

Maximizing productivity by using conditions in MadCap Flare projects

Contents

Summary	3
Introduction.....	3
Before you start	3
About the examples	4
Project platform	4
Example project.....	4
Defined conditions	5
Basic inclusion and exclusion of conditions	6
Example 1 – Generating a target without any Include and Exclude conditions.....	6
Example 2 – Actively including a condition	7
Example 3 – Actively excluding a condition	9
Working with multiple conditions on topics	10
Example 4 – Actively including a condition where a topic has more than one condition.....	11
Example 5 – Actively excluding a condition where a topic has more than one condition	12
Mixing includes and excludes in targets	13
Example 6 – Using includes and excludes on single condition topics	13
Example 7 – Using includes and excludes on multiple condition topics	14
Considering the nature of the content	16
Example 8 – Using exclusion conditions on partial common content.....	17
Example 9 – Using inclusion conditions on partial common content	19
Folder-level conditions	22
Example 10 – Applying a condition to a folder that contains unconditioned files.....	23
Example 11 – Applying a condition to a folder where its files have other conditions	25
Topic-level contra content-level conditions.....	27
Example 12 – Topic-level conditions override content conditions	27
Example 13 – Image conditions.....	29
Example 14 – Snippet conditions	31
Example 15 – ToC conditions	32
Implications of conditions on content versioning	35
Example 16 – File-level conditions and file header information.....	35
Example 17 – Folder-level conditions and file content	36

Example 18 - Snippet file-level conditions and file content	37
Example 19 - Snippet folder-level conditions and file content	37
Conditions and content complexity	38
Condition consistency across multiple projects	40
Summary of issues to consider	42
Concluding comments.....	43

Summary

This document describes some of the considerations encountered when applying conditions to topics and their content in MadCap Flare projects. Pragmatic examples are used throughout. The planning criteria and decisions required to correctly apply conditions are emphasized, with the ultimate aim of maximizing content usage and productivity.

Introduction

MadCap Flare includes a vast array of documentation functions, many of which can streamline work processes, reduce costs and increase productivity. One of the most significant of these is the ability to apply conditions to project content.

While not unique to MadCap Flare, the ability to apply conditions has significant value. In basic terms it means that a single set of content can be used to satisfy multiple audiences – this is commonly known as *single sourcing*. This is achieved by conditioning your project files to effectively include or exclude content from target output, thereby avoiding duplicate sets of content.

Before you start

There are two very significant issues that must be understood prior to conditioning project content.

- ***Conditions require careful definition.***
Conditioning is a double-edged sword and without understanding the consequences of conditioning you can seriously affect the output of your MadCap Flare targets. On the other hand, if you have a well-defined approach then you will reap the benefits and never regret having implemented conditions.
- ***Conditions update content.***
If you apply a condition to a file, for example to a topic, then a condition tag is entered into the file header. Because the file has been modified it receives a new timestamp and is effectively a new version. If your project is to be translated or even localized, then this updated file will be marked for reevaluation, even if the visible readable content therein has not changed! This can have significant economic consequences.

The following sections will help clarify these issues.

About the examples

Project platform

The examples used in this document are all based upon MadCap Flare v.7.1.0 running on a 32-bit Microsoft Windows 7 platform. For sake of simplicity only HTML Help (CHM) targets are used, though many of the concepts are valid for other types of targets.

The examples focus on the results achieved when conditioning project content. The procedures for creating conditions sets, actual conditions and the tagging of project content are all documented in detail in the MadCap Flare online help file, specifically in the *Features > Conditional Text* section.

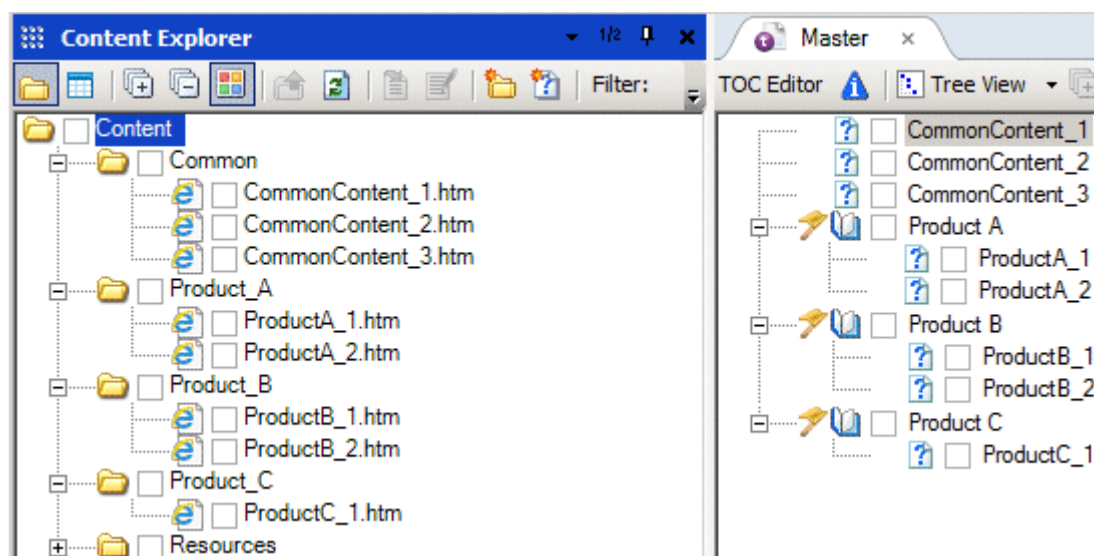
The examples in this document use conditions on topic files and in some specific cases actual topic content, snippets and images. Conditions can also be applied to other content such as tables, lists, relationship tables and many other types of files and content.

Example project

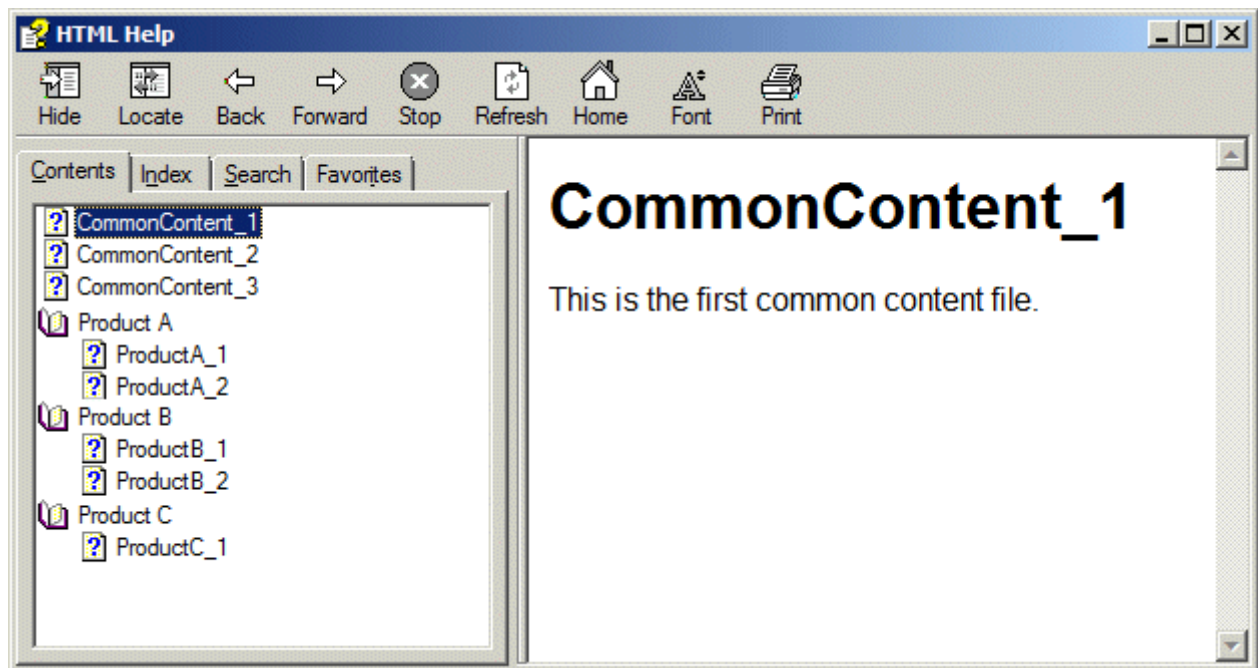
A very basic and simple *Products* project is used for the examples. In its basic form this has nine topics, where:

- Three topics are common to all products
- Two topics are for Product A
- Two topics are for Product B
- One topic is for Product C

This can be seen in the following image where each topic has been placed in its folder and the ToC has also been structured.

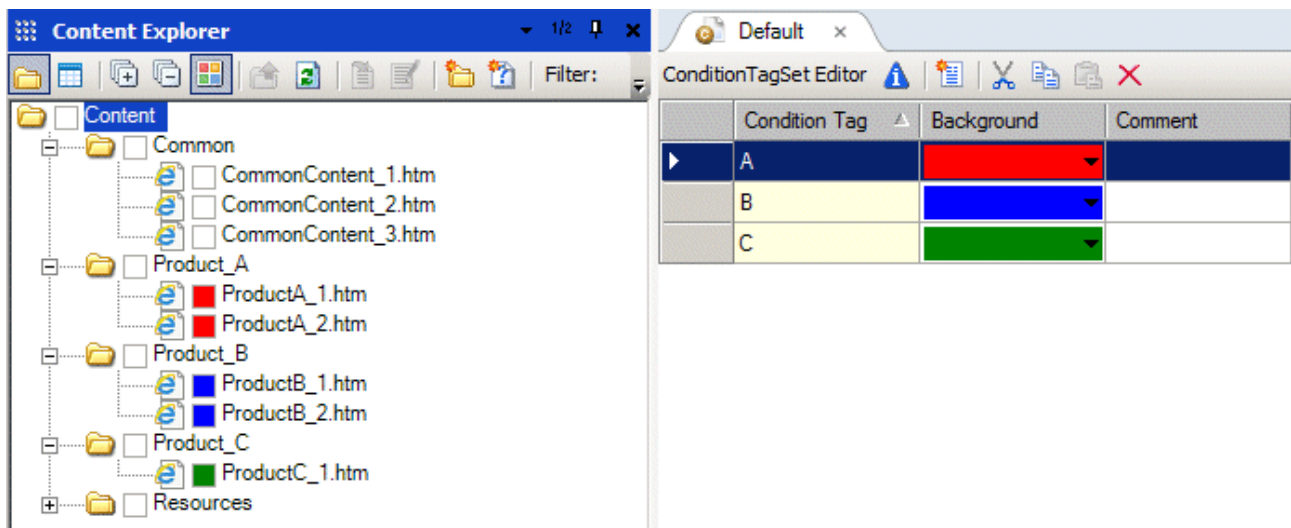


No conditions have been applied and generating the target, does as expected, include all the topics in the output.



Defined conditions

A set of conditions are defined. For simplicity these are named A, B and C and have applied them to the topics for Product A, Product B and Product C respectively.

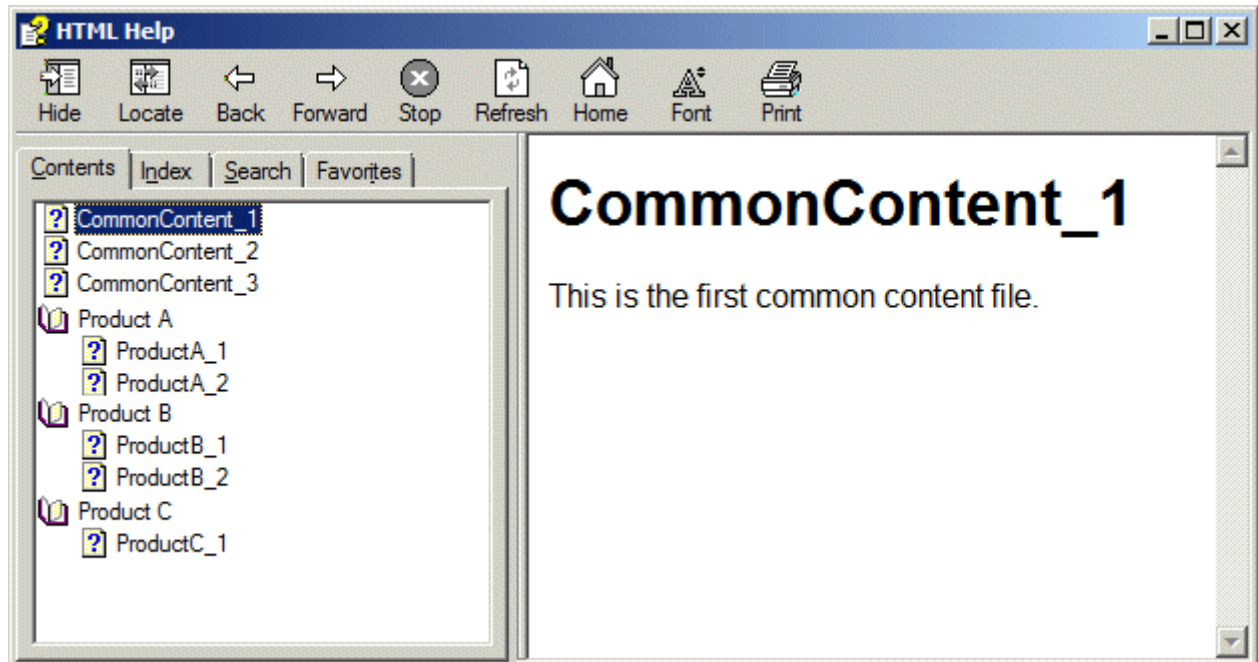


The three conditions appear in the target definition, but at the outset none are selected for inclusion or exclusion. The conditions will be applied in the examples, as will other conditions which will be defined and applied as necessary.

Basic inclusion and exclusion of conditions

Example 1 – Generating a target without any Include and Exclude conditions

Generating the target does as before, it includes all topics.

