madcap
software

USER GUIDE

MADCAP FLARE 2024

Context-Sensitive Help

MadCap Software
9171 Towne Center Drive, Suite 335
San Diego, California 92122
858-320-0387
www.madcapsoftware.com

THIS PDF WAS CREATED USING MADCAP FLARE.

CONTENTS

CHAPTER 1

CHAPTER 2

CHAPTER 3

CHAPTER 4

APPENDIX

CHAPTER 1

Introduction

Supported In:

HTML 5 | ECLIPSE HELP | MS HTML HELP | WEB HELP | WEB HELP PLUS

Context-sensitive Help (CSH) is a way to tie your topics or micro content with links or specific areas of a software interface. For example, when users click a particular part of the interface (e.g., a Help button), the topic or micro content pertaining to it opens.

General Information

- "Who is Involved?" on page 9
- "Planning Context-Sensitive Help" on page 10
- "Header Files" on page 11
- "Alias Files" on page 13
- "Context-Sensitive Help Features Per Output Type" on page 14

Process (If Author Provides Header File)

1. Author Add a header file to the project.
2. Author Add an alias file to the project. The alias file lets you populate the header file through a user-friendly interface.
3. Author Create and assign identifiers (IDs).

(Optional for HTML5) Create a skin to use in CSH, set an ID for it, associate it with the appropriate IDs, and in the target set all skins to be generated.

4. **Author** Associate the alias file with the target. This is necessary only if you have created more than one alias file.
5. **Author** Provide the developer with a copy of the header file (or at least the CSH IDs within it) and the skin ID (if you are using one).
6. **Author** Generate output and provide the developer with a copy of the output files.
7. **Developer** Make CSH calls using information in the header file and the skin ID, if applicable.
8. **Developer** Create a new build of the software application.
9. **Author** Install the new software build and test the CSH.

Process (If Developer Provides Header File)

1. **Developer** Create the header file (or at least individual CSH IDs) and send to the author.
2. **Author** Add the developer's header file to the project.
3. **Author** Add an alias file to the project. The alias file lets you populate the header file through a user-friendly interface.
4. **Author** Assign IDs for the header file.

   (Optional for HTML5) Create a skin to use in CSH, set an ID for it, associate it with the appropriate IDs, and in the target set all skins to be generated. Provide the developer with the skin ID.
5. **Author** Associate the alias file with the target. This is necessary only if you have created more than one alias file.
6. **Author** Generate output and provide the developer with a copy of the output files.
7. **Developer** Make CSH calls using information in the header file and the skin ID, if applicable.
8. **Developer** Create a new build of the software application.
9. **Author** Install the new software build and test the CSH.

☆ EXAMPLE You are creating an online Help system for your company's software application. This application contains a dialog called "Properties" that users open to specify settings for a particular element. In your Help project, you have written a topic to explain this Properties dialog. By creating CSH, users will be able to open that specific topic by clicking a Help button on the Properties dialog (or by pressing F1 when it is open).

CHAPTER 2

General Information for Context-Sensitive Help

There are various pieces of general information you should know if you plan to use this feature.

This chapter discusses the following:

Who is Involved?

Creating context-sensitive Help (CSH) is mostly a joint effort between *you (the author)* and the *software developer*. There are tasks that you must perform and tasks that the developer must perform in order for CSH to be implemented successfully. For this reason, it is essential that you communicate clearly with the developer when planning, creating, and implementing CSH. Other individuals (managers, other Help authors, etc.) may also be involved as well, particularly in the early planning stages.

However, there might be times when you function as both the author and the developer. For example, this might be the case if you are generating HTML5 output and simply want to create links on a website that open specific parts of your output. In that situation, you might first generate the online output with the CSH information. Then you might serve as the developer, modifying pages on a website to include CSH links pointing to your documentation.

Planning Context-Sensitive Help

For context-sensitive Help (CSH) to be successful, some initial planning is necessary. This includes making decisions such as:

- Which type of output is being created?
- Which dialogs and windows will be included in the CSH (if connecting to an application)?
- Where will the links be located (if connecting to simple web links)?
- What will the Help buttons look like on the dialogs or windows?
- Will the CSH topics open in a different window than the other topics in the Help system (different size, position, and appearance)?
- Will the CSH topics use a different skin depending on the situation?

Depending on how your company operates, questions such as these may be decided independently by you or the developer. Or they may be decided jointly by you, the software developer, and others (managers, other authors).

Another major decision that needs to be made at the beginning of the process is whether you or the developer will be responsible for providing the header file that is necessary for CSH. This decision is typically made jointly by you and the software developer. See "Header Files" on the next page.

Header Files

A header file (sometimes referred to as a "map file") is a simple text file that works in conjunction with an alias file (i.e., as you edit the alias file, the header file is populated automatically) in context-sensitive Help. The header file contains basic information about connecting areas of an interface to the corresponding topics or micro content in online output. Both you and the software developer need access to this file, or at least to the IDs that will be used to connect various parts of the interface to specific areas of the documentation.

A header file has an .h extension and is stored in the Project Organizer under the Advanced folder. You can export the Flare header file into other file formats (e.g., .bas, .properties, .inc, etc.) if necessary.

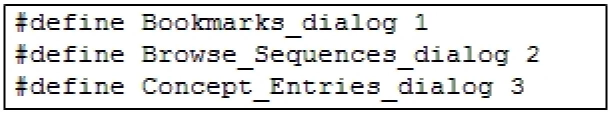
Who Develops the Header File and How?

Either you or the software developer is responsible for creating the header file. That is something you must decide with the developer.

If it is decided that you are responsible for creating the header file, you can do so by adding a header file to the project, adding an alias file to the project, and then creating and assigning identifiers (IDs).

What is Contained in a Header File?

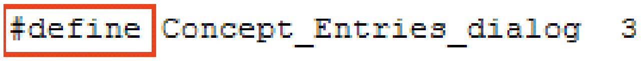
A completed header file contains one or more lines of text containing IDs. Each ID refers to a specific area of the interface that is linked to a corresponding topic or micro content response in the output. Here is part of a header file, showing three identifiers:



```
#define Bookmarks_dialog 1
#define Browse_Sequences_dialog 2
#define Concept_Entries_dialog 3
```

Preliminary Text

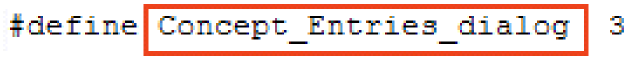
At the beginning of each line (before the header ID) is some necessary preliminary text (#define). Flare will add this text if you create IDs from within Flare. If you create the ID by using another text editor (such as Word or Notepad), you need to type this text manually.



```
#define Concept_Entries_dialog   3
```

ID Name

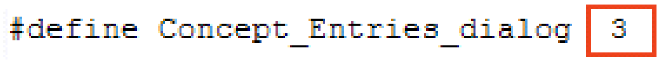
The next portion is the ID name for the topic or micro content. It tells you and the developer which part of the interface is associated with the line. This ID name is determined by you or the developer—whatever helps you to identify that area of the interface. Spaces are not allowed between words, so in the following example we used underscores.



```
#define Concept_Entries_dialog  3
```

ID Number

At the end of the line is a numerical value for the ID. Each ID must have a unique number assigned to it. Developers can "hook" the area of the interface in different ways (e.g., by pointing to the ID name or number). This way, the software application and your output can communicate with each other.



```
#define Concept_Entries_dialog  3
```

What’s Noteworthy?

NOTE If you are importing FrameMaker documents and you create topic alias markers in the source files, this file will be created automatically when you perform the import.

Alias Files

An alias file is used to populate a header file with the information necessary for producing context-sensitive Help (CSH). After opening an alias file, you can use the Alias Editor to create and assign identifiers (IDs) for the header file. You can use a single alias file in a project for multiple header files, or you can create a separate alias file to go with each header file.

Context-Sensitive Help Features Per Output Type

Following are context-sensitive Help (CSH) features supported in each output type.

	HTML5	PDF	Word	Clean XHTML	Eclipse Help	EPUB	HTML Help
CSH Supported	✓	✗	✗	✗	✓	✗	✓
Micro Content CSH	✓	✗	✗	✗	✗	✗	✗

CHAPTER 3

Process for Context-Sensitive Help

The process that you follow for context-sensitive Help (CSH) depends on whether you or the software developer creates the header file.

This chapter discusses the following:

If Author Creates Header File

1. Author Add a header file to the project.
2. Author Add an alias file to the project. The alias file lets you populate the header file through a user-friendly interface.
3. Author Create and assign identifiers (IDs).

   (Optional for HTML5) Create a skin to use in CSH, set an ID for it, associate it with the appropriate IDs, and in the target set all skins to be generated.
4. Author Associate the alias file with the target. This is necessary only if you have created more than one alias file.
5. Author Provide the developer with a copy of the header file (or at least the CSH IDs within it) and the skin ID (if you are using one).
6. Author Generate output and provide the developer with a copy of the output files.
7. Developer Make CSH calls using information in the header file and the skin ID, if applicable.
8. Developer Create a new build of the software application.
9. Author Install the new software build and test the CSH.

If Developer Creates Header File

1. **Developer** Create the header file (or at least individual CSH IDs) and send to the author.
2. **Author** Add the developer's header file to the project.
3. **Author** Add an alias file to the project. The alias file lets you populate the header file through a user-friendly interface.
4. **Author** Assign IDs for the header file.

   (Optional for HTML5) Create a skin to use in CSH, set an ID for it, associate it with the appropriate IDs, and in the target set all skins to be generated. Provide the developer with the skin ID.
5. **Author** Associate the alias file with the target. This is necessary only if you have created more than one alias file.
6. **Author** Generate output and provide the developer with a copy of the output files.
7. **Developer** Make CSH calls using information in the header file and the skin ID, if applicable.
8. **Developer** Create a new build of the software application.
9. **Author** Install the new software build and test the CSH.

Adding Header Files

Aside from planning the context-sensitive Help (CSH), the first step in this process is to add the header file to the Flare project. Exactly how you do this depends on whether you or the software developer is responsible for creating the header file.

How to Add a Header File (If Author Creates It)

1. Do one of the following, depending on the part of the user interface you are using:

   - **Ribbon** Select **Project > New > Advanced > Header File**.
   - **Right-Click** In the Project Organizer, right-click on the **Advanced** folder and from the context menu select **Add Header File**.

   The Add File dialog opens.

2. In the **File Type** field at the top, make sure **Header File** is selected.

3. In the **Source** area, choose to create the new file based on a template or an existing file.

   - **New From Template** Choose either a factory template file or one of your own custom template files as a starting point. The new file will take on all of the settings contained in the template. If you want to use the factory template provided by Flare, expand the **Factory Templates** folder and click on a template file. If you want to use your own custom template file, expand the appropriate folder and click on a file. For more information about templates, see the online Help.
   - **New From Existing** Choose an existing file of the same type as a starting point for your new file. As with template files, your new file will take on all of the settings contained in the file you select. To use this option, click ..., use the Open File dialog to find a file, and double-click it.

4. (Optional) If you want to place the file into a subfolder previously created in the Content Explorer or Project Organizer, in the **Folder** field click ... and select the subfolder. Otherwise, keep the default location.

5. In the **File Name** field, type a new name for the header file.

6. (Optional) If you want to apply condition tags to the file, expand the **Attributes** section at the bottom of the dialog. Next to the **Condition Tags** field, click ... and select the conditions you want to apply. Click **OK**.

7. (Optional) If you want to apply file tags, expand the **Attributes** section at the bottom of the dialog. Next to the **File Tags** field, click … and select the file tags you want to apply. Click **OK**.
8. Click **Add**. The header file is added to the Advanced folder in the Project Organizer. The Text Editor opens to the right, with the page for the new header file (including an initial identifier) shown.
9. You can close the Text Editor by clicking the **x** at the top-right corner of the tab.

   Most authors do not enter content into this editor directly. The file is populated automatically when you work in the alias file. See "Creating and Assigning Identifiers" on page 23.

How to Add a Header File (If Developer Creates It)

1. Do one of the following, depending on the part of the user interface you are using:
   - **Ribbon** Select **Project > New > Advanced > Header File**.
   - **Right-Click** In the Project Organizer, right-click on the **Advanced** folder and from the context menu select **Add Header File**.

   The Add File dialog opens.
2. In the **File Type** field at the top, make sure **Header File** is selected.
3. Select **New from existing** and click ....
4. Find and select the header file that you want to import.
5. Click **Open**. The Source File field now contains the path to the file that you are importing. Also, the name of the file is displayed in the File Name field.
6. If you want to give the header file a different name than that for the imported file, click in the **File name** field and replace the text.
7. (Optional) If you want to apply condition tags to the file, expand the **Attributes** section at the bottom of the dialog. Next to the **Condition Tags** field, click ... and select the conditions you want to apply. Click **OK**.
8. (Optional) If you want to apply file tags, expand the **Attributes** section at the bottom of the dialog. Next to the **File Tags** field, click ... and select the file tags you want to apply. Click **OK**.
9. Click **Add**. The header file is added.

Adding Alias Files

After you add a header file to your project, the next step in creating context-sensitive Help (CSH) is to add an alias file. The alias file will help you to populate the header file.

How to Add an Alias File

1. Do one of the following, depending on the part of the user interface you are using:
   - **Ribbon** Select **Project > New > Advanced > Alias File**.
   - **Right-Click** In the Project Organizer, right-click on the **Advanced** folder and from the context menu select **Add Alias File**.

   The Add File dialog opens.
2. In the **File Type** field at the top, make sure **Alias File** is selected.
3. In the **Source** area, choose to create the new file based on a template or an existing file.
   - **New From Template** Choose either a factory template file or one of your own custom template files as a starting point. The new file will take on all of the settings contained in the template. If you want to use the factory template provided by Flare, expand the **Factory Templates** folder and click on a template file. If you want to use your own custom template file, expand the appropriate folder and click on a file. For more information about templates, see the online Help.
   - **New From Existing** Choose an existing file of the same type as a starting point for your new file. As with template files, your new file will take on all of the settings contained in the file you select. To use this option, click …, use the Open File dialog to find a file, and double-click it.
4. (Optional) If you want to place the file into a subfolder previously created in the Content Explorer or Project Organizer, in the **Folder** field click … and select the subfolder. Otherwise, keep the default location.
5. In the **File Name** field, type a new name for the alias file.
6. (Optional) If you want to apply condition tags to the file, expand the **Attributes** section at the bottom of the dialog. Next to the **Condition Tags** field, click … and select the conditions you want to apply. Click **OK**.

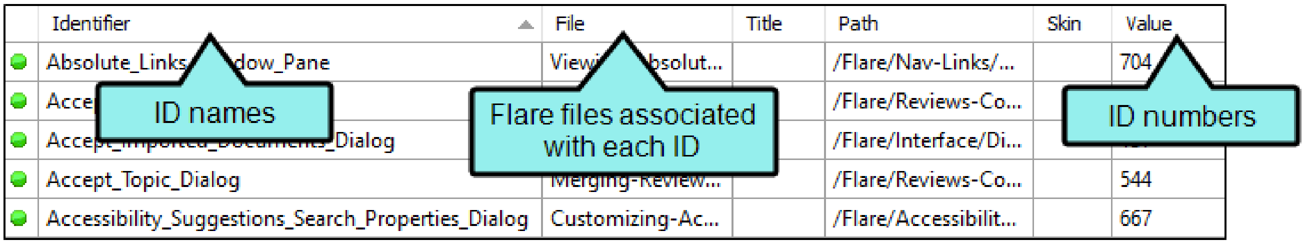
7. (Optional) If you want to apply file tags, expand the **Attributes** section at the bottom of the dialog. Next to the **File Tags** field, click … and select the file tags you want to apply. Click **OK**.

8. Click **Add**. The alias file is added to the Advanced folder in the Project Organizer. The Alias Editor opens to the right, with the page for the new alias file shown. The file includes an initial identifier for the header file that you created previously.

Creating and Assigning Identifiers

The first steps in developing context-sensitive Help (CSH) for your project are to add a header file and add an alias file. After this, you need to work on creating identifiers (IDs) for the header file.

A CSH ID is text (i.e., name) or a number (i.e., value) that connects a particular topic or micro content response with the corresponding part of the software interface. You do not need to use both the ID name and value; one is enough. By pointing to the ID (instead of the full path of the topic or micro content), you can be assured that the link will always work in case the documentation file is later renamed or moved.



	Identifier	File	Title	Path	Skin	Value
●	Absolute_Links dow_Pane	Viewi bsolut...		/Flare/Nav-Links/...		704
●	Acce			/Flare/Reviews-Co...		
●	Accep _Dialog			/Flare/Interface/Di...		
●	Accept_Topic_Dialog	Merging-Review...		/Flare/Reviews-Co...		544
●	Accessibility_Suggestions_Search_Properties_Dialog	Customizing-Ac...		/Flare/Accessibilit...		667

- If it is decided that you are responsible for creating the header file in CSH, you need to open the Alias Editor to create *and* assign IDs to topics or micro content phrases.
- If it is decided that the software developer is responsible for creating the header file in CSH, you only need to assign the IDs, which should already be contained in the header file that the developer provides for you. Before assigning the IDs to topics or micro content phrases, you must first import the header file from the developer so that it is located in the Advanced folder in the Project Organizer. See "Importing Header Files" on page 66.

How to Automatically Generate IDs

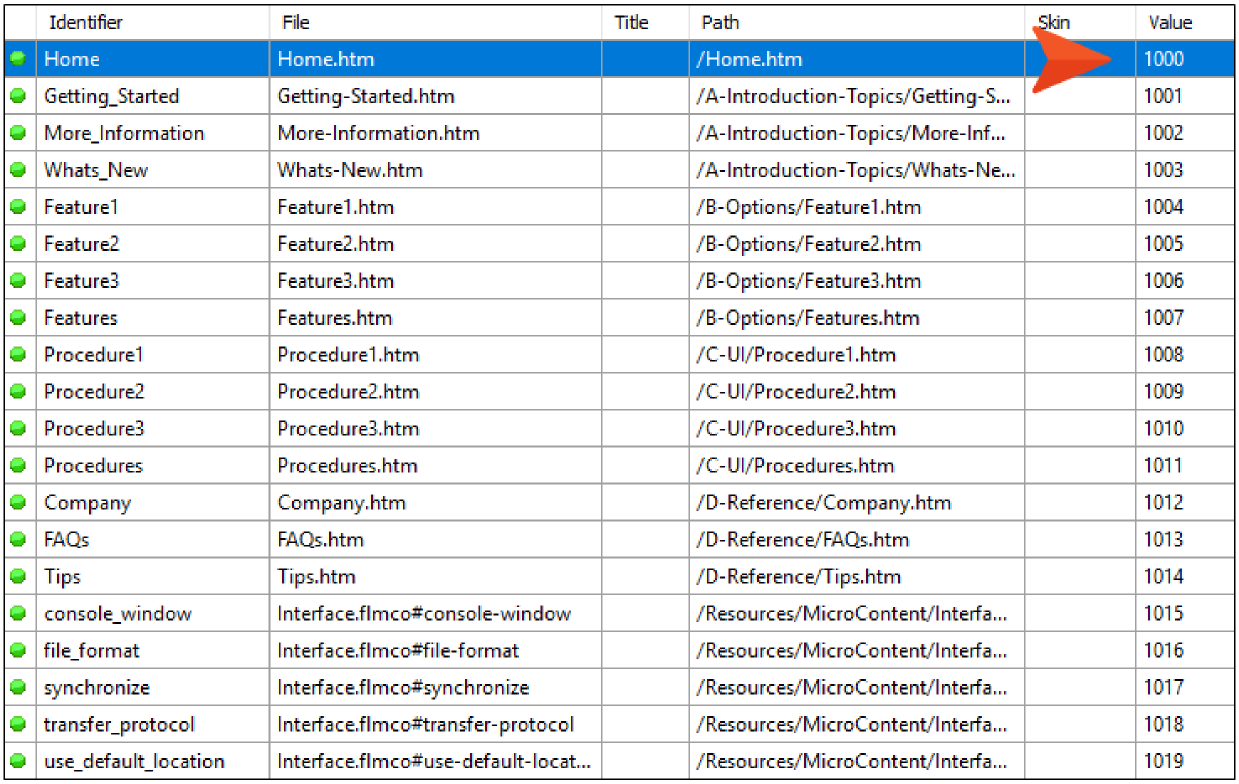
You might use these steps if you are just getting started with a header and alias file, and you need to create IDs for all or many topics or phrases.

1. Open the alias file that you created.
2. (Optional) You can set options for new IDs in advance. This will supply some of the information (e.g., starting value, prefix, include topic in ID name, assign skin) for you automatically as you create new IDs. See "Setting Identifier Options" on page 69.
3. Do one of the following:
   - In the local toolbar of the Alias Editor click .
   - Right-click in the **Identifiers** side of the editor, and from the context menu select **Auto Generate**.

