name installed in the operating system
DEFAULT VALUE	"Verdana"

This setting affects the font used in interface elements like menus and topic lists on the Contents and Index tabs. The content of topics displayed in the browser is rendered with the own font coded in topic HTMLs.

AppMainFontSize

Specifies the size of the font used in the interface of the application.

TYPE	Floating-point value
POSSIBLE VALUES	Any positive value specifying font size in points
DEFAULT VALUE	9

This setting affects the font used in interface elements like menus and topic lists on the Contents and Index tabs. The content of topics displayed in the browser is rendered with the own font coded in topic HTMLs.

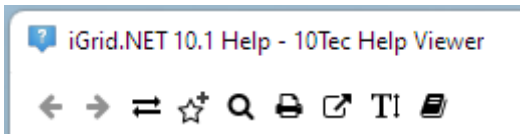
See also the MenuImageSize*dpi and ToolbarImageSize*dpi settings that can be used together with the **AppMainFontSize** settings to adjust the size of elements of the interface.

AppWindowTitleTemplate

Defines the template for the application's form title.

TYPE	String
POSSIBLE VALUES	A string that can contain Common Template Fields
DEFAULT VALUE	"{HelpTitle} - {AppName}"

This setting defines the template for the application's form title. This template is evaluated every time when a new help file is opened. As a rule, the **{HelpTitle}** and **{AppName}** common template fields are used in it. Below is an example of the Help Viewer window title when an iGrid.NET help file is opened and the default title template is used:

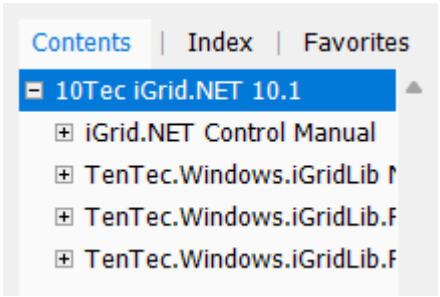
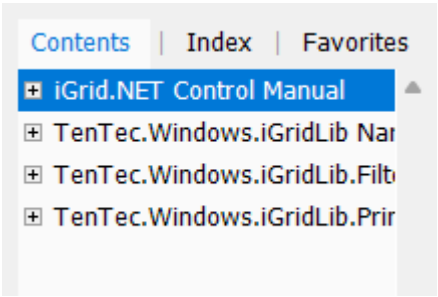


ContentsHideSingleRoot

Defines whether to hide a single root folder in the Contents tab.

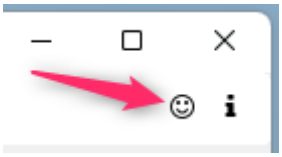
TYPE	Boolean
POSSIBLE VALUES	False, True
DEFAULT VALUE	False
EXAMPLE FOR IGRID	True

Many help files have the only root topic. This **ContentsHideSingleRoot** setting allows you to remove the only root topic from the tree on the Contents tab to save some space. Compare the look of contents when this option is True and False:

FALSE	TRUE
	

FeedbackAllowed

Specifies whether to show the Send Feedback button on the toolbar:



TYPE	Boolean
POSSIBLE VALUES	False, True
DEFAULT VALUE	True

FeedbackEmail

An email address placed in the To filed in a message composed automatically when the user presses the Send Feedback button on the toolbar.

TYPE	String
POSSIBLE VALUES	Any valid email
DEFAULT VALUE	"contact@10tec.com"

HelpFileNameTemplate

Specifies the file name template for the iGrid.NET help files.

TYPE	String
POSSIBLE VALUES	A string that may include {0} and {1} to substitute with the major and minor versions
DEFAULT VALUE	""
EXAMPLE FOR IGRID	iGrid.NET_{0}.{1}_Help

This setting is used solely by the infrastructure related to iGrid.NET help system. The value of this setting defines the template for the iGrid.NET help file name and allows Help Viewer to find the help file for a given version of iGrid.NET. The fields {0} and {1} in the help file template are integer values representing the major and minor version of iGrid.NET respectively.

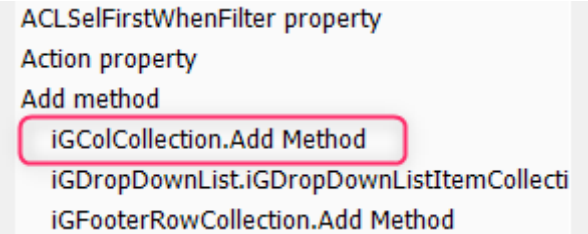
The template specifies the start part of the file name, which allows adding any suffix at the end of file name. For example, the template "iGrid.NET_{0}.{1}_Help" can be used to find a help file name like iGrid.NET_10.1_Help_-_Revision_1.mshc that may include a revision number.

IndexItemDetails

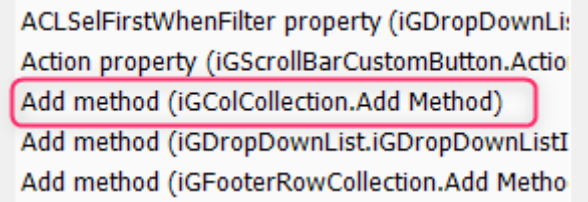
Defines how topic titles in the Index tab are shown in the **ListWithDetails**, **SimpleTree**, and **TreeWithKeywordSplit** views (see the IndexView setting).

TYPE	Enumeration
POSSIBLE VALUES	H1, Title, ElementId
DEFAULT VALUE	Title
EXAMPLE FOR IGRID	ElementId

In the **SimpleTree** and **TreeWithKeywordSplit** views, the value of this setting defines the source of topic titles displayed as sub-nodes for an Index entry:



In the **ListWithDetails** view, the value of this setting defines the source of the topic detail information defined by the template in the IndexItemExtendedTemplate setting:



The table below explains each available option:

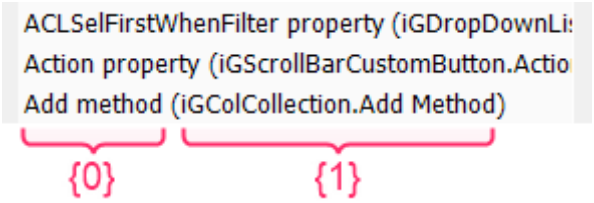
VALUE	DESCRIPTION
H1	The topic title is the contents of the <H1> element of the topic HTML, for example: `<h1>StartFromCurRow Property</h1>`
Title	The topic title is retrieved from the <title> element of the topic HTML, for example: `<title>StartFromCurRow Property</title>`
ElementId	The topic title is retrieved from the page element with the id attribute specified in the TopicTitleElementId setting of the configuration file, for example: `<span id="nsrTitle">iGSearchAsType.StartFromCurRow Property</span>`

IndexItemExtendedTemplate

Defines the look of the Index items in the **ListWithDetails** view (see the IndexView setting).