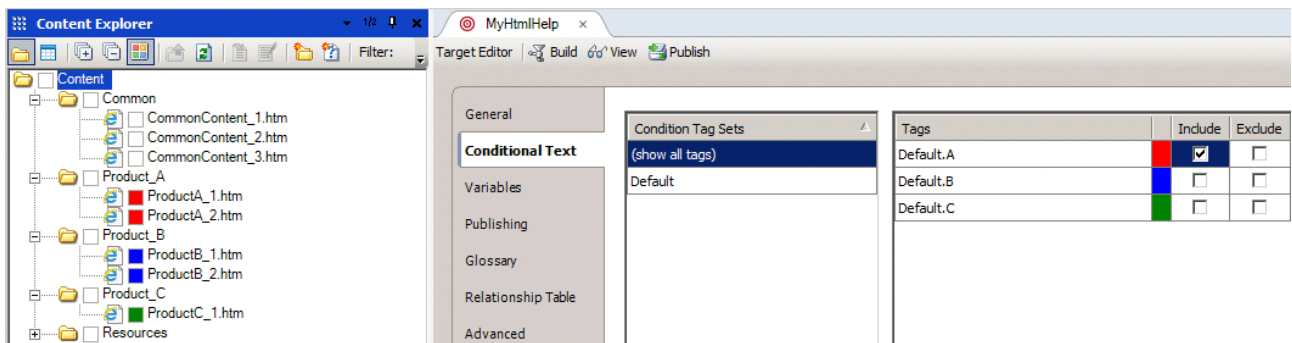


Conclusion

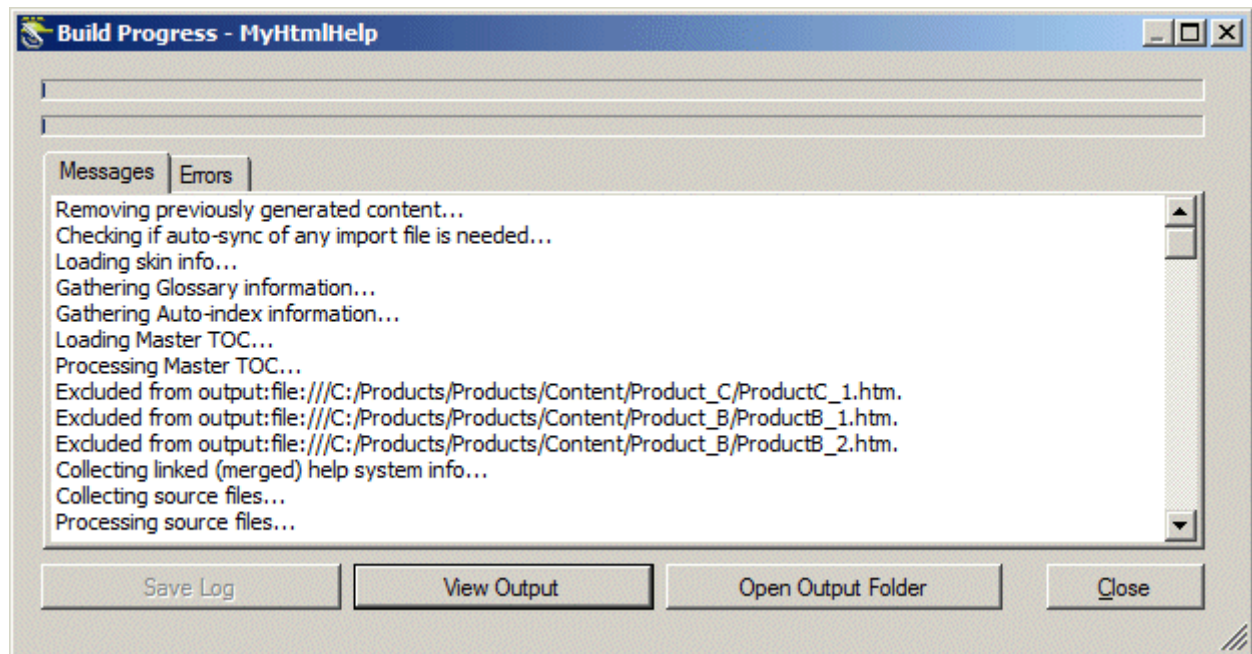
If you have defined conditions in your project, but have not selected Include or Exclude for any of these in your target, then this is equivalent to not having defined the conditions at all, even if they have been associated with topics.

Example 2 – Actively including a condition

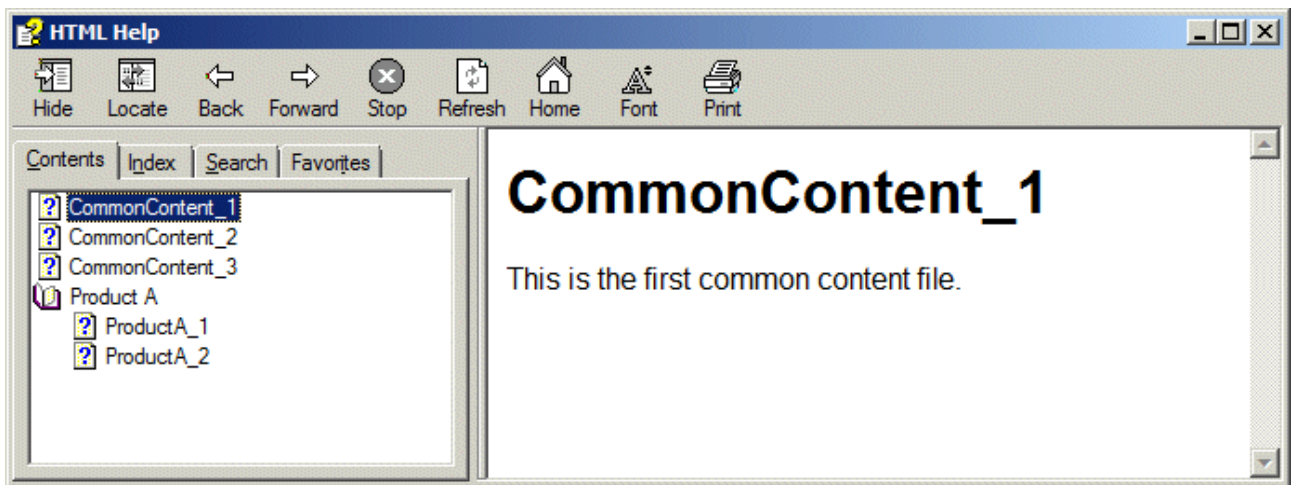
In this example we actively select condition A for inclusion.



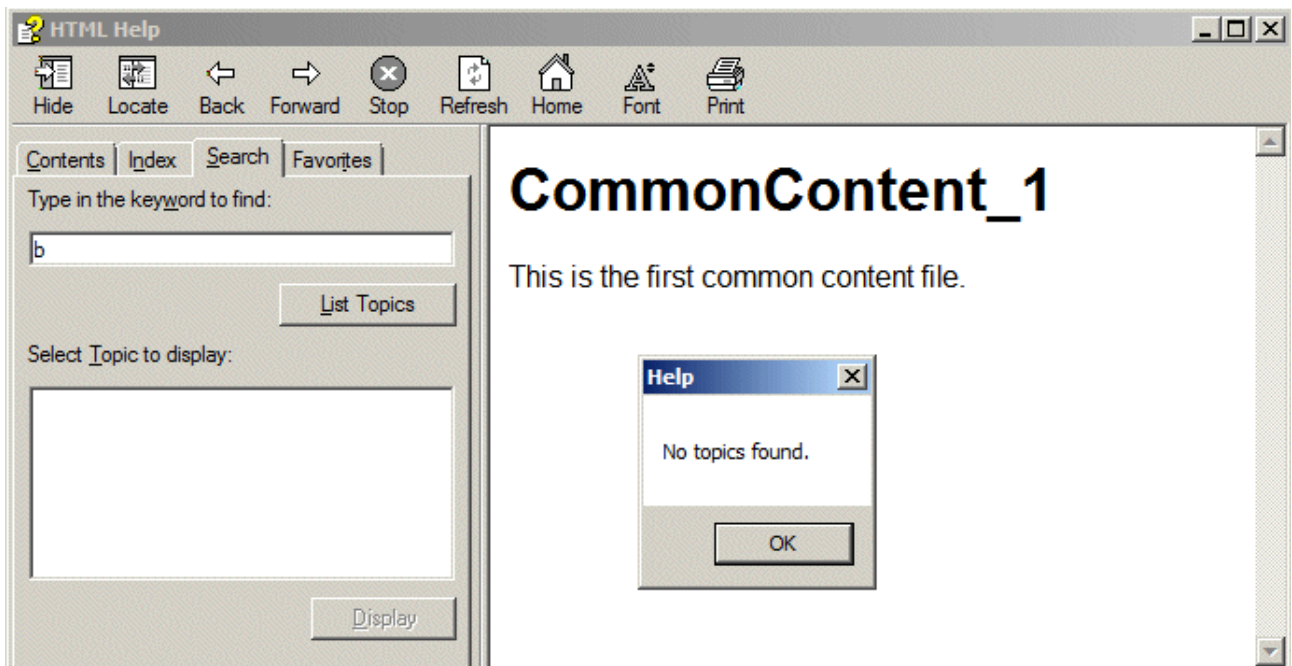
The output includes only the non-conditioned topics and those with condition A. All other conditioned topics (that do not have condition A) are excluded. This is clearly indicated in the build log.



The result can be seen in the output file where only the common and Product A topics appear.



Note that even a search for a Product B or a Product C topic yields no result; they are not included in the output file.

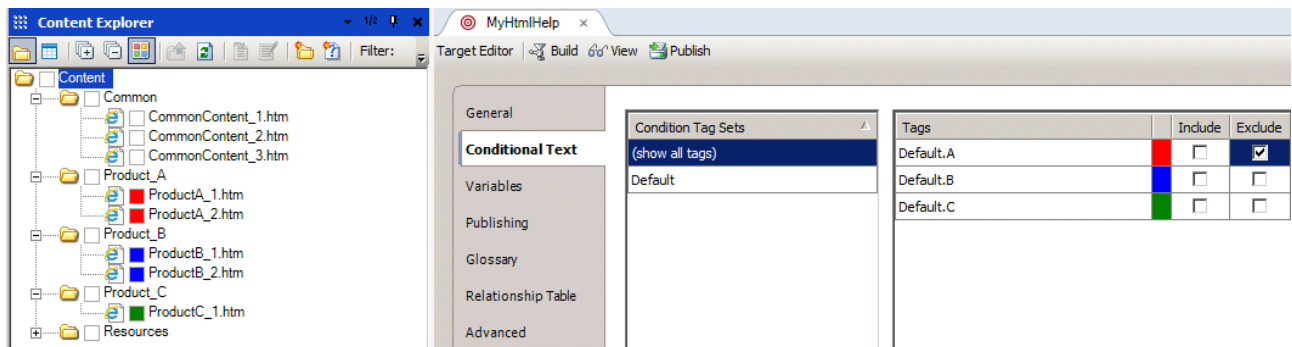


Conclusion

Actively selecting the inclusion of a condition in a target will include all topics that have that condition as well as all topics that have no conditions. All other conditioned topics that do not have the selected condition are excluded.

Example 3 – Actively excluding a condition

In this example we actively select condition A for exclusion.

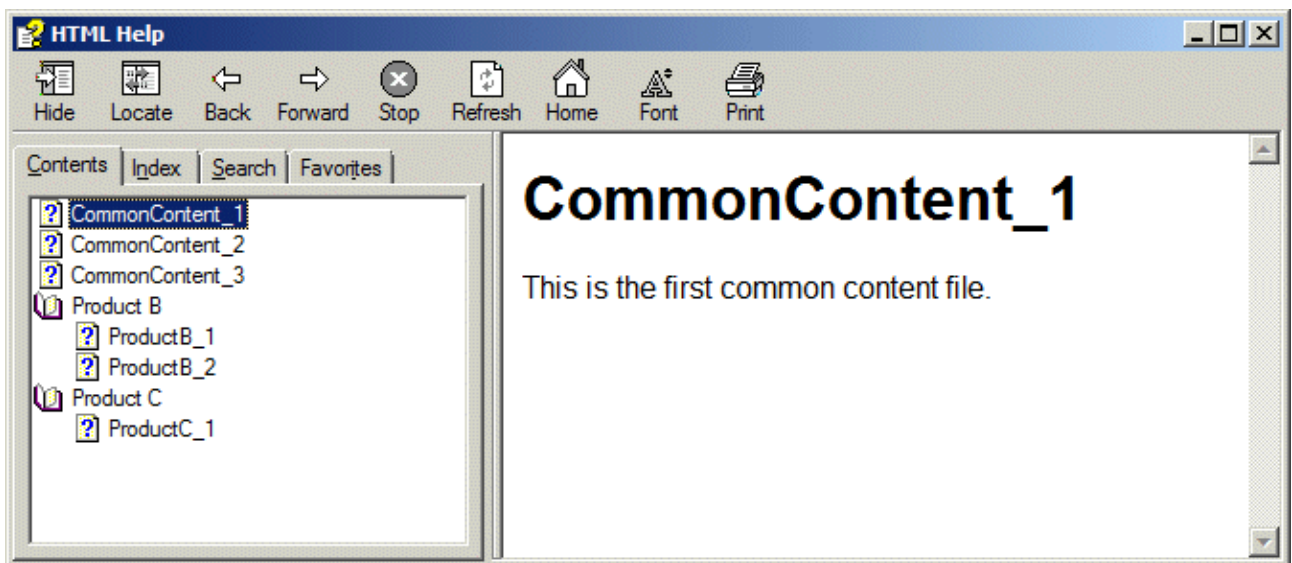


The output excludes topics marked with condition A. However, all topics marked with another condition as well as non-conditioned topics are included in the output file. This is listed in the build log.

Excluded from output:file:///C:/Products/Products/Content/Product_A/ProductA_1.htm.

Excluded from output:file:///C:/Products/Products/Content/Product_A/ProductA_2.htm.

The result can be seen in the output file where Product A topics are not included.



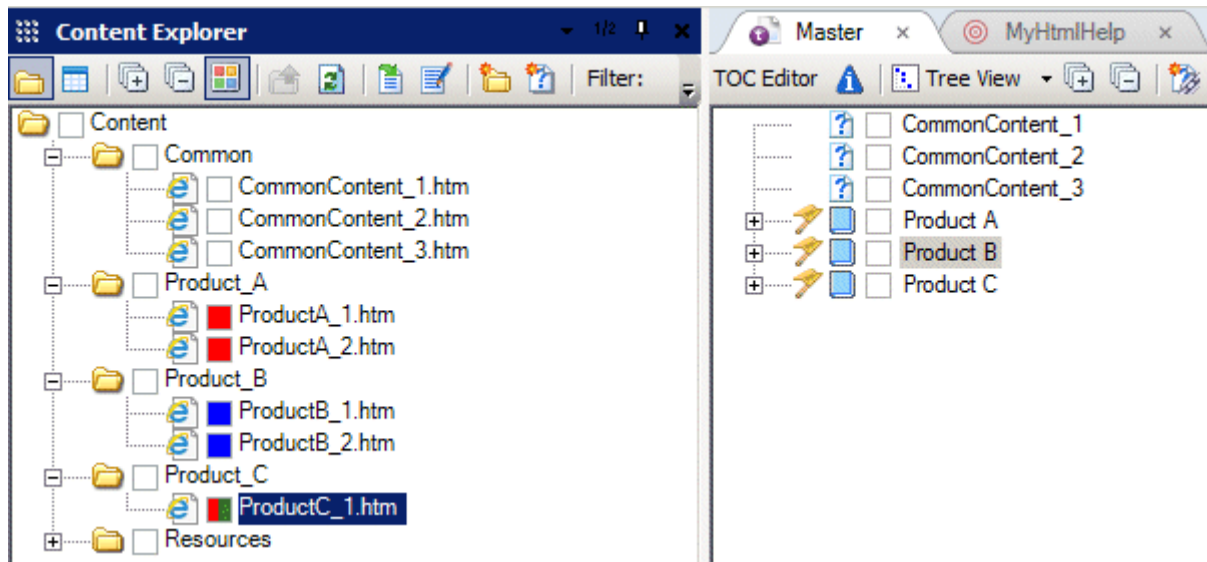
Note that a search for a Product A topic yields no result; they are not included in the output file.

Conclusion

Actively selecting the exclusion of a condition in a target will exclude all topics that have that condition. All topics that have no conditions as well as other topics that do not have the selected condition are included.

Working with multiple conditions on topics

Topics may have multiple applicability, but not necessarily common for all targets. For example, let us say that the topic ProductC_1 is also applicable to Product A (even though the topic's name, title and folder do not reflect this). We apply condition A to ProductC_1 and it is indicated by the red and green markings.