   The Generate Identifiers dialog opens.
4. In the **Header File** area on the left, select whether to create a new header file from a template or choose an existing header file.
5. (Optional) On the right side, in the **Identifier Options** area, you can override values that are already set in the Identifier Options dialog (see Step 2).

STARTING VALUE

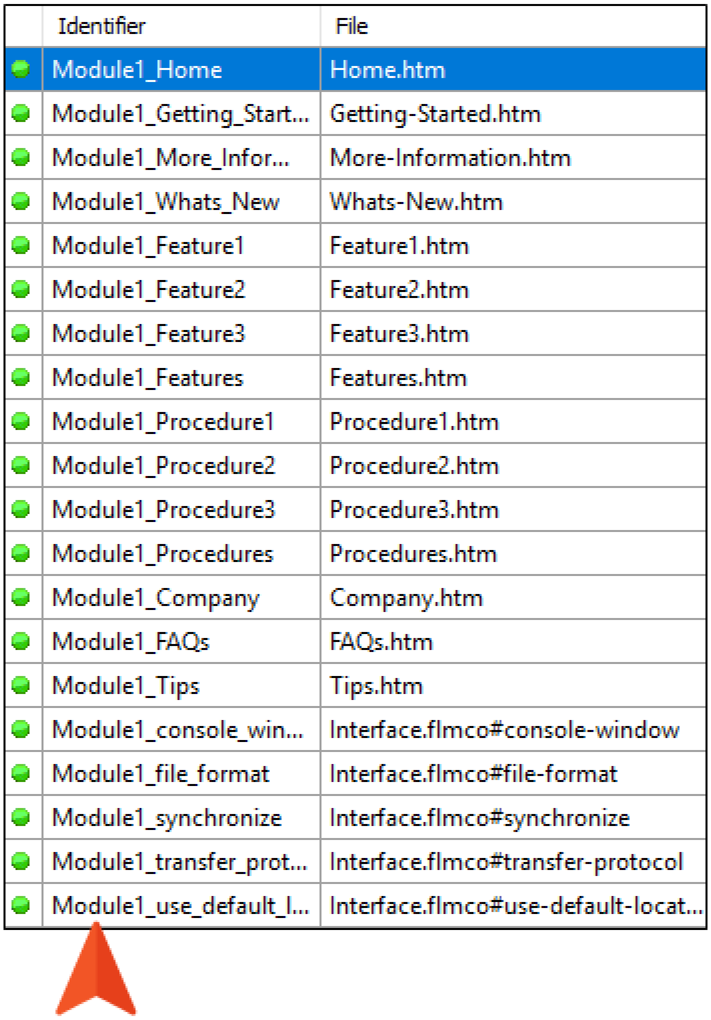
Enter the starting value for IDs that you create. Additional IDs that are created will be incremented automatically based on that starting value.



	Identifier	File	Title	Path	Skin	Value
●	Home	Home.htm		/Home.htm		1000
●	Getting_Started	Getting-Started.htm		/A-Introduction-Topics/Getting-S...		1001
●	More_Information	More-Information.htm		/A-Introduction-Topics/More-Inf...		1002
●	Whats_New	Whats-New.htm		/A-Introduction-Topics/Whats-Ne...		1003
●	Feature1	Feature1.htm		/B-Options/Feature1.htm		1004
●	Feature2	Feature2.htm		/B-Options/Feature2.htm		1005
●	Feature3	Feature3.htm		/B-Options/Feature3.htm		1006
●	Features	Features.htm		/B-Options/Features.htm		1007
●	Procedure1	Procedure1.htm		/C-UI/Procedure1.htm		1008
●	Procedure2	Procedure2.htm		/C-UI/Procedure2.htm		1009
●	Procedure3	Procedure3.htm		/C-UI/Procedure3.htm		1010
●	Procedures	Procedures.htm		/C-UI/Procedures.htm		1011
●	Company	Company.htm		/D-Reference/Company.htm		1012
●	FAQs	FAQs.htm		/D-Reference/FAQs.htm		1013
●	Tips	Tips.htm		/D-Reference/Tips.htm		1014
●	console_window	Interface.flmco#console-window		/Resources/MicroContent/Interfa...		1015
●	file_format	Interface.flmco#file-format		/Resources/MicroContent/Interfa...		1016
●	synchronize	Interface.flmco#synchronize		/Resources/MicroContent/Interfa...		1017
●	transfer_protocol	Interface.flmco#transfer-protocol		/Resources/MicroContent/Interfa...		1018
●	use_default_location	Interface.flmco#use-default-locat...		/Resources/MicroContent/Interfa...		1019

PREFIX

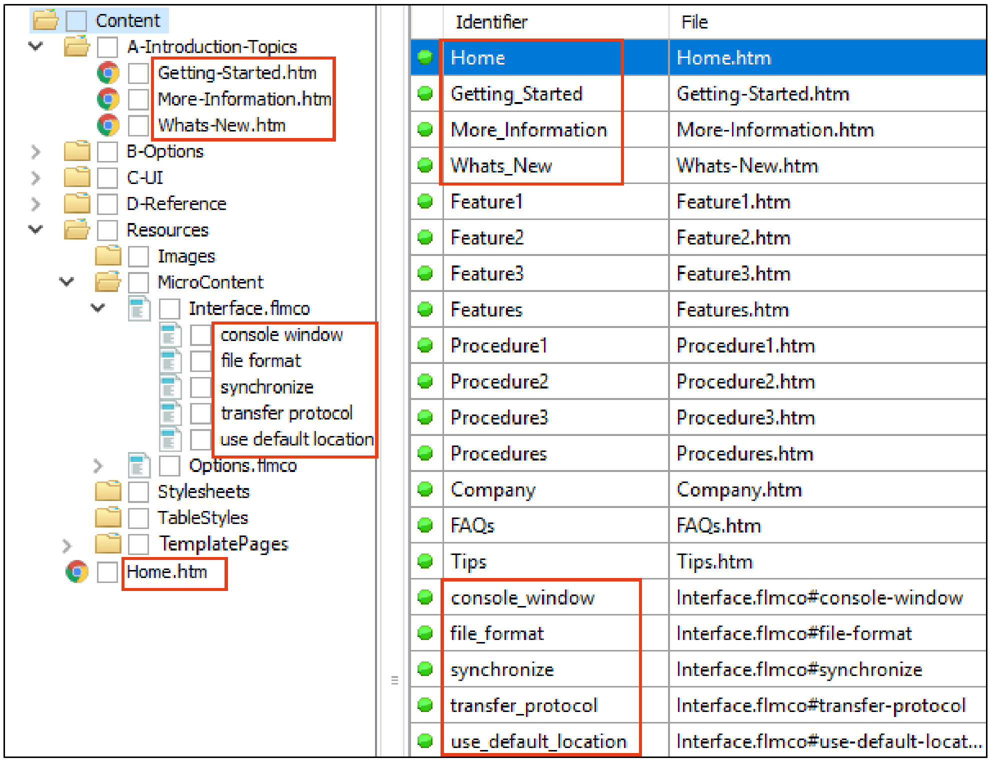
Specify a prefix to be added at the beginning of each new ID that you create (e.g., ID_, Dialog, Module1).



	Identifier	File
●	Module1_Home	Home.htm
●	Module1_Getting_Start...	Getting-Started.htm
●	Module1_More_Infor...	More-Information.htm
●	Module1_Whats_New	Whats-New.htm
●	Module1_Feature1	Feature1.htm
●	Module1_Feature2	Feature2.htm
●	Module1_Feature3	Feature3.htm
●	Module1_Features	Features.htm
●	Module1_Procedure1	Procedure1.htm
●	Module1_Procedure2	Procedure2.htm
●	Module1_Procedure3	Procedure3.htm
●	Module1_Procedures	Procedures.htm
●	Module1_Company	Company.htm
●	Module1_FAQs	FAQs.htm
●	Module1_Tips	Tips.htm
●	Module1_console_win...	Interface.flmco#console-window
●	Module1_file_format	Interface.flmco#file-format
●	Module1_synchronize	Interface.flmco#synchronize
●	Module1_transfer_prot...	Interface.flmco#transfer-protocol
●	Module1_use_default_l...	Interface.flmco#use-default-locat...

INCLUDE FILE OR PHRASE NAME IN IDENTIFIER NAME

Select this check box if you want the names of the assigned topics or micro content phrases to be included automatically in the names of the new IDs. Flare will add underscores if a name or phrase has more than one word.

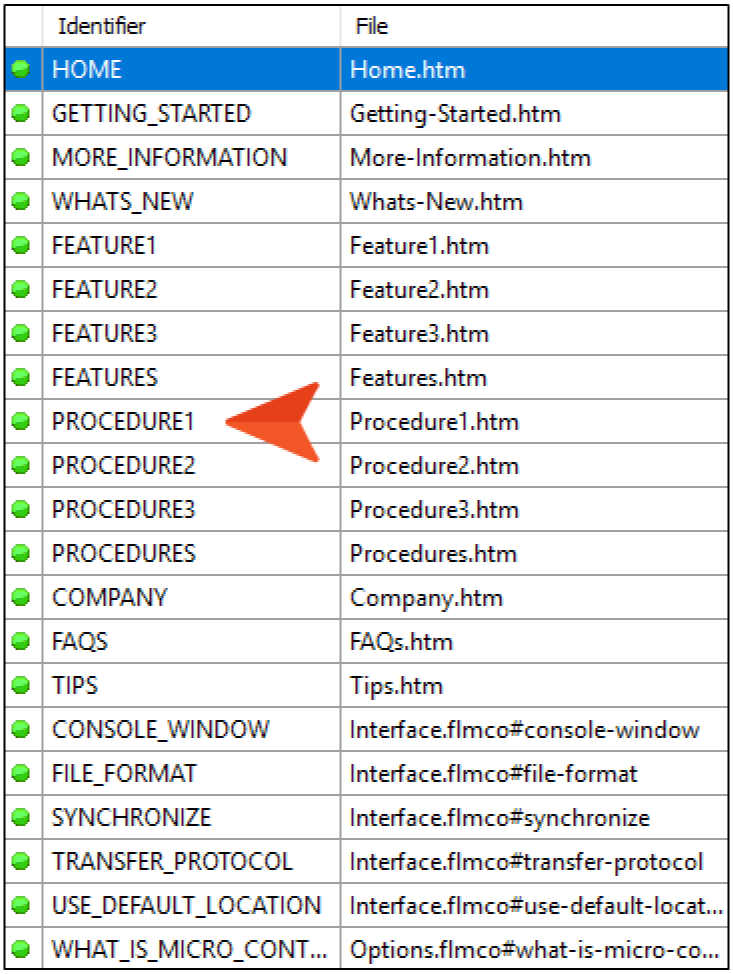


- □ Content
  - □ A-Introduction-Topics
    - □ Getting-Started.htm
    - □ More-Information.htm
    - □ Whats-New.htm
  - □ B-Options
  - □ C-UI
  - □ D-Reference
  - □ Resources
    - □ Images
    - □ MicroContent
      - □ Interface.flmco
        - □ console window
        - □ file format
        - □ synchronize
        - □ transfer protocol
        - □ use default location
      - □ Options.flmco
    - □ Stylesheets
    - □ TableStyles
    - □ TemplatePages
  - □ Home.htm

	Identifier	File
●	Home	Home.htm
●	Getting_Started	Getting-Started.htm
●	More_Information	More-Information.htm
●	Whats_New	Whats-New.htm
●	Feature1	Feature1.htm
●	Feature2	Feature2.htm
●	Feature3	Feature3.htm
●	Features	Features.htm
●	Procedure1	Procedure1.htm
●	Procedure2	Procedure2.htm
●	Procedure3	Procedure3.htm
●	Procedures	Procedures.htm
●	Company	Company.htm
●	FAQs	FAQs.htm
●	Tips	Tips.htm
●	console_window	Interface.flmco#console-window
●	file_format	Interface.flmco#file-format
●	synchronize	Interface.flmco#synchronize
●	transfer_protocol	Interface.flmco#transfer-protocol
●	use_default_location	Interface.flmco#use-default-locat...

CAPITALIZE IDENTIFIER NAMES

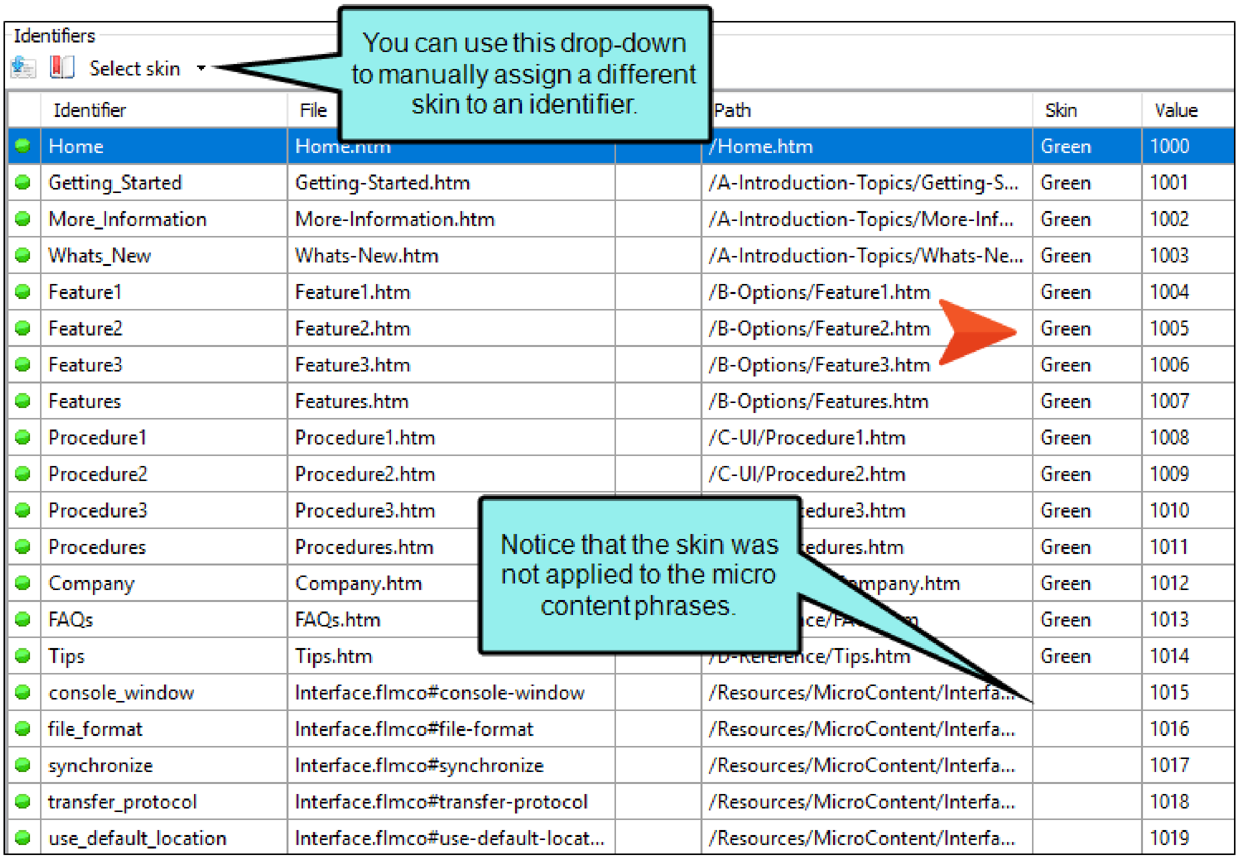
Select this check box if you want the name that is automatically added for new IDs to use all caps.



	Identifier	File
●	HOME	Home.htm
●	GETTING_STARTED	Getting-Started.htm
●	MORE_INFORMATION	More-Information.htm
●	WHATS_NEW	Whats-New.htm
●	FEATURE1	Feature1.htm
●	FEATURE2	Feature2.htm
●	FEATURE3	Feature3.htm
●	FEATURES	Features.htm
●	PROCEDURE1	Procedure1.htm
●	PROCEDURE2	Procedure2.htm
●	PROCEDURE3	Procedure3.htm
●	PROCEDURES	Procedures.htm
●	COMPANY	Company.htm
●	FAQS	FAQs.htm
●	TIPS	Tips.htm
●	CONSOLE_WINDOW	Interface.flmco#console-window
●	FILE_FORMAT	Interface.flmco#file-format
●	SYNCHRONIZE	Interface.flmco#synchronize
●	TRANSFER_PROTOCOL	Interface.flmco#transfer-protocol
●	USE_DEFAULT_LOCATION	Interface.flmco#use-default-locat...
●	WHAT_IS_MICRO_CONT...	Options.flmco#what-is-micro-co...

SKINS

Specify which skin should be assigned by default to new IDs that are created. You can always manually select a different skin for any ID afterward, but when you first create a new ID, it will initially be assigned to the default skin that you specified. This option applies to topics only, not to micro content phrases.



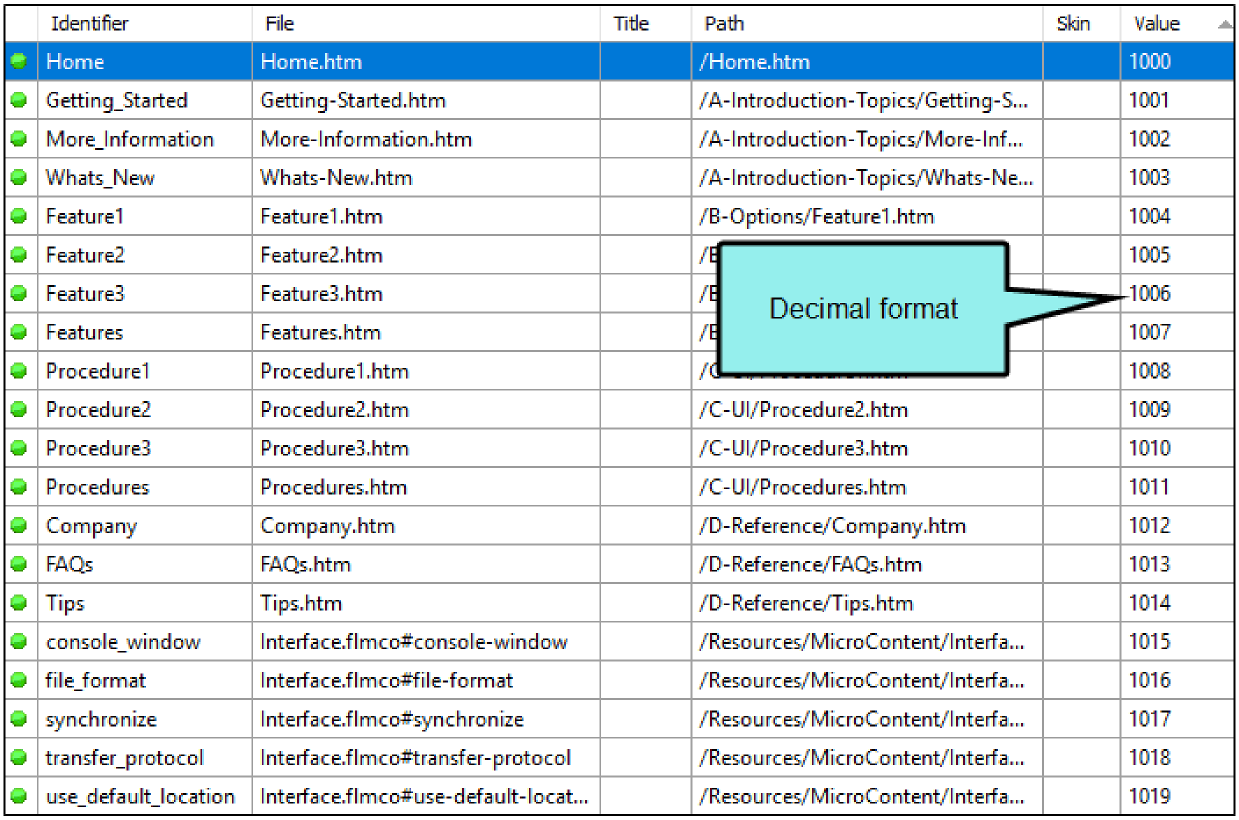
You can use this drop-down to manually assign a different skin to an identifier.

