



Welcome

Welcome to the Sandcastle MAML Guide. This is intended to be a reference that you can use to find out all that you need to know about Microsoft Assistance Markup Language (MAML) as used with Sandcastle to produce help files containing conceptual content. Currently, there is very little information about MAML and all of the features that it offers. As such, consider this a work in progress. If you have any information, tips, updates, or corrections that you would like to see added to the guide, feel free to submit them to the author using the **Send Feedback** link at the top of the page or the e-mail link in the page footer.

▲ What is Conceptual Content?

Conceptual content allows you to define non-API documentation that is included in the help file for usage notes, walkthroughs, tutorials, etc. Instead of HTML, conceptual content topics use Microsoft Assistance Markup Language (MAML). This is basically XML conforming to a well defined schema that describes the structure of the conceptual content much like XML comments describe the structure of the code comments. There is no layout or style information within the MAML files. Instead, they are ran through a series of XSL transformations using Sandcastle's **BuildAssembler** tool similar to the reference (API) content so that they match it in appearance and features. This allows you to utilize the various Sandcastle and third-party build components to alter and extend the presentation style used in the topics.

The term "conceptual content" is actually a catch-all term used to refer to the various types of content that may appear in the help file. MAML actually defines several different content types, each used for a specific type of document (conceptual, glossary, how-to, reference, etc.). Each type has a specific structure with various required and optional XML elements. This help file will let you get to know the various document and element types.

▲ Creating Conceptual Content

Creating the actual topic files is just a matter of entering the content using a text editor. Creating a help file from them requires several configuration files and the Sandcastle tools. If you are going to build a help file using Sandcastle alone, see the topics in the [Conceptual Content File Types](#) section. They cover the various file types that are used in the build process.

If you prefer a more visual and automated approach to managing and building help files, it is strongly recommended that you use one of the third-party tools such as the [Sandcastle Help File Builder](#). They contain all of the necessary features and tools required to make building help files with Sandcastle a much simpler process. This help file was produced using the [Sandcastle Help File Builder](#) and is an example of what the VS2013 presentation style looks like.

▲ See Also

Other Resources

[Known Issues and Bugs](#)

[Links to Resources](#)

[Conceptual Content File Types](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



License Agreement

This license governs use of the accompanying software. If you use the software, you accept this license. If you do not accept the license, do not use the software.

▲ Microsoft Public License (Ms-PL)

1. Definitions

The terms "reproduce," "reproduction," "derivative works," and "distribution" have the same meaning here as under U.S. copyright law.

A "contribution" is the original software, or any additions or changes to the software.

A "contributor" is any person that distributes its contribution under this license.

"Licensed patents" are a contributor's patent claims that read directly on its contribution.

2. Grant of Rights

(A) Copyright Grant - Subject to the terms of this license, including the license conditions and limitations in section 3, each contributor grants you a non-exclusive, worldwide, royalty-free copyright license to reproduce its contribution, prepare derivative works of its contribution, and distribute its contribution or any derivative works that you create.

(B) Patent Grant - Subject to the terms of this license, including the license conditions and limitations in section 3, each contributor grants you a non-exclusive, worldwide, royalty-free license under its licensed patents to make, have made, use, sell, offer for sale, import, and/or otherwise dispose of its contribution in the software or derivative works of the contribution in the software.

3. Conditions and Limitations

(A) No Trademark License - This license does not grant you rights to use any contributors' name, logo, or trademarks.

(B) If you bring a patent claim against any contributor over patents that you claim are infringed by the software, your patent license from such contributor to the software ends automatically.

(C) If you distribute any portion of the software, you must retain all copyright, patent, trademark, and attribution notices that are present in the software.

(D) If you distribute any portion of the software in source code form, you may do so only under this license by including a complete copy of this license with your distribution. If you distribute any portion of the software in compiled or object code form, you may only do so under a license that complies with this license.

(E) The software is licensed "as-is." You bear the risk of using it. The contributors give no express warranties, guarantees or conditions. You may have additional consumer rights under your local laws which this license cannot change. To the extent permitted under your local laws, the contributors exclude the implied warranties of merchantability, fitness for a particular purpose and non-infringement.

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Known Issues and Bugs

This topic covers known issues and bugs with Sandcastle and conceptual content. Currently, the VS2013 presentation style is the best choice for conceptual content as it contains no known issues beyond those noted in the **General Issues** section.

▲ General Issues

These are the currently known issues with Sandcastle in general when used to produce conceptual content regardless of presentation style.

- There does not appear to be any formatting for several of the **Whitepaper** topic elements.
- Several elements of the **XML Reference** topic type do not render correctly if at all.
- Sandcastle's XSL transformations remove inter-element whitespace. As such, if you place two MAML elements one after the other and require a separating space between them, use a non-breaking space entity (` `). See the [replaceable](#) topic for an example.

▲ See Also

Other Resources

[Welcome](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Conceptual Content File Types

The topics in this section define the various file types used to produce conceptual content help files. If you are using a tool such as the [Sandcastle Help File Builder](#), only the topic file information will be of interest to you. The other file types are used by the build process and may only be of interest if you want to see how everything fits together and is used to build the help topics.

▾ Conceptual Content File Types

Select one of the topics below to learn more about a particular file type.

- [Code Snippets File](#)
- [Media Content File](#)
- [Token File](#)
- [Topic File](#)
- [Companion File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Code Snippets File

The code snippets file allows you to define code samples outside of conceptual topic files. Each snippet has a unique ID that can be referenced in multiple topics using a [codeReference](#) MAML element so that the snippets do not have to be copied into each one. Automatic colorization and language filter synchronization is supported by specifying a language identifier on the code samples.

Currently supported identifiers are [VisualBasic](#), [CSharp](#), [ManagedCPlusPlus](#), [JSharp](#), and [JScript](#). Custom colorization rules can be added to the conceptual configuration file for the [ExampleComponent](#).

Note

When using the [Sandcastle Help File Builder](#) or the standalone build components from it, the **Code Block Component** offers the same set of features and several others for both inline and external code snippets. In addition, it supports many other languages and can import code from working source code files which allows you to pull in examples from working projects that you can maintain and build without having to update a separate example snippets file. As such, you may prefer to use it rather than the code snippets file.

Code Snippets File Format

Below is an example of a code snippets file.

Example Code Snippets File

[Copy](#)

```
<?xml version="1.0" encoding="utf-8" ?>
<!-- This is an example code snippets file -->
<examples>
  <item id="ClassDefinition#Define">
    <sampleCode language="CSharp">
      public class CSharpClass()
```

```

        {
            // Members go here
        }
    </sampleCode>
    <sampleCode language="VisualBasic">
        Public Class VBClass
            ' Members go here
        End Class
    </sampleCode>
</item>

<item id="CreateInstance#Local">
    <sampleCode language="CSharp">
        CSharpClass x = new CSharpClass();
    </sampleCode>
    <sampleCode language="VisualBasic">
        Dim x As VBClass = New VBClass()
    </sampleCode>
</item>

<item id="CreateInstance#Static">
    <sampleCode language="CSharp">
        public static CSharpClass sharedInstance =
    </sampleCode>
    <sampleCode language="VisualBasic">
        Public Shared sharedInstance As VBClass = N
    </sampleCode>
</item>
</examples>

```

The file consists of a root `examples` node that contains one or more `item` nodes. The `item` element has an `id` attribute that gives each one a unique identifier. This is used as the inner text of the `codeReference` element in a topic to represent the snippet.

Note

The ID value must consist of an example ID, a hash character (#), and a sample ID. The example and sample ID values are

not case-sensitive. The example ID does not have to be unique. However, when combined with a sample ID, the entire value must be unique.

Each item must contain one or more `sampleCode` elements with a `language` attribute identifying the language of the sample code. The valid language IDs are noted above. The inner XML of the `sampleCode` element is the code snippet.

▲ See Also

Other Resources

[Conceptual Content File Types
codeReference](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Media Content File

The media (images) used by conceptual content in a help file project are defined in a separate media content file. Each is assigned a unique ID that is used to refer to the image in conceptual topic links. When a help file is built, all images referenced by an image link are copied to the help file's *.Media* folder. Any unused images are omitted.

Note

The [Sandcastle Help File Builder](#) does not use a media content file directly. Instead, images are managed as project file items and the media content file is created for you automatically at build time based on the property values on those items.

Media Content File Format

Below is an example of a media content file.

Example Media Content File

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This is an example media content file -->
<stockSharedContentDefinitions>
  <item id="4cfacbe8-d100-471c-9d69-1dbcce50e316"
    <image file="..\ProjectImages\SmallInformatic
      <altText>Information</altText>
    </image>
  </item>
  <item id="6be7079d-a9d8-4189-9021-0f72d1642beb"
    <image file="..\ProjectImages\SandcastleLogo
      <altText>Sandcastle</altText>
    </image>
  </item>
  <item id="98a8a8b7-c374-40c7-902a-91c947bf107c'
```

```
<image file="..\ProjectImages\SandcastleLogo&#x2013;
</item>
<item id="b5d069f7-5daf-45ae-882c-4596ab2b724a'
  <image file="..\ProjectImages\SecuritySuccess&#x2013;
</item>
</stockSharedContentDefinitions>
```

The file consists of a root `stockSharedContentDefinitions` node that contains one or more `item` nodes that represent the images.

The `item` element has an `id` attribute that gives each one a unique identifier. This is used as the target of the media link elements in the topics. Within the item element is a required `image` element used to define the image. The `file` attribute should point to the location of the image file. A relative or absolute path can be used as long as it can be located at build time. If you would like to display alternate text when the mouse hovers over the image, nest an `altText` element within the image element as shown in the first item in the example above. The inner text for it defines the alternate text to display.

Note

The unique ID is not case-sensitive. The example above uses GUID values for the ID but you are free to use any unique value that you like.

Caution

Unlike topics, image files will retain their name when copied to the media folder. As such, if you add two images from different locations that have the same name, one will overwrite the other when the help file is built.

See Also

Other Resources

[Conceptual Content File Types](#)

[mediaLink](#)

[mediaLinkInline](#)

[v1.1.0.3] Sandcastle MAML Guide

Send comments on this topic to [Eric Woodruff](#)



Token File

A token is used as a replaceable tag within a topic and is represented using a `token` element. The inner text of the element is a token name. The tokens are defined in a separate token file. They are an easy way to represent common link items that you use regularly such as a common phrase or external link.

Token File Format

Below is an example of a token file.

Example Token File

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This is an example tokens file -->
<content xml:space="preserve"
  xmlns:dde="http://dde.schemas.microsoft.com/1.2.1/Content/Serializing"
  xmlns:xlink="http://www.w3.org/1999/xlink">

  <item id="Sandcastle"><dde:externalLink>
    <dde:linkText>Sandcastle</dde:linkText>
    <dde:linkUri>https://GitHub.com/EWSoftware/Sandcastle</dde:linkUri>
  </dde:externalLink></item>
  <item id="SHFB"><dde:externalLink>
    <dde:linkText>Sandcastle Help File Builder</dde:linkText>
    <dde:linkUri>https://GitHub.com/EWSoftware/Sandcastle</dde:linkUri>
  </dde:externalLink></item>
  <item id="autoOutline"><autoOutline /></item>
</content>
```

The file consists of a root `content` node that contains one or more `item` nodes. Be sure to add the `xml:space` attribute to preserve

whitespace and the two namespace attributes on the `content` element.