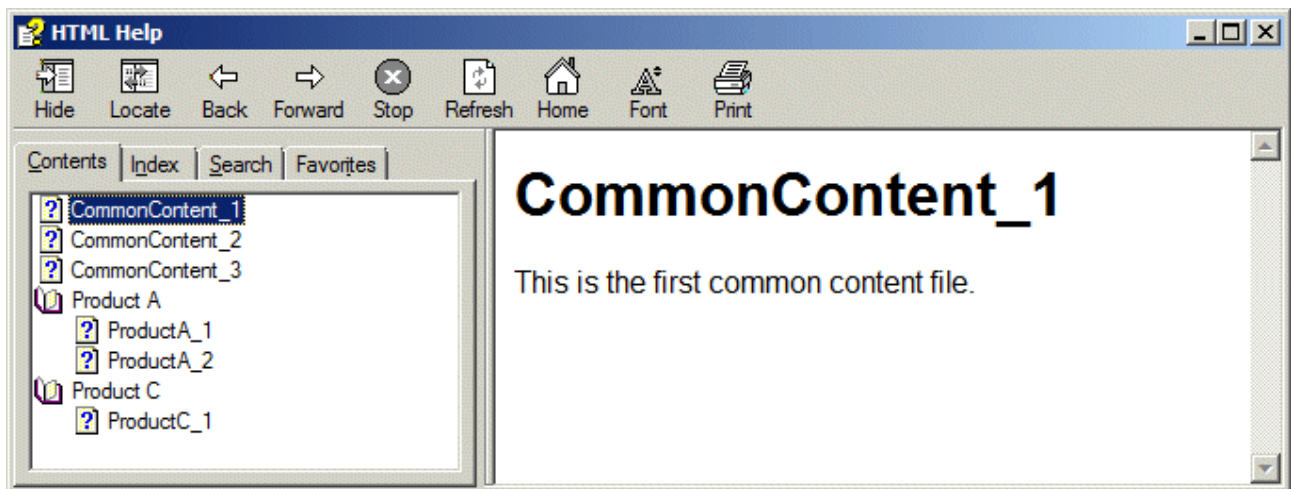


Example 4 – Actively including a condition where a topic has more than one condition

This target is identical to that used in Example 2. Because condition A is included, ProductC_1 is included. Only topics without condition A are excluded as seen in the build log:

Excluded from output:file:///C:/Products/Products/Content/Product_B/ProductB_1.htm.
Excluded from output:file:///C:/Products/Products/Content/Product_B/ProductB_2.htm.

This can be seen in the output file:



Conclusion

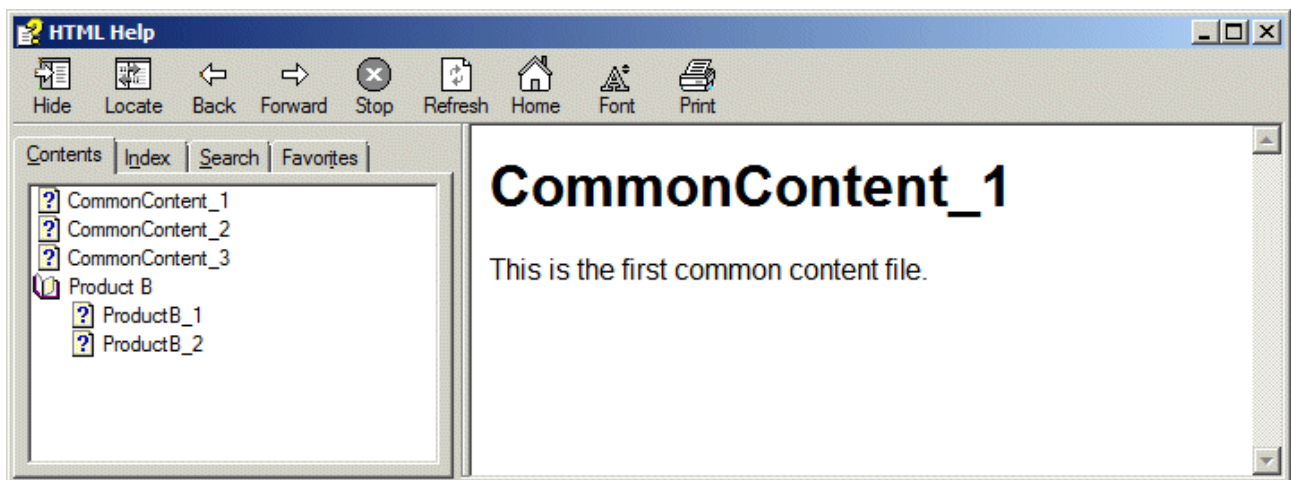
This is the same as for Example 2. Actively selecting the inclusion of a condition in a target will include all topics that have that condition as well as all topics that have no conditions. All other conditioned topics that do not have the selected condition are excluded.

Example 5 – Actively excluding a condition where a topic has more than one condition

The target is identical to that used in Example 3. Because condition A is excluded, all topics marked with another condition as well as non-conditioned topics are included in the output file. This is listed in the build log.

```
Excluded from output:file:///C:/Products/Products/Content/Product_A/ProductA_1.htm.  
Excluded from output:file:///C:/Products/Products/Content/Product_A/ProductA_2.htm.  
Excluded from output:file:///C:/Products/Products/Content/Product_C/ProductC_1.htm.
```

This can be seen in the output file:



Conclusion

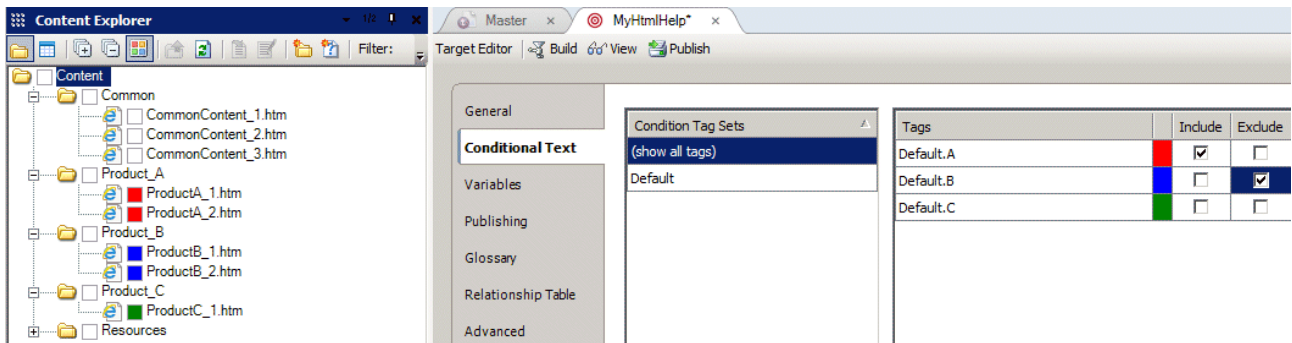
This is the same as for Example 3. Actively selecting the exclusion of a condition in a target will exclude all topics that have that condition. All topics that have no conditions as well as other topics that do not have the selected condition are included.

Mixing includes and excludes in targets

The previous examples all used simple includes or excludes in the target files. However, it is possible to specify includes and excludes in the same target.

Example 6 – Using includes and excludes on single condition topics

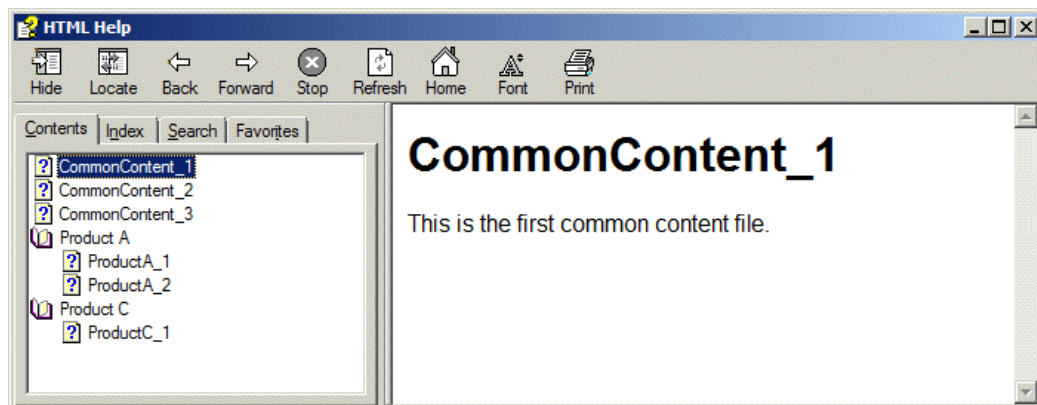
Consider the following target:



Condition A is marked for inclusion and condition B is marked for exclusion. The build log lists the exclusions as:

```
Excluded from output:file:///C:/Products/Products/Content/Product_B/ProductB_1.htm.  
Excluded from output:file:///C:/Products/Products/Content/Product_B/ProductB_2.htm.
```

As expected the topics with condition A are included and those with condition B are excluded. Unconditioned topics and those not marked with condition B are also included.

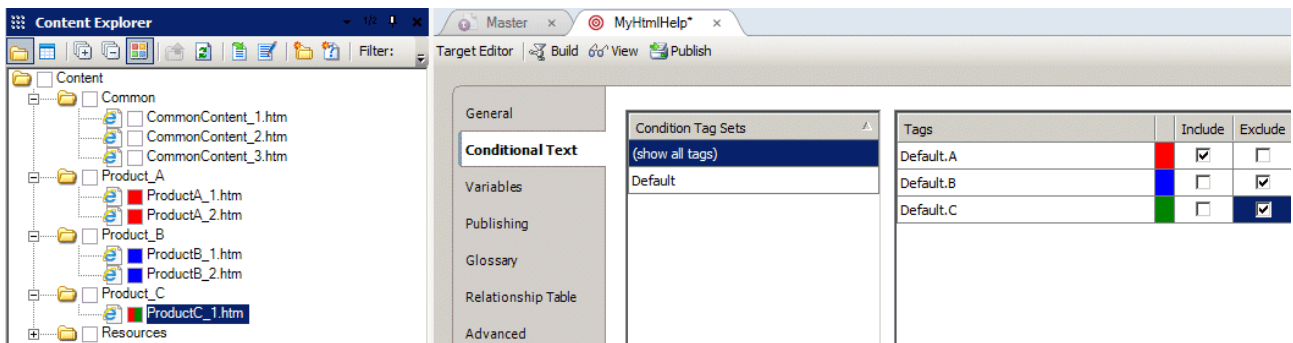


Conclusion

This is a mix of two previous examples. Actively selecting the inclusion of a condition in a target will include all topics that have that condition. Actively selecting the exclusion of a condition in a target will exclude all topics that have that condition. All topics that have no conditions as well as other topics that do not have the excluded condition are included.

Example 7 – Using includes and excludes on multiple condition topics

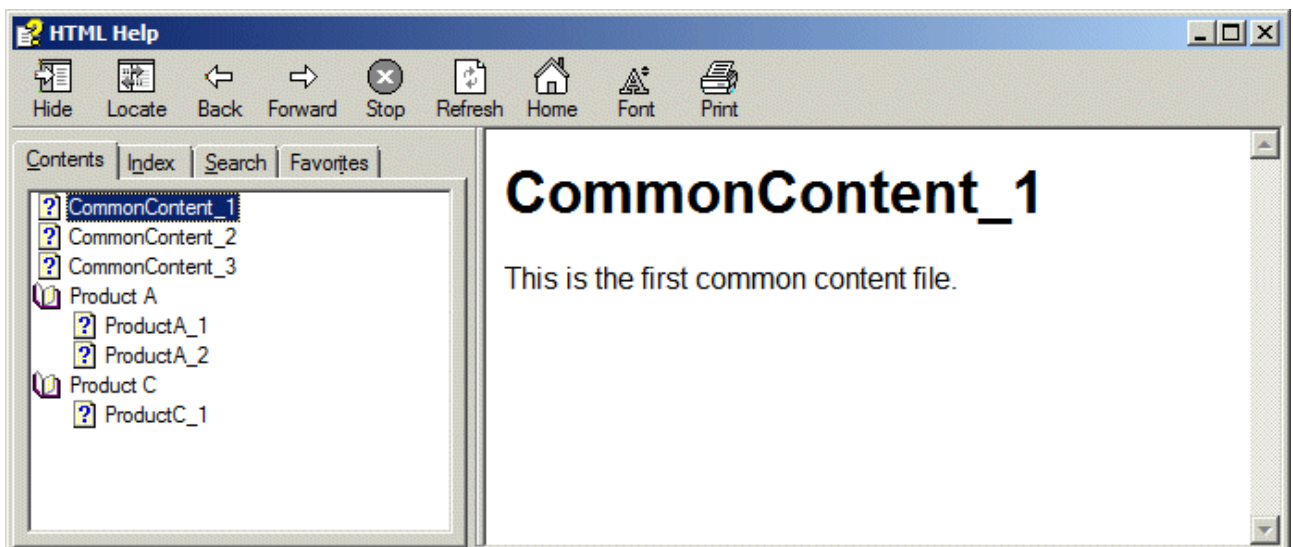
As an enhancement to Example 6, consider the following target:



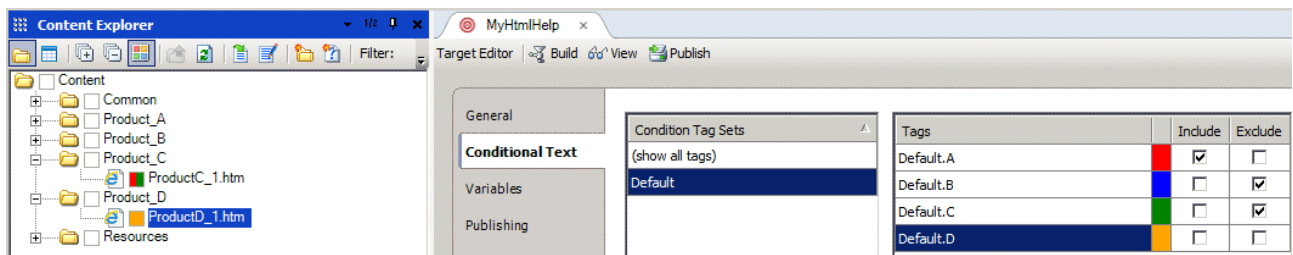
Note that ProductC_1 has been assigned conditions A and C. Condition A is marked for inclusion and condition B and C for exclusion. The build log lists the exclusions as:

```
Excluded from output:file:///C:/Products/Products/Content/Product_B/ProductB_1.htm.  
Excluded from output:file:///C:/Products/Products/Content/Product_B/ProductB_2.htm.
```

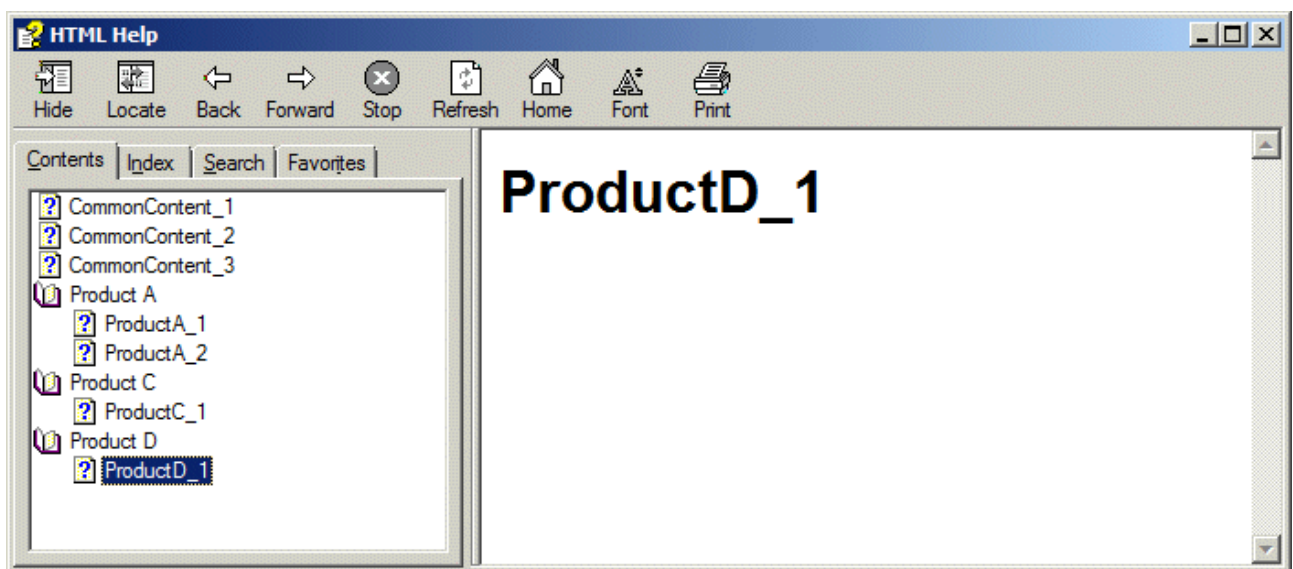
As expected the topics with condition B are excluded. Topics with condition A are included, as are unconditioned topics. Note how ProductC_1 is included even though it has condition C that is excluded, this is because it has condition A that overrides the exclusion.