Identifier	File	Path	Skin	Value
Home	Home.htm	/Home.htm	Green	1000
Getting_Started	Getting-Started.htm	/A-Introduction-Topics/Getting-S...	Green	1001
More_Information	More-Information.htm	/A-Introduction-Topics/More-Inf...	Green	1002
Whats_New	Whats-New.htm	/A-Introduction-Topics/Whats-Ne...	Green	1003
Feature1	Feature1.htm	/B-Options/Feature1.htm	Green	1004
Feature2	Feature2.htm	/B-Options/Feature2.htm	Green	1005
Feature3	Feature3.htm	/B-Options/Feature3.htm	Green	1006
Features	Features.htm	/B-Options/Features.htm	Green	1007
Procedure1	Procedure1.htm	/C-UI/Procedure1.htm	Green	1008
Procedure2	Procedure2.htm	/C-UI/Procedure2.htm	Green	1009
Procedure3	Procedure3.htm	edure3.htm	Green	1010
Procedures	Procedures.htm	edures.htm	Green	1011
Company	Company.htm	mpany.htm	Green	1012
FAQs	FAQs.htm	ce/FA	Green	1013
Tips	Tips.htm	/D-Reference/Tips.htm	Green	1014
console_window	Interface.flmco#console-window	/Resources/MicroContent/Interfa...		1015
file_format	Interface.flmco#file-format	/Resources/MicroContent/Interfa...		1016
synchronize	Interface.flmco#synchronize	/Resources/MicroContent/Interfa...		1017
transfer_protocol	Interface.flmco#transfer-protocol	/Resources/MicroContent/Interfa...		1018
use_default_location	Interface.flmco#use-default-locat...	/Resources/MicroContent/Interfa...		1019

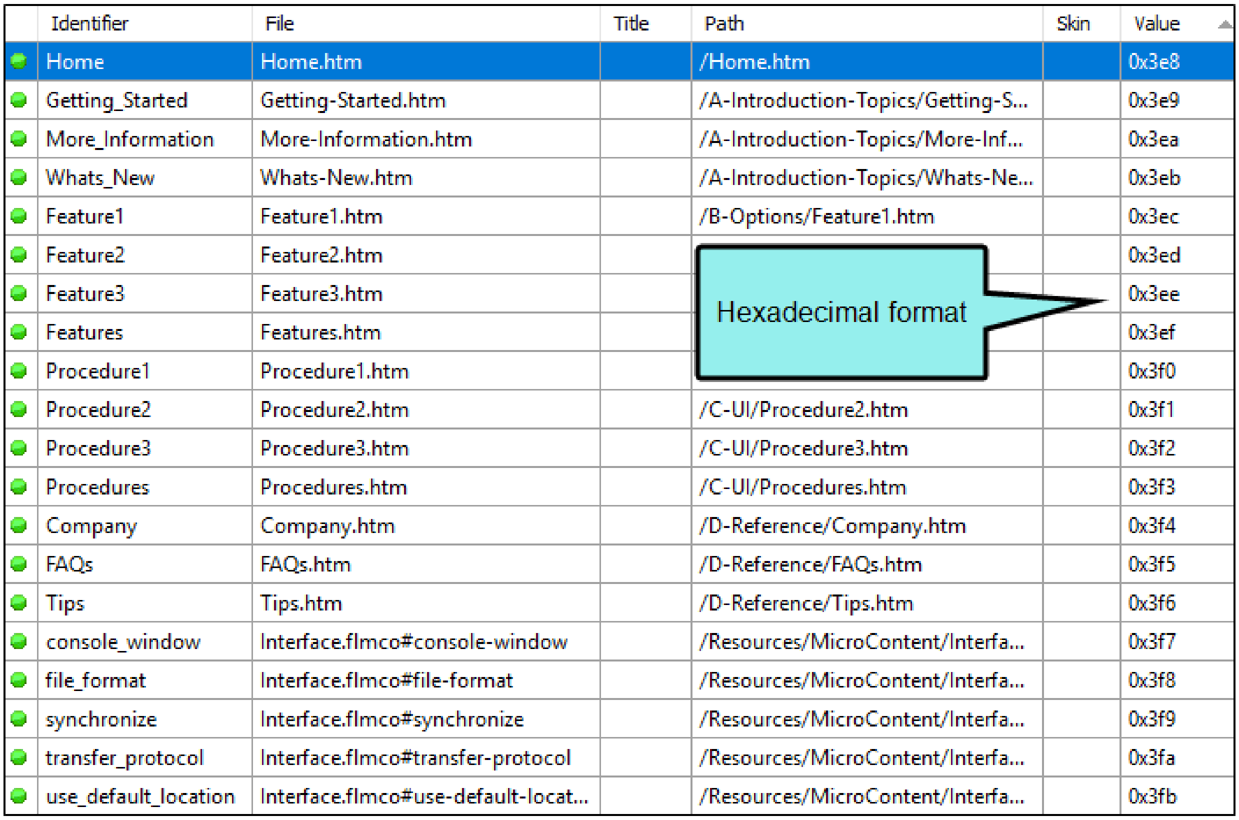
Notice that the skin was not applied to the micro content phrases.

VALUE FORMAT

Specify whether the ID values should use a decimal or hexadecimal format. Using hexadecimal values does not affect your CSH in a different way; it's simply another option in case your developers prefer that format.



	Identifier	File	Title	Path	Skin	Value
●	Home	Home.htm		/Home.htm		1000
●	Getting_Started	Getting-Started.htm		/A-Introduction-Topics/Getting-S...		1001
●	More_Information	More-Information.htm		/A-Introduction-Topics/More-Inf...		1002
●	Whats_New	Whats-New.htm		/A-Introduction-Topics/Whats-Ne...		1003
●	Feature1	Feature1.htm		/B-Options/Feature1.htm		1004
●	Feature2	Feature2.htm		/B		1005
●	Feature3	Feature3.htm		/B		1006
●	Features	Features.htm		/B		1007
●	Procedure1	Procedure1.htm		/C		1008
●	Procedure2	Procedure2.htm		/C-UI/Procedure2.htm		1009
●	Procedure3	Procedure3.htm		/C-UI/Procedure3.htm		1010
●	Procedures	Procedures.htm		/C-UI/Procedures.htm		1011
●	Company	Company.htm		/D-Reference/Company.htm		1012
●	FAQs	FAQs.htm		/D-Reference/FAQs.htm		1013
●	Tips	Tips.htm		/D-Reference/Tips.htm		1014
●	console_window	Interface.flmco#console-window		/Resources/MicroContent/Interfa...		1015
●	file_format	Interface.flmco#file-format		/Resources/MicroContent/Interfa...		1016
●	synchronize	Interface.flmco#synchronize		/Resources/MicroContent/Interfa...		1017
●	transfer_protocol	Interface.flmco#transfer-protocol		/Resources/MicroContent/Interfa...		1018
●	use_default_location	Interface.flmco#use-default-locat...		/Resources/MicroContent/Interfa...		1019



	Identifier	File	Title	Path	Skin	Value
●	Home	Home.htm		/Home.htm		0x3e8
●	Getting_Started	Getting-Started.htm		/A-Introduction-Topics/Getting-S...		0x3e9
●	More_Information	More-Information.htm		/A-Introduction-Topics/More-Inf...		0x3ea
●	Whats_New	Whats-New.htm		/A-Introduction-Topics/Whats-Ne...		0x3eb
●	Feature1	Feature1.htm		/B-Options/Feature1.htm		0x3ec
●	Feature2	Feature2.htm				0x3ed
●	Feature3	Feature3.htm				0x3ee
●	Features	Features.htm				0x3ef
●	Procedure1	Procedure1.htm				0x3f0
●	Procedure2	Procedure2.htm		/C-UI/Procedure2.htm		0x3f1
●	Procedure3	Procedure3.htm		/C-UI/Procedure3.htm		0x3f2
●	Procedures	Procedures.htm		/C-UI/Procedures.htm		0x3f3
●	Company	Company.htm		/D-Reference/Company.htm		0x3f4
●	FAQs	FAQs.htm		/D-Reference/FAQs.htm		0x3f5
●	Tips	Tips.htm		/D-Reference/Tips.htm		0x3f6
●	console_window	Interface.flmco#console-window		/Resources/MicroContent/Interfa...		0x3f7
●	file_format	Interface.flmco#file-format		/Resources/MicroContent/Interfa...		0x3f8
●	synchronize	Interface.flmco#synchronize		/Resources/MicroContent/Interfa...		0x3f9
●	transfer_protocol	Interface.flmco#transfer-protocol		/Resources/MicroContent/Interfa...		0x3fa
●	use_default_location	Interface.flmco#use-default-locat...		/Resources/MicroContent/Interfa...		0x3fb

Hexadecimal format

6. In the **Generate Identifiers for** field, select one of the options, depending on whether you want to create IDs for all topics and/or micro content, or only for those that are not yet assigned to IDs.

7. If you selected one of the "All" options in the previous step, choose one of the following in the **Existing Identifiers** field:

   - **Keep** Select this if you want to keep the IDs that you already have in the header file. As a result, you will have some topics that are assigned to multiple IDs (i.e., the old ID and the new one).

   - **Delete** Select this if you want to delete the existing IDs in the header file, creating new ones instead. This way, you will not have any topics that are assigned to multiple IDs.

8. Click **Create**. In the Alias Editor, the header file is automatically selected in the local toolbar. Also, new rows are added in the Alias Editor with ● (instead of ●) next to them. The green icon indicates that the IDs are assigned to topics.

9. (Optional) If you want a certain ID to point to a specific bookmark or header in the assigned topic, do the following:
    a. Select the ID row.
    b. Click the 🔖 button located above the ID list.
    c. In the dialog that opens, select a bookmark or header within the topic.
    d. Click **OK**.
10. (Optional) If you want to make changes (e.g., modify the ID name or number), click in the appropriate cell and type the new information.

    > NOTE Make sure you use underscores between words because spaces are not allowed.

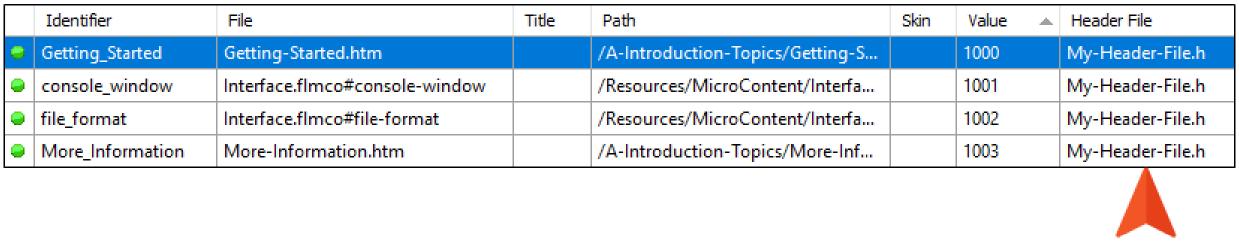
11. Click 💾 to save your work.

How to Create and Assign a New ID Manually

You might use these steps if you already have a header and alias file that you've been working on for some time. Instead of creating lots of new IDs automatically, you can just create them manually as necessary.

1. Open the alias file that you created.
2. (Optional) You can set options for new IDs in advance. This will supply some of the information (e.g., starting value, prefix, include topic in ID name, assign skin) for you automatically as you create new IDs. See "Setting Identifier Options" on page 69.
3. If you have more than one header file in your project, click the down arrow on the left side of the Alias Editor toolbar, and select the header file for which you want to create IDs. Otherwise, you can just leave *(all identifiers)* in the field.

   What if you have multiple header files in your project and you do not select a header file, leaving *(all identifiers)* shown in the drop-down field? In that case, the changes you made in the Alias Editor are applied to the primary header file that you have selected when setting ID options. See "Setting Identifier Options" on page 69.

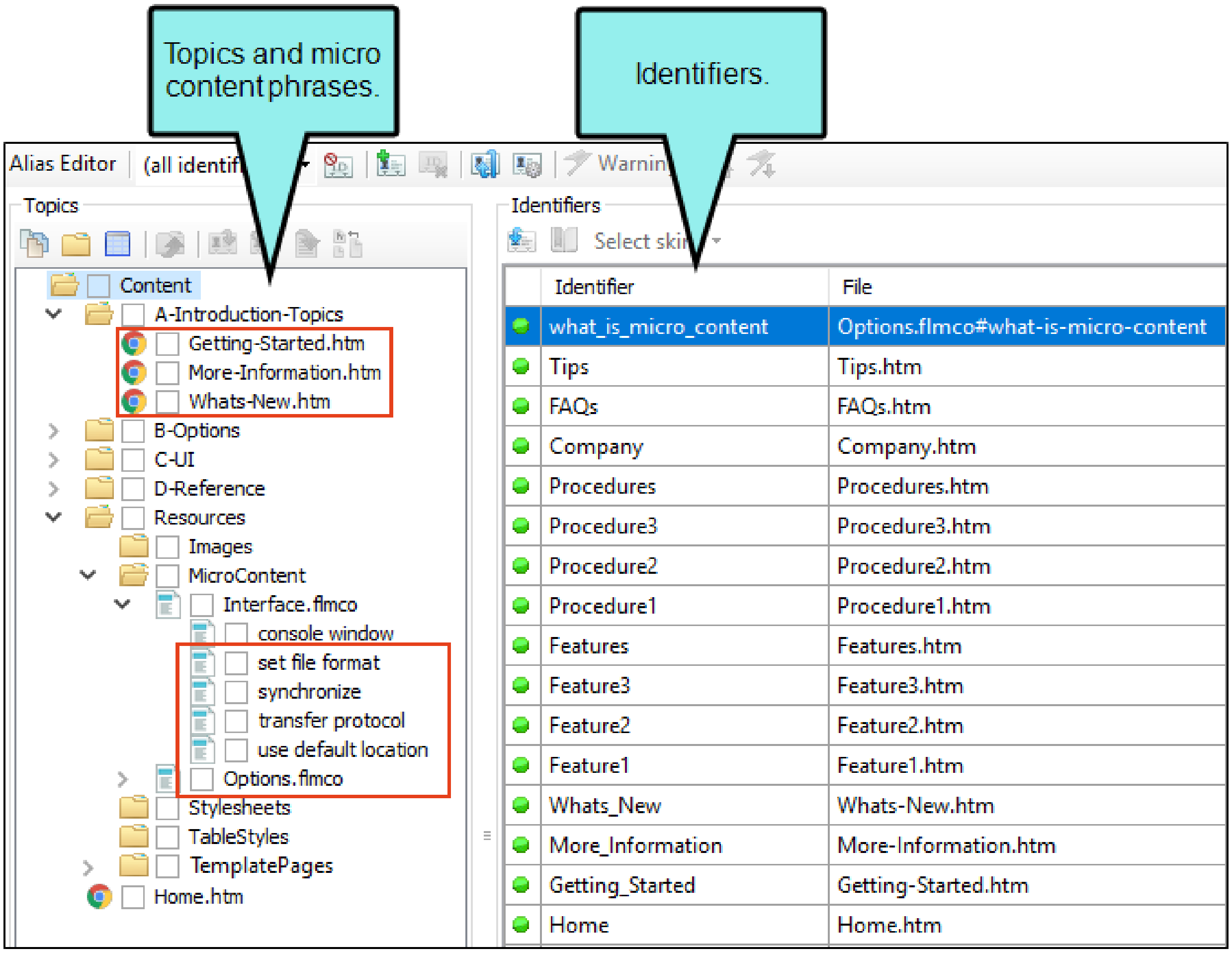


	Identifier	File	Title	Path	Skin	Value ▲	Header File
●	Getting_Started	Getting-Started.htm		/A-Introduction-Topics/Getting-S...		1000	My-Header-File.h
●	console_window	Interface.flmco#console-window		/Resources/MicroContent/Interfa...		1001	My-Header-File.h
●	file_format	Interface.flmco#file-format		/Resources/MicroContent/Interfa...		1002	My-Header-File.h
●	More_Information	More-Information.htm		/A-Introduction-Topics/More-Inf...		1003	My-Header-File.h

4. Do one of the following:

   - On the left side of the editor, select a topic or micro content phrase, and click the **Assign to a new identifier** button located above the file list.
   - On the left side of the editor, right-click a topic or micro content phrase, and select **Assign to New Identifier**.

   A new row is added with 🟢 (instead of 🟡) next to it. The green icon indicates that the ID is assigned to a topic or micro content phrase.



Topics and micro content phrases.

Identifiers.

- Content
  - A-Introduction-Topics
    - Getting-Started.htm
    - More-Information.htm
    - Whats-New.htm
  - B-Options
  - C-UI
  - D-Reference
  - Resources
    - Images
    - MicroContent
      - Interface.flmco
        - console window
        - set file format
        - synchronize
        - transfer protocol
        - use default location
      - Options.flmco
    - Stylesheets
    - TableStyles
    - TemplatePages
  - Home.htm

	Identifier	File
🟢	what_is_micro_content	Options.flmco#what-is-micro-content
🟢	Tips	Tips.htm
🟢	FAQs	FAQs.htm
🟢	Company	Company.htm
🟢	Procedures	Procedures.htm
🟢	Procedure3	Procedure3.htm
🟢	Procedure2	Procedure2.htm
🟢	Procedure1	Procedure1.htm
🟢	Features	Features.htm
🟢	Feature3	Feature3.htm
🟢	Feature2	Feature2.htm
🟢	Feature1	Feature1.htm
🟢	Whats_New	Whats-New.htm
🟢	More_Information	More-Information.htm
🟢	Getting_Started	Getting-Started.htm
🟢	Home	Home.htm

5. (Optional) If you want the ID to point to a specific bookmark or header in the topic, do the following:

    a. Select the ID row.

    b. Click the 🔖 button located above the ID list.

    c. In the dialog that opens, select a bookmark or header within the topic.

    d. Click **OK**.

6. (Optional) If you want to make changes (e.g., modify the ID name or number), click in the appropriate cell and type the new information.

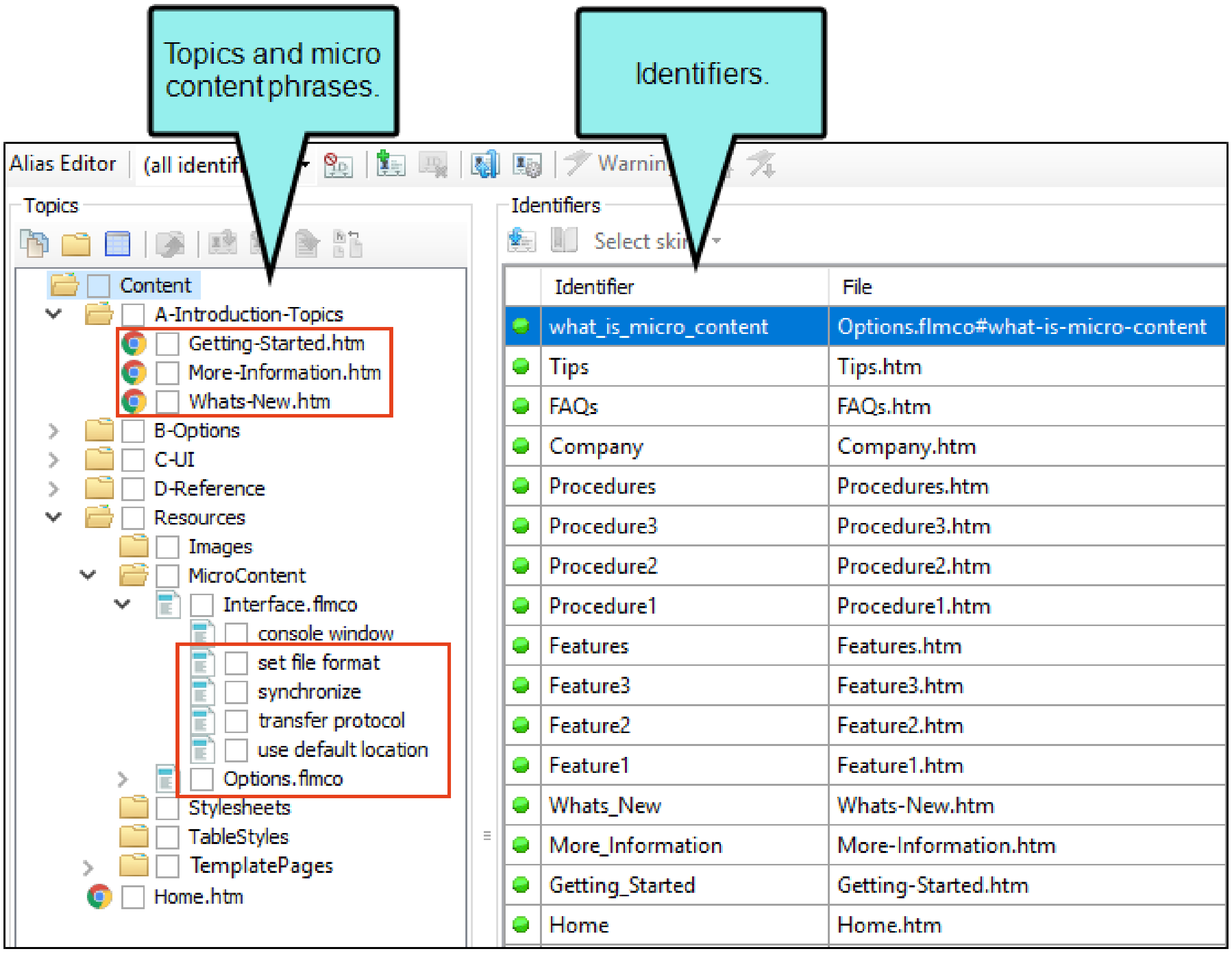
    > NOTE Make sure you use underscores between words because spaces are not allowed.