The `item` element has an `id` attribute that gives each one a unique identifier. This is used as the inner text of the `token` element in a topic to represent the token. The inner XML of the item will be used to replace the token at build-time.

Note

The token ID is not case-sensitive.

Token Content

The content of a token item can be anything you like and may include other MAML elements. In the example above, the first four elements are used to define common external links used by this project.

Note

If you embed MAML elements within a token, you must prefix each element name with `"ddue:"` (without the quotes) as shown in the example above. If not, the elements are inserted using the default empty namespace and they will not be processed by the Sandcastle transformations.

The `autoOutline` token is a special case used by earlier versions of Sandcastle to insert an auto-generated section outline within a topic's introduction or sub-sections. See the `autoOutline` topic for more information.

Tip

The `Sandcastle Help File Builder` adds the `autoOutline` token as part of the build automatically. As such, you do not need to define it in the project yourself.

Tip

Sandcastle contains an extension to the MAML schema and the `autoOutline` element can be added directly in the topic without the need for embedding it in a token.

▲ See Also

Other Resources

[Conceptual Content File Types](#)

[autoOutline](#)

[token](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Topic File

The topic file is used to contain a single conceptual topic. Instead of HTML, conceptual content topics use Microsoft Assistance Markup Language (MAML). This is basically XML conforming to a well defined schema that describes the structure of the conceptual content much like XML comments describe the structure of the code comments. There is no layout or style information within the MAML files. Instead, they are ran through a series of XSL transformations using Sandcastle's **BuildAssembler** tool similar to the reference (API) content so that they match it in appearance and features. This allows you to utilize the various build components to alter and extend the presentation style of the topics.

▲ Topic File Structure

MAML defines several different content types, each used for a specific type of document (conceptual, glossary, how-to, reference, etc.). Each type has a specific structure with various required and optional XML elements. When building conceptual content, each topic requires a unique ID in the form of a globally unique identifier (GUID) and a revision number. These are specified in a containing `topic` element in the content file as shown in this example.

Topic Example

[Copy](#)

```
<!-- The topic element contains the unique ID and
<topic id="303c996a-2911-4c08-b492-6496c82b3edb"
  <!-- This element name will change based on the
  <developerConceptualDocument
    xmlns="http://ddue.schemas.microsoft.com/aut
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!-- The content goes here -->

  </developerConceptualDocument>
</topic>
```



Note

The `topic` element is not part of the MAML schema but is required in order to build the help file. The example topics below include a `topic` element with placeholder `ID` and `RevisionNumber` values. If you copy these files, be sure to replace the ID in each topic with a unique value of your own for each file that you create.

Document Types

Select one of the topics below to learn more about a particular document type.

- [Conceptual](#)
- [Error Message](#)
- [Glossary](#)
- [How-To](#)
- [Orientation](#)
- [Reference](#)
- [Reference With Syntax](#)
- [Reference Without Syntax](#)
- [Sample](#)
- [SDK Technology Architecture](#)
- [SDK Technology Code Directory](#)
- [SDK Technology Orientation](#)
- [SDK Technology Scenarios](#)
- [SDK Technology Summary](#)

- [Troubleshooting](#)
- [User Interface Reference](#)
- [Walkthrough](#)
- [Whitepaper](#)
- [XML Reference](#)

▲ See Also

Other Resources

[Conceptual Content File Types](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Conceptual

This document type is used for general conceptual topics.

Conceptual File Format

Below is an example of a Conceptual topic file.

Conceptual Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerConceptualDocument
    xmlns="http://ddue.schemas.microsoft.com/autf
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <!-- Uncomment this to generate an outline
      titles. Specify a numeric value as th
      a specific number of sub-topics when c
      zero (0) to limit it to top-level sect
    <!-- <autoOutline /> -->

    <para>Required introduction</para>
  </introduction>

  <!-- Add one or more top-level section elemer
```

```

        If using <autoOutline />, add an address
        and specify a title so that it can be ju
<section address="Section1">
  <title>Optional section title</title>
  <content>
    <!-- Uncomment this to create a sub-secti
    <autoOutline /> -->
    <para>Add one or more sections with conte
  </content>
  <!-- If a section contains a sections eleme
  sub-sections. These are not collapsib
  <sections>
    <section address="SubSection1">
      <title>Sub-section 1</title>
      <content>
        <para>Sub-section content.</par
      </content>
    </section>
    <section address="SubSection2">
      <title>Sub-section 2</title>
      <content>
        <para>Sub-section content.</par
      </content>
    </section>
  </sections> -->
</section>

<relatedTopics>
  <!-- One or more of the following:
  - A local link
  - An external link
  - A code entity reference

  <link xlink:href="Other Topic's ID"/>
  <link xlink:href="Other Topic's ID">Link in

  <externalLink>
    <linkText>Link text</linkText>
    <linkAlternateText>Optional alternate ]

```

```
<linkUri>URI</linkUri>
</externalLink>
```

```
<codeEntityReference>API member ID</codeEnt
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5
<link xlink:href="00e97994-e9e6-46e0-b420-5
```

```
<externalLink>
  <linkText>Sandcastle Help File Builder
  <linkAlternateText>Go to GitHub</linkAl
  <linkUri>https://GitHub.com/EWSoftware/
</externalLink>
```

```
<codeEntityReference>T:TestDoc.TestClass</c
<codeEntityReference>P:TestDoc.TestClass.Sc
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.Tc
<codeEntityReference>M:TestDoc.TestClass.Fi
<codeEntityReference>M:TestDoc.TestClass.Se
-->
```

```
</relatedTopics>
```

```
</developerConceptualDocument>
```

```
</topic>
```



▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Error Message

This document type is used for error message topics.

▲ Error Message File Format

Below is an example of an Error Message topic file.

Error Message Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerErrorMessageDocument
    xmlns="http://ddue.schemas.microsoft.com/autf
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!-- <nonLocErrorTitle>Optional non-localizab
  <!-- <secondaryErrorTitle>Optional localizabl

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <para>Required introduction</para>
  </introduction>

  <!-- <procedure>Optional. Zero or more proce
  <!-- <codeExample>Optional code example.</coc

  <relatedTopics>
```

```
<!-- One or more of the following:
- A local link
- An external link
- A code entity reference

<link xlink:href="Other Topic's ID">Link to
<externalLink>
  <linkText>Link text</linkText>
  <linkAlternateText>Optional alternate ]
  <linkUri>URI</linkUri>
</externalLink>
<codeEntityReference>API member ID</codeEnt
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5
<externalLink>
  <linkText>Sandcastle Help File Builder
  <linkAlternateText>Go to GitHub</linkA]
  <linkUri>https://GitHub.com/EWSoftware/
</externalLink>

<codeEntityReference>T:TestDoc.TestClass</c
<codeEntityReference>P:TestDoc.TestClass.Sc
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.Tc
<codeEntityReference>M:TestDoc.TestClass.Fi
<codeEntityReference>M:TestDoc.TestClass.Se
-->
```

```
</relatedTopics>
</developerErrorMessageDocument>
</topic>
```



▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Glossary

This document type is used to create a glossary.

▲ Glossary File Format

Below is an example of a Glossary topic file.

Glossary Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerGlossaryDocument
    xmlns="http://ddue.schemas.microsoft.com/autf
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <glossary>
    <!-- <title>Optional title</title> -->

    <!-- Glossary division. You can use these
    have one group, you can omit it and li
    directly.

    If used, be sure to give each glossary
    and a title element. These will be us
    top of the glossary page.

    Glossary divisions will appear in the
    Glossary entries will be sorted by the
    Terms are sorted by their value. -->
  <glossaryDiv address="Group1">
    <title>Group #1</title>
```

```
<glossaryEntry>
  <terms>
    <!-- One or more terms associated with the
         attribute if you need to link to a
         term. Otherwise, it can be omitted. Term
         list above the definition. -->
    <term termId="xyz">1st term</term>
    <term>Other term (if any)</term>
  </terms>

  <definition>
    <para>The definition of the term(s)</para>
  </definition>

  <!-- One or more optional related entries
         line after the definition with link
         attribute -->
  <!-- <relatedEntry termId="xyz" /> -->
</glossaryEntry>

</glossaryDiv>

</glossary>
</developerGlossaryDocument>
</topic>
```

▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



How-To

This document type is used for "how to" topics.

How-To File Format

Below is an example of a How-To topic file.

How-To Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerHowToDocument
    xmlns="http://ddue.schemas.microsoft.com/aut
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <para>Required introduction</para>
  </introduction>

  <!-- Optional procedures followed by optional
    at least one procedure or code example -
  <procedure>
    <title>Procedure title</title>
    <steps class="ordered">
      <step>
        <content>
```

```
        <para>First step</para>
    </content>
</step>
<step>
    <content>
        <para>Second step</para>
    </content>
</step>
</steps>
<!-- <conclusion>Optional conclusion</concl
</procedure>
```

```
<!-- <codeExample>Optional code example</code
```

```
<!-- <buildInstructions>Optional instructions
    code example.</buildInstructions> -->
```

```
<!-- <robustProgramming>Optional discussion c
    issues related to writing solid code.</r
```

```
<!-- <security>Optional discussion of securit
```

```
<relatedTopics>
```

```
    <!-- One or more of the following:
        - A local link
        - An external link
        - A code entity reference
```

```
<link xlink:href="Other Topic's ID">Link to
<externalLink>
```

```
    <linkText>Link text</linkText>
```

```
    <linkAlternateText>Optional alternate ]
```

```
    <linkUri>URI</linkUri>
```

```
</externalLink>
```

```
<codeEntityReference>API member ID</codeEnt
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5
```

```
<externalLink>
  <linkText>Sandcastle Help File Builder
  <linkAlternateText>Go to GitHub</linkAl
  <linkUri>https://GitHub.com/EWSoftware/
</externalLink>

<codeEntityReference>T:TestDoc.TestClass</c
<codeEntityReference>P:TestDoc.TestClass.Sc
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.Tc
<codeEntityReference>M:TestDoc.TestClass.Fi
<codeEntityReference>M:TestDoc.TestClass.Se
-->
</relatedTopics>
</developerHowToDocument>
</topic>
```



▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Orientation

This document type is used for orientation topics.