Furthermore, if a topic is assigned a condition that is neither included nor excluded then it is also included in the output. For example ProductD_1 has been assigned condition D:



This appears in the output.

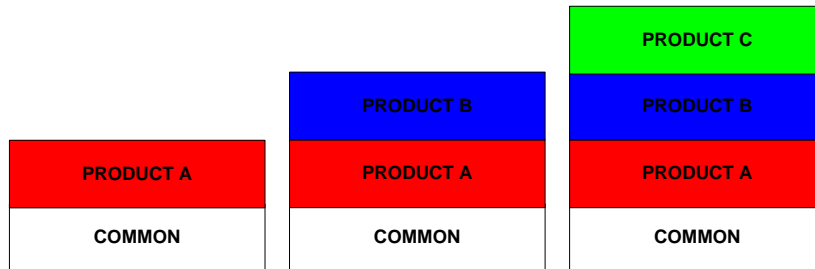


Conclusion

Actively selecting the inclusion of a condition in a target will include all topics that have that condition. Actively selecting the exclusion of a condition in a target will exclude all topics that have that condition, but an include overrides an exclude where multiple conditions have been assigned to the topic. All topics that have no conditions as well as other topics that do not have the excluded condition are included.

Considering the nature of the content

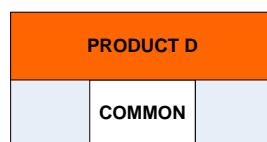
The previous examples all considered a classic documentation approach. For example, a basic set of common documents were relevant for the products. All the common documents would be applied for each product mix and can be visualized as building blocks, for example:



In the example project the following combinations could be used:

- **Common** + Product A
- **Common** + Product B
- **Common** + Product C
- **Common** + Product A + Product B
- **Common** + Product A + Product C
- **Common** + Product B + Product C
- **Common** + Product A + Product B + Product C

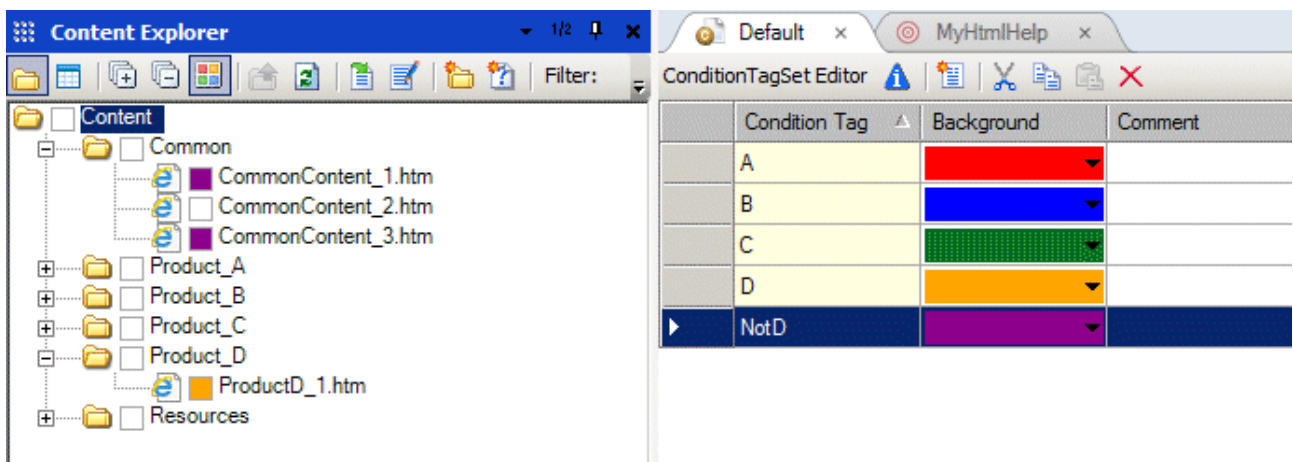
However, as is the nature of companies and organizations, sooner or later documentation is required where perhaps not all the common material is relevant for a specific product (or output). For example, a new Product D may only uses the CommonContent_2 topic, where CommonContent_1 and CommonContent_3 are not suitable for Product D.



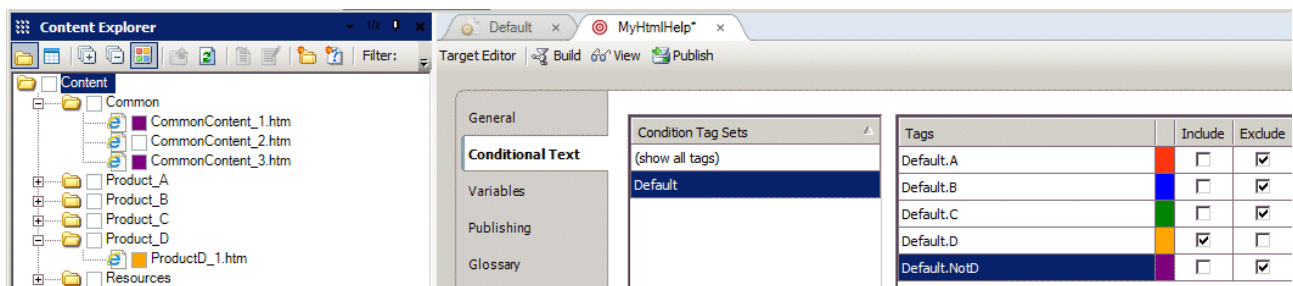
Note that for simplicity of explanation only one possible combination with Product D is shown, but there can be many more. There are various ways to deal with these situations.

Example 8 - Using exclusion conditions on partial common content

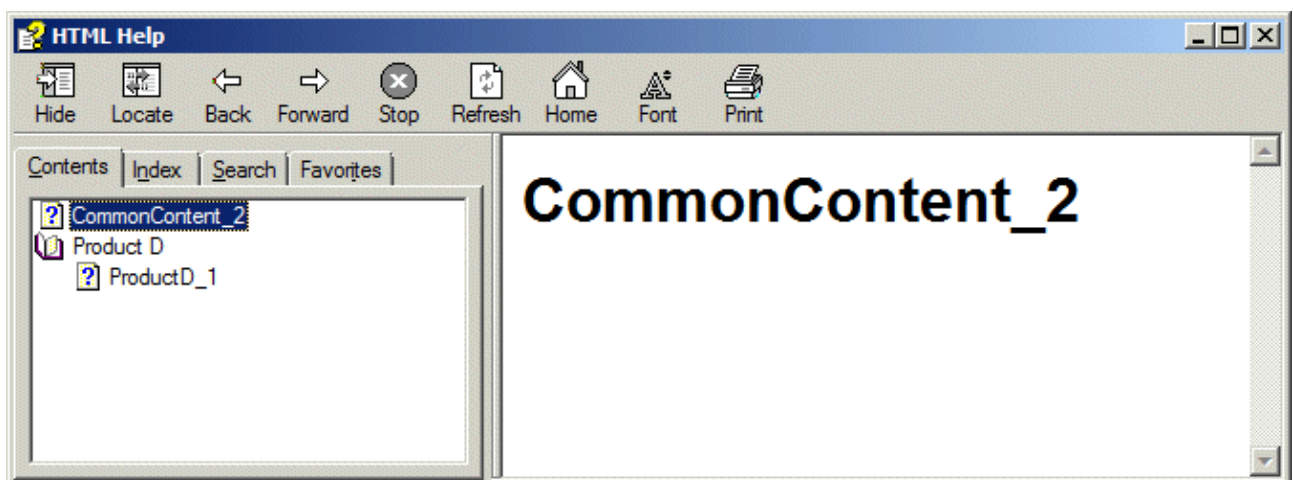
Define a new condition NotD and assign it to CommonContent_1 and CommonContent_3.



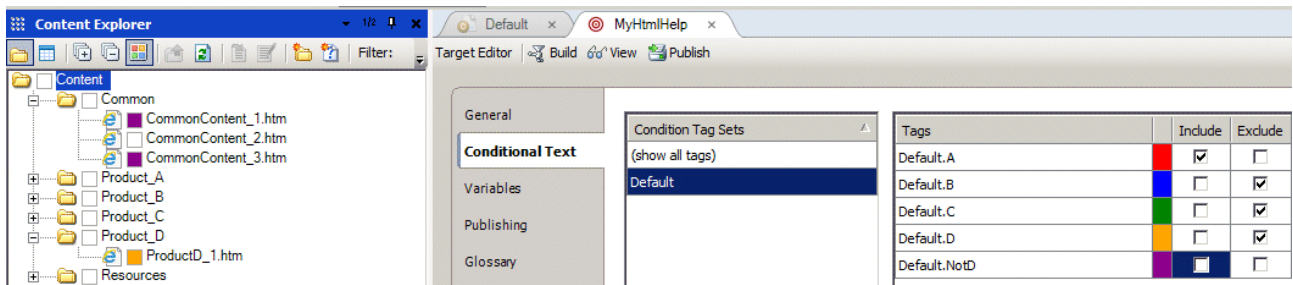
Exclude the condition in the target.



When generated, the output appears as expected CommonContent_2 and ProductD_1 are included, while CommonContent_1 and CommonContent_3 are not included.



However, this now implies that we have to update previous targets. In Example 2 the target would need to be as follows to ensure the correct content.



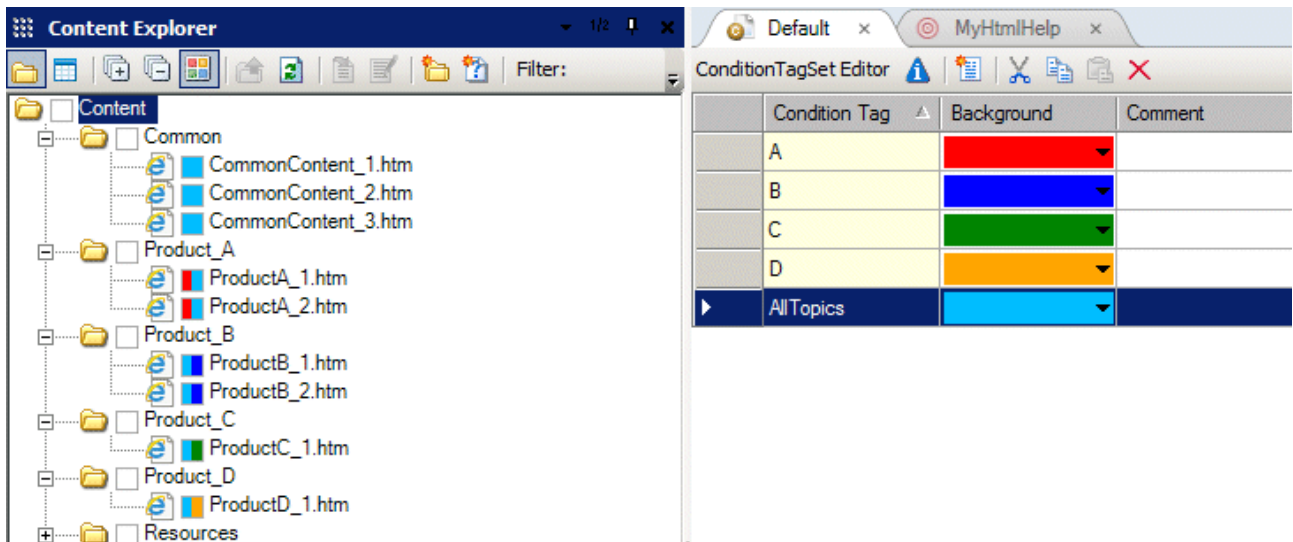
This can be complicated and somewhat confusing – ignoring the not include condition for a product that you are not really concerned with right now! This becomes even more complicated if other products also require a partial set of the common files. Ultimately you may spend time deciphering and revising targets to ensure the correct content is included.

Conclusion

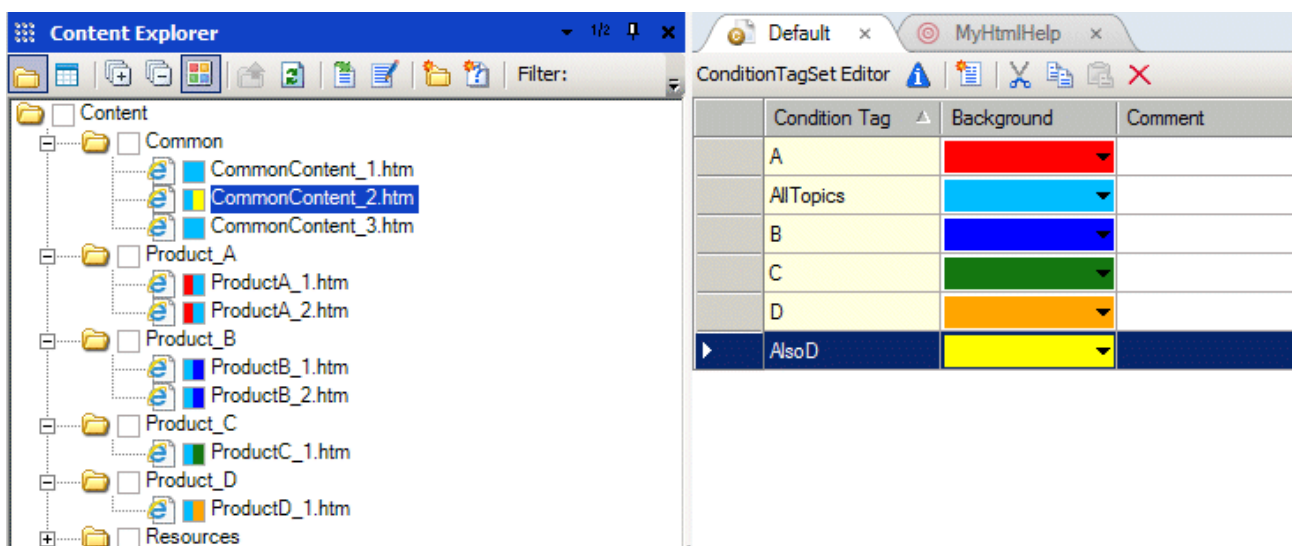
The use of a *Not condition* is only viable if there are few products that use a partial set of common files and only if the common files that require the Not condition are few in number.