7. Click 💾 to save your work.

How to Assign IDs

You might use these steps if the developer initially created the header file and you imported it. The IDs therefore already exist. You just need to assign them to the correct topics or micro content phrases.

1. Open the alias file that you created.
2. (Optional) If you want to see only the IDs that are not yet assigned to topics, click in the local toolbar of the Alias Editor. This hides IDs that are already assigned to topics in the editor until you click the button again.
3. On the right side of the Alias Editor, select an ID that you want to assign to a topic or phrase.
4. On the left side of the Alias Editor, find a topic or micro content phrase that you want to assign to the ID.



	Identifier	File
●	what_is_micro_content	Options.flmco#what-is-micro-content
●	Tips	Tips.htm
●	FAQs	FAQs.htm
●	Company	Company.htm
●	Procedures	Procedures.htm
●	Procedure3	Procedure3.htm
●	Procedure2	Procedure2.htm
●	Procedure1	Procedure1.htm
●	Features	Features.htm
●	Feature3	Feature3.htm
●	Feature2	Feature2.htm
●	Feature1	Feature1.htm
●	Whats_New	Whats-New.htm
●	More_Information	More-Information.htm
●	Getting_Started	Getting-Started.htm
●	Home	Home.htm

5. Do one of the following:
   - Double-click the topic or phrase.
   - Click the button located above the file list.
   - Right-click the topic or phrase, and select **Assign to Selected Identifier** from the context menu.

   The topic or phrase name is added to the ID row on the right side of the Alias Editor, and a green icon 🟢 (instead of 🟡) is displayed next to it. The green icon indicates that the ID is assigned.

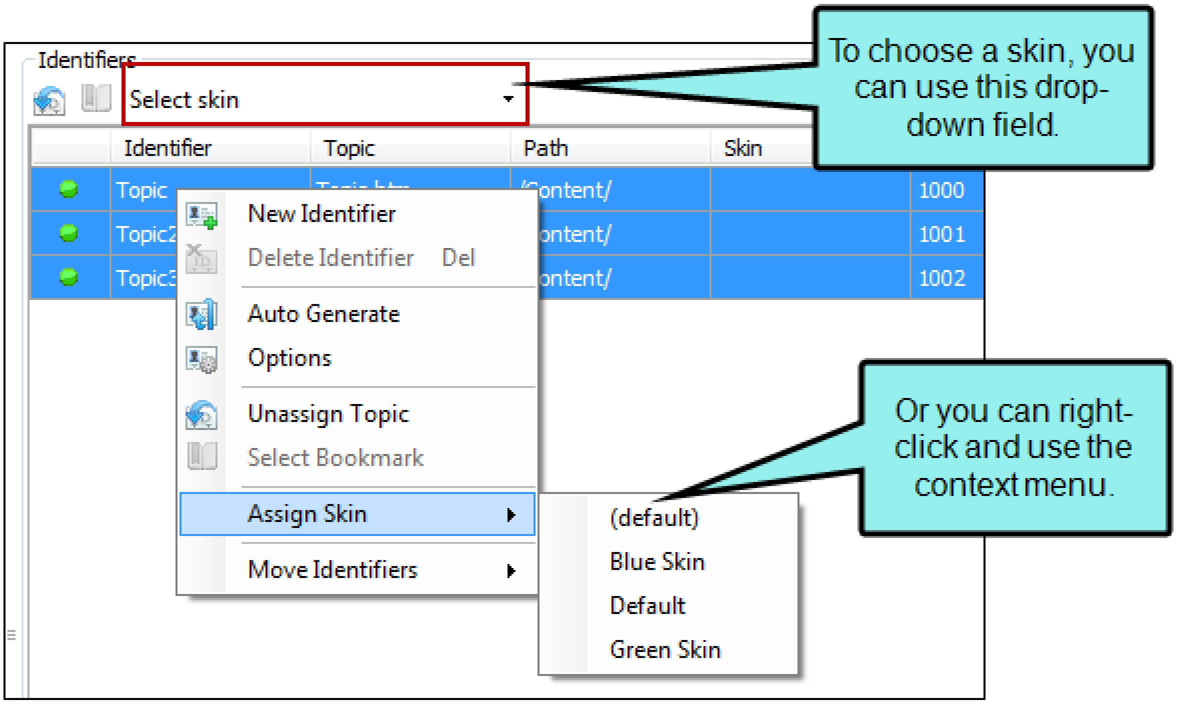
6. (Optional) If you want the ID to point to a specific bookmark or header in the topic, do the following:
   a. Select the ID row.
   b. Click the 📖 button located above the ID list.
   c. In the dialog that opens, select a bookmark or header within the topic.
   d. Click **OK**.

7. Click 💾 to save your work.

How to Associate Skins With IDs

You can use these steps for IDs assigned to topics only, not micro content phrases.

1. Open the alias file that you created.
2. Select the ID row(s). You can hold the **SHIFT** key to select a range, or you can hold the **CTRL** key to select individual items.
3. Do one of the following:
   - Click the arrow in the **Select skin** button Select skin ▾ (located above the ID list) and choose a skin from the drop-down.
   - Right-click somewhere in the list. From the context menu select **Assign Skin** and then from the submenu choose the name of the skin.



To choose a skin, you can use this drop-down field.

Or you can right-click and use the context menu.

4. Click 💾 to save your work.

Button and Context Menu Explanations

Following are explanations of the various buttons and context menu options in the Alias Editor, some of which have been included in the steps above.

Hide IDs that already have topics or phrases assigned to them. By doing this, you can concentrate solely on unassigned IDs.

Create a new ID.

> NOTE You can also perform this task by right-clicking in the **Identifiers** side of the editor, and from the context menu selecting **New Identifier**.

Delete the selected ID.

> NOTE You can also perform this task by right-clicking a selection in the **Identifiers** side of the editor, and selecting **Delete Identifier** from the context menu.

Automatically generate new IDs for all or many of the topics or phrases in your project.

> NOTE You can also perform this task by right-clicking in the **Identifiers** side of the editor, and from the context menu selecting **Auto Generate**.

	Specify options in advance for how new IDs are created. See "Setting Identifier Options" on page 69. NOTE You can also perform this task by right-clicking in the **Identifiers** side of the editor, and from the context menu selecting **Options**.
Unassigned Identifiers: 2	This lets you see if there are any issues with IDs in the editor, such as IDs that are not yet assigned to topics or phrases.
	This highlights the previous ID in the list that has an issue.
	This highlights the next ID in the list that has an issue.
	Assign the topic or phrase to an ID. NOTE You can also perform this task by right-clicking on the topic or phrase, and from the context menu selecting **Assign to Selected Identifier**.
	Create a new ID and assign the topic or phrase to it at the same time. NOTE You can also perform this task by right-clicking on the topic or phrase, and from the context menu selecting **Assign to New Identifier**.

	Open the selected topic or micro content file. NOTE You can also perform this task by right-clicking on the topic or phrase, and from the context menu selecting **Open**.
	This displays all header files where the selected topic or phrase is assigned to an ID. NOTE You can also perform this task by right-clicking on the topic or phrase, and from the context menu selecting **Locate the in Header Files**.
	Unassign the topic or phrase from the ID. NOTE You can also perform this task by right-clicking in the **Identifiers** side of the editor, and from the context menu selecting **Unassign**.
	Select a bookmark in the selected topic. First you can select an ID row that has already been assigned to a topic and then click this new button. A dialog opens, letting you select a bookmark within that topic. NOTE You can also perform this task by right-clicking in the **Identifiers** side of the editor, and from the context menu selecting **Select Bookmark**.

Select skin ▾	You can click the down arrow in the **Select Skin** button and choose the appropriate skin for the ID.
🟠	This icon displays next to any IDs that are not assigned to a topic or phrase.
🟢	This icon displays next to any IDs that have already been assigned to a topic or phrase.
🔴	This icon displays next to any IDs with broken links.

What’s Noteworthy?

NOTE You can move IDs from one header file to another by using the right-click context menu. See "Moving Identifiers to Different Headers" on page 83.

NOTE You cannot have the same identifier more than once in the same alias file.

Associating an Alias File With a Target

If you have added more than one alias file for your project, you need to associate the appropriate alias file with the target that you plan to build. If you do not specify an alias file in a target, Flare uses the first alias file listed in the Project Organizer.

How to Associate an Alias File With a Target

1. Open the target.
2. In the Target Editor, click the **Advanced** tab.
3. Scroll down to the **Alias File** field, click the drop-down, and select the alias file that you want to associate with the target.
4. Click 💾 to save your work.

Providing a Developer With Files and Information

Since producing context-sensitive Help (CSH) is a collaborative process, it is important that you maintain good communication with the developer, and provide the necessary information to that person. You can give this to the developer by emailing it or using a similar method. Alternatively, you can export it from the Flare project to a location where the developer can access it. The advantage of exporting it is that you can select a specific file format, depending on the developer's preference.

Exporting Header Files

If it is decided that you are responsible for creating the header file (i.e., the file containing an .h extension in your project), you must provide a copy of it to the software developer. You can do this by exporting the header file so that the developer has access to it. If you make further changes to the header file, you need to ensure that the developer receives a new copy of it.

How to Export Header Files

1. Do one of the following, depending on the part of the user interface you are using:

    - **Ribbon** Select **Tools > Export Header File(s)**.
    - **Right-Click** In the Project Organizer, right-click the **Advanced** folder and select **Export Header File(s)**.

    The Export Header Files dialog opens.

2. On the left side of the dialog, select the format(s) that you want to use for exporting the header file(s). Work with your developer to determine the appropriate type of format:
    - C/C++ (.h) If this option is selected, a copy of the header file will be created with an .h file extension.
    - Visual Basic (.bas) If this option is selected, a copy of the header file will be created with a .bas file extension.
    - Java (.properties) If this option is selected, a copy of the header file will be created with a .properties file extension.
    - Delphi Pascal (.inc) If this option is selected, a copy of the header file will be created with an .inc file extension.
3. On the right side of the dialog, select the header file(s) to be exported.
4. Click the Browse button and select the location where the exported file(s) will be sent.
5. Click Export.
6. (Optional) If the file(s) are not in a shared location where the developer can retrieve them, you need to copy the exported files(s) from that location and send them to the developer.

Information for the Developer

Following is additional information you need to provide to the developer depending on your output type. The developer can use this information to make CSH calls from the software application or website.

- HTML5 "CSH Calls for HTML5 Output" on page 47
- WebHelp or WebHelp Plus "CSH Calls for WebHelp and WebHelp Plus" on page 53
- HTML Help "CSH Calls for HTML Help" on page 59

If you are using a special skin for your CSH, provide the developer with the skin ID as well.

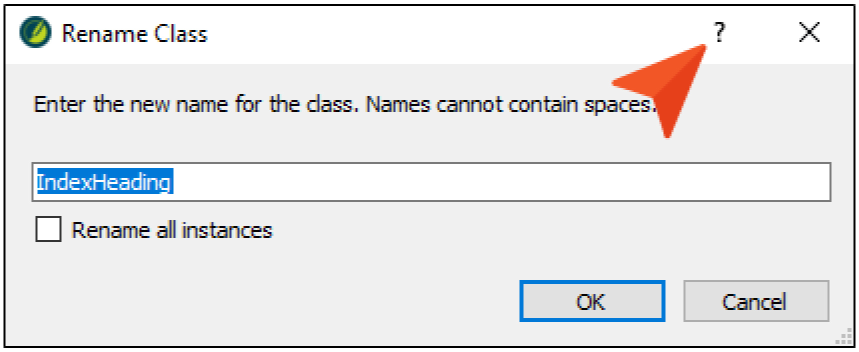
Making Context-Sensitive Help Calls

Supported In:

HTML 5 · MS HTML HELP · WEB HELP · WEB HELP PLUS

Regardless of who is responsible for creating the header file (the author or the developer) when producing CSH, there is some additional information you need to provide to the developer. This will help that person to complete the hooks to your company's software user interface.

For HTML5 and the WebHelp output types, there are two general methods the developer might use—JavaScript or URL. HTML Help is an older output type that requires somewhat different coding.



Enter the new name for the class. Names cannot contain spaces.

IndexHeading

□ Rename all instances

Information for Developers

- **HTML5** "CSH Calls for HTML5 Output" on the next page
- **WebHelp or WebHelp Plus** "CSH Calls for WebHelp and WebHelp Plus" on page 53
- **HTML Help** "CSH Calls for HTML Help" on page 59

CSH Calls for HTML5 Output

Use the following information if you are producing HTML5 and want to incorporate context-sensitive Help (CSH) into the software application.

There are two methods you can use:

- **Method 1—JavaScript** Using this method requires calling a JavaScript function that Flare provides.
- **Method 2—URL** Using this method, you can create a hyperlink to launch the Help system.

> NOTE If you are using multiple skins for CSH, the author should select the option on the Advanced tab of the Target Editor to generate all skins.

Which Method is Best for You?

Each method has its unique benefits. Generally speaking, the JavaScript method lets you have more control, whereas the URL method is a bit more quick and simple.

One reason to choose the JavaScript method is to better control the window size and location. With the URL method, the browser window automatically starts to open at the same size and location as the previous time that browser window was opened. But if you have specified a different size and location for your output window, the window will visibly move and resize accordingly. The JavaScript method prevents this type of behavior by opening the window directly to the size and location you specified. You would set the window size and location in the skin. Then in the JavaScript call you would specify the appropriate skin.

Another benefit to using the JavaScript method is that it is required in order for the Browser Settings option to take effect. This option can be found on the Setup tab of the Skin Editor.

If you use products like Fortify and Veracode to run security scans on your output, having JavaScript files in your HTML5 target might prompt security warnings when scanning it. If security warnings are displayed during these scans, you may want to exclude JavaScript files from your output. See "Excluding JavaScript for CSH Calls in HTML5 Output" on page 88.

How to Use the JavaScript Method for CSH Calls

1. Add a reference to the JavaScript file (which is created automatically when the author builds the output). This .js file should be named "csh.js." The reference to the JavaScript file should use the following format.

   ```
   <script type="text/javascript" src="[path of file]/csh.js"></script>
   ```

   > NOTE Make sure you use forward slashes (/) in the src path to the file, even if the file is referenced locally.

2. Create a trigger and add the command to let users open the Help system. Here is a format that you can use to add a button.

   ```
   <input type="button" value="Open Help System" onclick="MadCap.OpenHelp(ID, skin
   name, search string, first pick search string value );" />
   ```

   You can change the input type and the value if necessary. The most important parts that you will adjust are the elements within parentheses (ID, skin name, search string, first pick search string value).

   - ID This can be either the ID name or number. The topic (or micro content phrase) and skin associated with the ID will be used. If no skin is associated with the ID in Flare, the skin name that you provide in this command will be used.

     Alternatively, the ID may contain a topic path. In this case, the specified topic will be loaded with the skin that is specified in this command. The topic path must be relative to the Content folder of the Flare project. You also have the option of entering "null" instead of an ID to use the Help system's default starting topic.

   - Skin Name This is the name of the skin to use when opening the Help system. If a skin has been assigned to the ID in Flare *and* you enter a skin name in this command, the skin name in the command will take precedence. You also have the option of entering "null" instead of a skin name if you want to use the Help system's default skin or to use the skin that is associated with the CSH ID in Flare.

- **Search String** This is an optional element that automatically performs a search for a specific string.
- **First Pick Search String Value** This element can be used in conjunction with the search string. If you use the first pick option, you can include a true or false value. If the value is true, the first topic found with the specified search string will be opened automatically. If the value is false, the search results will simply be displayed; the first topic will not be opened automatically.

In the following example, the topic and skin associated with "Welcome" will be used. No search string information is included.

```
<input type="button" value="Open Help System" onclick="MadCap.OpenHelp('Welcome', null,
null, null );" />
```

In the following example, the topic associated with "Welcome" will be used. "BlueSkin" will override the skin associated with "Welcome." No search string information is included.

```
<input type="button" value="Open Help System" onclick="MadCap.OpenHelp('Welcome',
'BlueSkin', null, null );" />
```

In the following example, the topic and skin associated with the ID value 1000 will be used. No search string information is included.

```
<input type="button" value="Open Help System" onclick="MadCap.OpenHelp(1000, null,
null, null );" />
```

In the following example, the topic associated with the ID value 1000 will be used. "BlueSkin" will override the skin associated with ID value 1000. No search string information is included.

```
<input type="button" value="Open Help System" onclick="MadCap.OpenHelp(1000,
'BlueSkin', null, null );" />
```

In the following example, "Company/Employees.htm" will be used with the default skin. No search string information is included.

```
<input type="button" value="Open Help System" onclick="MadCap.OpenHelp
('Company/Employees.htm', null, null, null );" />
```

In the following example, both the default topic and skin will be used. A search will automatically be performed for the words "quarterly report," but the first topic found in the search *will not* be opened automatically.

```
<input type="button" value="Open Help System" onclick="MadCap.OpenHelp(null, null,
'quarterly report', false );" />
```

In the following example, the default topic will be used with "BlueSkin." A search will automatically be performed for the words "quarterly report," and the first topic found in the search *will* be opened automatically.

```
<input type="button" value="Open Help System" onclick="MadCap.OpenHelp(null,
'BlueSkin', 'quarterly report', true );" />
```

How to Use the URL Method for CSH Calls

Create a trigger and add a link to let users open a specific area of the Help system.

There is a certain amount of flexibility in terms of how you create the link and what you can include in it. Here is the basic structure of the link, although not all of the information shown below is required.

```
[main entry file].htm#cshid=[ID number, ID name, or topic path/name]&searchQuery=
[search string]&firstPick=true
```

After #, you can specify any combination of the parameters (cshid, searchQuery, firstPick), separated by ampersands (&). The order of the parameters does not matter.

- **Main entry file** Provide the path to the main entry file for the output. The file name is determined by whatever you enter into the **Output File** field in the **General** tab of the Target Editor. If you do not provide a name in this field, the name "Default" will be used.
- **cshid** This can be either the identifier name or number. The topic (or micro content phrase) and skin that is associated with the ID will be used (unless you override it in this link by specifying a different skin name).

  Alternatively, you can enter the path and name of the specific topic to which you want to link. If you use this element, an ID does not need to be created. The topic path must be relative to the Content folder of the Flare project.
- **searchQuery** This is an optional element that automatically performs a search for a specific string.
- **firstPick** This element can be used in conjunction with the search string. If you include the first pick option, the first topic found with the specified search string *will* be opened automatically. If you do not include this element, the search results will simply be displayed; the first topic *will not* be opened automatically.

In the following examples, these values are used:

- Default.htm = main entry file name
- 1000 = CSH ID value
- Soccer = CSH ID name
- Soccer.htm = topic in the project, at the root level of the Content Explorer
- World Cup Standings = search term

```
<a href="http://my.mycompany.com/Default.htm#cshid=Soccer">Click here to open</a>
```

```
<a href="http://my.mycompany.com/Default.htm#cshid=1000&searchQuery=World Cup
Standings&firstPick=true">Click here to open</a>
```

```
<a href="http://my.mycompany.com/Default.htm#cshid=Soccer&searchQuery=World Cup
Standings&firstPick=true">Click here to open</a>
```

```
<a href="http://my.mycompany.com/Default.htm#cshid=Soccer.htm&searchQuery=World Cup
Standings&firstPick=true">Click here to open</a>
```

CSH Calls for WebHelp and WebHelp Plus

IMPORTANT WebHelp and WebHelp Plus are deprecated in Flare, which means that they are slated to be removed in a future version.

Use the following information if you are producing WebHelp or WebHelp Plus and want to incorporate context-sensitive Help (CSH) into the software application.

There are two methods you can use:

- **Method 1—JavaScript** Using this method requires calling a JavaScript function that Flare provides.
- **Method 2—URL** Using this method, you can create a hyperlink to launch the Help system.

Which Method is Best for You?

Each method has its unique benefits. Generally speaking, the JavaScript method lets you have more control, whereas the URL method is a bit more quick and simple.