Orientation File Format

Below is an example of an Orientation topic file.

Orientation Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerOrientationDocument
    xmlns="http://ddue.schemas.microsoft.com/aut
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <para>Required introduction</para>
  </introduction>

  <!-- <inThisSection>Optional description of c
  <!-- <reference>Optional reference section</r
  <!-- <relatedSections>Optional related sectic
  <!-- <externalResources>Optional external res
```

```
<relatedTopics>
```

```
<!-- One or more of the following:  
- A local link  
- An external link  
- A code entity reference
```

```
<link xlink:href="Other Topic's ID">Link to  
<externalLink>  
  <linkText>Link text</linkText>  
  <linkAlternateText>Optional alternate ]  
  <linkUri>URI</linkUri>  
</externalLink>  
<codeEntityReference>API member ID</codeEnt
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5  
  
<externalLink>  
  <linkText>Sandcastle Help File Builder  
  <linkAlternateText>Go to GitHub</linkA]  
  <linkUri>https://GitHub.com/EWSoftware/  
</externalLink>
```

```
<codeEntityReference>T:TestDoc.TestClass</c  
<codeEntityReference>P:TestDoc.TestClass.Sc  
<codeEntityReference>M:TestDoc.TestClass.#c  
<codeEntityReference>M:TestDoc.TestClass.#c  
<codeEntityReference>M:TestDoc.TestClass.Tc  
<codeEntityReference>M:TestDoc.TestClass.Fi  
<codeEntityReference>M:TestDoc.TestClass.Se  
-->
```

```
</relatedTopics>
```

```
</developerOrientationDocument>
```

```
</topic>
```



▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Reference

This document type is used for reference topics.

Reference File Format

Below is an example of a Reference topic file.

```
Reference Topic Copy
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <codeEntityDocument
    xmlns="http://ddue.schemas.microsoft.com/autf
    xmlns:xlink="http://www.w3.org/1999/xlink">

    <!-- NOTE: This is similar to the API referer
      produced by the reference build. You sh
      instead of this. -->

    <assembly>
      <assemblyName>Assembly name</assemblyName>
    </assembly>
    <codeEntities>
      <codeEntity>
        <codeEntityReference>API member ID</codeE
      </codeEntity>
    </codeEntities>
  </codeEntityDocument>
</topic>
```

See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Reference With Syntax

This document type is used for reference with syntax topics.

Reference With Syntax File Format

Below is an example of a Reference With Syntax topic file.

Reference With Syntax Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerReferenceWithSyntaxDocument
    xmlns="http://ddue.schemas.microsoft.com/aut
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <!-- NOTE: This is similar to the API referer
    produced by the reference build. You sh
    instead of this. -->
  <introduction>
    <!-- Uncomment this to generate an outline
      titles. Specify a numeric value as th
      a specific number of sub-topics when c
      zero (0) to limit it to top-level sect
    <!-- <autoOutline /> -->

    <para>Required introduction</para>
```

```

</introduction>

<syntaxSection>
  <sections>
    <section>
      <title>Syntax section title</title>
      <content>
        <para>Syntax content</para>
      </content>
    </section>
  </sections>
  <legacySyntax>
    <parameterReference>Parameter reference</
  </legacySyntax>
</syntaxSection>

<!-- <parameters>Optional parameters info</pa
<!-- <returnValue>Optional return value info<
<!-- <exceptions>Optional exception info</exc
<!-- <languageReferenceRemarks>Optional langu
<!-- <codeExample>Optional code example</code
<!-- <dotNetFrameworkEquivalent>Optional .NET
<!-- <requirements>Optional requirements info

<!-- Add one or more top-level section elemer
      If using <autoOutline />, add an address
      it so that it can be jumped to with a hy
<section address="Section1">
  <title>Optional section title</title>
  <content>
    <!-- Uncomment this to create a sub-secti
    <autoOutline /> -->
    <para>Add one or more sections with conte
  </content>
</section>

<relatedTopics>
  <!-- One or more of the following:
    - A local link

```

- An external link
- A code entity reference

```
<link xlink:href="Other Topic's ID">Link to  
<externalLink>  
  <linkText>Link text</linkText>  
  <linkAlternateText>Optional alternate ]  
  <linkUri>URI</linkUri>  
</externalLink>  
<codeEntityReference>API member ID</codeEnt
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5  
<externalLink>  
  <linkText>Sandcastle Help File Builder  
  <linkAlternateText>Go to GitHub</linkA]  
  <linkUri>https://GitHub.com/EWSoftware/  
</externalLink>
```

```
<codeEntityReference>T:TestDoc.TestClass</c  
<codeEntityReference>P:TestDoc.TestClass.Sc  
<codeEntityReference>M:TestDoc.TestClass.#c  
<codeEntityReference>M:TestDoc.TestClass.#c  
<codeEntityReference>M:TestDoc.TestClass.Tc  
<codeEntityReference>M:TestDoc.TestClass.Fi  
<codeEntityReference>M:TestDoc.TestClass.Se  
-->
```

```
</relatedTopics>  
</developerReferenceWithSyntaxDocument>  
</topic>
```



See Also

Other Resources

[Topic File](#)

[v1.1.0.3] Sandcastle MAML Guide

Send comments on this topic to [Eric Woodruff](#)



Reference Without Syntax

This document type is used for reference without syntax topics.

Reference Without Syntax File Format

Below is an example of a Reference Without Syntax topic file.

Reference Without Syntax Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerReferenceWithoutSyntaxDocument
    xmlns="http://ddue.schemas.microsoft.com/aut
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <!-- NOTE: This is similar to the API referer
    produced by the reference build. You sh
    instead of this. -->
  <introduction>
    <!-- Uncomment this to generate an outline
      titles. Specify a numeric value as th
      a specific number of sub-topics when c
      zero (0) to limit it to top-level sect
    <!-- <autoOutline /> -->

    <para>Required introduction</para>
```

```
</introduction>
```

```
<!-- Add one or more top-level section elements here.
      If using <autoOutline />, add an address attribute
      and specify a title so that it can be justified.
```

```
<section address="Section1">
```

```
  <title>Optional section title</title>
```

```
  <content>
```

```
    <!-- Uncomment this to create a sub-section.
         <autoOutline /> -->
```

```
    <para>Add one or more sections with content here.
```

```
  </content>
```

```
</section>
```

```
<!-- <languageReferenceRemarks>Optional reference remarks </languageReferenceRemarks>
```

```
<!-- <codeExample>Optional code example </codeExample>
```

```
<relatedTopics>
```

```
  <!-- One or more of the following:
```

- A local link
- An external link
- A code entity reference

```
<link xlink:href="Other Topic's ID">Link text</link>
<externalLink>
```

```
  <linkText>Link text</linkText>
```

```
  <linkAlternateText>Optional alternate text</linkAlternateText>
```

```
  <linkUri>URI</linkUri>
```

```
</externalLink>
```

```
<codeEntityReference>API member ID</codeEntityReference>
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5" />
```

```
<externalLink>
```

```
  <linkText>Sandcastle Help File Builder</linkText>
```

```
  <linkAlternateText>Go to GitHub</linkAlternateText>
```

```
  <linkUri>https://GitHub.com/EWSoftware/
```

```
</externalLink>

<codeEntityReference>T:TestDoc.TestClass</c
<codeEntityReference>P:TestDoc.TestClass.Sc
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.Tc
<codeEntityReference>M:TestDoc.TestClass.Fi
<codeEntityReference>M:TestDoc.TestClass.Se
-->
</relatedTopics>
</developerReferenceWithoutSyntaxDocument>
</topic>
```



See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Sample

This document type is used for sample topics.

▲ Sample File Format

Below is an example of a Sample topic file.

Sample Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerSampleDocument
    xmlns="http://ddue.schemas.microsoft.com/autf
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <!-- Uncomment this to generate an outline
      titles. Specify a numeric value as th
      a specific number of sub-topics when c
      zero (0) to limit it to top-level sect
    <!-- <autoOutline /> -->

    <para>Required introduction</para>
  </introduction>

  <!-- <procedure>Optional procedures. See How
```

```
<!-- <requirements>Optional requirements sect
<!-- <demonstrates>Optional info about what i
<!-- <codeExample>Optional code example</code
```

```
<!-- Add one or more top-level section elemer
      If using <autoOutline />, add an address
      and specify a title so that it can be ju
```

```
<section address="Section1">
  <title>Optional section title</title>
  <content>
    <!-- Uncomment this to create a sub-secti
    <autoOutline /> -->
    <para>Add one or more sections with conte
  </content>
</section>
```

```
<relatedTopics>
```

```
<!-- One or more of the following:
      - A local link
      - An external link
      - A code entity reference
```

```
<link xlink:href="Other Topic's ID">Link te
<externalLink>
  <linkText>Link text</linkText>
  <linkAlternateText>Optional alternate ]
  <linkUri>URI</linkUri>
</externalLink>
<codeEntityReference>API member ID</codeEnt
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5
```

```
<externalLink>
  <linkText>Sandcastle Help File Builder
  <linkAlternateText>Go to GitHub</linkAl]
  <linkUri>https://GitHub.com/EWSoftware/
</externalLink>
```

```
<codeEntityReference>T:TestDoc.TestClass</c
<codeEntityReference>P:TestDoc.TestClass.Sc
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.Tc
<codeEntityReference>M:TestDoc.TestClass.Fi
<codeEntityReference>M:TestDoc.TestClass.Se
-->
</relatedTopics>
</developerSampleDocument>
</topic>
```



▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



SDK Technology Architecture

This document type is used for technology architecture topics.