TYPE	String
POSSIBLE VALUES	A string with the {0} and {1} fields
DEFAULT VALUE	"{0} ({1})"

This setting defines the look of the Index items in the **ListWithDetails** view. The special {0} and {1} fields are replaced with the keyword and details as follows:

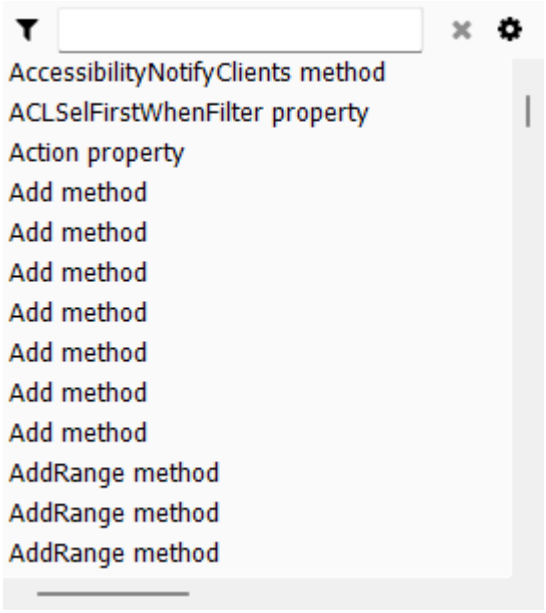
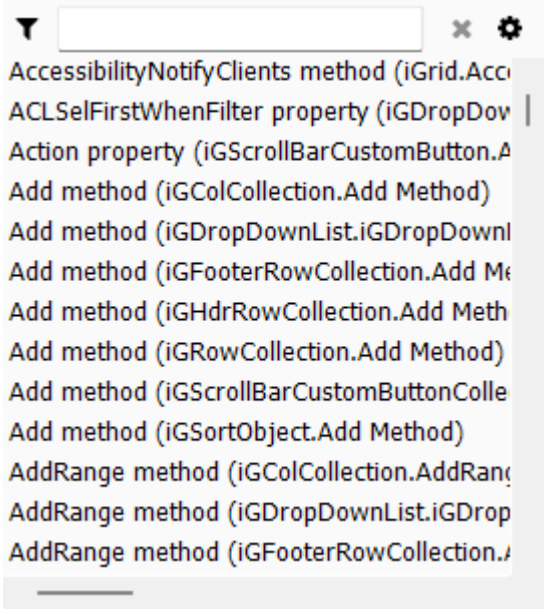


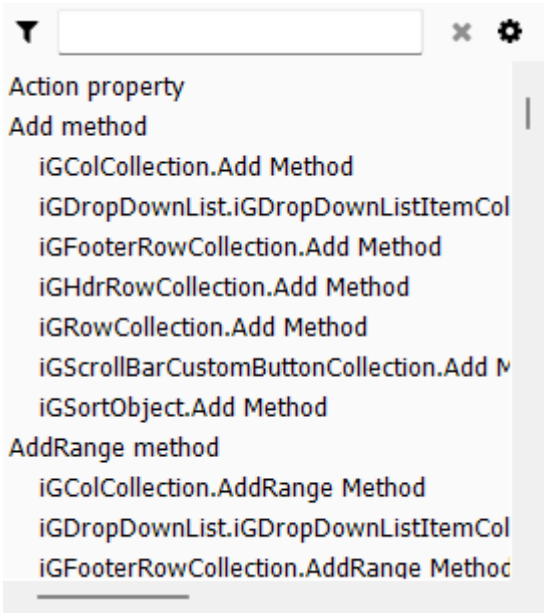
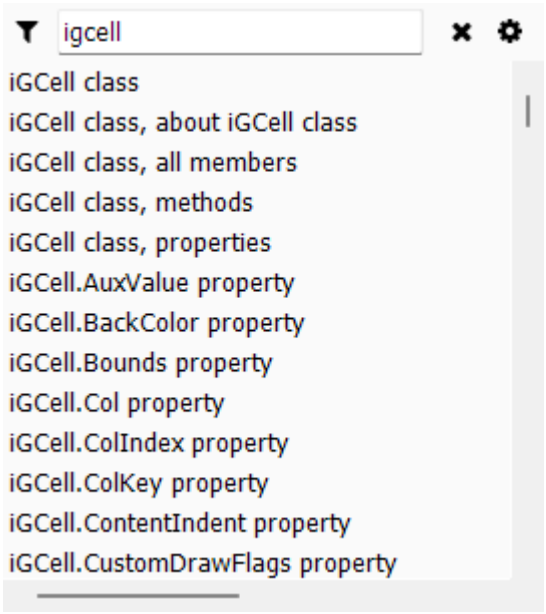
IndexView

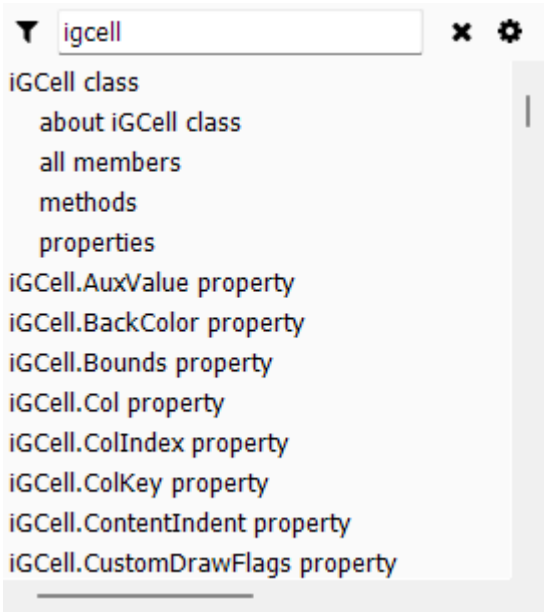
Defines how topics in the Index tab are shown.

TYPE	Enumeration
POSSIBLE VALUES	List, ListWithDetails, SimpleTree, TreeWithKeywordSplit
DEFAULT VALUE	TreeWithKeywordSplit

The table below explains each available option:

VALUE	DESCRIPTION
List	Topics in the Index tab will be shown as a simple list. Each item of this list is a keyword. Below is an example for the Add and AddRange methods. These methods are implemented in several classes and described in several topics, but they all are associated with the "Add method" and "AddRange method" keywords: 
ListWithDetails	Topics in the Index tab will be shown as a list. Each item of the list is a keyword supplemented with details about the topic. The details are determined by the **IndexItemDetails** and **IndexItemExtendedTemplate** settings. 

SimpleTree	In this mode same keywords are grouped in one hierarchical section. The corresponding topic titles are displayed with an indent below the keyword:  Pay attention to the fact that keywords may contain commas, for example: `<meta name="Microsoft.Help.Keywords" content="iGCell class, about iGCell class" />` Such keywords will not be displayed like one hierarchical section in the **SimpleTree** view:  To display them in one hierarchical section, use the **TreeWithKeywordSplit** view.

TreeWithKeywordSplit	This view is an extended version of the **SimpleTree** view. In addition to grouping of same keywords in hierarchical sections, keywords with commas are split and displayed in one hierarchical section as well: 

MenuImageSize*dpi

This group of settings consists of 4 parameters:

- MenuImageSize96dpi
- MenuImageSize144dpi
- MenuImageSize192dpi
- MenuImageSize288dpi

They define the size of images in context and drop-down menus for displays with corresponding DPI values. The size is the width and height of the image in pixels.

TYPE	Integer
POSSIBLE VALUES	16, 24, 32, 48
DEFAULT VALUES	MenuImageSize96dpi: 16 MenuImageSize144dpi: 24 MenuImageSize192dpi: 32 MenuImageSize288dpi: 48

Help Viewer determines the size of menu images on a particular display as follows:

```
DeviceDpiRatio = DeviceDpi / 96;
if (DeviceDpi <= 96 || DeviceDpiRatio < 1.5)
    return MenuImageSize96dpi;
else if (DeviceDpiRatio < 2)
```

```
    return MenuImageSize144dpi;
else if (DeviceDpiRatio < 3)
    return MenuImageSize192dpi;
else
    return MenuImageSize288dpi;
```

You can use this group of **MenuImageSize** settings to increase the size of images in drop-down menus for more comfortable use. The size of the check mark in the text size menu or help file list is also governed by this setting.