Example 9 – Using inclusion conditions on partial common content

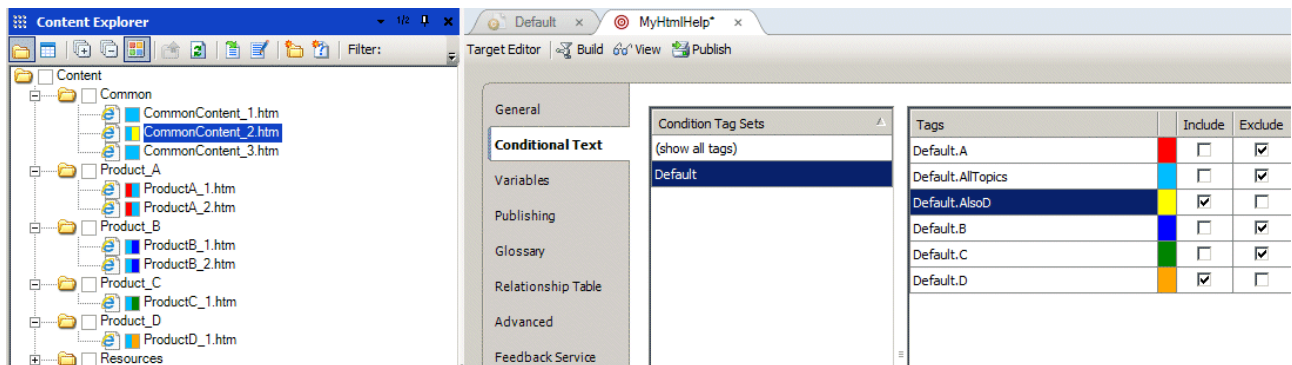
Sometimes it is easier to consider what is needed than what is not needed. This requires a fundamental change to condition tagging. All topics must be conditioned with a common condition, for example AllTopics.



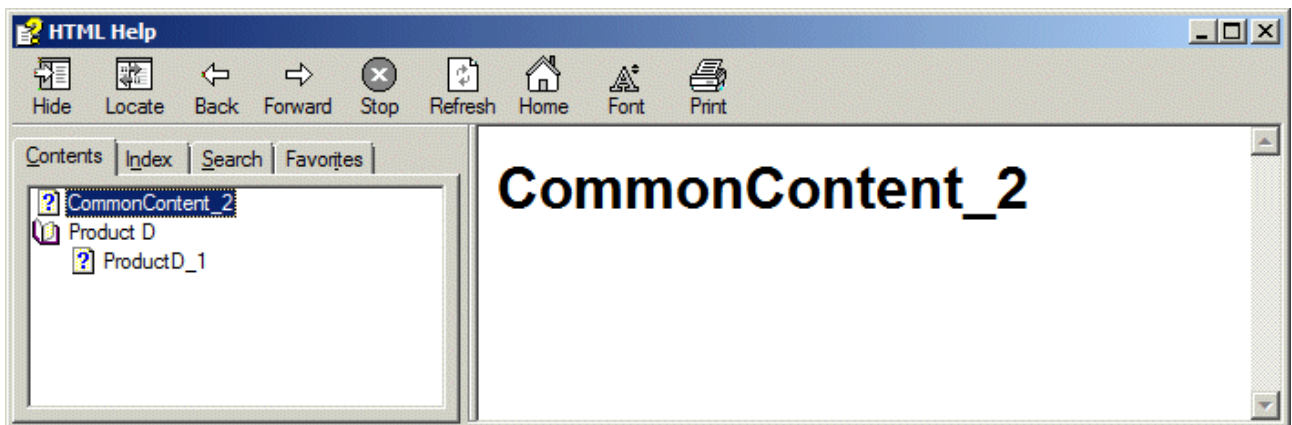
A new condition is created, for example AlsoD. This is applied to the common files that Product D requires, that is CommonContent_2.



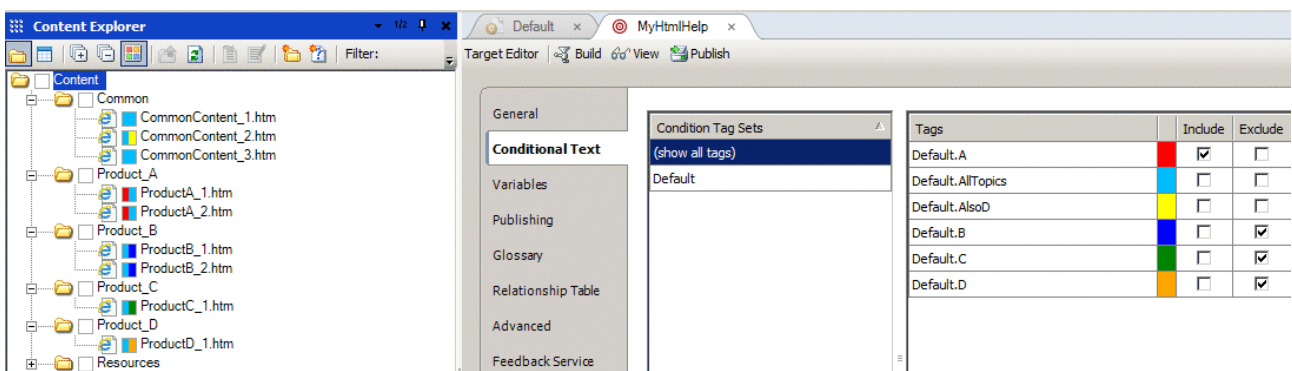
The target for Product D can be defined as:



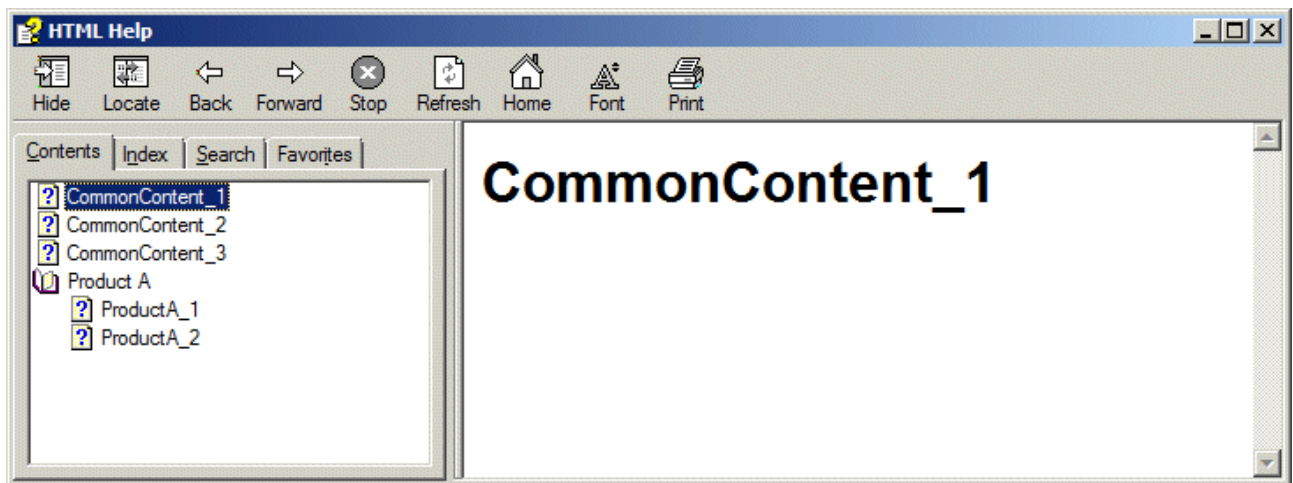
Note that everything is excluded by using an exclude on AllTopics. Thereafter the required topics added by including the AlsoD and D conditions. This gives the output as:



When using AllTopics, the target for the Product A would be:



This gives the desired output.



The result follows the conclusion from Example 3, where actively selecting the exclusion of a condition in a target will exclude all topics that have that condition. All topics that have no conditions as well as other topics that do not have the selected condition are included.

Conclusion

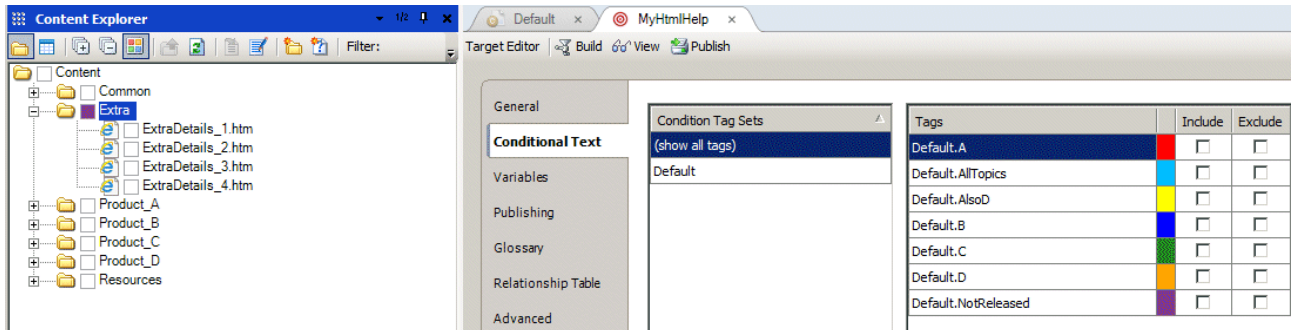
The advantage of this approach is realized if products only use a small number of common files. This is the inverse of using the Not condition described in Example 8, which proves unsuitable when many common files require the Not condition, that is when only a few of the common files are used by the new product.

Folder-level conditions

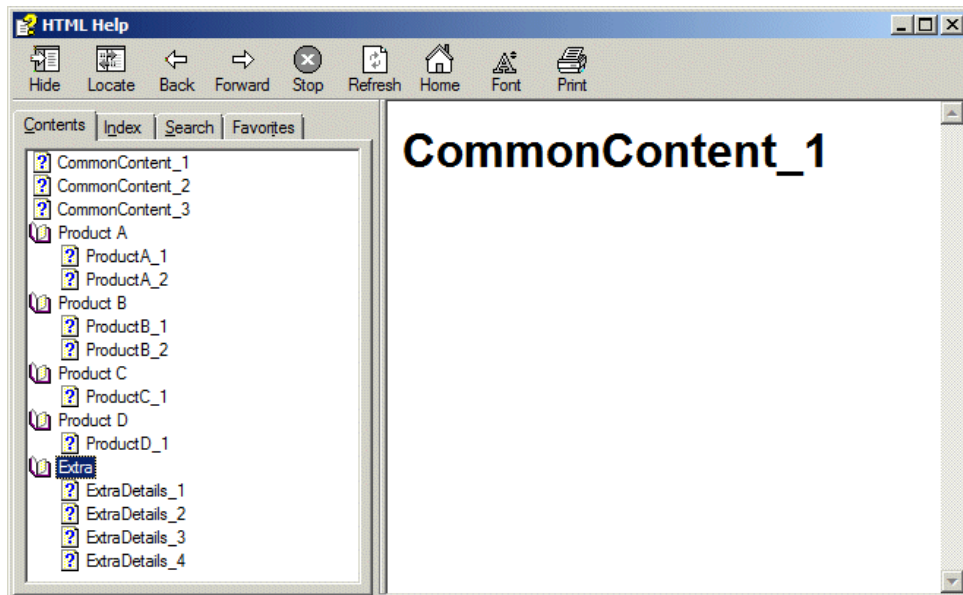
Folder-level conditions provide a quick and easy way to assign conditions to all the files therein. For example, content that is under development can be placed in a folder with a NotReleased condition and this set to Exclude in the target. When ready for release, the condition can be removed or a new condition can be assigned allowing the folder content to be released. Files can also be dragged and dropped into the folder, effectively releasing or restricting them.

Example 10 – Applying a condition to a folder that contains unconditioned files

Consider a folder named Extra where a condition NotReleased has been applied. Note that the topics do not display the color of the condition as it has been assigned at folder level.

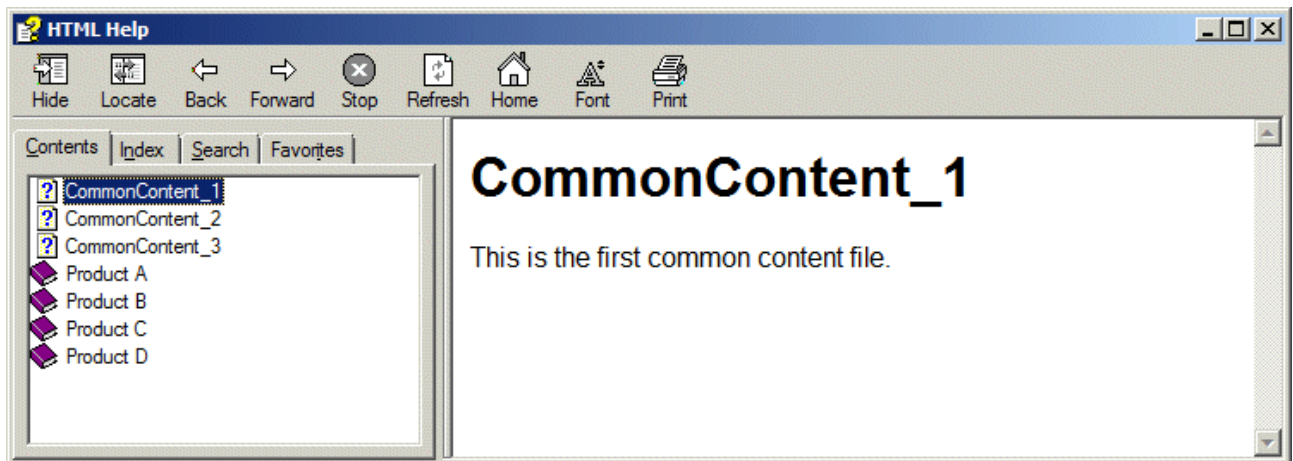


Generating this target creates an output with all files therein.