SDK Technology Architecture File Format

Below is an example of an SDK Technology Architecture topic file.

SDK Technology Architecture Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerSDKTechnologyOverviewArchitectureDocu
    xmlns="http://ddue.schemas.microsoft.com/aut
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <!-- Uncomment this to generate an outline
      titles. Specify a numeric value as th
      a specific number of sub-topics when c
      zero (0) to limit it to top-level sect
    <!-- <autoOutline /> -->

    <para>Required introduction</para>
  </introduction>

  <!-- Add one or more top-level section elemer
```

```
    If using <autoOutline />, add an address
    and specify a title so that it can be ju
<section address="Section1">
  <title>Optional section title</title>
  <content>
    <!-- Uncomment this to create a sub-secti
    <autoOutline /> -->
    <para>Add one or more sections with conte
  </content>
</section>
```

```
<relatedTopics>
```

```
  <!-- One or more of the following:
    - A local link
    - An external link
    - A code entity reference
```

```
<link xlink:href="Other Topic's ID">Link to
<externalLink>
  <linkText>Link text</linkText>
  <linkAlternateText>Optional alternate l
  <linkUri>URI</linkUri>
</externalLink>
<codeEntityReference>API member ID</codeEnt
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5
```

```
<externalLink>
  <linkText>Sandcastle Help File Builder
  <linkAlternateText>Go to GitHub</linkAl
  <linkUri>https://GitHub.com/EWSoftware/
</externalLink>
```

```
<codeEntityReference>T:TestDoc.TestClass</c
<codeEntityReference>P:TestDoc.TestClass.Sc
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.#c
```

```
<codeEntityReference>M:TestDoc.TestClass.Tc
<codeEntityReference>M:TestDoc.TestClass.Fi
<codeEntityReference>M:TestDoc.TestClass.Se
-->
</relatedTopics>
</developerSDKTechnologyOverviewArchitectureDoc
</topic>
```



▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



SDK Technology Code Directory

This document type is used for UI reference topics.

▲ SDK Technology Code Directory File Format

Below is an example of an SDK Technology Code Directory topic file.

SDK Technology Code Directory Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerSDKTechnologyOverviewCodeDirectoryDoc
    xmlns="http://ddue.schemas.microsoft.com/autl
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <para>Required introduction</para>
  </introduction>

  <codeList>
    <content>
      <para>Code list content</para>
    </content>
  </codeList>
```

```
<relatedTopics>
```

```
<!-- One or more of the following:  
- A local link  
- An external link  
- A code entity reference
```

```
<link xlink:href="Other Topic's ID">Link to  
<externalLink>  
  <linkText>Link text</linkText>  
  <linkAlternateText>Optional alternate ]  
  <linkUri>URI</linkUri>  
</externalLink>  
<codeEntityReference>API member ID</codeEnt
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5  
  
<externalLink>  
  <linkText>Sandcastle Help File Builder  
  <linkAlternateText>Go to GitHub</linkA]  
  <linkUri>https://GitHub.com/EWSoftware/  
</externalLink>
```

```
<codeEntityReference>T:TestDoc.TestClass</c  
<codeEntityReference>P:TestDoc.TestClass.Sc  
<codeEntityReference>M:TestDoc.TestClass.#c  
<codeEntityReference>M:TestDoc.TestClass.#c  
<codeEntityReference>M:TestDoc.TestClass.Tc  
<codeEntityReference>M:TestDoc.TestClass.Fi  
<codeEntityReference>M:TestDoc.TestClass.Se  
-->
```

```
</relatedTopics>
```

```
</developerSDKTechnologyOverviewCodeDirectoryDc
```

```
</topic>
```



▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



SDK Technology Orientation

This document type is used for technology orientation topics.

SDK Technology Orientation File Format

Below is an example of an SDK Technology Orientation topic file.

SDK Technology Orientation Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerSDKTechnologyOverviewOrientationDocun
    xmlns="http://ddue.schemas.microsoft.com/autf
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <para>Required introduction</para>
  </introduction>

  <inThisSection>
    <content>
      <para>"In This Section" info</para>
    </content>
  </inThisSection>

  <!-- <reference>Optional reference info</refe
```

```
<!-- <relatedSections>Optional related section
<!-- <externalResources>Optional external resources
</developerSDKTechnologyOverviewOrientationDocu
</topic>
```



▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



SDK Technology Scenarios

This document type is used for technology scenario topics.

SDK Technology Scenarios File Format

Below is an example of an SDK Technology Scenarios topic file.

SDK Technology Scenarios Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerSDKTechnologyOverviewScenariosDocumer
    xmlns="http://ddue.schemas.microsoft.com/aut
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <!-- Uncomment this to generate an outline
      titles. Specify a numeric value as th
      a specific number of sub-topics when c
      zero (0) to limit it to top-level sect
    <!-- <autoOutline /> -->

    <para>Required introduction</para>
  </introduction>

  <!-- Add one or more top-level section elemer
```

```
    If using <autoOutline />, add an address
    and specify a title so that it can be ju
<section address="Section1">
  <title>Optional section title</title>
  <content>
    <!-- Uncomment this to create a sub-secti
    <autoOutline /> -->
    <para>Add one or more sections with conte
  </content>
</section>
```

```
<relatedTopics>
```

```
  <!-- One or more of the following:
    - A local link
    - An external link
    - A code entity reference
```

```
<link xlink:href="Other Topic's ID">Link to
<externalLink>
  <linkText>Link text</linkText>
  <linkAlternateText>Optional alternate l
  <linkUri>URI</linkUri>
</externalLink>
<codeEntityReference>API member ID</codeEnt
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5
<externalLink>
  <linkText>Sandcastle Help File Builder
  <linkAlternateText>Go to GitHub</linkAl
  <linkUri>https://GitHub.com/EWSoftware/
</externalLink>
```

```
<codeEntityReference>T:TestDoc.TestClass</c
<codeEntityReference>P:TestDoc.TestClass.Sc
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.#c
```

```
<codeEntityReference>M:TestDoc.TestClass.Tc
<codeEntityReference>M:TestDoc.TestClass.Fi
<codeEntityReference>M:TestDoc.TestClass.Se
-->
</relatedTopics>
</developerSDKTechnologyOverviewScenariosDocume
</topic>
```



▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



SDK Technology Summary

This document type is used for technology summary topics.

SDK Technology Summary File Format

Below is an example of an SDK Technology Summary topic file.

SDK Technology Summary Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerSDKTechnologyOverviewTechnologySummar
    xmlns="http://ddue.schemas.microsoft.com/autf
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <para>Required introduction</para>
  </introduction>

  <keywords>
    <content>
      <para>Keyword content</para>
    </content>
  </keywords>

  <namespaces>
```

```

    <content>
      <para>Namespaces content</para>
    </content>
  </namespaces>

<!-- <relatedTechnologies>Optional related te

<background>
  <content>
    <para>Background content</para>
  </content>
</background>

<implementingTechnologyNameClasses>
  <title>"Implementing [Technology Name] Clas
  <content>
    <para>Implementation content</para>
  </content>
</implementingTechnologyNameClasses>

<technologyNameClassesAtAGlance>
  <title>"[Technology Name] Classes at a Glar
  <content>
    <para>At a Glance content</para>
  </content>
</technologyNameClassesAtAGlance>

<!-- <whatsNew>Optional "what's new" informat

<relatedTopics>
  <!-- One or more of the following:
    - A local link
    - An external link
    - A code entity reference

  <link xlink:href="Other Topic's ID">Link te
  <externalLink>
    <linkText>Link text</linkText>
    <linkAlternateText>Optional alternate ]

```

```
<linkUri>URI</linkUri>
</externalLink>
<codeEntityReference>API member ID</codeEnt
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5
```

```
<externalLink>
  <linkText>Sandcastle Help File Builder
  <linkAlternateText>Go to GitHub</linkAl
  <linkUri>https://GitHub.com/EWSoftware/
</externalLink>
```

```
<codeEntityReference>T:TestDoc.TestClass</c
<codeEntityReference>P:TestDoc.TestClass.Sc
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.Tc
<codeEntityReference>M:TestDoc.TestClass.Fi
<codeEntityReference>M:TestDoc.TestClass.Se
-->
```

```
</relatedTopics>
```

```
</developerSDKTechnologyOverviewTechnologySumma
</topic>
```



See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Troubleshooting

This document type is used for troubleshooting topics.

▴ Troubleshooting File Format

Below is an example of a Troubleshooting topic file.

Troubleshooting Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerTroubleshootingDocument
    xmlns="http://ddue.schemas.microsoft.com/autf
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <!-- Uncomment this to generate an outline
      titles. Specify a numeric value as th
      a specific number of sub-topics when c
      zero (0) to limit it to top-level sect
    <!-- <autoOutline /> -->

    <para>Required introduction</para>
  </introduction>

  <!-- Zero or more sections. These are collap
```

```

    <autoOutline />, add an address attribute
    a title so that it can be jumped to with
<section address="Section1">
  <title>Optional section title</title>
  <content>
    <!-- Uncomment this to create a sub-section
    <autoOutline /> -->
    <para>Section content</para>
    <!-- <procedure>Optional procedures. See
  </content>
</section>

```

```

<relatedTopics>

```

```

  <!-- One or more of the following:
    - A local link
    - An external link
    - A code entity reference

```

```

  <link xlink:href="Other Topic's ID">Link text
</externalLink>
  <linkText>Link text</linkText>
  <linkAlternateText>Optional alternate text
  <linkUri>URI</linkUri>
</externalLink>
<codeEntityReference>API member ID</codeEntityReference>

```

Examples:

```

<link xlink:href="00e97994-e9e6-46e0-b420-5
</externalLink>
  <linkText>Sandcastle Help File Builder
  <linkAlternateText>Go to GitHub</linkAlternateText>
  <linkUri>https://GitHub.com/EWSoftware/
</externalLink>

```

```

<codeEntityReference>T:TestDoc.TestClass</codeEntityReference>
<codeEntityReference>P:TestDoc.TestClass.Sc
<codeEntityReference>M:TestDoc.TestClass.#c

```

```
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.Tc
<codeEntityReference>M:TestDoc.TestClass.Fi
<codeEntityReference>M:TestDoc.TestClass.Se
-->
</relatedTopics>
</developerTroubleshootingDocument>
</topic>
```



▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



User Interface Reference

This document type is used for user interface reference topics.