See also the ToolbarImageSize*dpi and AppMainFontSize settings that can be used together with the **MenuImageSize** settings to adjust the size of elements of the interface.

ToolbarImageSize*dpi

This group of settings consists of 4 parameters:

- ToolbarImageSize96dpi
- ToolbarImageSize144dpi
- ToolbarImageSize192dpi
- ToolbarImageSize288dpi

They define the size of toolbar images for displays with corresponding DPI values. The size is the width and height of the image in pixels.

TYPE	Integer
POSSIBLE VALUES	16, 24, 32, 48
DEFAULT VALUES	ToolbarImageSize96dpi: 16 ToolbarImageSize144dpi: 24 ToolbarImageSize192dpi: 32 ToolbarImageSize288dpi: 48

Help Viewer determines the size of toolbar images on a particular display as follows:

```
DeviceDpiRatio = DeviceDpi / 96;
if (DeviceDpi <= 96 || DeviceDpiRatio < 1.5)
    return ToolbarImageSize96dpi;
else if (DeviceDpiRatio < 2)
    return ToolbarImageSize144dpi;
else if (DeviceDpiRatio < 3)
    return ToolbarImageSize192dpi;
else
    return ToolbarImageSize288dpi;
```

You can use this group of **ToolbarImageSize** settings to increase the size of toolbar buttons for more comfortable use.

See also the MenuImageSize*dpi and AppMainFontSize settings that can be used together with the **ToolbarImageSize** settings to adjust the size of elements of the interface.

TopicAllowJSFiles

Allows external JavaScript files in topic HTMLs.

TYPE	Boolean
POSSIBLE VALUES	False, True
DEFAULT VALUE	True

When this setting is set to False, JavaScript files will not be loaded and executed when topics are opened in the topic browser. Note that this setting affects only scripts located in external JS files. Internal scripts inside HTML pages are not affected by this setting.

TopicAllowMSHelpLinks

Specifies whether to convert “mshelp:link” HTML elements into bold text.

TYPE	Boolean
POSSIBLE VALUES	False, True
DEFAULT VALUE	True
EXAMPLE FOR IGRID	False

Specific “mshelp:link” HTML elements may not work in Help Viewer. The problem is that such an element looks like a real hyperlink, but nothing happens when the user clicks it. To avoid misleading the user, you can convert such links into non-clickable bold text with the help of this setting.

TopicBuiltWithSandcastle

Defines whether Help Viewer uses a special heuristic to provide human-readable page titles.

TYPE	Boolean
POSSIBLE VALUES	False, True
DEFAULT VALUE	False
EXAMPLE FOR IGRID	True

If topic HTMLs were generated by Sandcastle, topic titles are stored in the element with the id attribute set to “nsrTitle”. However, they are not suitable for reading

by the user because they contain language-specific elements dynamically made visible or invisible by page scripts, for example:

```
<span id="nsrTitle">MyClass<span class="languageSpecificText"><span
class="cs">.</span><span class="vb">.</span><span class="cpp">::</span><span
class="nu">.</span><span class="fs">.</span></span>Item Property </span>
```

Even if we read only the inner text of this element, we will get "MyClass..::..Item Property". The heuristics enabled by the **TopicBuiltWithSandcastle** setting converts this into "MyClass.Item Property".

Below is another more complex example for a generic member:

```
<span id="nsrTitle">Class1<span class="languageSpecificText"><span
class="cs">&lt;</span><span class="vb">(Of </span><span
class="cpp">&lt;</span><span class="nu">(</span><span
class="fs">&lt;'</span></span><span class="typeparameter">T</span><span
class="languageSpecificText"><span class="cs">&gt;</span><span
class="vb">)</span><span class="cpp">&gt;</span><span class="nu">)</span><span
class="fs">&gt;</span></span><span class="languageSpecificText"><span
class="cs">.</span><span class="vb">.</span><span class="cpp">::</span><span
class="nu">.</span><span class="fs">.</span></span>Item Property </span>
```

If we read the text contents of this element, we will get "Class1<(Of <(<'T>)>)>..::..Item Property". The heuristics enabled with the **TopicBuiltWithSandcastle** setting yields "Class1(T).Item Property" for it.

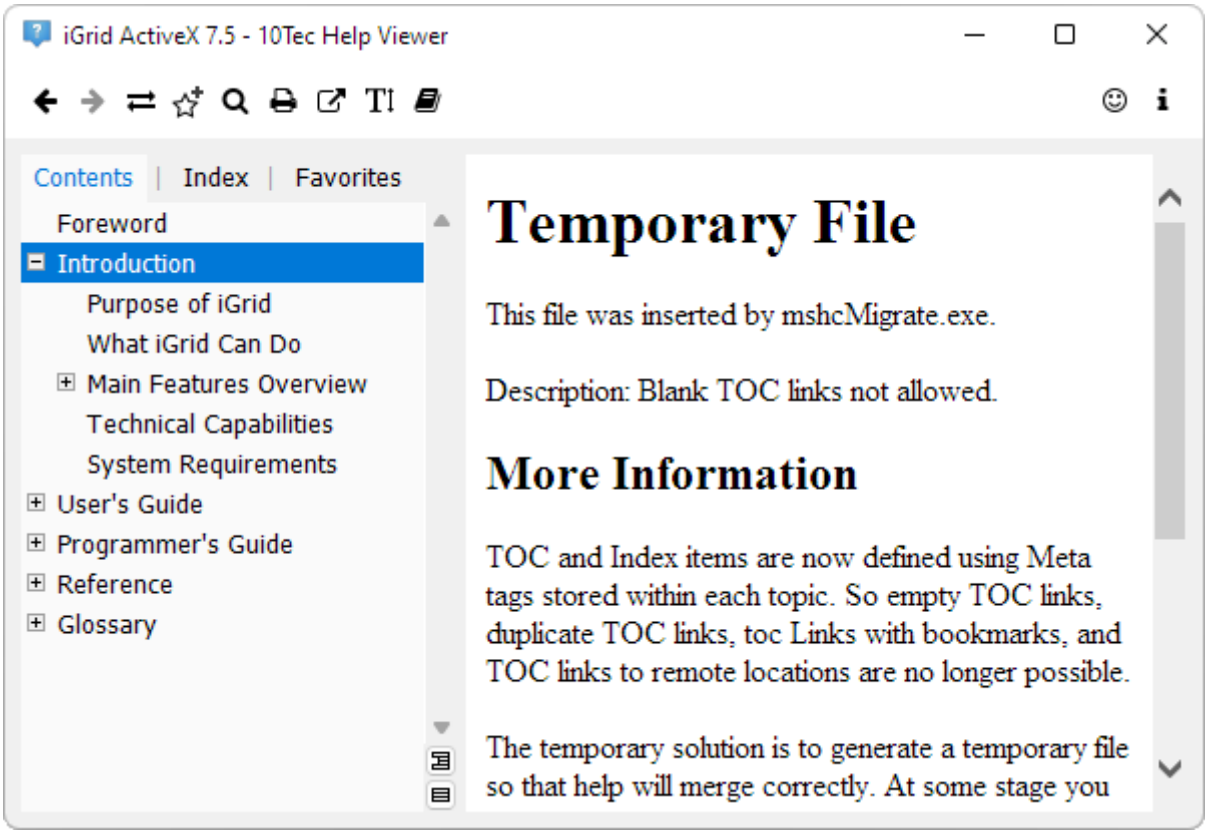
TopicFoldersToIgnore

Defines the list of help archive folders to skip.