One reason to choose the JavaScript method is to better control the window size and location. With the URL method, the browser window automatically starts to open at the same size and location as the previous time that browser window was opened. But if you have specified a different size and location for your output window, the window will visibly move and resize accordingly. The JavaScript method prevents this type of behavior by opening the window directly to the size and location you specified. You would set the window size and location in the skin. Then in the JavaScript call you would specify the appropriate skin.

Another benefit to using the JavaScript method is that it is required in order for the Browser Settings option to take effect. This option can be found on the WebHelp Setup tab of the Skin Editor.

How to Use the JavaScript Method

1. Add a reference to the JavaScript file (which is created automatically when the author builds the output). This .js file is named after the WebHelp or WebHelp Plus output file and placed at the root level of the output folder. For example, if the output file is named "MyFirstHelp.htm," the JavaScript file is called "MyFirstHelp.js." The reference to the JavaScript file should use the following format.

   ```
   <script type="text/javascript" src="[path of file]/csh.js"></script>
   ```

   > NOTE Make sure you use forward slashes (/) in the src path to the file, even if the file is referenced locally.

2. Create a trigger and add the command to let users open the Help system. Here is a format that you can use to add a button.

   ```
   <input type="button" value="Open Help System" onclick="FMCOpenHelp(ID, skin
   name, search string, first pick search string value );" />
   ```

   You can change the input type and the value if necessary. The most important parts that you will adjust are the elements within parentheses (ID, skin name, search string, first pick search string value).

   - ID This can be either the identifier name or value. The topic and skin associated with the ID will be used. If no skin is associated with the ID in Flare, the skin name that you provide in this command will be used.

     Alternatively, the ID may contain a topic path. In this case, the specified topic will be loaded with the skin that is specified in this command. The topic path must be relative to the Content folder of the Flare project. You also have the option of entering "null" instead of an ID to use the Help system's default starting topic.

   - Skin Name This is the name of the skin to use when opening the Help system. If a skin has been assigned to the ID in Flare *and* you enter a skin name in this command, the skin name in the command will take precedence. You also have the option of entering "null" instead of a skin name if you want to use the Help system's default skin or to use the skin that is associated with the CSH ID in Flare.

- Search String This is an optional element that automatically performs a search for a specific string.
- First Pick Search String Value This element can be used in conjunction with the search string. If you use the first pick option, you can include a true or false value. If the value is true, the first topic found with the specified search string will be opened automatically. If the value is false, the search results will simply be displayed; the first topic will not be opened automatically.

In the following example, the topic and skin associated with "Welcome" will be used. No search string information is included.

```
<input type="button" value="Open Help System" onclick="FMCOpenHelp('Welcome', null,
null, null );" />
```

In the following example, the topic associated with "Welcome" will be used. "BlueSkin" will override the skin associated with "Welcome." No search string information is included.

```
<input type="button" value="Open Help System" onclick="FMCOpenHelp('Welcome',
'BlueSkin', null, null );" />
```

In the following example, the topic and skin associated with the ID value 1000 will be used. No search string information is included.

```
<input type="button" value="Open Help System" onclick="FMCOpenHelp(1000, null, null,
null );" />
```

In the following example, the topic associated with the ID value 1000 will be used. "BlueSkin" will override the skin associated with ID value 1000. No search string information is included.

```
<input type="button" value="Open Help System" onclick="FMCOpenHelp(1000, 'BlueSkin',
null, null );" />
```

In the following example, "Company/Employees.htm" will be used with the default skin. No search string information is included.

```
<input type="button" value="Open Help System" onclick="FMCOpenHelp
('Company/Employees.htm', null, null, null );" />
```

In the following example, both the default topic and skin will be used. A search will automatically be performed for the words "quarterly report," but the first topic found in the search *will not* be opened automatically.

```
<input type="button" value="Open Help System" onclick="FMCOpenHelp(null, null,
'quarterly report', false );" />
```

In the following example, the default topic will be used with "BlueSkin." A search will automatically be performed for the words "quarterly report," and the first topic found in the search *will* be opened automatically.

```
<input type="button" value="Open Help System" onclick="FMCOpenHelp(null, 'BlueSkin',
'quarterly report', true );" />
```

How to Use the URL Method

Create a trigger and add a link to let users open a specific area of the Help system.

There is a certain amount of flexibility in terms of how you create the link and what you can include in it. Here is the basic structure of the link.

```
[main entry file]_CSH.htm?[search string]|FirstPick#[ID or topic file name.htm]|[skin
name]
```

- Main Entry file_CSH.htm This is the main entry file for your output. The file name is determined by whatever you enter into the **Output File** field in the **General** tab of the Target Editor. If you do not provide a name in this field, the name "Default" will be used. After the file name, it is important that you add an underscore followed by "CSH."

> NOTE You do not need to include the "_CSH" portion in the file name of the topic in Flare. You only need to add " _CSH" to the web page hyperlink connecting to that topic.

- Search String This is an optional element that automatically performs a search for a specific string.
- First PickThis element can be used in conjunction with the search string. If you include the first pick option, the first topic found with the specified search string will be opened automatically. If you do not include this element, the search results will simply be displayed; the first topic will not be opened automatically.
- ID This can be either the ID name or number. The topic and skin that is associated with the ID will be used, unless you override it in this link by specifying a different skin name. If you do not want to use a map ID, you can use the topic file name.
- Topic Name This is the name of the specific topic to which you want to link. If you use this element, an ID does not need to be created. The topic path must be relative to the Content folder of the Flare project. If you do not want to use the topic name, you can use an ID.
- Skin Name If you use an ID in the link, the skin associated with that ID (if any) will be used to display the Help. However, you can specify a skin directly in the link, which will override the skin associated with that topic in the alias file.

In the following examples, these values are used:

- Default.htm = main entry file name
- 1000 = CSH ID value
- Soccer = CSH ID name
- Soccer.htm = topic in the project, at the root level of the Content Explorer
- World Cup Standings = search term
- Green = skin name

```
<a href="http://my.mycompany.com/Default_CSH.htm#Soccer|Green">Click here to open</a>
```

```
<a href="http://my.mycompany.com/Default_CSH.htm#1000">Click here to open</a>
```

```
<a href="http://my.mycompany.com/Default_CSH.htm#Soccer.htm">Click here to open</a>
```

```
<a href="http://my.mycompany.com/Default_CSH.htm?World Cup Standings#1000">Click here
to open</a>
```

```
<a href="http://my.mycompany.com/Default_CSH.htm?World Cup
Standings|FirstPick#Soccer">Click here to open</a>
```

CSH Calls for HTML Help

Information for Developers

Use the following information to connect Microsoft HTML Help to an application.

- Use the following syntax to call a topic using a map number:

```
HWND HtmlHelp(Window(), "c:\path\MyHelp.chm", HH_HELP_CONTEXT, Number);
```

- Use the following syntax to call a topic using a file name:

```
HWND HtmlHelp (Window(), "c:\path\MyHelp.chm", HH_DISPLAY_TOPIC,
"welcome.htm");
```

To hook context-sensitive Help (CSH) to an application, the code looks something like this:

```
HtmlHelp(hWnd, /*Window handle of program or dialog*/
"CSHHelp.chm", /*Name of the CHM file*/
HH_HELP_CONTEXT,
dwMapNumber); /*Number from header file*/
```

HtmlHelp(hWnd	This is the window handle of a program or dialog. A window handle helps identify a window so the HTML Help engine knows which application is performing the specific action.
MyHelp.chm	This is the name of a compiled HTML Help file (CHM) that includes the CSH. The actual name of the CHM file is determined by the Help author.
HH_HELP_CONTEXT	This is the command sent to the HTML Help engine for window-level Help.
dwMapNumber)	A map number from the header file.

Testing Context-Sensitive Help

After you create context-sensitive Help (CSH), you should test it to verify that it works. You can do this a couple of different ways, and it is not a bad idea to do both.

First, you can test your CSH identifiers from inside your project. If you are testing Microsoft HTML Help, WebHelp, or WebHelp Plus, you can use the Context-sensitive Help API Tester dialog. If you are testing HTML5 output, you can use the CSH Test page.

Second, you can test the CSH using a new build of the software application.

How to Test Context-Sensitive Help From Inside a Project

There are different ways to test CSH calls from inside a project. The method you use depends on the output type.

To Use the CSH Test Page—HTML5

1. Do one of the following, depending on the part of the user interface you are using:

   - **Ribbon** If you want to test the CSH for the primary target, select the **Tools** ribbon. In the **Context Sensitive Help** section, select **Test CSH API Calls**.

   - **Right-Click** In the Project Organizer, right-click a target under the **Targets** folder, and select **Test CSH API Calls**.

   > NOTE If the target's output is not up-to-date, click **Yes** when prompted to regenerate it.

   The CSH Test page opens in a web browser. For Chrome and Firefox users, the page opens in a new tab. For Safari users, the page opens in a new window.

2. Test the desired topic(s) using one of these options:

   - **ID Name or Number** In the **Topic ID** box, enter a an ID name or number (on the left side) or select it from the drop-down. You can find this information in the Identifier and Value columns of your alias file.

     OR

   - **Search for a Topic** Type a keyword in the **Search** box. Then under the **First Pick** label select **None**, **True**, or **False**.

3. (Optional) In the **Skin** drop-down list, select the desired skin for viewing the topic.

4. If you published your output to a web server and want to test a CSH call, type the web site path in the **Path to Help System** box. If you leave this box blank, Flare will test the CSH call to the Output folder.

> ☆ EXAMPLE If you published your help system to the root directory on a web server, you might type:
>
> ```
> http://help.example.com
> ```
>
> OR
>
> ```
> https://help.example.com
> ```
>
> If you are publishing multiple outputs on the same web server, you might type one of the following (where [Flare Target Name] = the publishing destination folder of the target):
>
> ```
> http://help.example.com/[Flare Target Name]
> ```
>
> OR
>
> ```
> https://help.example.com/[Flare Target Name]
> ```

5. Click **Show Help (Javascript)**. If the correct topic opens, the CSH link works.

6. When you have finished testing your topics, close your web browser.

To Use the Test CSH API Call Dialog—HTML Help, WebHelp, WebHelp Plus

1. Do one of the following, depending on the part of the user interface you are using:
   - **Ribbon** If you want to test the CSH for the primary target, select the **Tools** ribbon. In the **Context Sensitive Help** section, select **Test CSH API Calls**.
   - **Right-Click** In the Project Organizer, right-click a target under the **Targets** folder, and select **Test CSH API Calls**.

   > NOTE If the target's output is not up-to-date, click **Yes** when prompted to regenerate it.

   The Context-sensitive Help API Tester dialog opens.
2. (Optional) In the **Skin** drop-down list, select the desired skin for viewing the topic.
3. Next to each identifier, click Test. If the correct Help topic opens, the CSH link works.
4. When you have finished testing your topics, click **Close**.

How to Test Context-Sensitive Help Using a Build From the Application

1. After the developer finishes "hooking" the identifiers in the application, obtain a new build from the developer.
2. Launch the application build that you obtained from the developer.
3. Open a dialog or window that should be tied to a CSH topic.
4. Click the **Help** button in the dialog or window, or press **F1** on your keyboard. The topic associated with the dialog or window should open accordingly.
5. Test all of the CSH-related dialogs or windows in this same way.

CHAPTER 4

Other Activities and Information for Context-Sensitive Help

There are some additional tasks you might perform regarding this feature.

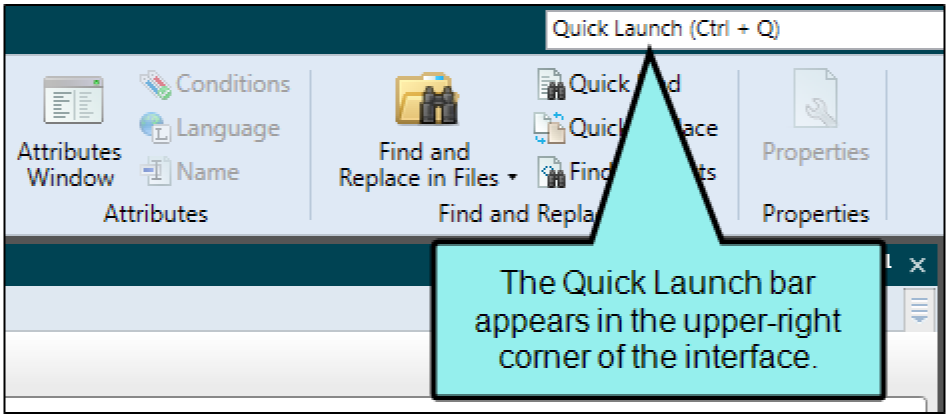
This chapter discusses the following:

Opening Header Files

When you want to work on an existing header file directly (as opposed to populating it by using the alias file), use the following steps to open it.

How to Open a Header File From the Quick Launch Bar

The Quick Launch bar lets you search for any Flare file or command. It is located in the upper-right corner of the interface. You can press CTRL+Q on your keyboard to move focus to the Quick Launch bar so you can begin typing.



The Quick Launch bar appears in the upper-right corner of the interface.

1. In the Quick Launch bar, type a few letters of the name of the file you want to open. Any available results appear in a drop-down list.
2. From the list, click the name of the file.

How to Open a Header File From the Project Organizer

1. Open the Project Organizer.
2. Double-click the **Advanced** folder. The header file(s) in your project are displayed (next to any other files that you have created, such as browse sequences, search filter sets, or alias files).
3. Double-click the header file that you want to open. The Text Editor opens to the right, with the header file page shown.

Importing Header Files

Not only can you add new header files to Flare, but you can also import existing header files (e.g., files with .h or .hh extensions).

How to Import a Header File

1. Do one of the following, depending on the part of the user interface you are using:
   - **Ribbon** Select **Project > New > Advanced > Header File**.
   - **Right-Click** In the Project Organizer, right-click on the **Advanced** folder and from the context menu select **Add Header File**.

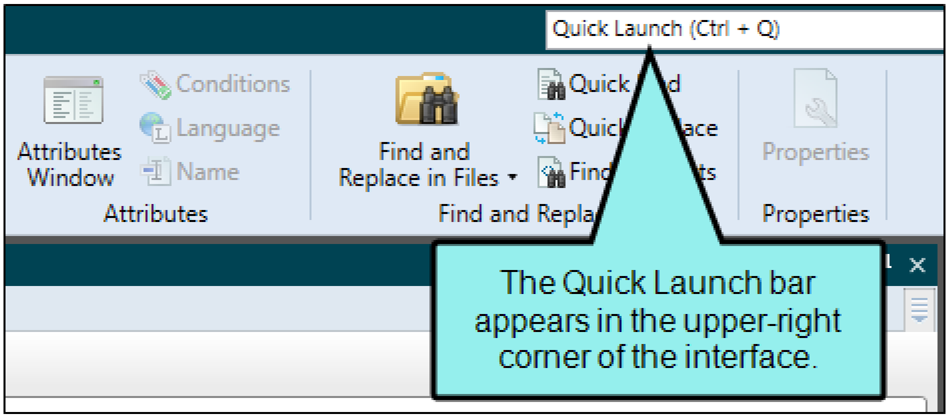
   The Add File dialog opens.
2. In the **File Type** field at the top, make sure **Header File** is selected.
3. Select **New from existing** and click ....
4. Find and select the header file that you want to import.
5. Click **Open**. The Source File field now contains the path to the file that you are importing. Also, the name of the file is displayed in the File Name field.
6. If you want to give the header file a different name than that for the imported file, click in the **File name** field and replace the text.
7. (Optional) If you want to apply condition tags to the file, expand the **Attributes** section at the bottom of the dialog. Next to the **Condition Tags** field, click ... and select the conditions you want to apply. Click **OK**.
8. (Optional) If you want to apply file tags, expand the **Attributes** section at the bottom of the dialog. Next to the **File Tags** field, click ... and select the file tags you want to apply. Click **OK**.
9. Click **Add**. The header file is added.

Opening Alias Files

When you want to work on an existing alias file, use the following steps to open it.

How to Open an Alias File From the Quick Launch Bar

The Quick Launch bar lets you search for any Flare file or command. It is located in the upper-right corner of the interface. You can press CTRL+Q on your keyboard to move focus to the Quick Launch bar so you can begin typing.



1. In the Quick Launch bar, type a few letters of the name of the file you want to open. Any available results appear in a drop-down list.
2. From the list, click the name of the file.

How to Open an Alias File From the Project Organizer

1. Open the Project Organizer.
2. Double-click the Advanced folder. The alias file(s) in your project are displayed (next to any other files that you have created, such as browse sequences, search filter sets, or header files).
3. Double-click the alias file that you want to open. The Alias Editor opens to the right, with the alias file page shown.

Setting Identifier Options

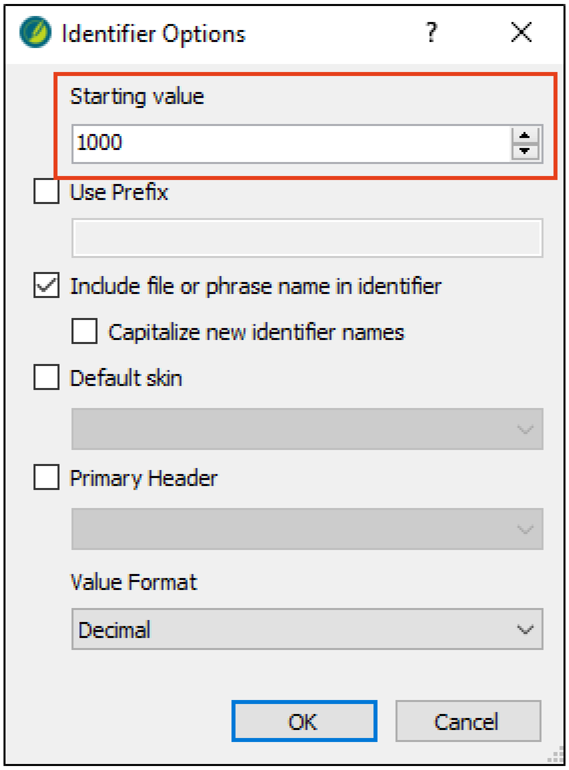
When creating context-sensitive Help (CSH), you can set options for new identifiers (IDs) in advance. Doing this will supply some of the information (e.g., starting value, prefix, include topic or phrase in ID name, assign skin) for you automatically as you create new IDs. See "Creating and Assigning Identifiers" on page 23.

How to Set ID Options

1. Open an alias file.
2. In the local toolbar of the Alias Editor click .
3. In the Identifier Options dialog, complete any of the options as necessary.

STARTING VALUE

Enter the starting value for IDs that you create. Additional IDs that are created will be incremented automatically based on that starting value.