▲ User Interface Reference File Format

Below is an example of a User Interface Reference topic file.

User Interface Reference Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerUIReferenceDocument
    xmlns="http://ddue.schemas.microsoft.com/autf
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <!-- Uncomment this to generate an outline
      titles. Specify a numeric value as th
      a specific number of sub-topics when c
      zero (0) to limit it to top-level sect
    <!-- <autoOutline /> -->

    <para>Required introduction</para>
  </introduction>

  <taskList>
```

```

    <content>
      <para>Task list content</para>
    </content>
  </taskList>

  <!-- Zero or more sections. These are collapsed by default. To collapse a section, add the attribute
    <autoOutline />, add an address attribute to the section, and
    a title so that it can be jumped to with the <link> element.
  <section address="Section1">
    <title>Optional section title</title>
    <content>
      <!-- Uncomment this to create a sub-section. This is collapsed by default.
        <autoOutline /> -->
      <para>Section content</para>
    </content>
  </section>

  <uiElementList>
    <content>
      <para>UI element list content</para>
    </content>
  </uiElementList>

  <relatedTopics>
    <!-- One or more of the following:
      - A local link
      - An external link
      - A code entity reference
    </-->

    <link xlink:href="Other Topic's ID">Link to other topic</link>
    <externalLink>
      <linkText>Link text</linkText>
      <linkAlternateText>Optional alternate text</linkAlternateText>
      <linkUri>URI</linkUri>
    </externalLink>
    <codeEntityReference>API member ID</codeEntityReference>
  </relatedTopics>

```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5

<externalLink>
  <linkText>Sandcastle Help File Builder
  <linkAlternateText>Go to GitHub</linkA]
  <linkUri>https://GitHub.com/EWSoftware/
</externalLink>

<codeEntityReference>T:TestDoc.TestClass</c
<codeEntityReference>P:TestDoc.TestClass.Sc
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.Tc
<codeEntityReference>M:TestDoc.TestClass.Fi
<codeEntityReference>M:TestDoc.TestClass.Se
-->
</relatedTopics>
</developerUIReferenceDocument>
</topic>
```



▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Walkthrough

This document type is used for walkthrough topics.

Walkthrough File Format

Below is an example of a Walkthrough topic file.

Walkthrough Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerWalkthroughDocument
    xmlns="http://ddue.schemas.microsoft.com/aut
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <!-- Uncomment this to generate an outline
      titles. Specify a numeric value as th
      a specific number of sub-topics when c
      zero (0) to limit it to top-level sect
    <!-- <autoOutline /> -->

    <para>Required introduction</para>
  </introduction>

  <!-- <prerequisites><content>Optional prerequ
```

```
<!-- One or more procedure or section with pr
<procedure>
  <title>Procedure title</title>
  <steps class="ordered">
    <step>
      <content>
        <para>First step</para>
      </content>
    </step>
    <step>
      <content>
        <para>Second step</para>
      </content>
    </step>
  </steps>
  <!-- <conclusion>Optional conclusion</concl
</procedure>
```

```
<!-- Optional additional procedure or section
<section address="Section1">
  <title>Optional section title</title>
  <content>
    <procedure>
      <title>Procedure #2</title>
      <steps class="ordered">
        <step>
          <content>
            <para>First step</para>
          </content>
        </step>
        <step>
          <content>
            <para>Second step</para>
          </content>
        </step>
      </steps>
      <!-- <conclusion>Optional conclusion</c
    </procedure>
```

```
</content>
</section>
```

```
<!-- Optional next steps info
<nextSteps>
  <content><para>Next steps info goes here</p>
</nextSteps>
-->
```

```
<relatedTopics>
```

```
<!-- One or more of the following:
  - A local link
  - An external link
  - A code entity reference
```

```
<link xlink:href="Other Topic's ID">Link to
<externalLink>
  <linkText>Link text</linkText>
  <linkAlternateText>Optional alternate ]
  <linkUri>URI</linkUri>
</externalLink>
<codeEntityReference>API member ID</codeEnt
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5
<externalLink>
  <linkText>Sandcastle Help File Builder
  <linkAlternateText>Go to GitHub</linkAl]
  <linkUri>https://GitHub.com/EWSoftware/
</externalLink>
```

```
<codeEntityReference>T:TestDoc.TestClass</c
<codeEntityReference>P:TestDoc.TestClass.Sc
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.#c
<codeEntityReference>M:TestDoc.TestClass.Tc
<codeEntityReference>M:TestDoc.TestClass.Fi
```

```
<codeEntityReference>M:TestDoc.TestClass.Sc  
-->  
</relatedTopics>  
</developerWalkthroughDocument>  
</topic>
```



▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Whitepaper

This document type is used for whitepapers.

Whitepaper File Format

Below is an example of a Whitepaper topic file.

Whitepaper Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerWhitePaperDocument
    xmlns="http://ddue.schemas.microsoft.com/autf
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <byline>
    <para>Byline</para>
  </byline>

  <date>
    <para>Date</para>
  </date>

  <summarySection>
    <content>
      <para>Summary content</para>
```

```

    </content>
</summarySection>

<!-- <downloadCenterLink>Optional info about
<!-- <appliesTo>Optional "applies to" info</a
<!-- <contents>Optional contents info</conter

<introduction>
    <!-- Uncomment this to generate an outline
         titles. Specify a numeric value as th
         a specific number of sub-topics when c
         zero (0) to limit it to top-level sect
    <!-- <autoOutline /> -->

    <para>Required introduction</para>
</introduction>

<!-- Add one or more top-level section elemer
         If using <autoOutline />, add an address
         and specify a title so that it can be ju
<section address="Section1">
    <title>Optional section title</title>
    <content>
        <!-- Uncomment this to create a sub-secti
             <autoOutline /> -->
        <para>Add one or more sections with conte
    </content>
</section>

<!-- <codeExample>Optional code example.</coc

<conclusion>
    <content>
        <para>Conclusion content</para>
    </content>
</conclusion>

```

```
</relatedTopics>
```

```
<!-- One or more of the following:  
- A local link  
- An external link  
- A code entity reference
```

```
<link xlink:href="Other Topic's ID">Link text  
<externalLink>  
  <linkText>Link text</linkText>  
  <linkAlternateText>Optional alternate text</linkAlternateText>  
  <linkUri>URI</linkUri>  
</externalLink>  
<codeEntityReference>API member ID</codeEntityReference>
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5f481b121381">Link text  
<externalLink>  
  <linkText>Sandcastle Help File Builder  
  <linkAlternateText>Go to GitHub</linkAlternateText>  
  <linkUri>https://GitHub.com/EWSoftware/SandcastleHelpFileBuilder</linkUri>  
</externalLink>
```

```
<codeEntityReference>T:TestDoc.TestClass</codeEntityReference>  
<codeEntityReference>P:TestDoc.TestClass.Scenarios</codeEntityReference>  
<codeEntityReference>M:TestDoc.TestClass.#ctor</codeEntityReference>  
<codeEntityReference>M:TestDoc.TestClass.#ctor</codeEntityReference>  
<codeEntityReference>M:TestDoc.TestClass.TestMethod</codeEntityReference>  
<codeEntityReference>M:TestDoc.TestClass.Fixture</codeEntityReference>  
<codeEntityReference>M:TestDoc.TestClass.Setup</codeEntityReference>  
-->
```

```
</relatedTopics>
```

```
<!-- <additionalInformation>Optional additional information</additionalInformation>
```

```
<!-- <authorBio>Optional author bio</authorBio>
```

```
</developerWhitePaperDocument>
```

`</topic>`

▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



XML Reference

This document type is used for XML reference topics.

XML Reference File Format

Below is an example of an XML Reference topic file.

XML Reference Topic

[Copy](#)

```
<?xml version="1.0" encoding="utf-8"?>
<topic id="00000000-0000-0000-0000-000000000000"
  <developerXmlReference
    xmlns="http://ddue.schemas.microsoft.com/autf
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <!--
  <summary>
    <para>Optional summary abstract</para>
  </summary>
  -->

  <introduction>
    <para>Required introduction</para>
  </introduction>

  <!-- Optional schema hierarchy
  <schemaHierarchy>
    <link xlink:href="Other Topic's ID">Link 1<
    <link xlink:href="Other Topic's ID">Link 2<
  </schemaHierarchy> -->

  <syntaxSection>
```

```
<sections>
  <section>
    <title>Syntax section title</title>
    <content>
      <para>Syntax content</para>
    </content>
  </section>
</sections>
<legacySyntax>
  <parameterReference>Parameter reference</
</legacySyntax>
</syntaxSection>

<attributesandElements>
  <para>Attributes and elements syntax intro</
<attributes>
  <para>Attributes syntax</para>
</attributes>
<attribute>
  <title>Syntax title</title>
  <table>
    <tableHeader>
      <row>
        <entry>
          <para>Attribute header</para>
        </entry>
      </row>
    </tableHeader>
    <row>
      <entry>
        <para>Attribute value</para>
      </entry>
    </row>
  </table>
</attribute>
<childElement>
  <para>Child element syntax</para>
</childElement>
<parentElement>
```

```
<para>Parent element syntax</para>
</parentElement>
</attributesandElements>
```

```
<!-- <codeExample>Optional code example.</codeExample>
```

```
<!-- <elementInformation>Optional element information</elementInformation>
```

```
<relatedTopics>
```

```
<!-- One or more of the following:
```

- A local link
- An external link
- A code entity reference

```
<link xlink:href="Other Topic's ID">Link text</link>
<externalLink>
```

```
<linkText>Link text</linkText>
```

```
<linkAlternateText>Optional alternate text</linkAlternateText>
```

```
<linkUri>URI</linkUri>
```

```
</externalLink>
```

```
<codeEntityReference>API member ID</codeEntityReference>
```

Examples:

```
<link xlink:href="00e97994-e9e6-46e0-b420-5" data-bbox="252 589 866 609"/>
```

```
<externalLink>
```

```
<linkText>Sandcastle Help File Builder
```

```
<linkAlternateText>Go to GitHub</linkAlternateText>
```

```
<linkUri>https://GitHub.com/EWSoftware/
```

```
</externalLink>
```

```
<codeEntityReference>T:TestDoc.TestClass</codeEntityReference>
```

```
<codeEntityReference>P:TestDoc.TestClass.Sc
```

```
<codeEntityReference>M:TestDoc.TestClass.#c
```

```
<codeEntityReference>M:TestDoc.TestClass.#c
```

```
<codeEntityReference>M:TestDoc.TestClass.Tc
```

```
<codeEntityReference>M:TestDoc.TestClass.Fi
```

```
<codeEntityReference>M:TestDoc.TestClass.Se
```

```
-->
</relatedTopics>
</developerXmlReference>
</topic>
```

▲ See Also

Other Resources

[Topic File](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Companion File

A companion file is used during the build process to obtain the topic title and link text for a particular conceptual topic. There will be one companion file for each conceptual content topic.