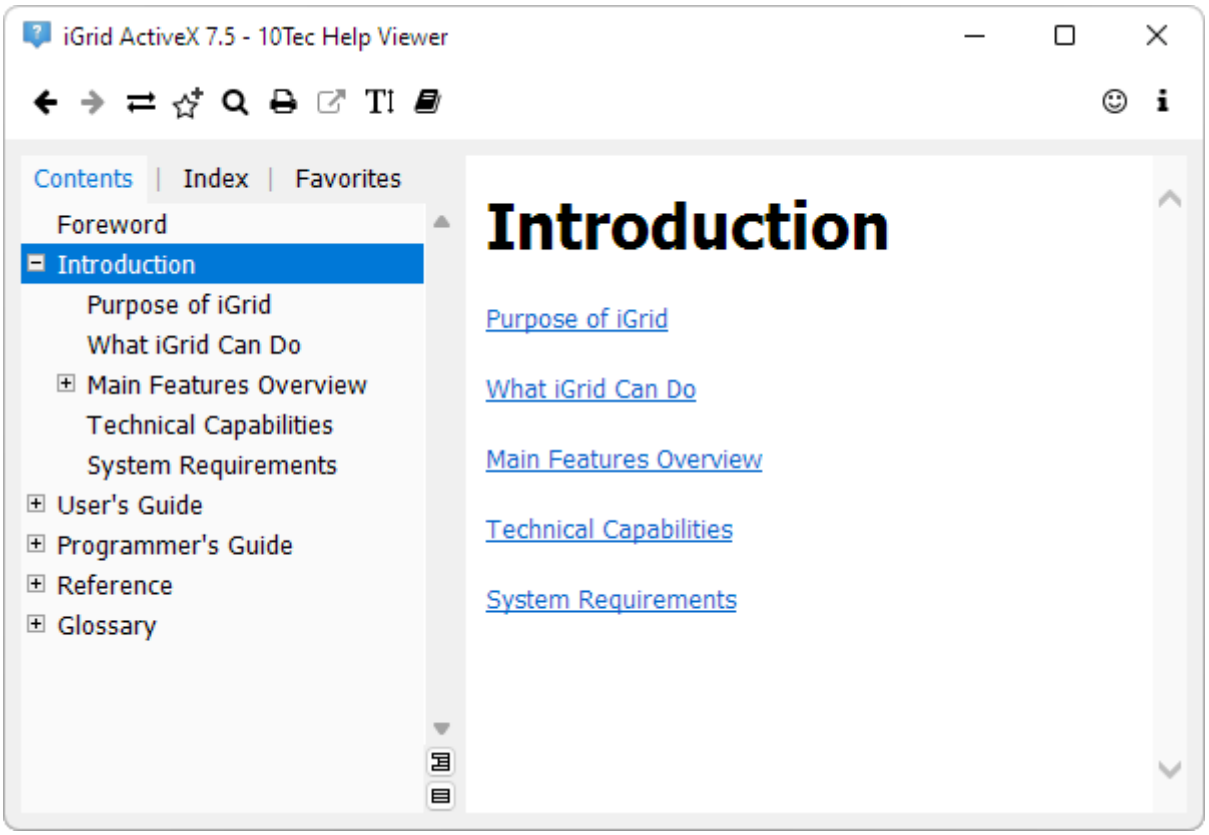
TYPE	String
POSSIBLE VALUES	A comma-separated list of folder names or an empty string
DEFAULT VALUE	""

All HTML files located in the folders specified in the **TopicFoldersToIgnore** setting will be ignored by Help Viewer. If the user selects a topic from the ignored folder on the Contents tab, they will see an automatically generated page with the list of child topics.

This setting is very useful for MSHC files generated automatically from CHM files with the Helpware mshcMigrate utility. This utility creates special temporary files in the resulting MSHC because of the differences in the CHM and MSHC help archive formats. The contents of these topics may not be suitable for viewing by the end user, especially in a commercial solution:



All these cases must be processed additionally after conversion from CHM to MSHC before deploying to the end user. The **TopicFoldersToIgnore** setting helps to avoid doing this tedious work every time after conversion. mshcMigrate stores all temporary files like on the picture above in a special folder named "_AutoGenerate". If you specify this string (without quotes) in the **TopicFoldersToIgnore** setting, Help Viewer will display friendly pages like the following one instead of ignored temporary topics:



TopicHiddenClasses

Defines the list of HTML classes to hide.

TYPE	String
POSSIBLE VALUES	A comma-separated list of classes or an empty string
DEFAULT VALUE	“”
EXAMPLE FOR IGRID	toggle

The classes specified in this setting will be hidden by Help Viewer when a topic is displayed in the browser. For example, if “toggle” is specified, all HTML elements with “toggle” in the “class” attribute will be hidden. Below is an example of such an element:

```
<img class="toggle" src="icons/collapse_all.gif" />
```

TopicHiddenIds

Defines the list of HTML element identifiers to hide.

TYPE	String
POSSIBLE VALUES	A comma-separated list of identifiers or an empty string
DEFAULT VALUE	“”
EXAMPLE FOR IGRID	topTable,footer,gradientTable,headerTableRow1

The HTML element with the specified identifiers will be hidden by Help Viewer when a topic is displayed in the browser. For example, if “topTable” is specified here, the HTML element with “topTable” in the “id” attribute will be hidden:

```
<table id="topTable" cellspacing="0" cellpadding="0">
```

TopicInjectCopyCode

Specifies whether to use own script to copy code from code snippet blocks generated by SandCastle.

TYPE	Boolean
POSSIBLE VALUES	False, True
DEFAULT VALUE	False
EXAMPLE FOR IGRID	True

This setting is useful for help files generated by SandCastle because its script to copy code from code snippet blocks may not work in Help Viewer. SandCastle’s script is used

in HTML "span" elements with the class name "copyCode". If the **TopicInjectCopyCode** setting is set to True, Help Viewer will change the behavior of these HTML elements by attaching its own script to copy code into the clipboard.

TopicTitleElementId

Specifies the identifier of the HTML element from which Help Viewer retrieves topic titles when the IndexItemDetails setting is set to **ElementId**.

TYPE	String
POSSIBLE VALUES	A string representing an HTML element identifier
DEFAULT VALUE	""
EXAMPLE FOR IGRID	"nsrTitle"

This option is useful for help files with topic titles stored not in the <H1> or <title> element, for example:

```
<span id="nsrTitle">iGFilterAppliedEventArgs Members</span>
```

Such topic titles can be generated by help compilers like Sandcastle. Help Viewer provides an additional option allowing you to enable post-processing of such titles to get human readable titles generated by Sandcastle. For more info, see the TopicBuiltWithSandcastle setting.

UILanguage

Specifies the language of the Help Viewer interface.

TYPE	String
POSSIBLE VALUES	A two-character language code or the special "auto" value
DEFAULT VALUE	"auto"

The interface of Help Viewer can be localized. This setting allows you to select the language of the user interface. When Help Viewer starts, it loads the corresponding localization if it exists or uses the default built-in English interface strings.

You can use the special "auto" value in this setting. In this case Help Viewer will use the operating system's user interface language.

For a more detailed description of the localization process and the list of possible language codes, read Chapter User Interface Localization.