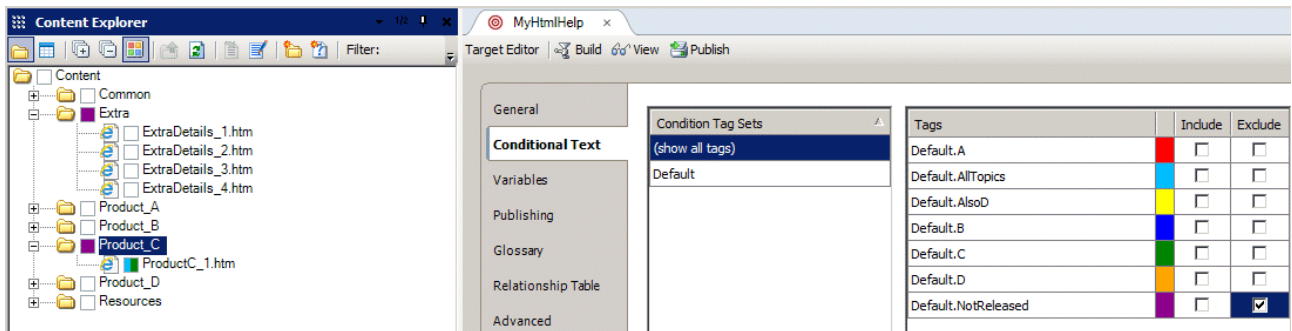
Placing an exclude on NotReleased causes the entire set of files in the folder to be excluded from the output, as indicated in the build log.

```
Excluded from output:file:///C:/Products/Products/Content/Extra/ExtraDetails_1.htm.  
Excluded from output:file:///C:/Products/Products/Content/Extra/ExtraDetails_2.htm.  
Excluded from output:file:///C:/Products/Products/Content/Extra/ExtraDetails_3.htm.  
Excluded from output:file:///C:/Products/Products/Content/Extra/ExtraDetails_4.htm.
```



Example 11 – Applying a condition to a folder where its files have other conditions

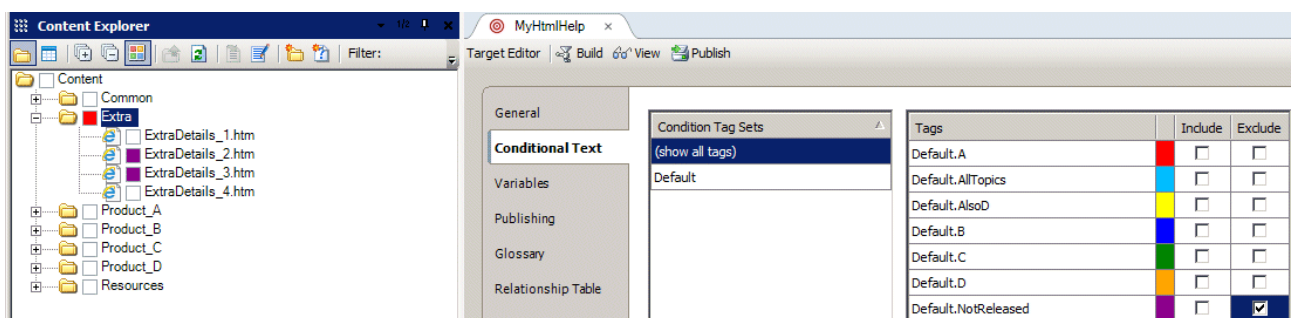
Apply NotReleased to ProductC folder. This includes the ProductC_1 topic that has other conditions applied it.



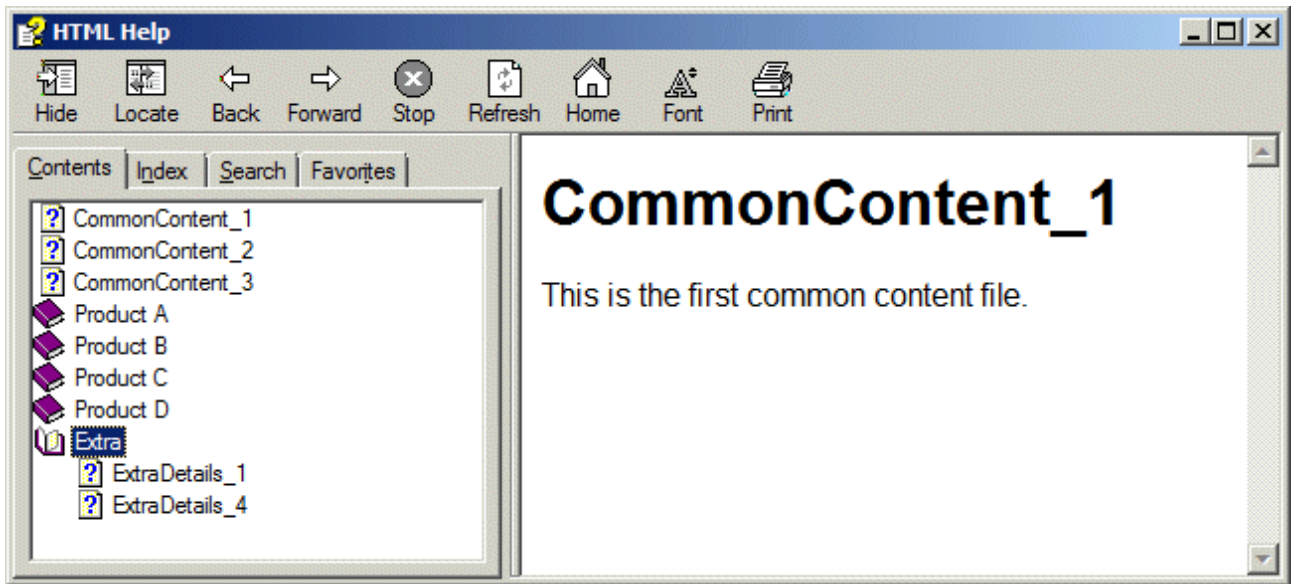
As NotReleased is set for exclusion in the target, the build log clearly shows that ProductC_1 is excluded.

Excluded from output:file:///C:/Products/Products/Content/Product_C/ProductC_1.htm.
Excluded from output:file:///C:/Products/Products/Content/Extra/ExtraDetails_1.htm.
Excluded from output:file:///C:/Products/Products/Content/Extra/ExtraDetails_2.htm.
Excluded from output:file:///C:/Products/Products/Content/Extra/ExtraDetails_3.htm.
Excluded from output:file:///C:/Products/Products/Content/Extra/ExtraDetails_4.htm.

Consider an example where condition A is applied to the Extra folder. The two topics therein (ExtraDetails_2 and ExtraDetails_3) have the NotReleased exclusion condition.



The two topics are excluded from the output.



Conclusion

A condition at folder level does not override the conditions of the topics therein, unless the folder level condition is actively set to Include or Exclude.

Topic-level contra content-level conditions

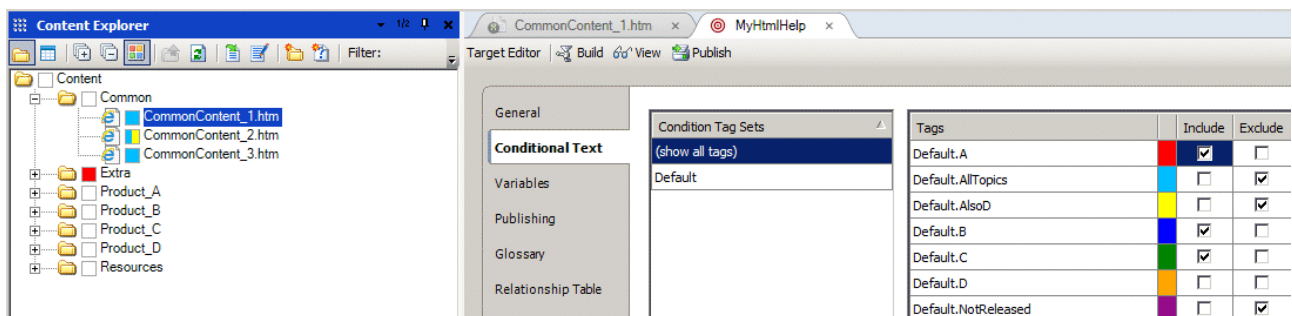
Conditions applied to topic content never determine the inclusion of the topic itself. The inclusion of a topic is always determined by topic-level and/or file-level conditions.

Example 12 – Topic-level conditions override content conditions

Consider a topic that has four paragraphs each with their own condition.



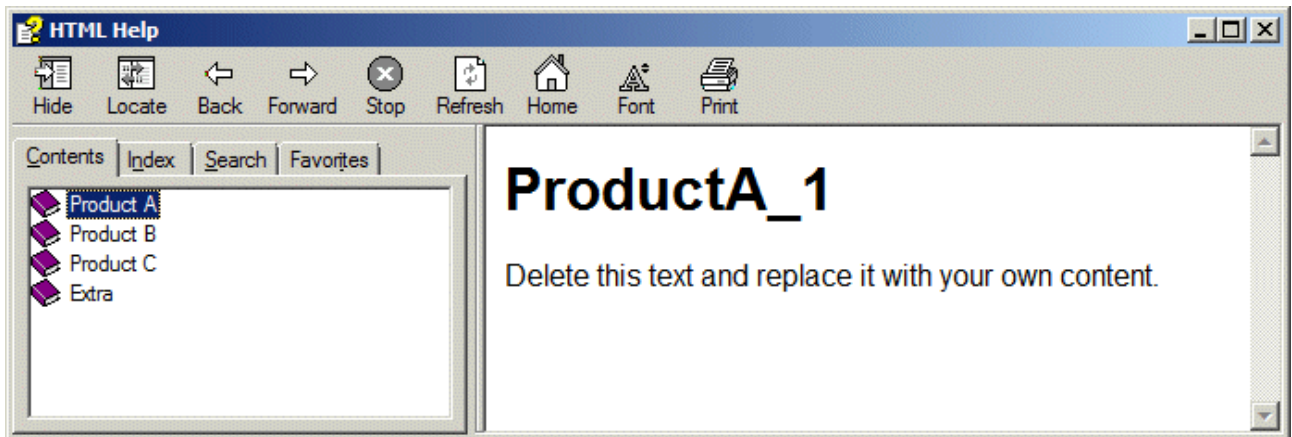
The target is set to include the conditions assigned to the paragraphs, but the file condition is set for exclusion.



The build log indicates the topic is excluded.

```
Excluded from output:file:///C:/Products/Products/Content/Common/CommonContent_1.htm.  
Excluded from output:file:///C:/Products/Products/Content/Common/CommonContent_2.htm.  
Excluded from output:file:///C:/Products/Products/Content/Common/CommonContent_3.htm.  
Excluded from output:file:///C:/Products/Products/Content/Product_D/ProductD_1.htm.
```

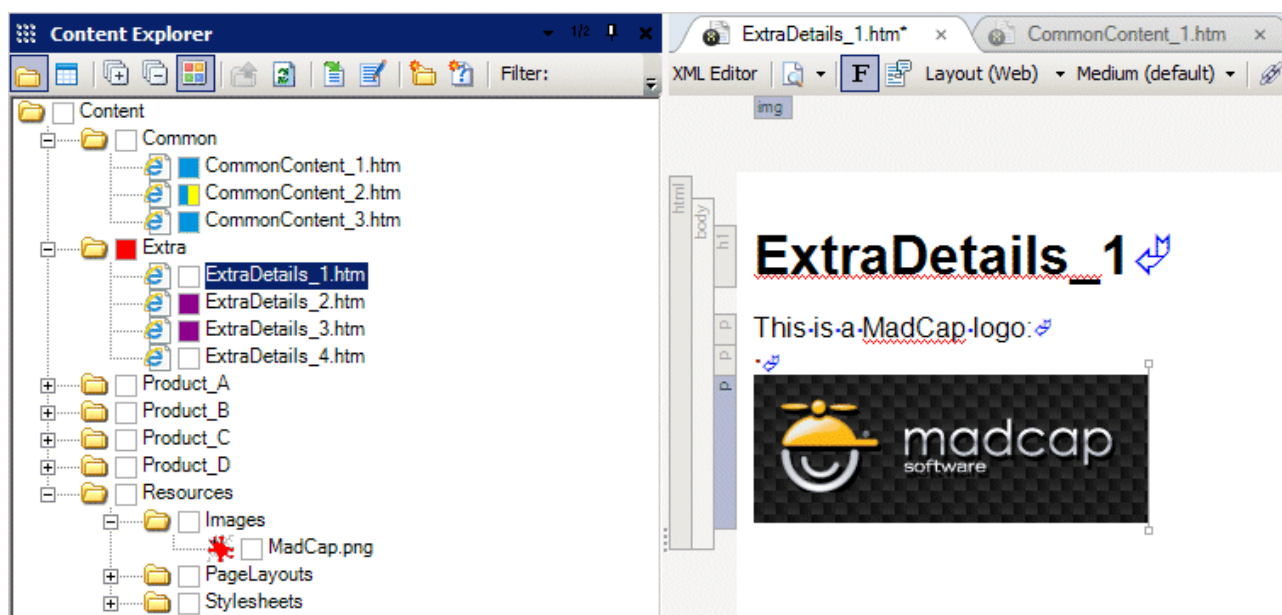
The resulting output does not include the topic as the topic level condition is set for exclusion.



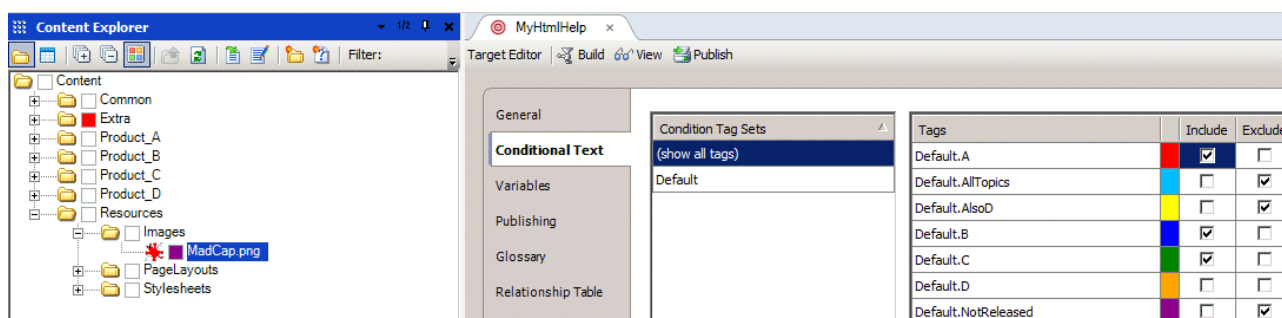
Example 13 – Image conditions

Conditions can be assigned to images. Images that are not assigned conditions are included in the CHM output file and results in bloated files. While these unused images cannot be searched in the HTML Help reader, the CHM can be decompiled and the images can be accessed. This is a serious consideration if undistributed or unreleased content should not be available.

Consider a topic that contains an image.