Note

If using a build tool such as the [Sandcastle Help File Builder](#), the companion files are created for you automatically as part of the build process and you do not need to maintain them yourself.

Companion File Format

Below is an example of a companion file.

```
Example Companion File Copy  
<?xml version="1.0" encoding="utf-8"?>  
<metadata>  
  <topic id="303c996a-2911-4c08-b492-6496c82b3edk"  
    <title>Welcome to the Sandcastle MAML Guide</  
    <linkText>welcome page</linkText>  
  </topic>  
</metadata>
```

The file consists of a root `metadata` node that contains one `topic` node with an `id` attribute that refers to the topic's GUID.

The `title` element is required and contains the title to use for the topic. This will be used as the default link text. In cases where the title is not appropriate or is too long for link text and you use the topic in a link frequently, you can specify a `linkText` element that contains different text to use for the link. Note that if a `link` element has inner text specified in the topic, it will always be used instead of the values from the companion file.

 **Note**

The ID value must be in the form of a Globally Unique Identifier (GUID).

▲ See Also

Other Resources

[Conceptual Content File Types](#)
link

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Block Elements

The topics in this section define the block elements that can be used in conceptual topics.

▲ Block Elements

Select one of the topics below to learn more about a particular block element.

- [alert](#)
- [code](#)
- [codeReference](#)
- [definitionTable](#), [definedTerm](#), [definition](#)
- [introduction](#)
- [list](#), [listItem](#)
- [para](#)
- [quote](#)
- [relatedTopics](#)
- [table](#), [tableHeader](#), [row](#), [entry](#)
- [title](#)

[v1.1.0.3] Sandcastle MAML Guide

Send comments on this topic to [Eric Woodruff](#)



alert

This element can be used to create a note-like section within a conceptual topic to draw attention to some important information.

Usage

Enter the element in the topic and specify the type of alert using the `class` attribute. The content of the alert element is usually one or more paragraphs of text describing the important information.

Note

The `class` attribute value is case-sensitive. Unrecognized values will default to the "note" alert style.

The `class` attribute can be any of the following values which fall into four different categories that use different icons to the left of the alert title. The title will generally be the alert class name or an expanded form of it.

| Category | Class Value |
|---------------|---|
| General note | note, tip, implement, caller, inherit |
| Cautionary | caution, warning, important |
| Security | security, security note |
| Language note | cs, CSharp, c#, C#, visual c# note, cpp, CPP, c++, C++, visual c++ note, vb, VB, VisualBasic, visual basic note, JSharp, j#, J#, visual j# note |

Example

```
<alert class="note">
  <para>This is a note</para>
</alert>

<alert class="caution">
  <para>The class attribute value is case-sensiti
</alert>

<alert class="security">
  <para>Do not tape your password to the undersic
</alert>

<alert class="cs">
  <para>This could contain C#-specific usage note
</alert>
```



Example Output

Note

This is a note

Caution

The class attribute value is case-sensitive.

Security Note

Do not tape your password to the underside of your keyboard.

C# Note

This could contain C#-specific usage notes.

See Also

Other Resources

Block Elements

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



code

This element can be used to display a multi-line code fragment in a conceptual topic. For single lines or keywords, you can use the `codeInline` element.

- [Usage](#)
- [Literal XML And Other Special Characters](#)
- [Attributes](#)
 - [language](#)
 - [title](#)
 - [source, region, and removeRegionMarkers](#)
 - [tabSize](#)
 - [numberLines](#)
 - [outlining](#)
 - [keepSeeTags](#)
- [Nested Code Elements](#)
- [See Also](#)

Usage

Enter the element in the topic and specify a language using the `language` attribute.

[Copy](#)

```
<code language="languageId"  
  [title="Optional title"]  
  [source="externalCodeFile" [region="importRegio  
  [tabsize="###"]  
  [numberLines="true | false"]  
  [outlining="true | false"]  
  [keepSeeTags="true | false"]>
```

```
///  
/// Code to display
```

```
</code>
```

Note

The `language` and `title` attributes are supported by Sandcastle. All other attributes are implemented by the **Code Block Component** supplied with the [Sandcastle Help File Builder](#). Refer to its help file for more information about the build component.

Literal XML And Other Special Characters

If you want to include XML or other unencoded information you can do one of the following:

- Utilize the `source` attribute to read it in from an external file instead.
- Use a `CDATA` section to encapsulate the code so that special characters and formatting are preserved. For example:

CDATA Example

[Copy](#)

```
/// <example>
/// <code language="xml" title="Example Config" />
/// <![CDATA[
/// <?xml version="1.0" encoding="utf-8"?>
/// <config>
///   <path>C:\Test\</path>
///   <filespec mask="*.txt" />
/// </config>]]>
/// </code>
/// </example>
```

Be sure to start the example code on a new line after the opening `<![CDATA[` tag as shown so that the leading whitespace can be normalized correctly.

Note

The inclusion of a `CDATA` tag anywhere in the code comments can cause Visual Studio 2005 to lock up when you type an angle bracket within a comment block. If you are affected by this issue, there is a hot fix <https://connect.microsoft.com/VisualStudio/Downloads/DownloadID=9646>.

Attributes

The following attributes can be used to control how the code block is handled.

language

This attribute allows you to define the code language. Although optional, it is recommended that you always specify a language attribute if using Sandcastle alone. Many of the presentation styles rely on the language attribute to classify the code block and handle it in some way such as attaching it to a language filter or grouping it with like code blocks. The Sandcastle Help File Builder's code block component will apply a default language if one is not specified. The possible language values are as follows:

| Language ID (case-insensitive) | Language Syntax Used |
|--|----------------------|
| cs, C#, CSharp | C# |
| cpp, cpp#, C++, CPlusPlus | C++ |
| c | C |
| fs, f#, FSharp, fscript | F# |
| EcmaScript, js, JavaScript | JavaScript |
| jscript, jscript#, jscriptnet, JScript.NET | JScript.NET |
| VB, VB#, vbnet, VB.NET | VB/VB.NET |
| vbs, vbscript | VBScript |

| | |
|-----------------------------|-----------------------------------|
| htm, html, xml, xsl | XML |
| XAML | XAML |
| jsharp, J# | J# |
| sql, sql server, sqlserver | SQL script |
| pshell, powershell, ps1 | PowerShell script |
| py, python | Python |
| Anything else (i.e. "none") | No language, no special handling. |

Important

The full name "**language**" should be used for the attribute name rather than the abbreviated form "**lang**". The reason is that the XSL transformations used by the presentation styles expect the full name when they search for the language attribute. The abbreviated form is a legacy syntax that should be avoided. The code block component will auto-correct the attribute name so that it is not missed by the transformations.

It is recommended that you always specify a language attribute.

title

This attribute allows you to add a title that appears before the code block. An example of its use would be to label the example with a description. If omitted, the language name will appear for the title. If you do not want a title on a particular block, set the **title** attribute to a single space (" ").

source, region, and removeRegionMarkers

These three attributes are extensions implemented by the code block build component. The **source** attribute is used to specify that the code block's content should be read from an external source code file. If used alone, the entire file is imported. The optional **region** attribute can be used to limit the code to a specific section of the file delimited with the named **#region** (**#pragma region** for C++). The **#region** and **#endregion** tags are excluded from the

extracted section of code.

This is not to be confused with the XML comments `include` element. This extension is intended to extract code from actual source files. This allows you to manage your code examples in buildable projects to test them for correctness as a project is developed and altered in the future. It also saves you from managing the code in the XML comments and does not require that the code be HTML encoded as it is when written in the comments. The code will be HTML encoded when it is read in for processing. When used, it is assumed that there is no code within the code element itself and thus it will always be self-closing. Here are some examples:

Retrieve all code from an external file and use the VB.NET syntax to color it. The path is relative to the code block component's `basePath` configuration element. Typically, the base path is the path of the documentation project file. So, if your documentation project is in a subfolder and the code is in the parent folder in an *Examples* subfolder, the path would look like the following example.

[Copy](#)

```
<code source="..\Examples\WholeDemo.vb" language=
```

Retrieve a specific `#region` from an external file.

[Copy](#)

```
<code source="..\Examples\SeveralExamples.vb"
  region="Example 1" language="vbnet" title="Exan
```

Note that VB.NET does not allow `#Region` and `#End Region` within a method body. Other file types such as XML, XAML, and SQL script do not understand `#region` statements. However, if you want to extract a region from a VB.NET method body or a section of another file type, you can add the region statements in comments to workaround the limitation. The component will still find it and extract the region.