User Settings

All options in this section are automatically updated by the application after they are changed by the user. Changed values of these settings are not stored in the original

config file. They are stored in the config file deeply hidden by the operating system. Each user of the operating system has own copy of these values.

The TenTec.HelpViewer.exe.config file contains only the default values for the options of this section.

AppWindowLocation

Specifies the top left corner of the application window on the desktop.

TYPE	Two integer values separated by a comma
POSSIBLE VALUES	The first value is the position of the left border. The second value is the position of the top border. The values are in pixels.
DEFAULT VALUE	0, 0

AppWindowSize

Specifies the width and height of the application window.

TYPE	Two integer values separated by a comma
POSSIBLE VALUES	The first value is width, the second value is height. The values are in pixels.
DEFAULT VALUE	0, 0

If at least one of these values equals zero, Help Viewer will determine the size and location of its window automatically.

AppWindowState

Specifies the default main form state (minimized, maximized, or normal).

TYPE	Enumeration
POSSIBLE VALUES	Normal, Minimized, Maximized
DEFAULT VALUE	Normal

NavigationPaneWidth

Specifies the width of navigation pane.

TYPE	Integer
POSSIBLE VALUES	Any positive integer value or zero
DEFAULT VALUE	0

This setting specifies the width of the navigation pane containing the Contents, Index, and Favorites tab when the application starts for the first time. The zero value means that the width will be determined automatically to show these 3 tabs.

UpgradeRequired

Specifies whether to upgrade configuration options from the previous version of the application.

TYPE	Boolean
POSSIBLE VALUES	False, True
DEFAULT VALUE	True

This setting is processed by Help Viewer and should not be changed manually. It is used to copy user settings from the previous version of the application if it was updated to a newer version. Once the configuration settings have been updated, this option is automatically changed to False.

COMMAND LINE PARAMETERS

General Information

Help Viewer supports the following command line syntax:

```
TenTec.HelpViewer.exe [<HelpFileName>] [-<parameter> …]
```

You can specify one or more parameters separated with spaces when launching the application, for example:

```
TenTec.HelpViewer.exe -WatchFolder -WatchNoDel
```

If a parameter does not start with "-", it is treated as the help file name to open. All other parameters starting with "-" are used to modify the application behavior. They are also called “switches” and fall into two categories: main switches and switches used for debug purpose (service switches). They are described below in the corresponding sections.

<HelpFileName> Parameter

The **<HelpFileName>** argument is the file name of the help file to open in the application when it starts. This should be only a file name without path. It will be used to find the help file in the help library folder. If the specified file does not exist, Help Viewer will start without loading any help file.

If **<HelpFileName>** is not specified, Help Viewer will try to load the last viewed file. If it is not possible, the first help file in the help library folder (if any) will be loaded.

Main Switches

-LibFolder

Allows you to specify a help library folder that differs from the application folder. The help library folder is specified after the "=" character, for example:

```
TenTec.HelpViewer.exe -LibFolder=D:\CustomHelpLib
```

If the path to the help library folder contains spaces, it must be enclosed into quotes:

```
TenTec.HelpViewer.exe -LibFolder="D:\Custom Help Library"
```

If the folder path ends with a backslash, it must be escaped with another backslash:

```
TenTec.HelpViewer.exe -LibFolder="D:\Custom Help Library\\"
```

-LocalizationSample

Creates a localization sample that can be used as a template to localize the application. This process is described in greater details in Chapter User Interface Localization Localization.

If this switch has been specified, Help Viewer does not show its window and finishes execution after creating the localization sample.

-WatchFolder

This command line parameter defines whether to watch a folder for Help Viewer command files. When a command file appears in the watched folder, the application automatically executes the command described in it. After the command has been executed, the command file is deleted unless the **-WatchNoDel** command line parameter was specified.

Help Viewer command files are special XML files with the .hvcmd extension. They describe commands for the application. For example, you can specify the help file and optionally the topic in it to open, custom configuration settings for the opened help file, and the like. Help Viewer command files are described in Chapter Command Files.

You can specify the folder to watch after "=" like this:

```
TenTec.HelpViewer.exe -WatchFolder=C:\SomeFolder
```

If the path to the folder contains spaces, it must be enclosed into quotes:

```
TenTec.HelpViewer.exe -WatchFolder="C:\Another Folder"
```

If the folder path ends with a backslash, it must be escaped with another backslash:

```
TenTec.HelpViewer.exe -WatchFolder="C:\Another Folder\\"
```

If no folder is specified, the application watches its own folder:

```
TenTec.HelpViewer.exe -WatchFolder
```

If the **-WatchFolder** parameter was specified, Help Viewer does not load any help file even if the **<HelpFileName>** argument was specified.

-WatchNoDel

This parameter specifies that the application doesn't delete HVCMD files after commands have been executed. You can use this command for testing purposes.

Service Switches

-ExecCmd

This parameter specifies a command or a list of commands to execute after application start. A command name or commands are specified after "=", for example:

```
TenTec.HelpViewer.exe -ExecCmd=OpenTempFolder
```

If you need to run several commands, use a comma separated list in quotes:

```
TenTec.HelpViewer.exe -ExecCmd="OpenTempFolder,OpenLogFile"
```

The available service commands are listed below:

COMMAND	DESCRIPTION
OpenAppFolder	Opens application folder in the File Explorer.
OpenHelpFile	Opens a help file from the hard drive. The file is selected in the system Open File dialog.
OpenLogFile	Opens log file in default application assigned to edit ".log" files.
OpenTempFolder	Opens the application's temporary folder in the File Explorer. This temporary folder is used by Help Viewer for storing temporary data, such as unpacked viewed HTML pages.

-Log

This parameter specifies whether to output detailed technical information about application execution into a log file. The output is saved to the file TenTec.HelpViewer.log in the application folder.

If an error or exception occurs during application execution, it is automatically logged regardless of whether the **-Log** parameter was specified.

COMMAND FILES

General Information

Command files let you to control the application the way you need. As a rule, they are used to control Help Viewer from another application. One of good examples of this is the context help opened when the user presses F1 in an application. In this case command files can be used to open the corresponding topic in Help Viewer to provide context-sensitive help.

The key aspect in the command files functionality is that they allow you to control Help Viewer without relaunching it. In this case Help Viewer works as a single-instance application and executes command files as they appear on a hard drive.

To turn this mode on, specify the **-WatchFolder** switch when launching Help Viewer. After that Help Viewer will be watching for command files in a specified folder and will execute them as they appear. This folder is specified after the **-WatchFolder** argument.

Command files are processed one by one. A command file is deleted after it has been processed unless the **- WatchNoDel** switch was specified in the command line.

Command files are special XML files with the .hvcmd extension. Below is an example of such a file:

```
<HelpContextAttributes version="1.0">
    <Key name="10tec-params">
        <Item>HelpFileName=SamplePanda.mshc</Item>
        <Item>TopicAllowJSFiles=False</Item>
    </Key>
    <Key name="keyword">
        <Item>topic2</Item>
    </Key>
</HelpContextAttributes>
```

A Help Viewer command file has the root node **<HelpContextAttributes>** in which one or more **<Key>** child nodes (sections) reside. To use Help Viewer's features from another app, you will create the **Key** sections with the names “10tec-params” and “keyword”. They are described in greater detail below.

A side note regarding the format of Help Viewer command files. The format of this file is based on the value returned by the Window.DTE.ContextAttributes property of the Visual Studio Extensibility Interface. It is used by 10Tec Help Viewer to provide context-sensitive help for iGrid.NET. An example of such a file can be found in Chapter Addendum: Examples of Files.