Starting value: 1000

☐ Use Prefix

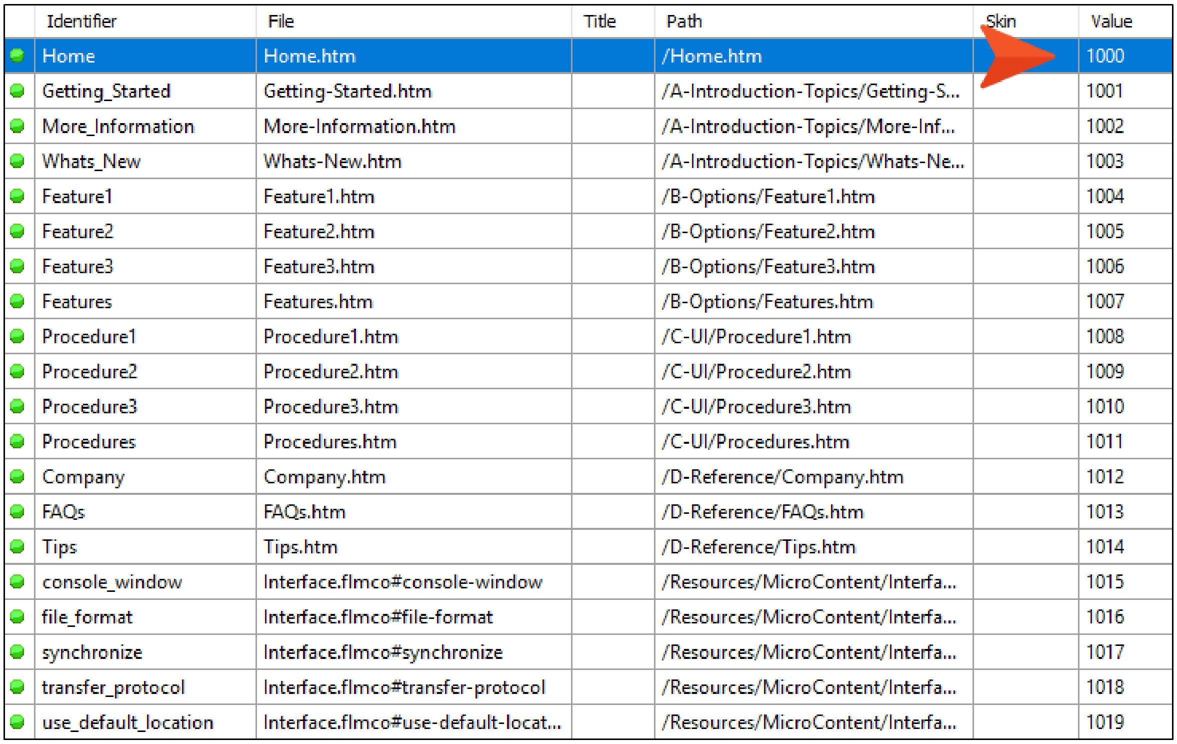
☑ Include file or phrase name in identifier

☐ Capitalize new identifier names

☐ Default skin

☐ Primary Header

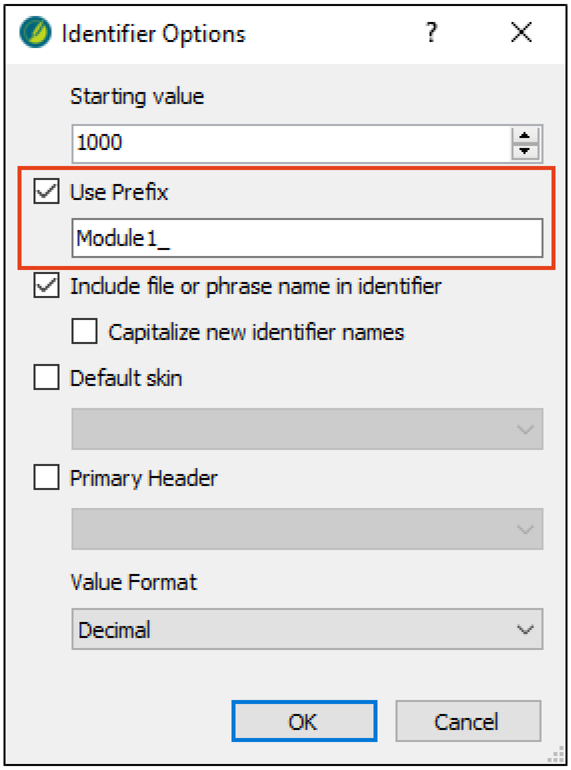
Value Format: Decimal



	Identifier	File	Title	Path	Skin	Value
●	Home	Home.htm		/Home.htm		1000
●	Getting_Started	Getting-Started.htm		/A-Introduction-Topics/Getting-S...		1001
●	More_Information	More-Information.htm		/A-Introduction-Topics/More-Inf...		1002
●	Whats_New	Whats-New.htm		/A-Introduction-Topics/Whats-Ne...		1003
●	Feature1	Feature1.htm		/B-Options/Feature1.htm		1004
●	Feature2	Feature2.htm		/B-Options/Feature2.htm		1005
●	Feature3	Feature3.htm		/B-Options/Feature3.htm		1006
●	Features	Features.htm		/B-Options/Features.htm		1007
●	Procedure1	Procedure1.htm		/C-UI/Procedure1.htm		1008
●	Procedure2	Procedure2.htm		/C-UI/Procedure2.htm		1009
●	Procedure3	Procedure3.htm		/C-UI/Procedure3.htm		1010
●	Procedures	Procedures.htm		/C-UI/Procedures.htm		1011
●	Company	Company.htm		/D-Reference/Company.htm		1012
●	FAQs	FAQs.htm		/D-Reference/FAQs.htm		1013
●	Tips	Tips.htm		/D-Reference/Tips.htm		1014
●	console_window	Interface.flmco#console-window		/Resources/MicroContent/Interfa...		1015
●	file_format	Interface.flmco#file-format		/Resources/MicroContent/Interfa...		1016
●	synchronize	Interface.flmco#synchronize		/Resources/MicroContent/Interfa...		1017
●	transfer_protocol	Interface.flmco#transfer-protocol		/Resources/MicroContent/Interfa...		1018
●	use_default_location	Interface.flmco#use-default-locat...		/Resources/MicroContent/Interfa...		1019

USE PREFIX

Specify a prefix to be added at the beginning of each new ID that you create (e.g., ID_, Dialog, Module1).



Starting value

1000

☑ Use Prefix

Module1_

☑ Include file or phrase name in identifier

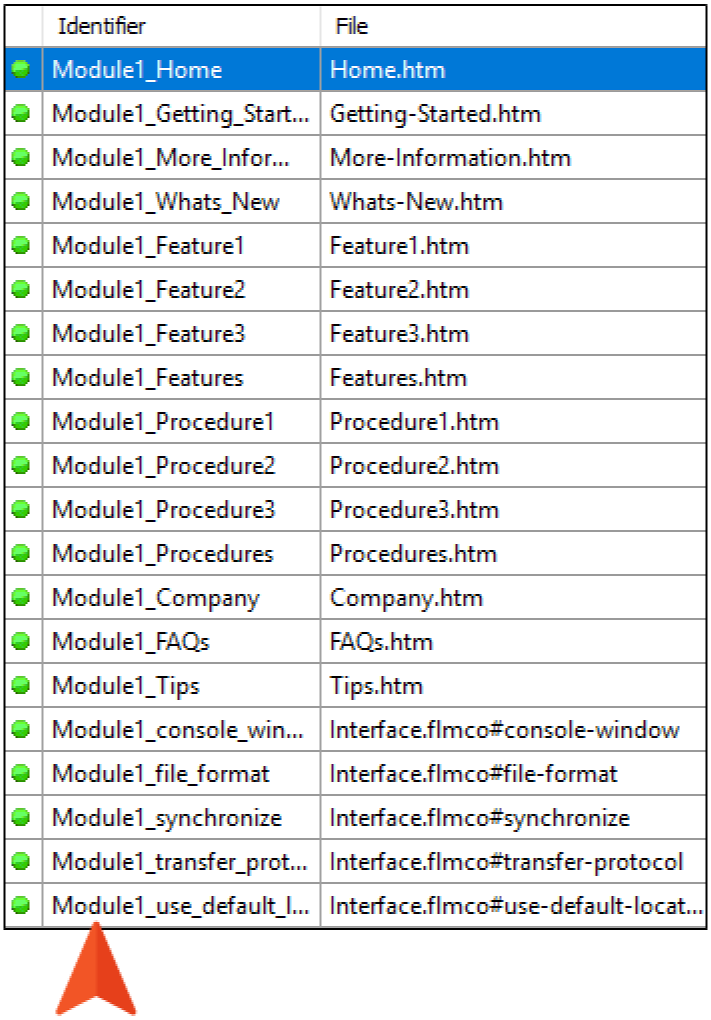
□ Capitalize new identifier names

□ Default skin

□ Primary Header

Value Format

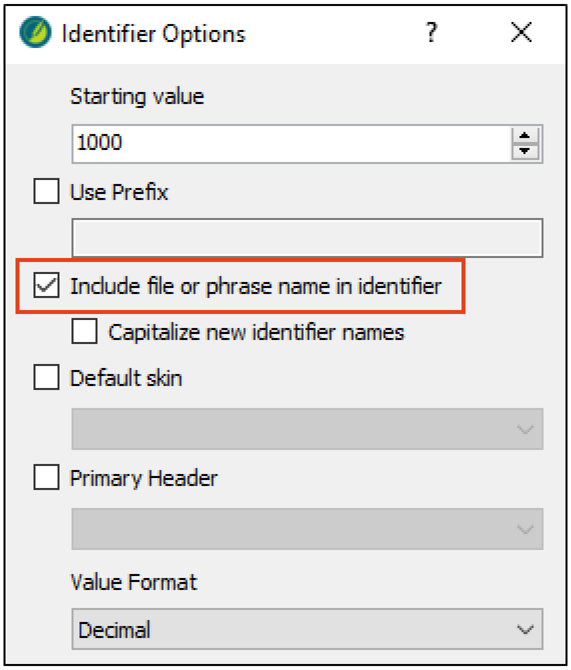
Decimal



	Identifier	File
●	Module1_Home	Home.htm
●	Module1_Getting_Start...	Getting-Started.htm
●	Module1_More_Infor...	More-Information.htm
●	Module1_Whats_New	Whats-New.htm
●	Module1_Feature1	Feature1.htm
●	Module1_Feature2	Feature2.htm
●	Module1_Feature3	Feature3.htm
●	Module1_Features	Features.htm
●	Module1_Procedure1	Procedure1.htm
●	Module1_Procedure2	Procedure2.htm
●	Module1_Procedure3	Procedure3.htm
●	Module1_Procedures	Procedures.htm
●	Module1_Company	Company.htm
●	Module1_FAQs	FAQs.htm
●	Module1_Tips	Tips.htm
●	Module1_console_win...	Interface.flmco#console-window
●	Module1_file_format	Interface.flmco#file-format
●	Module1_synchronize	Interface.flmco#synchronize
●	Module1_transfer_prot...	Interface.flmco#transfer-protocol
●	Module1_use_default_l...	Interface.flmco#use-default-locat...

INCLUDE FILE OR PHRASE NAME IN IDENTIFIER NAME

Select this check box if you want the names of the assigned topics or micro content phrases to be included automatically in the names of the new IDs. Flare will add underscores if a name or phrase has more than one word.



Identifier Options

Starting value

1000

□ Use Prefix

☑ Include file or phrase name in identifier

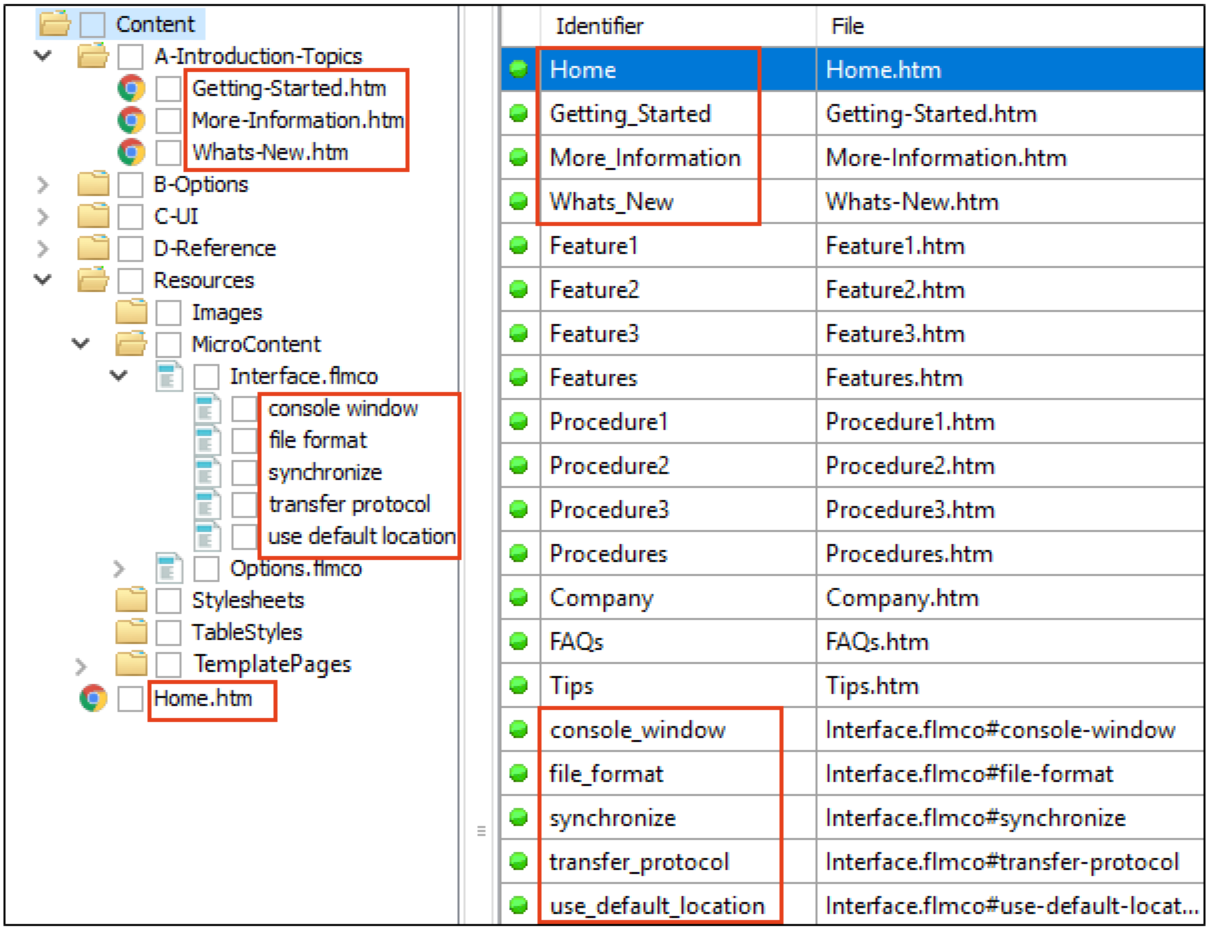
□ Capitalize new identifier names

□ Default skin

□ Primary Header

Value Format

Decimal

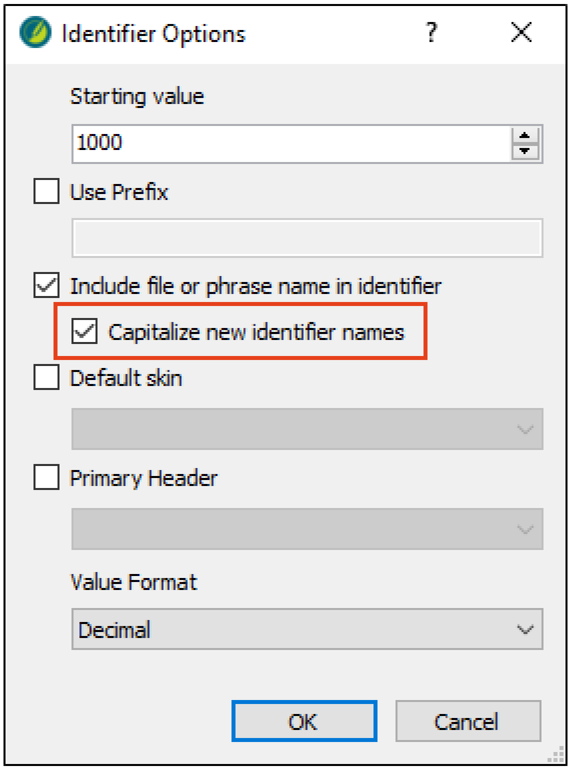


- □ Content
  - □ A-Introduction-Topics
    - □ Getting-Started.htm
    - □ More-Information.htm
    - □ Whats-New.htm
  - □ B-Options
  - □ C-UI
  - □ D-Reference
  - □ Resources
    - □ Images
    - □ MicroContent
      - □ Interface.flmco
        - □ console window
        - □ file format
        - □ synchronize
        - □ transfer protocol
        - □ use default location
      - □ Options.flmco
    - □ Stylesheets
    - □ TableStyles
    - □ TemplatePages
  - □ Home.htm

	Identifier	File
●	Home	Home.htm
●	Getting_Started	Getting-Started.htm
●	More_Information	More-Information.htm
●	Whats_New	Whats-New.htm
●	Feature1	Feature1.htm
●	Feature2	Feature2.htm
●	Feature3	Feature3.htm
●	Features	Features.htm
●	Procedure1	Procedure1.htm
●	Procedure2	Procedure2.htm
●	Procedure3	Procedure3.htm
●	Procedures	Procedures.htm
●	Company	Company.htm
●	FAQs	FAQs.htm
●	Tips	Tips.htm
●	console_window	Interface.flmco#console-window
●	file_format	Interface.flmco#file-format
●	synchronize	Interface.flmco#synchronize
●	transfer_protocol	Interface.flmco#transfer-protocol
●	use_default_location	Interface.flmco#use-default-locat...

CAPITALIZE IDENTIFIER NAMES

Select this check box if you want the name that is automatically added for new IDs to use all caps.



Starting value
1000

☐ Use Prefix

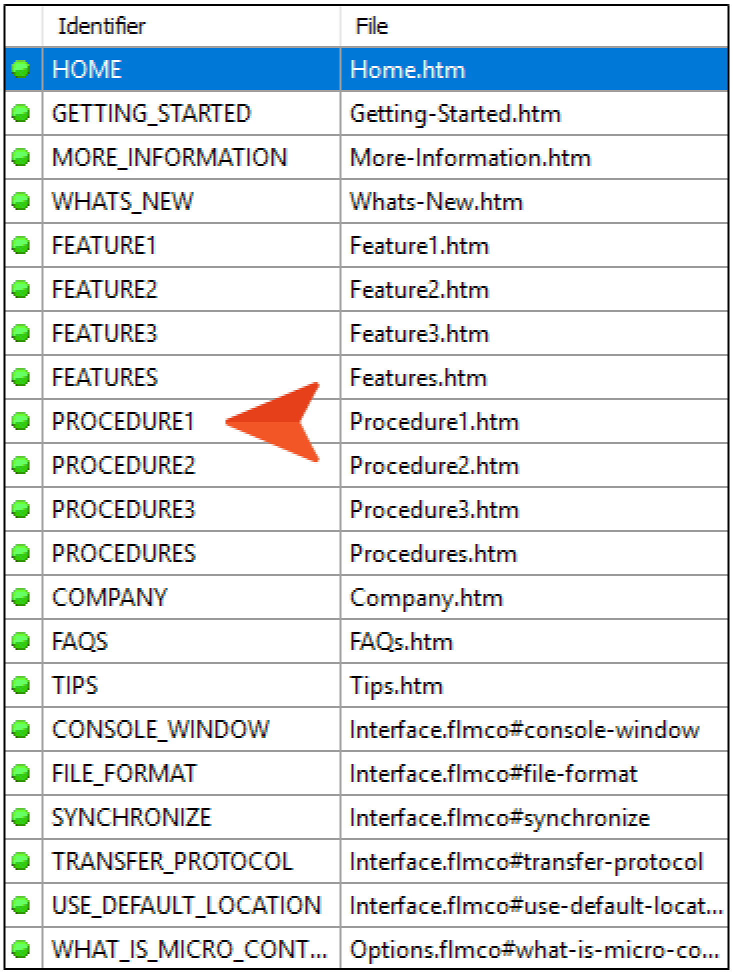
☑ Include file or phrase name in identifier

☑ Capitalize new identifier names

☐ Default skin

☐ Primary Header

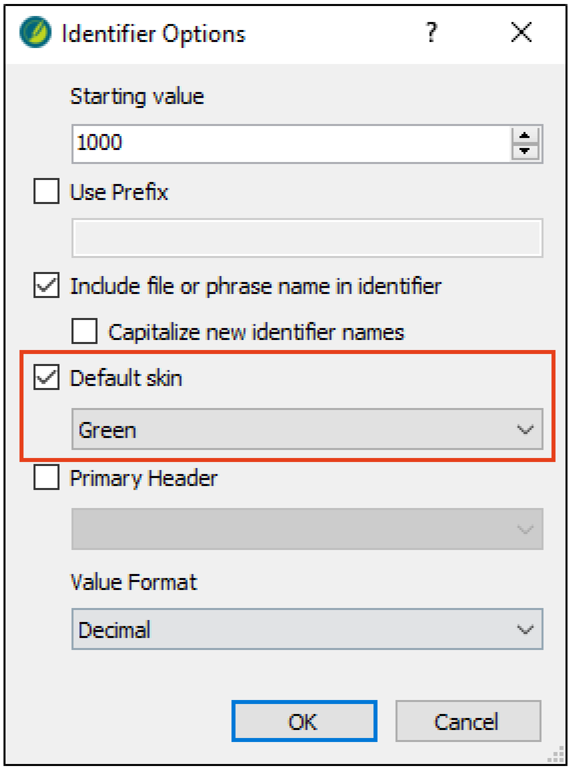
Value Format
Decimal



	Identifier	File
●	HOME	Home.htm
●	GETTING_STARTED	Getting-Started.htm
●	MORE_INFORMATION	More-Information.htm
●	WHATS_NEW	Whats-New.htm
●	FEATURE1	Feature1.htm
●	FEATURE2	Feature2.htm
●	FEATURE3	Feature3.htm
●	FEATURES	Features.htm
●	PROCEDURE1	Procedure1.htm
●	PROCEDURE2	Procedure2.htm
●	PROCEDURE3	Procedure3.htm
●	PROCEDURES	Procedures.htm
●	COMPANY	Company.htm
●	FAQS	FAQs.htm
●	TIPS	Tips.htm
●	CONSOLE_WINDOW	Interface.flmco#console-window
●	FILE_FORMAT	Interface.flmco#file-format
●	SYNCHRONIZE	Interface.flmco#synchronize
●	TRANSFER_PROTOCOL	Interface.flmco#transfer-protocol
●	USE_DEFAULT_LOCATION	Interface.flmco#use-default-locat...
●	WHAT_IS_MICRO_CONT...	Options.flmco#what-is-micro-co...