Commented Region Examples [Copy](#)

```
-----
VB.NET Example
-----
Public Sub SomeMethod()
```

```
' #Region "VB.NET Snippet"
Dim x As Integer

For x = 1 To 10
    Console.WriteLine(x)
Next x
' #End Region
End Sub
```

XAML Example

```
<Style x:Key="SpecialButton" TargetType="{x:Type
    <Style.Triggers>
        <!-- #region XAML Snippet -->
        <Trigger Property="Button.IsMouseOver" Value=
            <Setter Property = "Background" Value="Red"
        </Trigger>
        <!-- #endregion -->
    </Style.Triggers>
</Style>
```

SQL Example

```
-- #region SQL snippet
Select *
From tblName
Where LastName = 'SMITH'
-- #endregion

/* #region Another snippet */
Select *
From tblName
Where FirstName = 'JOHN'
/* #endregion */
```



The `removeRegionMarkers` attribute can be used to specify

whether or not region markers within the imported code file or region are removed. If not specified or set to false, any nested region markers are left in the imported code block. If specified and set to true, nested region markers are removed from the imported code. This is useful for removing nested region markers from larger code samples that contain smaller snippets of code used in other examples. The default setting (false) can be overridden via the component configuration.

tabSize

When the code blocks are formatted by the build component, tab characters are replaced with a set number of spaces to preserve formatting. This attribute can be used to override the default setting for a language which is specified in the syntax file. For example, if the default tab size for a language is four, adding `tabSize="8"` will force it to use eight spaces instead. If set to zero, the syntax file setting is used. This attribute sets the default tab size for unknown languages when used in the component's configuration.

numberLines

This attribute allows you to override the default setting in the component's configuration. For example, if the default setting is false to turn off line numbering, you can add `numberLines="true"` to enable numbering on a specific code example.

outlining

This attribute allows you to override the default setting in the component's configuration. For example, if the default setting is false to not add collapsible regions, you can add `outlining="true"` to enable collapsible regions on a specific code example. Note that if a code block contains no `#region` or `#if` blocks, outlining is automatically disabled and it will not reserve space in the margin for the markers.

keepSeeTags

When set to true, this attribute allows you to tell the build component's code colorizer to preserve `see` tags within the code so that they can be rendered as clickable links to the related topic. If set to false, the default, any `see` tags within the code will be colored and passed through as literal text. When using this option, you may find that you need to specify inner text for the `see` tag so that the link text appears as you want it. If the self-closing version of the tag is used, Sandcastle will generally set the link text to the name of the item plus any parameters if it is a generic type or takes parameters which may not be appropriate within a code sample.

▸ Nested Code Elements

You can import multiple code snippets into one common colored code block by nesting `code` elements within a parent `code` element. On nested `code` elements, only the `source` and `region` attributes will be utilized. All other attributes that control colorization of the merged code block will be retrieved from the parent `code` element.

```
Nested Code Elements Example Copy  
<code title="Nested code elements example" language="vb">  
<code source="..\Class1.vb" region="Snippet #1" />  
<code source="..\Class1.vb" region="Snippet #2" />  
</code>
```

Literal code can also be mixed in between the nested `code` elements. However, there are some limitations. All elements and literal code should be flush left within the parent `code` element or you will end up with additional spaces before the first line of each nested `code` element. There may still be additional spaces before the literal code. This is caused by the indentation included when the XML comments file is created by the compiler. If this is a problem, move the example to an include file and remove the excess leading whitespace. Use an `include` element to bring it into the XML comments. When done this way, the compiler preserves the spacing when writing the example out to the XML comments file.

```
Nested Code Elements Example Copy  
<code title="Nested code elements example" language="vb">  
<code source="..\Class1.vb" region="Snippet #1" />  
  
' ... Some stuff happens here ...  
  
<code source="..\Class1.vb" region="Snippet #2" />  
</code>
```

▸ See Also

Other Resources

[Block Elements](#)

[codeInline](#)

[codeEntityReference](#)

[codeReference](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



codeReference

A `codeReference` element is used to insert a commonly used code snippet into a conceptual topic. The code snippets are stored in an [external file](#).

Note

When using the [Sandcastle Help File Builder](#) or the standalone build components from it, the **Code Block Component** offers the same set of features and several others for both inline and external code snippets. In addition, it supports many other languages and can import code from working source code files which allows you to pull in examples from working projects that you can maintain and build without having to update a separate example snippets file. As such, you may prefer to use it rather than the code snippets file.

Usage

To use a code reference, simply insert its ID as the inner text of a `codeReference` element anywhere within a topic. You can combine multiple snippets with a common example ID into one code reference by specifying their sample IDs as a comma-separated list after the hash character.

Example

This example uses the samples from the example file above. The first three are the individual snippets. The last one combines the "create instance" snippets into one example.

[Copy](#)

```
Define a class  
<codeReference>ClassDefinition#Define</codeReference>
```

```
Create a local instance
```

`<codeReference>CreateInstance#Local</codeReference>`

Create a static (shared) instance

`<codeReference>CreateInstance#Static</codeReference>`

An example of combining snippets with a common example ID

`<codeReference>CreateInstance#Local,Static</codeReference>`



Example Output

Define a class

C# VB

Copy

```
public class CSharpClass()  
{  
    // Members go here  
}
```

Create a local instance

C# VB

Copy

```
CSharpClass x = new CSharpClass();
```

Create a static (shared) instance

C# VB

Copy

```
public static CSharpClass sharedInstance = new CSharpClass();
```



An example of combining snippets with a common example ID

C# VB

Copy

```
CSharpClass x = new CSharpClass();
```

...

```
public static CSharpClass sharedInstance = new CS
```

▲ See Also

Other Resources

[Block Elements](#)

[Code Snippets File](#)

[code](#)

[codeInline](#)

[codeEntityReference](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



definitionTable, definedTerm, definition

This element describes a list of terms and their definitions. Technically, the MAML schema defines the definition table as a two-column table without a header or footer. However, Sandcastle renders the element as an HTML definition list ([dl/dt/dd](#)).

Usage

Enter the element in the topic and specify one or more nested pairs of [definedTerm](#) and [definition](#) elements. The [definedTerm](#) element describes the term being defined. The [definition](#) element supplies the definition. Each [definition](#) element can contain other MAML elements, nested lists, tables, etc. Text should be contained in a [para](#) element.

Tip

The [definedTerm](#) element supports an [address](#) attribute that can be used to link to a specific defined term from another location.

Example

[Copy](#)

```
<para>This link takes you to <link xlink:href="#1
</link></para>

<definitionTable>
  <definedTerm>Term 1</definedTerm>
  <definition><para>Definition 1</para></definition>

  <definedTerm>Term 2</definedTerm>
  <definition><para>Definition 2</para></definition>

  <definedTerm>Term 3</definedTerm>
```

```
<definition><para>Definition 3</para></definiti

<definedTerm>Term 4</definedTerm>
<definition><para>Definition 4</para></definiti

<definedTerm>Term 5</definedTerm>
<definition><para>Definition 5</para></definiti

<definedTerm>Term 6</definedTerm>
<definition><para>Definition 6</para></definiti

<definedTerm>Term 7</definedTerm>
<definition><para>Definition 7</para></definiti

<definedTerm address="Term8">Term 8</definedTer
<definition><para>Definition 8. The entry's te
attribute that can be used as a link target.</par
</definitionTable>
```



Example Output

This link takes you to [Term #8](#).

Term 1
Definition 1

Term 2
Definition 2

Term 3
Definition 3

Term 4
Definition 4

Term 5
Definition 5

Term 6
Definition 6

Term 7

Definition 7

Term 8

Definition 8. The entry's term has an [address](#) attribute that can be used as a link target.

▲ See Also

Other Resources

[Block Elements](#)

[list](#), [listItem](#)

[table](#), [tableHeader](#), [row](#), [entry](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



introduction

This element contains an introduction to some part of the document. Most conceptual document types have a required introduction section.

Usage

Enter the element in the topic and specify the text to use as the document's introduction. An optional `address` attribute can be added to make the introduction accessible via an in-page link.

Example

[Copy](#)

```
<introduction address="#intro">
  <para>This element contains an introduction t
Most conceptual document types have a required ir
</introduction>

<section>
  <title>Example Output</title>
  <content>
    <para>See the <link xlink:href="#intro">Intro
section above.</para>
  </content>
</section>
```

Example Output

See the [Introduction](#) section above for an example of the output.

▲ See Also

Other Resources

[Block Elements](#)

[link](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



list, listItem

This element describes content that should be displayed as a list.

Usage

Enter the element in the topic and specify the type of `list` using the `class` attribute. The content of the list element is one or more `listItem` elements containing the content of the list items. The list items can themselves contain other MAML elements, nested lists, tables, etc. Text should be contained in a `para` element. The `class` attribute can be any of the following values:

- **bullet** - A standard bulleted list of items.
- **ordered** - A numbered list of items.
- **nobullet** - A list of items with no symbol preceding the items.

Note

The `class` attribute value is case-sensitive. Unrecognized values will default to the `nobullet` list style.

Tip

The `listItem` element supports an `address` attribute that can be used to link to a specific list item from another location.

Example

[Copy](#)

```
<para>This link takes you to <link xlink:href="#§>  
<list class="bullet">
```

```

<listItem><para>Item 1</para></listItem>
<listItem><para>Item 2</para></listItem>
<listItem><para>Item 3</para></listItem>
<listItem><para>Item 4 - This has an nested
<codeInline>ordered</codeInline> list.</para>

<list class="ordered">
  <listItem><para>Sub-Item 1</para></listItem>
  <listItem address="SubItem2"><para>Sub-Item 2
<codeInline>address</codeInline> attribute and ca
link target.</para></listItem>
  <listItem><para>Sub-Item 3</para></listItem>
  <listItem><para>Sub-Item 4</para></listItem>
  <listItem><para>Sub-Item 5 - This has a neste
<codeInline>nobullet</codeInline> list.</para>

  <list class="nobullet">
    <listItem><para>No-bullet item 1</para></li>
    <listItem><para>No-bullet item 2</para></li>
    <listItem><para>No-bullet item 3</para></li>
  </list>
</listItem>

  <listItem><para>Sub-Item 6</para></listItem>
</list>
</listItem>

<listItem><para>Item 5</para></listItem>
<listItem><para>Item 6</para></listItem>
<listItem><para>Item 7</para></listItem>
<listItem><para>Item 8</para></listItem>
<listItem><para>Item 9</para></listItem>
<listItem><para>Item 10</para></listItem>
</list>

```

▲ Example Output

This link takes you to [Sub-Item #2](#).