Actually the **Key** section with the name "10tec-params" is an extension for the Visual Studio context attributes collection. It was developed to control Help Viewer from any application but not only from Visual Studio. It allows developers to specify help files and topic to open in a friendly and compact format.

Command file sections

Section: 10tec-params

This section can contain the following options and commands to customize the application behavior:

COMMAND	DESCRIPTION
HelpFileName	The file name of the help file to load. The help file must exist in the application folder. The file name must not include the path, only the file name itself.
OpenTopicId	Optional string parameter. If specified, the application tries to find the topic to open searching for the specified value in the "content" attribute of the "Microsoft.Help.Id" meta tag in topic HTMLs: `<meta name="Microsoft.Help.Id" content="P:TenTec.Windows.iGridLib.iGrid.Cols" />`
OpenTopicUri	Optional string parameter. If specified, the application tries to find the topic to open using the specified value as a relative path to the topic inside the MSHC help archive. Below is an example: `html/topic1subtopic1.htm`
SelectContentsNode	Optional Boolean parameter (True or False). Specifies whether to find and select the specified topic in the table of contents tree on the **Contents** tab. If this parameter is not specified, the True value is in effect.
TopicAllowJSFiles	Optional Boolean parameter (True or False). Allows you to override the **TopicAllowJSFiles** option in the Help Viewer configuration file. See Chapter Configuration Options for description.
TopicHiddenClasses	Optional Boolean parameter (True or False). Allows you to override the **TopicHiddenClasses** option in the Help Viewer configuration file. See Chapter Configuration Options for description.
TopicHiddenIds	Optional Boolean parameter (True or False). Allows you to override the **TopicHiddenIds** option in the Help Viewer configuration file. See Chapter Configuration Options for description.

Section: keyword

The keyword section lets you specify one or more keywords to find a topic and open it in the application. If you specify more than one keyword, do it like this:

```
<Key name="keyword">
```

```
    <Item>topic2</Item>
    <Item>topic3</Item>
</Key>
```

When Help Viewer finds the **keyword** section in the command file, it tries to find the topic using the specified keywords one by one. If the topic is found, all remaining keywords are ignored.

Keywords are searched in the "Microsoft.Help.F1" meta tags listed in topic HTMLs:

```
<meta name="Microsoft.Help.F1"
content="TenTec.Windows.iGridLib.iGrid.TextRenderingHint" />
```

Use of Command File Sections

Both **keyword** and **10tec-params** sections are optional. However, in the vast majority of cases you need to specify the help file to open, and the **10tec-param** section will be present in your command files because you can specify the help file only with the **HelpFileName** parameter from this section.

As for the topic to open in Help Viewer, you can do it 3 different ways:

1) With the **OpenTopicId** parameter in the **10tec-params** section.
2) With the **OpenTopicUri** parameter in the **10tec-params** section.
3) With the child item(s) of the **keyword** section.

Use whatever method that works for you. Note that you can specify the topic to open using several methods listed above, but only one of them will be used to find the topic. The order in which Help Viewer checks the specified methods to find the topic correspond to the aforementioned order.

Note that you can simply open a help file without selecting a specific topic in it. To do that, does not specify a topic to open using one of the methods listed above, for example:

```
<HelpContextAttributes version="1.0">
    <Key name="10tec-params">
        <Item>HelpFileName=SamplePanda.mshc</Item>
    </Key>
</HelpContextAttributes>
```

Guide on command files for C# developers

This topic describes the recommended process of automating Help Viewer from a .NET application. The approach implements the single-instance application behavior, which is the behavior of a help system expected by users in most cases. The approach provides the fastest and most optimal way to control Help Viewer from another application.

For simplicity, we imply that the Help Viewer executable file (TenTec.HelpViewer.exe) is located in the same folder in which the current executable is being executing. Command files will be created in the same folder.

First of all, import the following namespaces to make our code snippets shorter:

```
using System.Diagnostics;
using System.IO;
```

Declare the following variable related to the Help Viewer process:

```
Process HelpViewerProcess;
```

This object must exist between launches of Help Viewer, so that the best accessibility scope for it is the form or module level.

Now let's write a method to open the topic with a given Id in a given help file. The method definition may look like the following:

```
private void OpenHelpTopic(string fileName, string topicId)
```

The first task of this method is to generate the corresponding contents of the future command file for Help Viewer:

```
string cmd =
    "<HelpContextAttributes version=\"1.0\">" +
    "<Key name=\"10tec-params\">" +
    $"<Item>HelpFileName={fileName}</Item>" +
    $"<Item>OpenTopicId={topicId}</Item>" +
    "</Key>" +
    "</HelpContextAttributes>";
```

Having it, we can create the command file on the disk:

```
string guid = Guid.NewGuid().ToString();
string TmpFileName = guid + ".hvcmdtmp";
string CmdFileName = guid + ".hvcmd";
File.WriteAllText(TmpFileName, cmd);
File.Move(TmpFileName, CmdFileName);
```

Pay attention to the fact that the Help Viewer command file is first created with an extension different from .hvcmd. If we created an HVCMD file without this temporary file, the Windows API functions would create an empty .hvcmd file in the beginning of the File.WriteAllText() call. As a result, it could be processed by the running instance of Help Viewer and the application would do nothing because the command file contents were empty on that moment.

The last task of the OpenHelpTopic() method is to launch Help Viewer. We need to check if we already have a running instance of TenTec.HelpViewer.exe, and if not – launch it:

```
if (HelpViewerProcess == null || HelpViewerProcess.HasExited)
{
    HelpViewerProcess = Process.Start(
        "TenTec.HelpViewer.exe", "-WatchFolder");
}
```

The Help Viewer executable file is launched with the **-WatchFolder** parameter to turn on command file processing mode. The **-WatchFolder** parameter is used without specifying a folder to watch, which means that Help Viewer will be watching its own folder.

There are two important points in this implementation. The first of them is related to the logic of processing command files. If there is no running instance of Help Viewer, it will

be launched after we created a command file for it. In this case Help Viewer will process the created command file immediately after initialization. But when we create a new command file the next time while having a running instance of Help Viewer, the command file will be processed right after its creation by the running instance of Help Viewer. This will happen before executing the code that checks whether Help Viewer is running that does not instantiate a new Help Viewer process in this situation.

The second important point is related to the activation of the Help Viewer window. The user expects that the help application must appear above all other windows when the user wants to get help. This automatically happens when we launch the Help Viewer process because it works like launching a new executable file in Windows. However, if Help Viewer is already running and we do not launch a new process, Help Viewer does its best to activate its window and bring it to the front. The result of this process depends on the architecture of the application calling it, but in the general case the existing Help Viewer window may remain on its place in the Z-order of opened windows and Windows will only flash the taskbar button of Help Viewer to notify the user. This happens because in Windows an application cannot force a window to the foreground while the user is working with another window.

To fix this issue and allow Help Viewer to bring its window to top, we must allow this from the calling process. This is done with the help of the API function called AllowSetForegroundWindow. You can add the declaration of this function next to the declaration of the HelpViewerProcess variable like this:

```
[System.Runtime.InteropServices.DllImport("user32.dll")]
static extern bool AllowSetForegroundWindow(int dwProcessId);
```

And call it at the end of the OpenHelpTopic() method we are building:

```
AllowSetForegroundWindow(HelpViewerProcess.Id);
```

Note that we must do this every time when OpenHelpTopic() is called to allow foreground window change from another process the next time when Help Viewer will process a new command file.