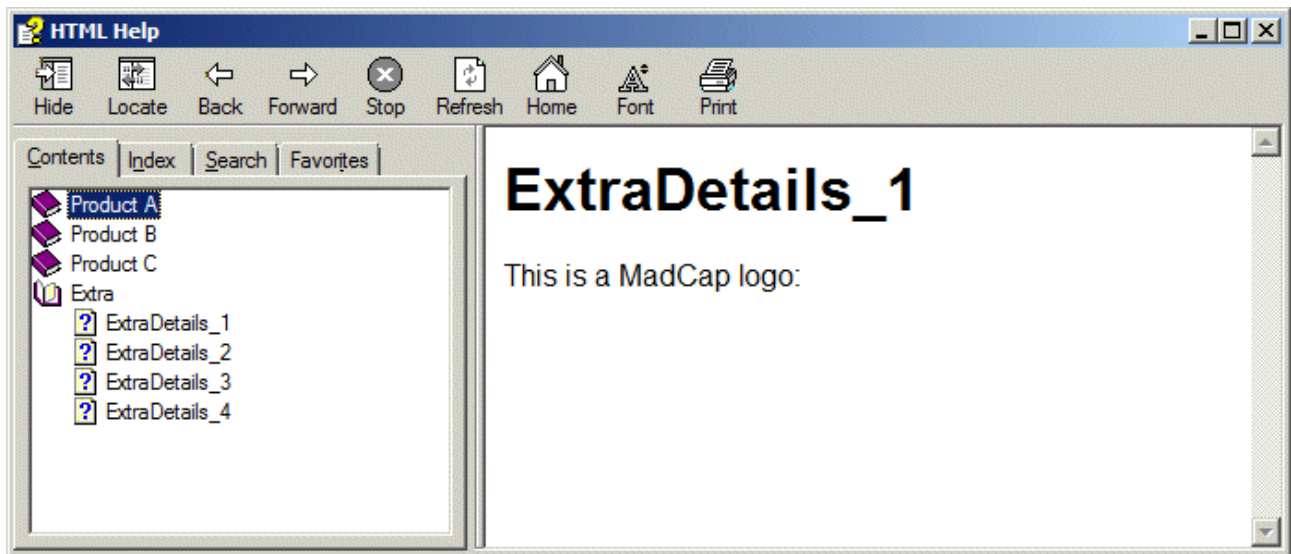
Simply tagging the image in the topic with the NotReleased condition and then ensuring the condition is set to Exclude will remove the image from the output. However, if the image is used in numerous topics and/or snippets then this procedure must be repeated for each occurrence thereof across the entire project. It is however easiest is to tag the image file itself, or where possible the folder, with the NotReleased condition.



The build log indicates the image was excluded:

```
Excluded from output:file:///C:/Products/Products/Content/Resources/Images/MadCap.png.
```

In the output the image never appears and is not part of the output file.



Obviously for print output this is not an issue as unused files are simply not included.

Conclusion

An exclusion condition overrides the inclusion condition, irrespective of where the inclusion and exclusion conditions are placed. For example, if tagged for inclusion at file level, but exclusion in the topic content, or vice-versa, the image is always excluded.

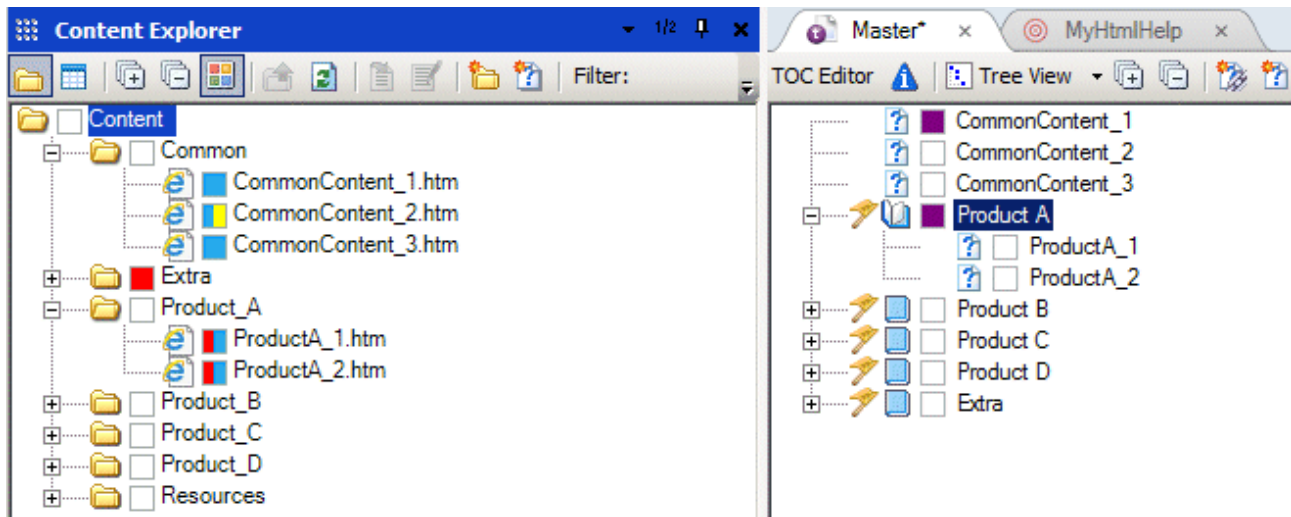
As with images, snippets have reusability across an entire project and it is most effective to apply conditioning at snippet (.flsnp) file level or folder level. It is of course possible to condition snippets in the topics itself or even in other snippets where nested content is applied. Unlike images, snippets are compiled into the topics when the target is generated. Unused snippets are simply not included in the output (in these examples CHM files) and have no affect on size of the output file.

As with images, an exclusion condition overrides the inclusion condition, irrespective of where the inclusion and exclusion conditions are placed. For example, if tagged for inclusion at file level, but exclusion in the topic content, or vice-versa, the snippet is always excluded.

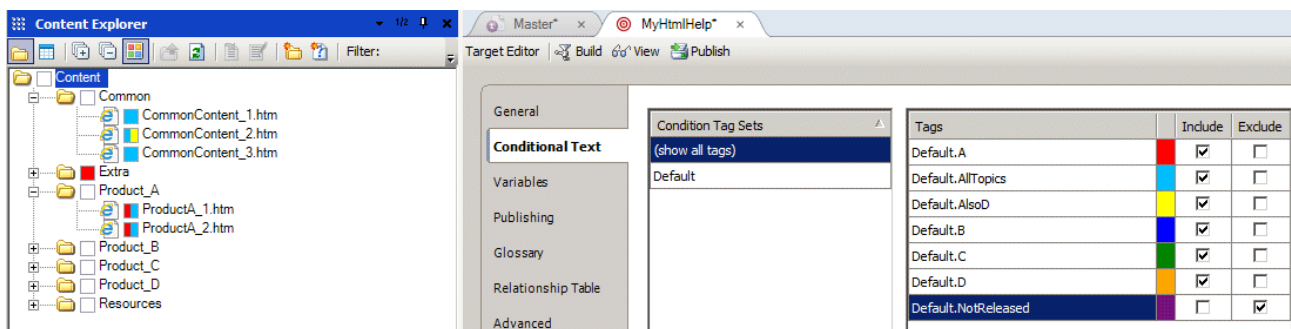
Example 15 – ToC conditions

Books and items in the project's ToC files can also be conditioned. Applying a condition to a ToC entry simply affects the appearance of the entry in the ToC of the output. It does not affect the inclusion of the associated file, for example a topic, in the output file.

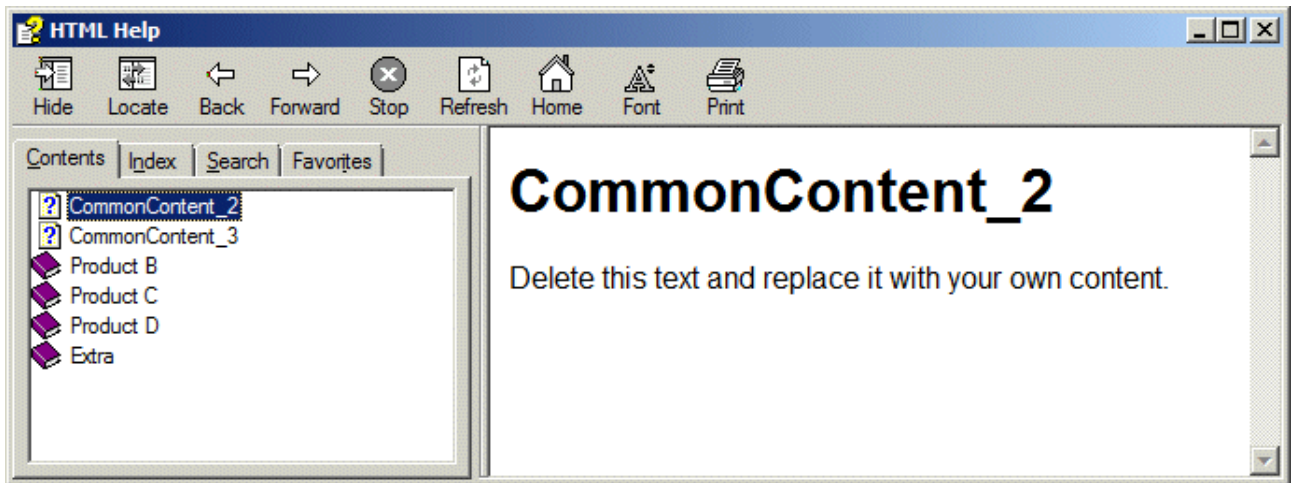
In this example the CommonContent1 item and Product A book have been conditioned with a NotReleased condition.



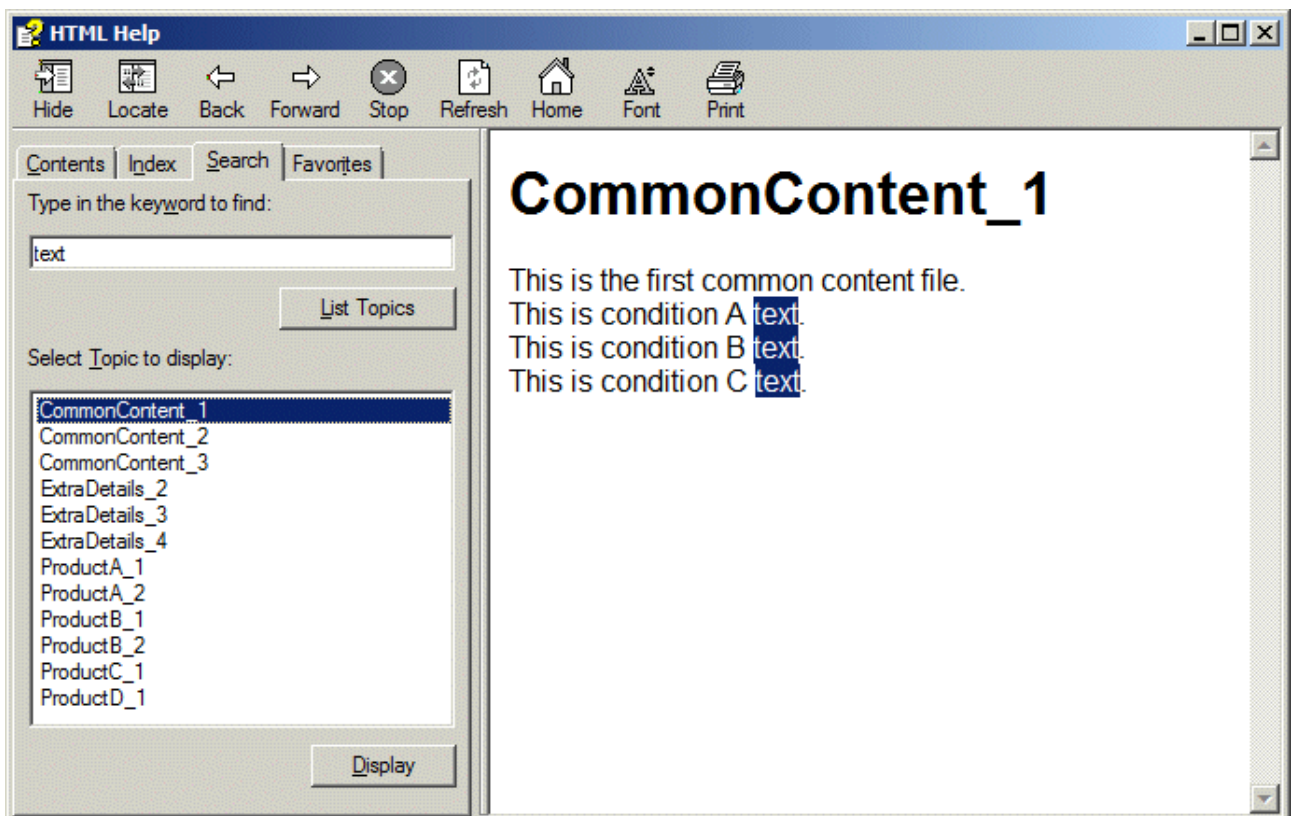
The target is defined so that NotReleased is excluded.



The output shows that the CommonContent_1 and Product A ToC entries have been excluded.



However, only the ToC entries are hidden, the content can be searched.



Conclusion

Applying conditions to ToC entries does not necessarily exclude the content from the output, for example in CHM files and WebHelp. The content can still be searched and read. In printed output, for example PDF and MS Word, the content is obviously not included.

Implications of conditions on content versioning

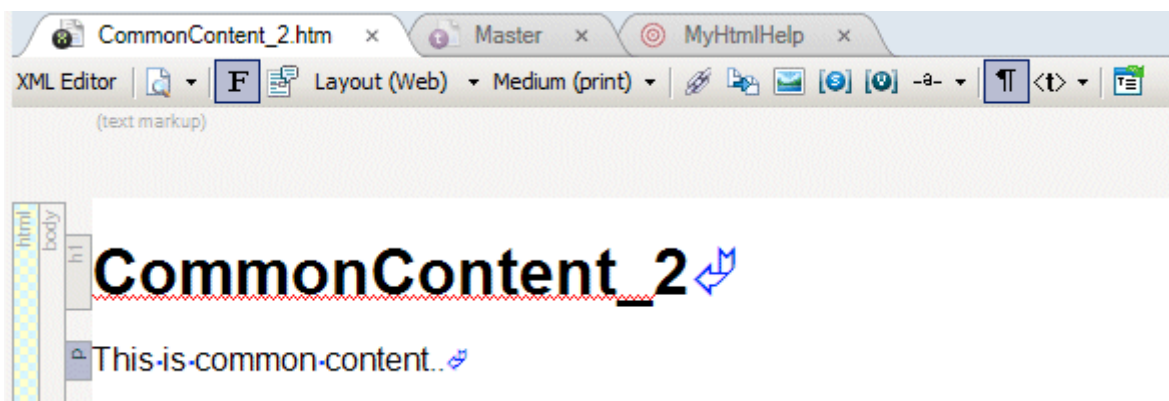
Applying conditions to file content will modify the file's content. Saving the file updates the time stamp and implies that a new version of the file has been created. Even though the visible readable content of the file has not been altered, the file will still be regarded as having been modified and therefore eligible for revision and retranslation. Having a translation team check what is most likely unchanged content, is not only time consuming, but extremely costly.