DEFAULT SKIN

Specify which skin should be assigned by default to new IDs that are created. You can always manually select a different skin for any ID afterward, but when you first create a new ID, it will initially be assigned to the default skin that you specified. This option applies to topics only, not to micro content phrases.



Identifier Options

Starting value

1000

☐ Use Prefix

☑ Include file or phrase name in identifier

☐ Capitalize new identifier names

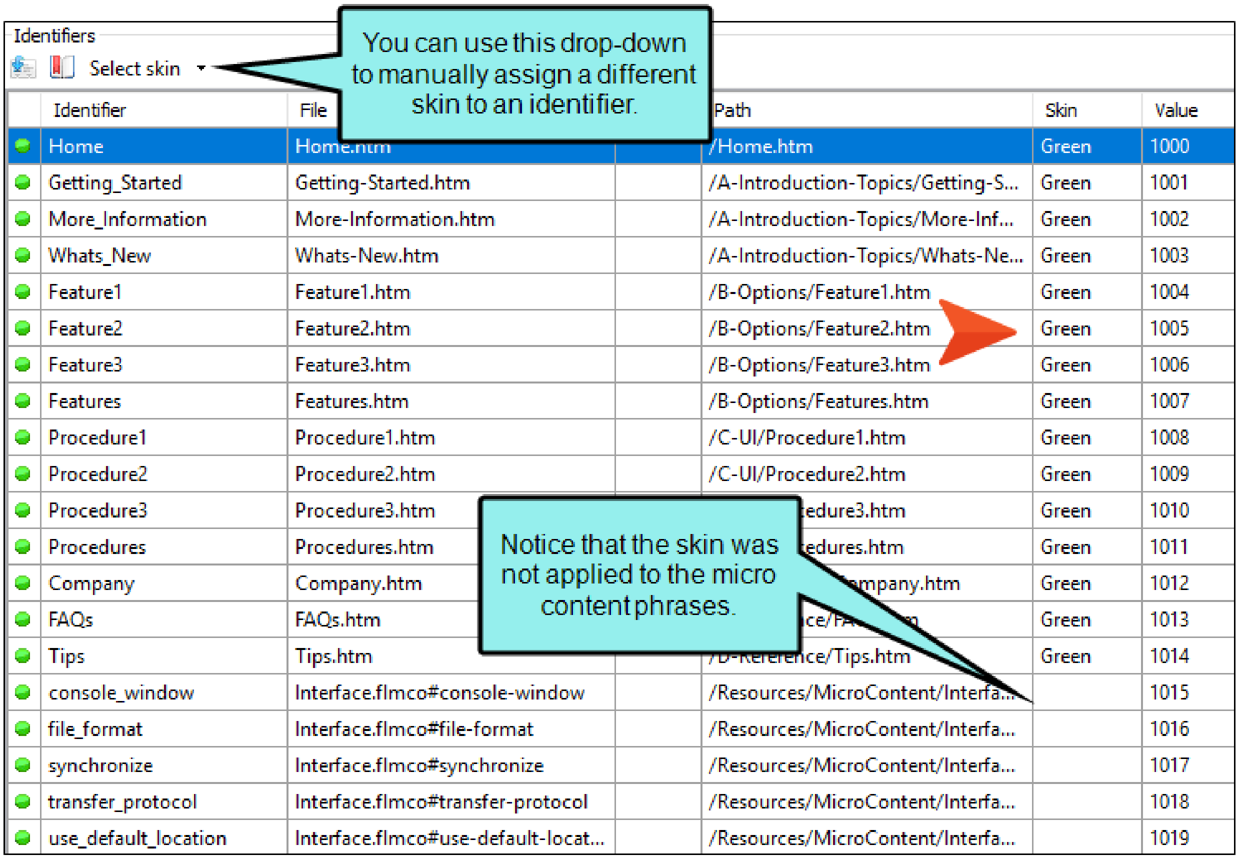
☑ Default skin

Green

☐ Primary Header

Value Format

Decimal



You can use this drop-down to manually assign a different skin to an identifier.

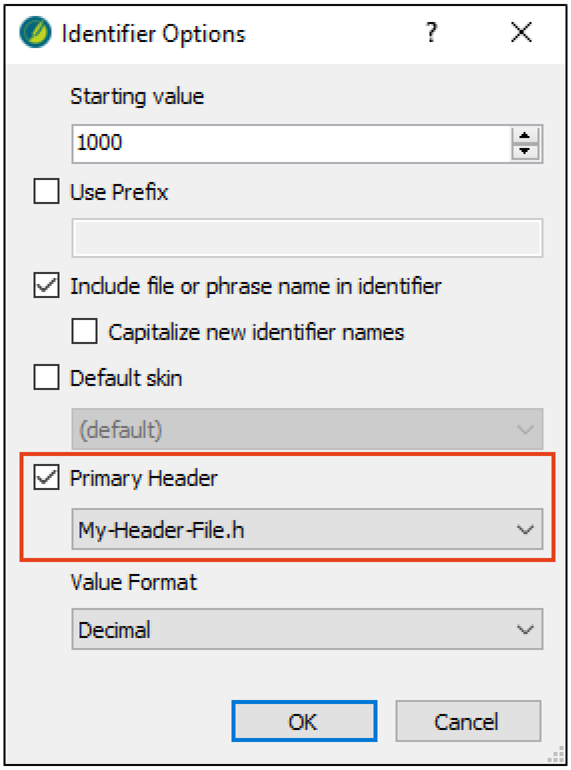
Notice that the skin was not applied to the micro content phrases.

	Identifier	File		Path	Skin	Value
●	Home	Home.htm		/Home.htm	Green	1000
●	Getting_Started	Getting-Started.htm		/A-Introduction-Topics/Getting-S...	Green	1001
●	More_Information	More-Information.htm		/A-Introduction-Topics/More-Inf...	Green	1002
●	Whats_New	Whats-New.htm		/A-Introduction-Topics/Whats-Ne...	Green	1003
●	Feature1	Feature1.htm		/B-Options/Feature1.htm	Green	1004
●	Feature2	Feature2.htm		/B-Options/Feature2.htm	Green	1005
●	Feature3	Feature3.htm		/B-Options/Feature3.htm	Green	1006
●	Features	Features.htm		/B-Options/Features.htm	Green	1007
●	Procedure1	Procedure1.htm		/C-UI/Procedure1.htm	Green	1008
●	Procedure2	Procedure2.htm		/C-UI/Procedure2.htm	Green	1009
●	Procedure3	Procedure3.htm		edure3.htm	Green	1010
●	Procedures	Procedures.htm		edures.htm	Green	1011
●	Company	Company.htm		mpany.htm	Green	1012
●	FAQs	FAQs.htm		ce/FA	Green	1013
●	Tips	Tips.htm		/D-Reference/Tips.htm	Green	1014
●	console_window	Interface.flmco#console-window		/Resources/MicroContent/Interfa...		1015
●	file_format	Interface.flmco#file-format		/Resources/MicroContent/Interfa...		1016
●	synchronize	Interface.flmco#synchronize		/Resources/MicroContent/Interfa...		1017
●	transfer_protocol	Interface.flmco#transfer-protocol		/Resources/MicroContent/Interfa...		1018
●	use_default_location	Interface.flmco#use-default-locat...		/Resources/MicroContent/Interfa...		1019

PRIMARY HEADER

Choose a header file, if you have more than one in your project.

When you are working in the Alias Editor, you can select a header file in the local toolbar. But what if you do not select one and "(all identifiers)" is shown in the drop-down field in the Alias Editor? In that case, the changes you make in the Alias Editor are applied to the primary header file that you have selected in the Identifier Options dialog. This setting will not be applied when you auto-generate IDs, but only when you manually create new IDs in the Alias Editor.



Identifier Options

Starting value

1000

□ Use Prefix

☑ Include file or phrase name in identifier

□ Capitalize new identifier names

□ Default skin

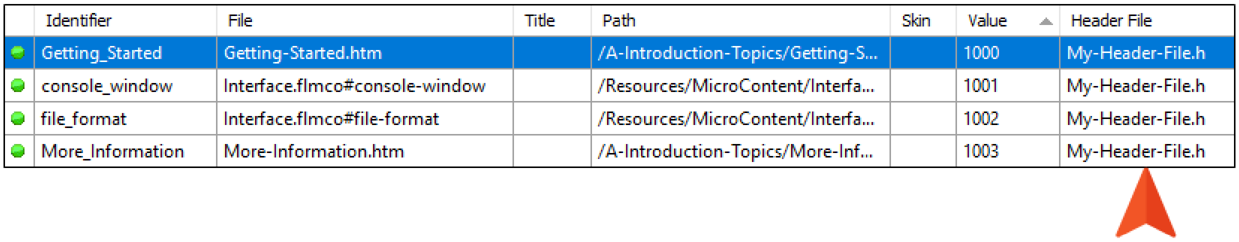
(default)

☑ Primary Header

My-Header-File.h

Value Format

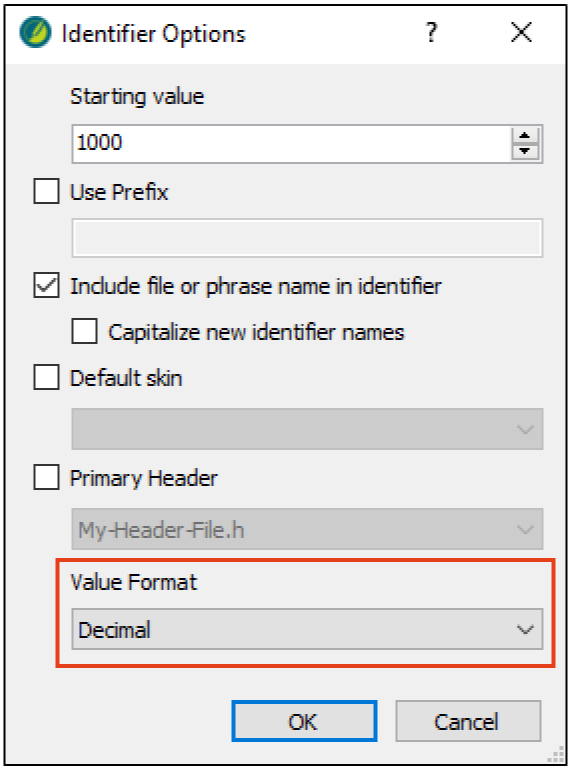
Decimal



	Identifier	File	Title	Path	Skin	Value	Header File
●	Getting_Started	Getting-Started.htm		/A-Introduction-Topics/Getting-S...		1000	My-Header-File.h
●	console_window	Interface.flmco#console-window		/Resources/MicroContent/Interfa...		1001	My-Header-File.h
●	file_format	Interface.flmco#file-format		/Resources/MicroContent/Interfa...		1002	My-Header-File.h
●	More_Information	More-Information.htm		/A-Introduction-Topics/More-Inf...		1003	My-Header-File.h

VALUE FORMAT

Specify whether the ID values should use a decimal or hexadecimal format. Using hexadecimal values does not affect your CSH in a different way; it's simply another option in case your developers prefer that format.



Identifier Options

Starting value

1000

□ Use Prefix

☑ Include file or phrase name in identifier

□ Capitalize new identifier names

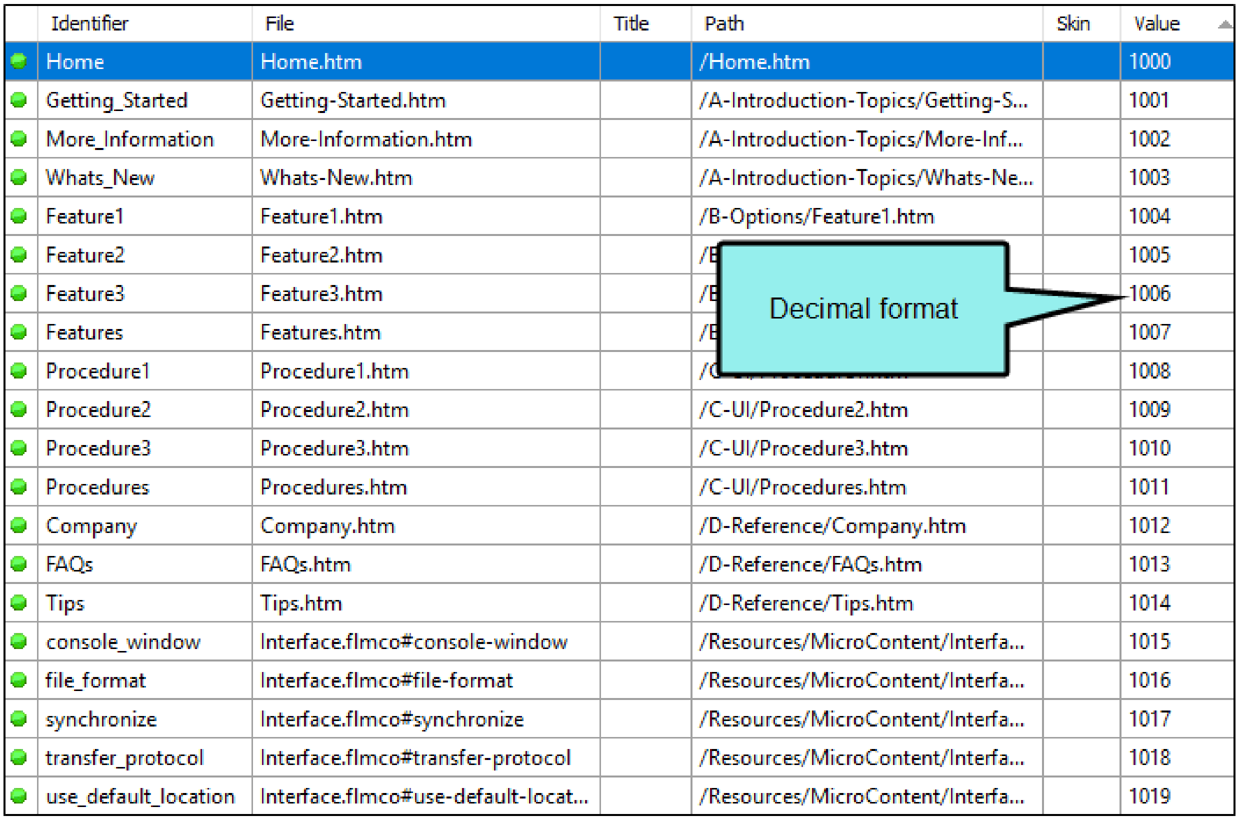
□ Default skin

□ Primary Header

My-Header-File.h

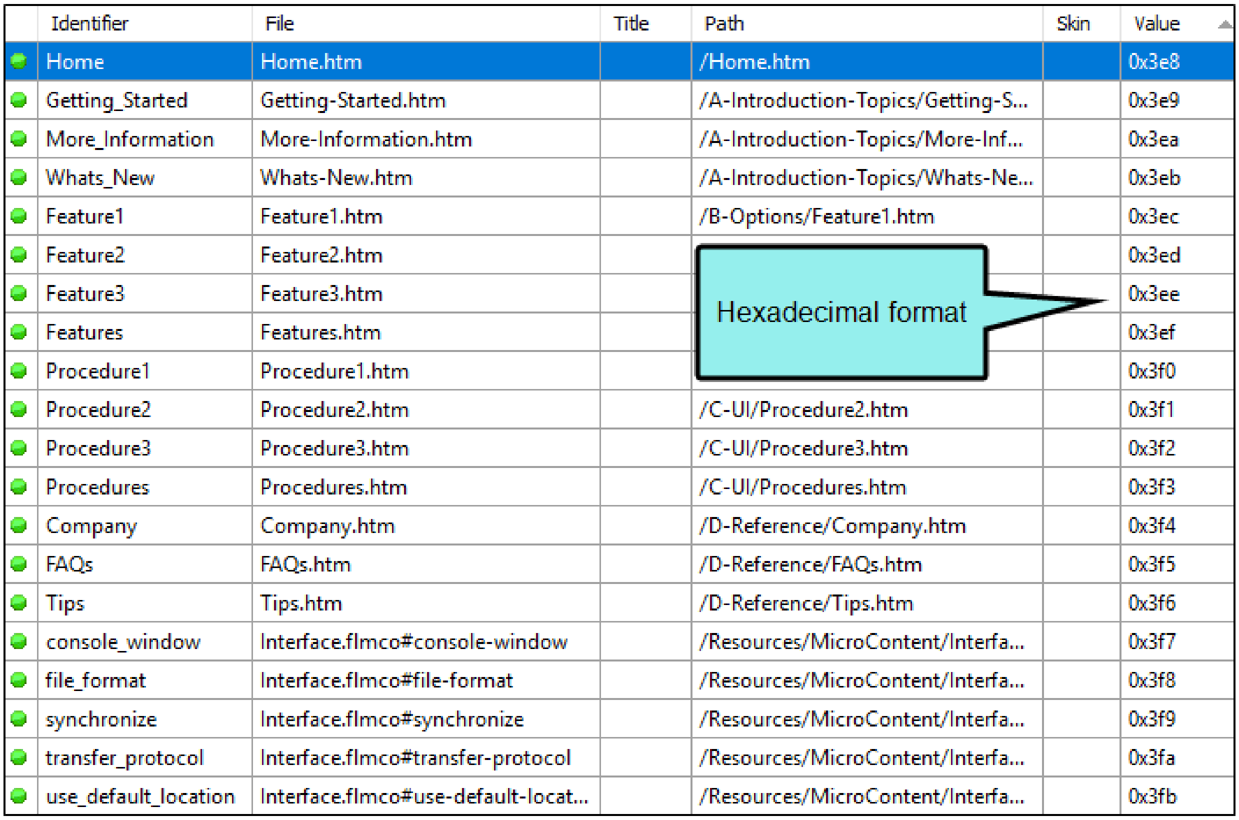
Value Format

Decimal



	Identifier	File	Title	Path	Skin	Value
●	Home	Home.htm		/Home.htm		1000
●	Getting_Started	Getting-Started.htm		/A-Introduction-Topics/Getting-S...		1001
●	More_Information	More-Information.htm		/A-Introduction-Topics/More-Inf...		1002
●	Whats_New	Whats-New.htm		/A-Introduction-Topics/Whats-Ne...		1003
●	Feature1	Feature1.htm		/B-Options/Feature1.htm		1004
●	Feature2	Feature2.htm		/B		1005
●	Feature3	Feature3.htm		/B		1006
●	Features	Features.htm		/B		1007
●	Procedure1	Procedure1.htm		/C		1008
●	Procedure2	Procedure2.htm		/C-UI/Procedure2.htm		1009
●	Procedure3	Procedure3.htm		/C-UI/Procedure3.htm		1010
●	Procedures	Procedures.htm		/C-UI/Procedures.htm		1011
●	Company	Company.htm		/D-Reference/Company.htm		1012
●	FAQs	FAQs.htm		/D-Reference/FAQs.htm		1013
●	Tips	Tips.htm		/D-Reference/Tips.htm		1014
●	console_window	Interface.flmco#console-window		/Resources/MicroContent/Interfa...		1015
●	file_format	Interface.flmco#file-format		/Resources/MicroContent/Interfa...		1016
●	synchronize	Interface.flmco#synchronize		/Resources/MicroContent/Interfa...		1017
●	transfer_protocol	Interface.flmco#transfer-protocol		/Resources/MicroContent/Interfa...		1018
●	use_default_location	Interface.flmco#use-default-locat...		/Resources/MicroContent/Interfa...		1019

Decimal format



	Identifier	File	Title	Path	Skin	Value
●	Home	Home.htm		/Home.htm		0x3e8
●	Getting_Started	Getting-Started.htm		/A-Introduction-Topics/Getting-S...		0x3e9
●	More_Information	More-Information.htm		/A-Introduction-Topics/More-Inf...		0x3ea
●	Whats_New	Whats-New.htm		/A-Introduction-Topics/Whats-Ne...		0x3eb
●	Feature1	Feature1.htm		/B-Options/Feature1.htm		0x3ec
●	Feature2	Feature2.htm				0x3ed
●	Feature3	Feature3.htm				0x3ee
●	Features	Features.htm				0x3ef
●	Procedure1	Procedure1.htm				0x3f0
●	Procedure2	Procedure2.htm		/C-UI/Procedure2.htm		0x3f1
●	Procedure3	Procedure3.htm		/C-UI/Procedure3.htm		0x3f2
●	Procedures	Procedures.htm		/C-UI/Procedures.htm		0x3f3
●	Company	Company.htm		/D-Reference/Company.htm		0x3f4
●	FAQs	FAQs.htm		/D-Reference/FAQs.htm		0x3f5
●	Tips	Tips.htm		/D-Reference/Tips.htm		0x3f6
●	console_window	Interface.flmco#console-window		/Resources/MicroContent/Interfa...		0x3f7
●	file_format	Interface.flmco#file-format		/Resources/MicroContent/Interfa...		0x3f8
●	synchronize	Interface.flmco#synchronize		/Resources/MicroContent/Interfa...		0x3f9
●	transfer_protocol	Interface.flmco#transfer-protocol		/Resources/MicroContent/Interfa...		0x3fa
●	use_default_location	Interface.flmco#use-default-locat...		/Resources/MicroContent/Interfa...		0x3fb

Hexadecimal format

4. Click **OK**.

Moving Identifiers to Different Headers

In the Alias Editor, you can move identifiers to a different header file (if you have multiple header files already). If you do not yet have a header file, you first need to add one (see "Adding Header Files" on page 18).