The full source code of the OpenHelpTopic() method an example of its call from a WinForms form can be found in Addendum - see Help Viewer Call Sample Form Module.

USER INTERFACE LOCALIZATION

You can localize the Help Viewer interface. The **-LocalizationSample** command line parameter is used for that:

```
TenTec.HelpViewer.exe -LocalizationSample
```

After executing this command, Help Viewer creates the following set of template files you can use as a starting point for localization:

FILE	DESCRIPTION
error-nomshc.html	This web page is shown when there are no help files in the application folder.
error-nosuchmshc.html	This web page is shown when there is no specific help file with a particular name.
error-notopic.html	This web page is shown when there is no such topic in the help file.
ui-strings.xml	This file contains all strings used in the interface and feedback email.

These files are created in the Localization\en subfolder. Help Viewer also creates the LanguageCodes.txt file in the Localization subfolder. This text file contains all supported language codes together with language locale names.

To create the localized files for a language, first find the corresponding two-letter language code in LanguageCodes.txt. For example, if you want to create a German localization, find the following record for it:

```
TwoLetterISOLanguageName: de
                    Name: de
             EnglishName: German
             DisplayName: German
              NativeName: Deutsch
```

The code is "de" next to the **TwoLetterISOLanguageName** field. Create the subfolder Localization\de and copy the template files into that folder. Now you can translate the contents of these files to your language.

To use the localized HTML files and interface strings, specify the corresponding language code in the **UILanguage** setting of Help Viewer (see Chapter Configuration Options).

The localization template HTML files and the string constants related to the 'send feedback' functionality may contain Common Template Fields enclosed in curly brackets. These parameters are changed to the corresponding values automatically by Help Viewer before these HTMLs and strings are shown to the user. You can also use the **{TopicTitle}** field representing the title of the currently viewed topic in addition to the common template fields.

There are also specific fields used only in localizable HTML file templates:

NAME	DESCRIPTION
{UseDefaultStyles}	This special field is used to change the font name and font size of the HTML file so that the page will look like other interface parts of Help Viewer (the settings from the **AppMainFontName** and **AppMainFontSize** configuration options). Place this field right after the <body> element as it is done in the HTML template files.
{HelpFileName}	Name of the help file. It is used only in the file error-nosuchmshc.html.
{TopicId}	Identifier of the topic. It is used only in the file error- notopic.html.

Note that it is also possible to customize the default interface strings and HTML pages if you work with one language. When the application starts, it searches for the Localization\<current language code> subfolder and uses the XML and HTML localization files from it if they are present. Thus, you can do the following to modify the default strings in the interface:

- Create the Localization subfolder Localization\<lang> for your language.
- Copy the ui-strings.xml and *.html files from the Localization\en subfolder into it.
- Edit the files as required.

MISCELANEOUS

Common Template Fields

Strings displayed in various parts of Help Viewer can be changed through the configuration file or localization files. Some of these adjustable strings support fields with dynamic values that will be replaced by Help Viewer with their actual values when these strings are shown to the user. Such fields are enclosed into curly brackets inside traditional strings.

A good example is the **{HelpTitle}** field you can use in the **AppWindowTitleTemplate** configuration setting. This field is replaced with the title of the currently viewed help file in the title of the Help Viewer window.

All common fields that can be used everywhere where fields are supported are listed below:

FIELD	VALUE
AppName	The name of the Help Viewer application ("10Tec Help Viewer").
HelpTitle	The title of the currently opened help file.

Supported Help File Features

- The CHM viewer redistributed with Microsoft Windows and Microsoft Help Viewer redistributed with Microsoft Visual Studio use the system HTML rendering engine based on Internet Explorer 7. 10Tec Help Viewer uses the same rendering engine for compatibility to provide the same look of CHM and MSHC help topics. Thus, topics of all help files viewed in 10Tec Help Viewer must be compatible with Internet Explorer 7.
- Help topics of Microsoft documentation designated for viewing in Microsoft Help Viewer from Visual Studio can be marked as not “self-branded”. This is done with the help of the following flag in topic HTMLs:

  <meta name="Microsoft.Help.SelfBranded" content="false" />

  If a topic is self-branded, it contains all required CSS and JavaScripts for correct rendering in the browser. If not, then the hosting environment (help viewer) must provide the corresponding CSS and JS files.

  10Tec Help Viewer does not provide the proprietary branding package for Microsoft documentation. Thus, if you decide to view Microsoft MSHC help files in 10Tec Help Viewer, some of them may look incorrectly.
- Not all link types from Microsoft's MSHC files are supported. Below are some examples:
  - F1 links:
    <a href="ms-

xhelp:///?method=f1&query=MyTopicKeyword&product=vs&productversion=100&locale=en-us">

- Search queries:
  <a href="ms-xhelp:///?method=search&query=SomeText&ProductVersion=100&Product=vs&PageSize=10&PageNumber=1&locale=en-us">
- Direct links:
  <a href="ms-xhelp:///?C:\\programData\\Microsoft\\Help Library\\store\\dev10.mshc;/VS-logo.gif">
- URLs with the onlineFallbackBase attribute:
  <a href="ms-xhelp:///?Id=VS.dotnet-api-docs%2Fapi%2Faccessibility.iaccpropservices.clearhmenuprops%3Fview%3Dnet-5.0&onlineFallbackBase=https://docs.microsoft.com/en-us/dotnet/api/accessibility.iaccpropservices.clearhmenuprops?view=net-5.0">

Limitations of the Free Version

10Tec Help Viewer is provided in two editions – the free edition for personal use and the paid version for commercial use. Changing some application settings is disabled in the free edition. These settings and their hard-coded values in the free version are listed below:

SETTING	VALUE IN FREE VERSION
AppWindowTitleTemplate	"{HelpTitle} - {AppName}"
FeedbackAllowed	True
FeedbackEmail	contact@10tec.com
TopicFoldersToIgnore	"" (empty string)
UILanguage	"en"

Notes:

- The unchangeable True value of the **FeedbackAllowed** settings means that the Send Feedback button cannot be removed from the toolbar. When the user clicks it, a message to the email from the **FeedbackEmail** setting (always "contact@10tec.com" in the free version) will be created.
- The unchangeable "en" value in the **UILanguage** setting also implies that the localization feature is not available in the free version. The hard-coded English strings will be always used in the interface.