Example 16 – File-level conditions and file header information

The following is the XML content of the ComonContent_2 topic.

```
<?xml version="1.0" encoding="utf-8"?>
<html xmlns:MadCap="http://www.madcapsoftware.com/Schemas/MadCap.xsd"
MadCap:conditions="Default.AllTopics,Default.AlsoD" MadCap:lastBlockDepth="2"
MadCap:lastHeight="100" MadCap:lastWidth="728">
  <head>
  </head>
  <body>
    <h1>CommonContent_2</h1>
    <p>This is common content..</p>
  </body>
</html>
```

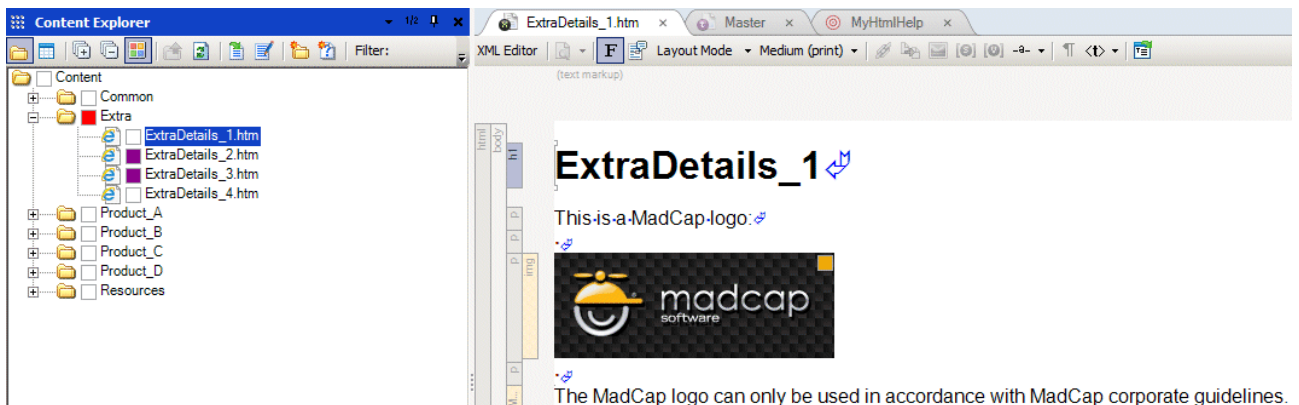
Note that the conditions are applied in the `<html>` tag which is not obvious when viewing the topic in the WYSIWYG editor or target output.



Removing or adding a condition will update the content of the `<html>` tag forcing a save of the file and applying a new time stamp.

Example 17 – Folder-level conditions and file content

Consider the ExtraDetails_1 topic.



The following is the XML content of the ExtraDetails_1 topic.

```
<?xml version="1.0" encoding="utf-8"?>
<html xmlns:MadCap="http://www.madcapsoftware.com/Schemas/MadCap.xsd"
MadCap:lastBlockDepth="4" MadCap:lastHeight="255" MadCap:lastWidth="702">
  <head>
  </head>
  <body>
    <h1>ExtraDetails_1</h1>
    <p>This is a MadCap logo:</p>
    <p>&#160;</p>
    <p>
      
    </p>
    <p>&#160;</p>
    <MadCap:snippetBlock src="../../Resources/Snippets/SnippetLogoUseRestriction.flshp"
MadCap:conditions="Default.D" />
  </body>
</html>
```

The only condition tagging in the content is that applied to the snippet and image.

Example 18 - Snippet file-level conditions and file content

The association of conditions on snippet files follows the same criteria as seen with topics, the file's `<html>` tag is affected.

Example 19 - Snippet folder-level conditions and file content

The association of conditions on snippet folders follows the same criteria as seen with topics, the content is not affected.

Conclusion

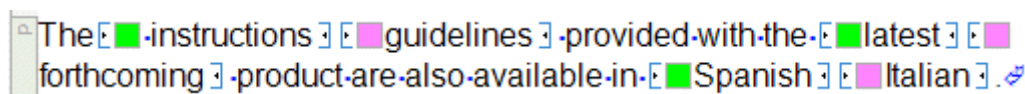
Folder-level conditions do not affect file content. This is a major reason for using folder-level conditions for content that requires translation and/or localization. Namely, the conditions do not force new versions of a file.

Conditions and content complexity

Assigning conditions to topic content can adversely affect the readability of the text. Consider the following example where words in a sentence have been conditioned with two conditions – USA and Europe. The conditions are defined as:

	Europe		
	USA		

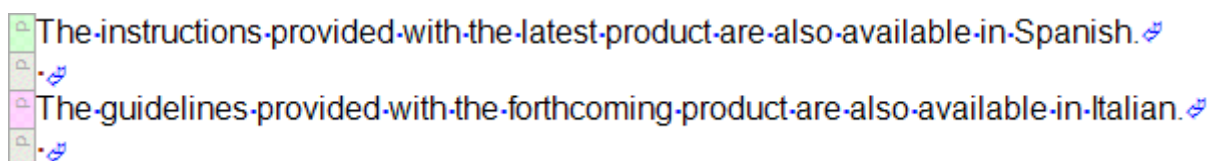
The conditioned text appears as:



When viewing the XML of the sentence it is even more complex and somewhat unreadable.

```
<p>The<MadCap:conditionalText MadCap:conditions="Default.Europe">
instructions</MadCap:conditionalText><MadCap:conditionalText
MadCap:conditions="Default.USA">guidelines</MadCap:conditionalText> provided with the
<MadCap:conditionalText
MadCap:conditions="Default.Europe">latest</MadCap:conditionalText><MadCap:conditionalText
MadCap:conditions="Default.USA">forthcoming</MadCap:conditionalText> product are also
available in <MadCap:conditionalText
MadCap:conditions="Default.Europe">Spanish</MadCap:conditionalText><MadCap:conditionalText
t MadCap:conditions="Default.USA">Italian</MadCap:conditionalText>.</p>
```

This can pose some serious translation and/or localization challenges. Due to incoherency words can be misplaced, resulting in a need for many reviews and corrections. To avoid this, the sentence (or paragraph) could be replaced by two independent sentences, each with their own condition, for example:

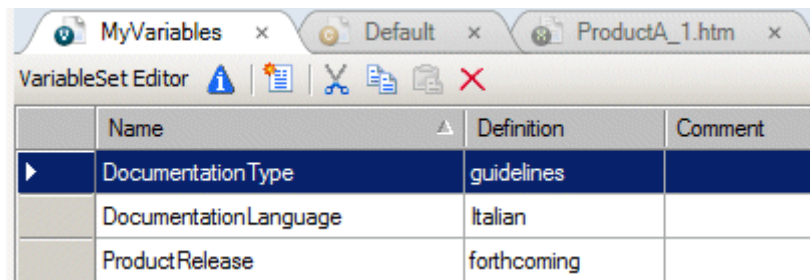


While duplicating content, the XML is much more simple and will certainly save time and expense as regards localization:

```
<p MadCap:conditions="Default.Europe">The instructions provided with the latest product
are also available in Spanish.</p>

<p MadCap:conditions="Default.USA">The guidelines provided with the forthcoming
product are also available in Italian.</p>
```

Another possibility is to use variables, for example:



The screenshot shows a web browser window with three tabs: 'MyVariables', 'Default', and 'ProductA_1.htm'. The 'MyVariables' tab is active, displaying the 'VariableSet Editor' interface. It contains a table with four columns: 'Name', 'Definition', and 'Comment'. The table lists three variables: 'DocumentationType' with definition 'guidelines', 'DocumentationLanguage' with definition 'Italian', and 'ProductRelease' with definition 'forthcoming'.

Name	Definition	Comment
DocumentationType	guidelines	
DocumentationLanguage	Italian	
ProductRelease	forthcoming	

This appears as:

The Document... guidelines provided with the ProductRe... forthcoming product are also available in Document... Italian.

The XML code for this appears as:

```
<p>The <MadCap:variable name="MyVariables.DocumentationType" /> provided with the  
<MadCap:variable name="MyVariables.ProductRelease" /> product are also available in  
<MadCap:variable name="MyVariables.DocumentationLanguage" />.</p>
```

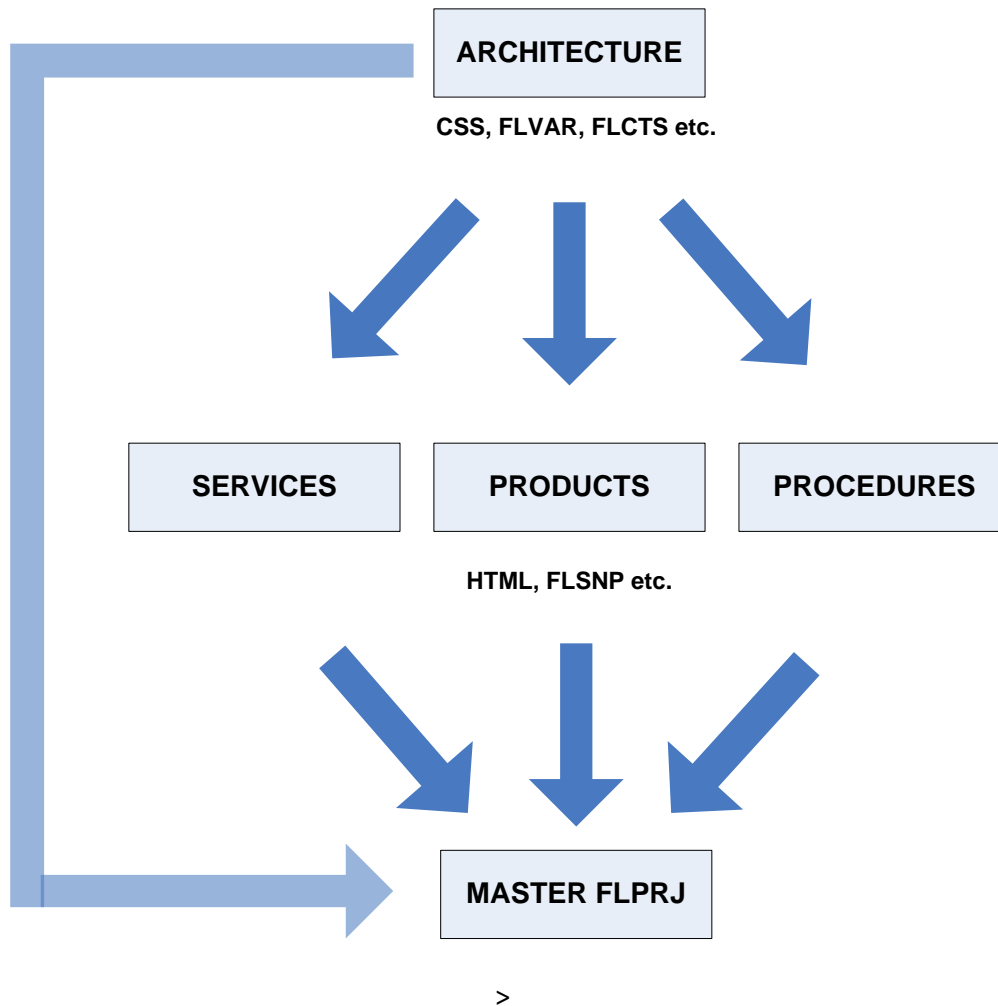
This is still less complex than that seen with conditions.

Conclusion

Conditions can be applied at word, sentence or paragraph level. There are however consequences when using multiple conditions, especially regarding the readability of the content for translation and localization. Whether or not the creation of limited duplicate content and/or the use of variables is viable, depends upon the nature of the project and the complexity of its content.

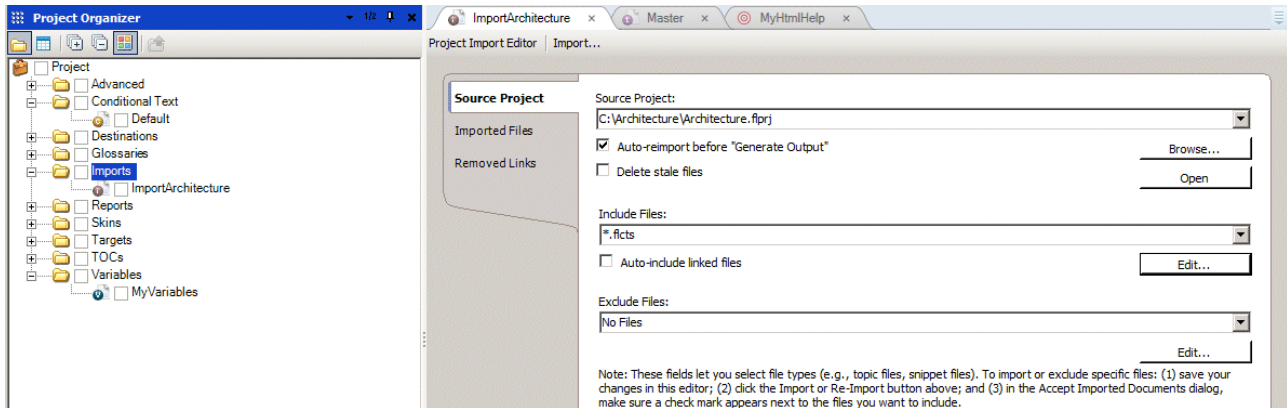
Condition consistency across multiple projects

Flare Import Project Files can be used to ensure that a consistent set of conditions is used across all projects. For example, a common Architecture project can be created as a repository for corporate or organizational templates, style sheets, common images, condition sets, variables and more. An Import Project file in each subproject can be used to import these definitions from the Architecture project.



A Master project can be used for compilation of the target output. This is by importing the content from the various subprojects, while also receiving the condition sets from the Architecture project.

The following depicts an example of an Import Project File that has been configured to import the condition set (.flcts) files from an Architecture project.



When using this approach it is not necessary to manually create each new condition in each project. Simply performing the import will make new and updated condition available for use. This also avoids the likelihood of condition naming errors. The use of Import Project Files is described in detail in the MadCap Flare online help in Features > Global Project Linking

Summary of issues to consider

The previous sections result in a basic checklist of issues that should be considered when using conditions in Madcap Flare projects.

☒ **Is the structure of the project amenable to conditioning?**

Essentially the project should be as atomized as possible. This means the optimal use of snippets and variables and having these structured so they can be applied across project topics as and where necessary. The project's topics should be no more than skeletons that have placeholders for snippets and other shared resources. Where possible, conditions are applied at folder level, then file level and, if necessary, on the placeholders in the topic content.

☒ **Does the project include a set of common basic files for all targets?**

If yes, can I expect future developments where only part of these common files will be applicable for certain targets?

If yes, consider inclusion or exclusion tagging on these common files, based upon the number of expected targets and the quantity of common files that will be relevant.

☒ **Is the project subject to translation and/or localization?**

If yes, consider the use of folder-level conditions as opposed to topic-level conditions.

If yes, consider the use of variables in sentences or even correctly conditioned repeating sentences (or paragraphs) as opposed to in-line conditions. This is to avoid complex and ambiguous sentence constructions.

☒ **Does the output type include unreferenced content, for example HTML Help CHM files including unused images?**

If yes, consider the conditioning of image files to avoid their unnecessary inclusion in the output . this will reduce output file sizes.

- ☒ **Does the output type allow searching of content that is not referenced from the ToC, for example as in CHM files?**

If yes, consider that the conditioning of ToC entries for exclusion does not necessarily exclude the content from the output file. Exclusion of content in specific types of targets can only be assured by conditioning at folder and/or topic level.

- ☒ **Does the project form part of a larger more comprehensive project?**

If yes, consider the use of import files to ensure consistent conditions definitions across all projects. This avoids manually managing the conditions in each and every project.

There are undoubtedly many other issues that could come into consideration, but many of these are dependent on the nature of the project and the future strategy that it is subject to.

Concluding comments

The conditioning of MadCap Flare projects cannot be generalized as companies, projects and demands differ. This document has attempted to explain some of the most common and pertinent issues that could arise when conditioning content in MadCap Flare projects. By being aware of these issues some of the basic and most important decisions can be taken, or as a minimum, be aware of the consequences of such decisions.