How to Move Identifiers to a Different Header

1. Open an alias file.
2. Select the identifiers that you want to move to another header file. You can hold the SHIFT key to select a range, or you can hold the CTRL key to select individual items.
3. Right-click and from the context menu select Move Identifiers > [Name of Header File].
4. Click 💾 to save your work.

Importing Alias Files

Not only can you add new alias files to Flare, but you can also import existing alias files (FLALI files).

How to Import an Alias File

1. Do one of the following, depending on the part of the user interface you are using:
   - **Ribbon** Select **Project > New > Advanced > Alias File**.
   - **Right-Click** In the Project Organizer, right-click on the **Advanced** folder and from the context menu select **Add Alias File**.

   The Add File dialog opens.
2. In the **File Type** field at the top, make sure **Alias File** is selected.
3. Select **New from existing** and click ....
4. Find and select the FLALI file that you want to import.
5. Click **Open**. The Source File field now contains the path to the file that you are importing. Also, the name of the file is displayed in the File Name field.
6. If you want to give the alias file a different name than that for the imported file, click in the **File name** field and replace the text.
7. (Optional) If you want to apply condition tags to the file, expand the **Attributes** section at the bottom of the dialog. Next to the **Condition Tags** field, click ... and select the conditions you want to apply. Click **OK**.
8. (Optional) If you want to apply file tags, expand the **Attributes** section at the bottom of the dialog. Next to the **File Tags** field, click ... and select the file tags you want to apply. Click **OK**.
9. Click **Add**. The alias file is added and opens in the Alias Editor.

Dynamic Help

The Dynamic Help window pane in Flare automatically displays the Help topic related to the active element in the interface. If you want to "lock" the window pane so that it continues to show the current Help topic while you move around in the interface, click 🔒 in the local toolbar of the window pane.

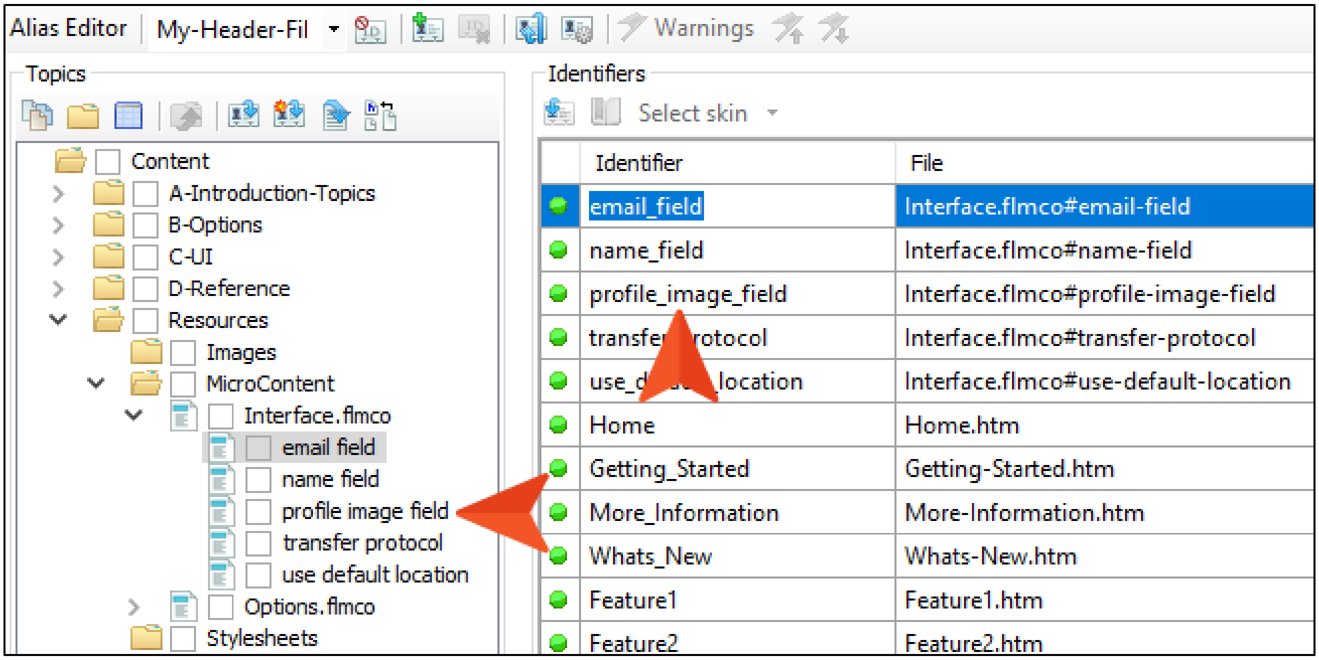
If you want to create dynamic Help for your own product(s), the process is essentially the same as it is for context-sensitive Help (CSH) in general. Work with your developer to come up with a plan. As the author, your job is still to create CSH IDs and link them to topics the same way you normally would, and make sure your developer uses the appropriate IDs wherever CSH is needed. The difference is that your developer might need to integrate a window pane in the software to display your Help output (if that is not being done already). Then, other parts of the product's interface (e.g., entire window panes) would be programmed to make the CSH call when a user clicks any part of it, as opposed to, say, clicking an isolated button.

Micro Content and Context-Sensitive Help

You can set context-sensitive Help (CSH) identifiers (IDs) on micro content phrases, therefore connecting the corresponding responses to specific parts of a software interface. And since micro content is intended to be quite short, this makes it ideal for field-level Help, as opposed to say, window- or dialog-level Help. Flare handles the connection between your micro content phrase/response combinations and CSH IDs; however, to link these IDs to a desktop application, your programmers would need to use whatever third-party software they normally use to complete their part of the work.

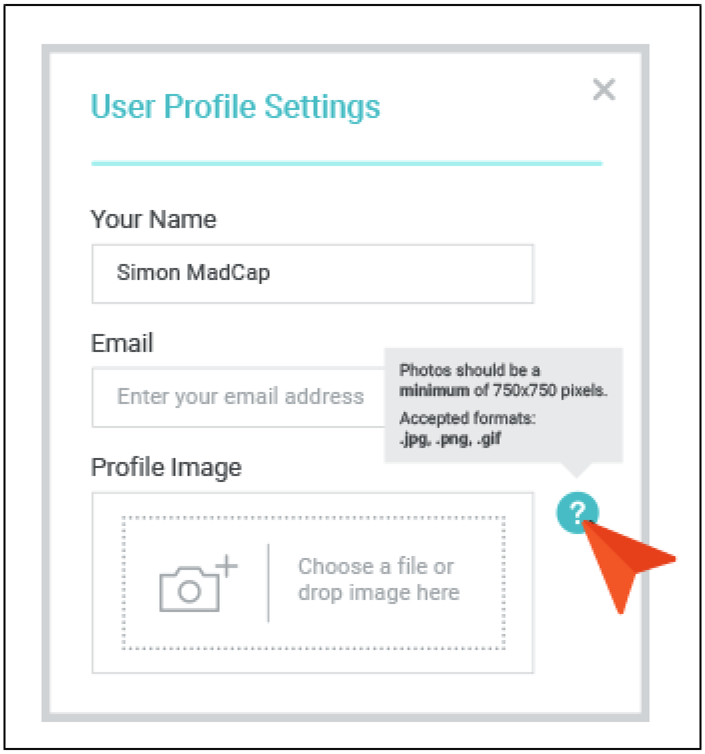
The process of creating CSH for micro content is essentially the same as it is for topics. See "Process for Context-Sensitive Help" on page 15.

- Author After assigning the IDs to the appropriate micro content phrases, you need to build the output and make it available to the developer. You also need to share the header file with the developer if you are responsible for creating it.
- Developer Based on the information in the header file, the developer associates the IDs with each area of your software's interface. The developer can use a JavaScript or URL method. See "CSH Calls for HTML5 Output" on page 47.



Content
A-Introduction-Topics
B-Options
C-UI
D-Reference
Resources
Images
MicroContent
Interface.flmco
email field
name field
profile image field
transfer protocol
use default location
Options.flmco
Stylesheets

	Identifier	File
●	email_field	Interface.flmco#email-field
●	name_field	Interface.flmco#name-field
●	profile_image_field	Interface.flmco#profile-image-field
●	transfer_protocol	Interface.flmco#transfer-protocol
●	use_default_location	Interface.flmco#use-default-location
●	Home	Home.htm
●	Getting_Started	Getting-Started.htm
●	More_Information	More-Information.htm
●	Whats_New	Whats-New.htm
●	Feature1	Feature1.htm
●	Feature2	Feature2.htm



User Profile Settings

Your Name: Simon MadCap

Email: Enter your email address

Profile Image

Photos should be a minimum of 750x750 pixels.

Accepted formats: .jpg, .png, .gif

For additional details, see the online Help.

Excluding JavaScript for CSH Calls in HTML5 Output

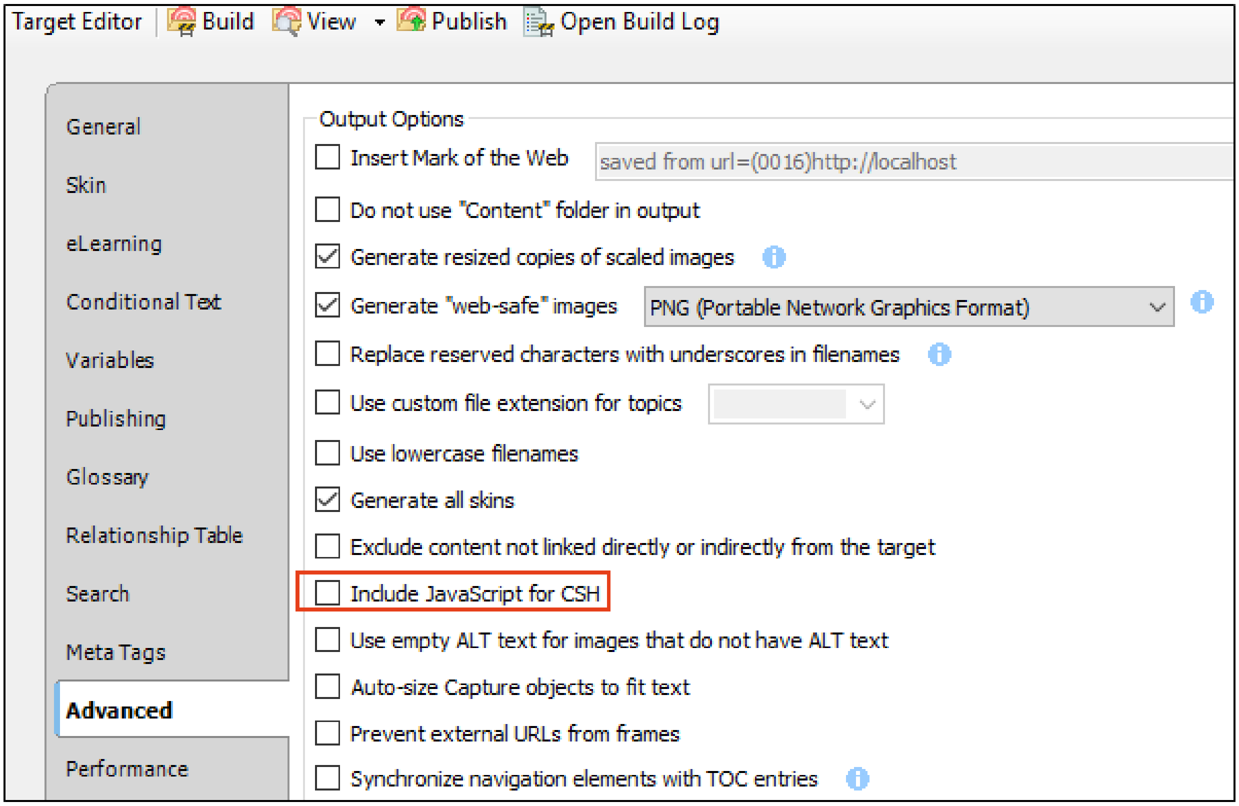
If you are using context-sensitive help (CSH) in your HTML5 output, you can either use JavaScript or a URL to open the output.

In the Target Editor for HTML5 targets, there is an option on the Advanced tab to include JavaScript for CSH calls. This option is enabled by default for HTML5 output.

If you use products like Fortify and Veracode to run security scans on your output, having JavaScript files in your HTML5 target might prompt security warnings when scanning it. If security warnings are displayed during these scans, you may want to exclude JavaScript files from your output.

How to Exclude JavaScript Files From Output

1. In the Project Organizer, expand **Targets** and double-click your HTML5 target. The Target Editor opens.
2. Click the **Advanced** tab.
3. Remove the check mark next to the **Include JavaScript for CSH** field.



Output Options

□ Insert Mark of the Web saved from url=(0016)http://localhost

□ Do not use "Content" folder in output

☑ Generate resized copies of scaled images

☑ Generate "web-safe" images PNG (Portable Network Graphics Format)

□ Replace reserved characters with underscores in filenames

□ Use custom file extension for topics

□ Use lowercase filenames

☑ Generate all skins

□ Exclude content not linked directly or indirectly from the target

□ Include JavaScript for CSH

□ Use empty ALT text for images that do not have ALT text

□ Auto-size Capture objects to fit text

□ Prevent external URLs from frames

□ Synchronize navigation elements with TOC entries

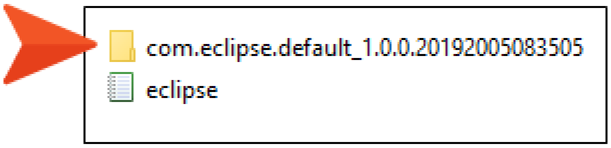
4. Click 💾 to save your work.

> NOTE If you choose not to include JavaScript files in your HTML5 output, you will not be able to perform CSH calls using the JavaScript method. However, you can still make CSH calls using URLs. See the Flare online Help for more information.

Additional Steps for Eclipse Help

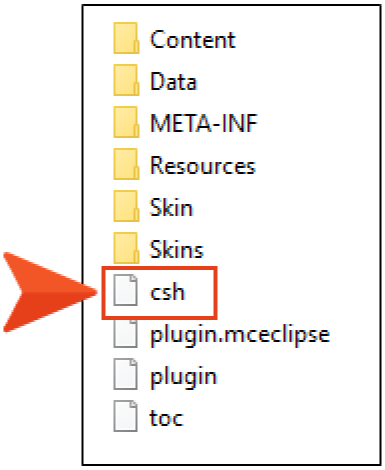
If you are generating Eclipse Help, there are additional steps that you need to complete in Eclipse.

1. After you create an alias and header file, and then generate an Eclipse target, select **Project > Open Output Folder**.
2. In Windows double-click the Eclipse output folder, which might look something like this:



com.eclipse.default_1.0.0.20192005083505
eclipse

3. Within that folder, an XML file named “csh” is included.



Content
Data
META-INF
Resources
Skin
Skins
csh
plugin.mceclipse
plugin
toc

Go to help.eclipse.org, then find and follow the instructions for packaging context-sensitive Help. When doing this, use the context-sensitive Help XML file produced by Flare.

Viewing Topics Not Linked by a Map ID

You can view a list of all topics that are not linked to context-sensitive Help (CSH) map IDs. You can also double-click a row to open the topic in question.

How to View Topics Not Linked by a Map ID

1. Open a project.
2. Do one of the following, depending on the part of the user interface you are using:
   - **Ribbon** Select **Analysis > More Reports > Topics Not Linked By Map ID**.
   - **Summary Window Pane** Double-click the row that mentions these items.

   The Topics Not Linked By Map ID window pane opens.
3. To see more of the information in the window pane, drag the divider bar to make the pane wider.
   - **File** Displays the name of the file.
   - **Title** Displays the properties title of the file (if any).
   - **Folder** Displays the folder where the file is found.
4. If a certain number of items have been found, page navigation buttons in the local toolbar may be enabled. You can use these buttons to go to additional pages to display more items. You can also click **View All** in the local toolbar to see all results on a single page. Keep in mind that the more items you have in the project, the longer it will take to load this view.

Viewing Unused CSH IDs

You can view a list of all context-sensitive Help (CSH) identifiers that have been created in your project but have not yet been assigned. You can double-click a row to open the Flare header file. For more information, see the *Flare CSH Guide* or the Flare online Help.

How to View Unused CSH IDs

1. Select **Analysis > More Reports > Unused Items**.

   The Unused Items window pane opens, which lets you view various unused elements in the project (such as bookmarks, condition tags, content files, CSH IDs, images, variables, styles, file tags, meta tags, and topics not linked).

2. Click the drop-down field at the top of the window pane and select **Unused CSH IDs**. The unused CSH IDs in the project are listed.

3. To see more of the information in the window pane, drag the divider bar to make the pane wider.

   - **CSH ID** Displays the CSH identifier that is not being used.
   - **File** Displays the alias file where the CSH ID is located. You can use the alias file to assign the unused CSH ID to a topic or micro content phrase, unless you prefer to delete the CSH ID.
   - **Definition** Displays the unique numerical value associated with the CSH ID.

4. If a certain number of items have been found, page navigation buttons in the local toolbar may be enabled. You can use these buttons to go to additional pages to display more items. You can also click **View All** in the local toolbar to see all results on a single page. Keep in mind that the more items you have in the project, the longer it will take to load this view.

5. If you want to remove any unused CSH IDs from Flare, as well as from the project being analyzed, select the CSH ID in the list and click ✖ in the local toolbar.

NOTE You can open the alias file for any CSH ID in the list to view or modify it in the Alias Editor.

APPENDIX

PDFs

The following PDFs are available for download from the online Help.

Tutorials

Getting Started Tutorial

Autonumbers Tutorial

Back-to-Top Button Tutorial

Context-Sensitive Help Tutorial

Custom Toolbar Tutorial

eLearning Tutorial—Basic

eLearning Tutorial—Advanced

Image Tooltips Tutorial

Lists Tutorial

Meta Tags Tutorial

Micro Content Tutorial—Basic

Micro Content Tutorial—Advanced

Responsive Output Tutorial

Single-Sourcing Tutorial

Snippet Conditions Tutorial

Styles Tutorials

Tables Tutorial

Word Import Tutorial

Cheat Sheets

Context-Sensitive Help Cheat Sheet

Folders and Files Cheat Sheet

Learning & Development Cheat Sheet

Lists Cheat Sheet

Micro Content Cheat Sheet

Print-Based Output Cheat Sheet

Search Cheat Sheet

Shortcuts Cheat Sheet

Structure Bars Cheat Sheet

Styles Cheat Sheet

User Guides

Accessibility Guide

Analysis and Reports Guide

Architecture Guide

Autonumbers Guide

Branding Guide

Condition Tags Guide

Context-Sensitive Help Guide

Eclipse Help Guide

eLearning Guide

Getting Started Guide

Global Project Linking Guide

HTML5 Guide

Images Guide

Import Guide

Indexing Guide

Key Features Guide

Lists Guide

MadCap Central Integration Guide

Meta Tags Guide

Micro Content Guide

Navigation Links Guide

Plug-In API Guide

Print-Based Output Guide

Project Creation Guide

QR Codes Guide

Reviews & Contributions With Contributor Guide

Scripting Guide

Search Guide

SharePoint Guide

Skins Guide

Snippets Guide

Source Control Guide: Git

Source Control Guide: Perforce Helix Core

Source Control Guide: Subversion

Source Control Guide: Team Foundation Server

Styles Guide

Tables Guide

Tables of Contents Guide

Targets Guide

Template Pages Guide

Templates Guide

Topics Guide

Touring the Workspace Guide

Transition From FrameMaker Guide

Translation and Localization Guide

Variables Guide

Videos Guide

What's New Guide