ADDENDUM

Typical Contents of Configuration File

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <configSections>
    <sectionGroup name="applicationSettings"
type="System.Configuration.ApplicationSettingsGroup, System, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" >
      <section name="TenTec.HelpViewer.Properties.Settings"
type="System.Configuration.ClientSettingsSection, System, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />
    </sectionGroup>
    <sectionGroup name="userSettings"
type="System.Configuration.UserSettingsGroup, System, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089">
      <section name="TenTec.HelpViewer.Properties.Settings"
type="System.Configuration.ClientSettingsSection, System, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089"
allowExeDefinition="MachineToLocalUser" requirePermission="false" />
    </sectionGroup>
  </configSections>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
  </startup>
  <System.Windows.Forms.ApplicationConfigurationSection>
    <add key="DpiAwareness" value="PerMonitorV2" />
  </System.Windows.Forms.ApplicationConfigurationSection>
  <applicationSettings>
    <TenTec.HelpViewer.Properties.Settings>
      <setting name="AppMainFontName" serializeAs="String">
        <value>Verdana</value>
      </setting>
      <setting name="AppMainFontSize" serializeAs="String">
        <value>9</value>
      </setting>
      <setting name="AppWindowTitleTemplate" serializeAs="String">
        <value>{HelpTitle} - {AppName}</value>
      </setting>
      <setting name="ContentsHideSingleRoot" serializeAs="String">
        <value>True</value>
      </setting>
      <setting name="FeedbackAllowed" serializeAs="String">
        <value>True</value>
      </setting>
      <setting name="FeedbackEmail" serializeAs="String">
        <value>contact@10tec.com</value>
      </setting>
      <setting name="HelpFileNameTemplate" serializeAs="String">
        <value>iGrid.NET_{0}.{1}_Help</value>
      </setting>
      <setting name="IndexItemDetails" serializeAs="String">
        <value>H1</value>
```

```
</setting>
<setting name="IndexItemExtendedTemplate" serializeAs="String">
  <value>{0} ({1})</value>
</setting>
<setting name="IndexView" serializeAs="String">
  <value>TreeWithKeywordSplit</value>
</setting>
<setting name="MenuImageSize96dpi" serializeAs="String">
  <value>16</value>
</setting>
<setting name="MenuImageSize144dpi" serializeAs="String">
  <value>24</value>
</setting>
<setting name="MenuImageSize192dpi" serializeAs="String">
  <value>32</value>
</setting>
<setting name="MenuImageSize288dpi" serializeAs="String">
  <value>48</value>
</setting>
<setting name="ToolbarImageSize96dpi" serializeAs="String">
  <value>16</value>
</setting>
<setting name="ToolbarImageSize144dpi" serializeAs="String">
  <value>24</value>
</setting>
<setting name="ToolbarImageSize192dpi" serializeAs="String">
  <value>32</value>
</setting>
<setting name="ToolbarImageSize288dpi" serializeAs="String">
  <value>48</value>
</setting>
<setting name="TopicAllowJSFiles" serializeAs="String">
  <value>True</value>
</setting>
<setting name="TopicAllowMSHelpLinks" serializeAs="String">
  <value>False</value>
</setting>
<setting name="TopicFoldersToIgnore" serializeAs="String">
  <value>_AutoGenerate</value>
</setting>
<setting name="TopicHiddenClasses" serializeAs="String">
  <value>toggle</value>
</setting>
<setting name="TopicHiddenIds" serializeAs="String">
  <value>topTable,footer,gradientTable,headerTableRow1</value>
</setting>
<setting name="TopicInjectCopyCode" serializeAs="String">
  <value>True</value>
</setting>
<setting name="TopicTitleElementId" serializeAs="String">
  <value>nsrTitle</value>
</setting>
<setting name="UILanguage" serializeAs="String">
  <value>auto</value>
</setting>
```

```
    </TenTec.HelpViewer.Properties.Settings>
  </applicationSettings>
  <userSettings>
    <TenTec.HelpViewer.Properties.Settings>
      <setting name="AppWindowLocation" serializeAs="String">
        <value>0, 0</value>
      </setting>
      <setting name="AppWindowSize" serializeAs="String">
        <value>0, 0</value>
      </setting>
      <setting name="AppWindowState" serializeAs="String">
        <value>Maximized</value>
      </setting>
      <setting name="NavigationPaneWidth" serializeAs="String">
        <value>0</value>
      </setting>
      <setting name="UpgradeRequired" serializeAs="String">
        <value>True</value>
      </setting>
    </TenTec.HelpViewer.Properties.Settings>
  </userSettings>
</configuration>
```

F1 Context Help Command File Example

```
<?xml version="1.0" encoding="UTF-8"?>
<HelpContextAttributes version="1.0">
    <Key name="keyword">
        <Item>TenTec.Windows.iGridLib.iGrid.Cols</Item>
        <Item>VS.TextEditor</Item>
        <Item>VS.Ambient</Item>
    </Key>
    <Key name="product">
        <Item>VS</Item>
        <Item>csharp</Item>
        <Item>VS</Item>
        <Item>C#</Item>
    </Key>
    <Key name="devlang">
        <Item>csharp</Item>
    </Key>
    <Key name="subtype">
        <Item>Form</Item>
    </Key>
    <Key name="item">
        <Item>cs</Item>
        <Item>Project</Item>
    </Key>
    <Key name="targetframeworkmoniker">
        <Item>.NETFramework,Version=v4.7.2</Item>
    </Key>
    <Key name="assemblyreferences">
        <Item>TenTec.Windows.iGridLib.10.1.0.net4=10.1</Item>
        <Item>System.Xml=4.0</Item>
        <Item>System.Windows.Forms=4.0</Item>
```

```
        <Item>System.Net.Http=4.2</Item>
        <Item>System.Drawing=4.0</Item>
        <Item>System.Deployment=4.0</Item>
        <Item>System.Data=4.0</Item>
        <Item>Microsoft.CSharp=4.0</Item>
        <Item>System.Data.DataSetExtensions=4.0</Item>
        <Item>System.Xml.Linq=4.0</Item>
        <Item>System.Core=4.0</Item>
        <Item>System=4.0</Item>
    </Key>
    <Key name="uniquefiles">
        <Item>settings</Item>
        <Item>cs</Item>
        <Item>config</Item>
        <Item>resx</Item>
    </Key>
    <Key name="sourcecontrol">
        <Item>FALSE</Item>
    </Key>
    <Key name="projtype">
        <Item>LocalProj</Item>
    </Key>
    <Key name="project">
        <Item>Exe</Item>
    </Key>
    <Key name="shellmode">
        <Item>Design</Item>
    </Key>
    <Key name="solution">
        <Item>Any</Item>
        <Item>Single</Item>
    </Key>
    <Key name="lcid">
        <Item>1033</Item>
    </Key>
    <Key name="applicationid">
        <Item>VisualStudio</Item>
    </Key>
</HelpContextAttributes>
```

C# Sample of Help Viewer Call

```
using System;
using System.Diagnostics;
using System.IO;
using System.Runtime.InteropServices;
using System.Windows.Forms;

namespace CallHelpViewerSample
{
    public partial class Form1 : Form
    {
        [DllImport("user32.dll")]
        static extern bool AllowSetForegroundWindow(int dwProcessId);
```

```
        Process HelpViewerProcess;

        public Form1()
        {
            InitializeComponent();
        }

        private void OpenHelpTopic(string fileName, string topicId)
        {
            string cmd =
                "<HelpContextAttributes version=\"1.0\">" +
                "<Key name=\"10tec-params\">" +
                $"<Item>HelpFileName={fileName}</Item>" +
                $"<Item>OpenTopicId={topicId}</Item>" +
                "</Key>" +
                "</HelpContextAttributes>";

            string guid = Guid.NewGuid().ToString();
            string TmpFileName = guid + ".hvcmdtmp";
            string CmdFileName = guid + ".hvcmd";
            File.WriteAllText(TmpFileName, cmd);
            File.Move(TmpFileName, CmdFileName);

            if (HelpViewerProcess == null ||
HelpViewerProcess.HasExited)
            {
                HelpViewerProcess = Process.Start(
                    "TenTec.HelpViewer.exe", "-WatchFolder");
            }

            AllowSetForegroundWindow(HelpViewerProcess.Id);
        }

        private void buttonCallHelpViewer_Click(object sender, EventArgs
e)
        {
            OpenHelpTopic("SamplePanda.mshc", "topic2_id");
        }
    }
}
```