- Item 1
- Item 2
- Item 3
- Item 4 - This has an nested [ordered](#) list.
 1. Sub-Item 1
 2. Sub-Item 2 - This has an [address](#) attribute and can be used as a link target.
 3. Sub-Item 3
 4. Sub-Item 4
 5. Sub-Item 5 - This has a nested [nobullet](#) list.
 - No-bullet item 1
 - No-bullet item 2
 - No-bullet item 3
 6. Sub-Item 6
- Item 5
- Item 6
- Item 7
- Item 8
- Item 9
- Item 10

▲ See Also

Other Resources

Block Elements
definitionTable, definedTerm, definition
table, tableHeader, row, entry

[v1.1.0.3] Sandcastle MAML Guide

Send comments on this topic to [Eric Woodruff](#)



para

This element can be used to create paragraphs in conceptual content.

Usage

Enter the element in the topic and specify the paragraph text as the inner text. Making the element self-closing or empty will cause the element to be ignored unless it creates an initial paragraph break.

Example

[Copy](#)

```
<para>Paragraph 1.</para>  
<para>Paragraph 2.</para>  
<para/>  
<para></para>  
<para>Self-closing and empty paragraphs are ignor  
<para>Paragraph 4.</para>
```

Example Output

Paragraph 1.

Paragraph 2.

Self-closing and empty paragraphs are ignored.

Paragraph 4.

See Also

Other Resources

Block Elements

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



quote

This element describes a quotation.

Usage

Enter the element in the topic and specify the quotation as the inner text.

Example

[Copy](#)

```
<para>To quote the schema:</para>  
<quote>The quote element describes a quotation.</
```

Example Output

To quote the schema:

The quote element describes a quotation.

See Also

Other Resources

[Block Elements](#)

[quoteInline](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



relatedTopics

This element is used to define a list of links to other topics that may be of interest to the reader.

Usage

Enter the element in the topic and specify one or more local links, external links, or code entity references.

Note

In topics in which this element appears, it is a required element. However, if the topic has no links of interest, you can leave it empty and the related topics section will not be included in the topic.

Note

The links within the element can appear in any order. They will be grouped by type when the topic is built.

Controlling Link Grouping

When used in a `relatedTopics` element, an optional `topicType_id` attribute can be added to `link` elements that specifies a GUID value used to classify the link type so that it is grouped with other similar links. If not specified, the links will be placed in the "Other Resources" group. The GUID values correspond to the various MAML topic file types and are as follows:

| Topic Type | GUID Value | Group Section |
|------------|--------------------------------------|---------------|
| Conceptual | 1FE70836-AA7D-4515-B54B-E10C4B516E50 | Concepts |

| | | |
|--|--------------------------------------|-----------------|
| Code Entity | 4A273212-0AC8-4D72-8349-EC11CD2FF8CD | Other Resources |
| Error Message | A635375F-98C2-4241-94E7-E427B47C20B6 | Reference |
| Glossary | A689E19C-2687-4881-8CE1-652FF60CF46C | Other Resources |
| How-To | DAC3A6A0-C863-4E5B-8F65-79EFC6A4BA09 | Tasks |
| Orientation | B137C930-7BF7-48A2-A329-3ADCAEF8868E | Other Resources |
| Reference Without Syntax | F9205737-4DEC-4A58-AA69-0E621B1236BD | Reference |
| Reference With Syntax | 95DADC4C-A2A6-447A-AA36-B6BE3A4F8DEC | Reference |
| Sample | 069EFD88-412D-4E2F-8848-2D5C3AD56BDE | Tasks |
| SDK Technology Overview Architecture | 68F07632-C4C5-4645-8DFA-AC87DCB4BD54 | Concepts |
| SDK Technology Overview Code Directory | 4BBAAF90-0E5F-4C86-9D31-A5CAEE35A416 | Concepts |
| SDK Technology Overview Orientation | CDB8C120-888F-447B-8AF8-F9540562E7CA | Other Resources |
| SDK Technology Overview Scenarios | 356C57C4-384D-4AF2-A637-FDD6F088A033 | Concepts |
| SDK Technology Overview Technology Summary | 19F1BB0E-F32A-4D5F-80A9-211D92A8A715 | Concepts |

| | | |
|-----------------|--------------------------------------|-----------------|
| Troubleshooting | 38C8E0D1-D601-4DBA-AE1B-5BEC16CD9B01 | Tasks |
| UI Reference | B8ED9F21-39A4-4967-928D-160CD2ED9DCE | Reference |
| Walkthrough | 4779DD54-5D0C-4CC3-9DB3-BF1C90B721B3 | Tasks |
| Whitepaper | 56DB00EC-28BA-4C0D-8694-28E8B244E236 | Other Resources |
| XML Reference | 3272D745-2FFC-48C4-9E9D-CF2B2B784D5F | Reference |

Example

Copy

```
<relatedTopics>
  <link topicType_id="1FE70836-AA7D-4515-B54B-E16
    xlink:href="b32a73b8-fc26-4c98-912c-d595fc1a1
  <link topicType_id="1FE70836-AA7D-4515-B54B-E16
    xlink:href="bc490dbf-7d46-432d-a816-3ae16ab6e
  <link topicType_id="1FE70836-AA7D-4515-B54B-E16
    xlink:href="d0c9ce6d-115b-4c70-891c-69b4af944
  <link topicType_id="1FE70836-AA7D-4515-B54B-E16
    xlink:href="53a2b7c6-a269-4916-9313-bdc301ee9

  <externalLink>
    <linkText>Sandcastle Help File Builder</linkText>
    <linkUri>https://GitHub.com/EWSoftware/SHFB</linkUri>
    <linkTarget>_blank</linkTarget>
  </externalLink>

  <codeEntityReference qualifyHint="true">T:System
  <codeEntityReference qualifyHint="true" autoUpd
  M:System.IO.FileStream.#ctor(System.String, Syst
```

```
</codeEntityReference>
<codeEntityReference qualifyHint="true" autoUpd
M:System.IO.FileStream.Write(System.Byte[], Syst
</codeEntityReference>
<codeEntityReference qualifyHint="false">P:Syst
</codeEntityReference>
<codeEntityReference qualifyHint="false" autoUpd
M:System.IO.FileStream.Flush</codeEntityReferer
</relatedTopics>
```



▲ Example Output

See below.

▲ See Also

Reference

[System.IOFileStream](#)
[FileStreamFileStream](#)
[FileStreamWrite](#)
[Length](#)
[Flush](#)

Concepts

[Block Elements](#)
[codeEntityReference](#)
[externalLink](#)
[link](#)

Other Resources

[Sandcastle Help File Builder](#)

[v1.1.0.3] [Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



table, tableHeader, row, entry

This element is used to arrange data in a table format with rows and columns. An optional title and table header can also be included.

Usage

A table consists of the following elements:

table

This is the main element. All other table elements are nested within it and it has no attributes.

title

This is an optional element. If specified, it defines a title that will appear above the table.

tableHeader

This is an optional element. If specified, it should contain a single **row** element that defines the header row. This can be used to add titles to each column. The table header, if specified, will be styled differently in the rendered topic.

row

This element is used to define each row in the table. A table must contain at least one row. Each row must contain one or more **entry** elements that define the columns in the row.

entry

This element is used to define the columns in each row. The entry items can themselves contain other MAML elements, nested lists, tables, etc. Text should be contained in a **para** element.

Tip

The **entry** element supports an **address** attribute that can be used to link to a specific entry from another

location.

Tip

To render an empty cell, use a non-breaking space entity (` `).

Note

There are no provisions for spanning rows or columns as there is in HTML tables. As such, each row must contain the same number of columns. However, you can nest tables within an `entry` element which can be used to achieve a similar effect.

Example

[Copy](#)

```
<para>This link takes you to <link xlink:href="#"#F
in the first table.</para>
```

```
<table>
  <title>A Simple Table with Title and Headers</t
  <tableHeader>
    <row>
      <entry><para>Header 1</para></entry>
      <entry><para>Header 2</para></entry>
      <entry><para>Header 3</para></entry>
    </row>
  </tableHeader>
  <row>
    <entry><para>Row 1, Cell 1</para></entry>
    <entry><para>Row 1, Cell 2</para></entry>
    <entry><para>Row 1, Cell 3</para></entry>
  </row>
  <row>
    <entry><para>Row 2, Cell 1</para></entry>
    <entry><para>Row 2, Cell 2</para></entry>
    <entry><para>Row 2, Cell 3</para></entry>
  </row>
```

```
<row>
  <entry address="Row3Cell1"><para>This entry
attribute that can be used as a link target.</par
  <entry><para>Row 3, Cell 2</para></entry>
  <entry><para>Row 3, Cell 3</para></entry>
</row>
</table>
```

```
<table>
  <tableHeader>
    <row>
      <entry><para>&#160;</para></entry>
      <entry><para>A Nested Table Example</para></e
    </row>
  </tableHeader>
  <row>
    <entry><mediaLink><image
      xlink:href="98a8a8b7-c374-40c7-902a-91c947
    </mediaLink></entry>
    <entry>
      <table>
        <row>
          <entry><para>Cell 1</para></entry>
          <entry><para>Cell 2</para></entry>
          <entry><para>Cell 3</para></entry>
          <entry><para>Cell 4</para></entry>
        </row>
        <row>
          <entry><para>Cell 5</para></entry>
          <entry><para>Cell 6</para></entry>
          <entry><para>Cell 7</para></entry>
          <entry><para>Cell 8</para></entry>
        </row>
        <row>
          <entry><para>Cell 9</para></entry>
          <entry><para>Cell 10</para></entry>
          <entry><para>Cell 11</para></entry>
          <entry><para>Cell 12</para></entry>
        </row>
```

```

</table>

<para>The table above doesn't have a
<codeInline>tableHeader</codeInline>.</para>
</entry>
</row>
</table>

```



Example Output

This link takes you to [Row 3, Cell 1](#) in the first table.

A Simple Table with Title and Headers

| Header 1 | Header 2 | Header 3 |
|--|---------------|---------------|
| Row 1, Cell 1 | Row 1, Cell 2 | Row 1, Cell 3 |
| Row 2, Cell 1 | Row 2, Cell 2 | Row 2, Cell 3 |
| This entry has an address attribute that can be used as a link target. | Row 3, Cell 2 | Row 3, Cell 3 |

A Nested Table Example



| | | | |
|--------|---------|---------|---------|
| Cell 1 | Cell 2 | Cell 3 | Cell 4 |
| Cell 5 | Cell 6 | Cell 7 | Cell 8 |
| Cell 9 | Cell 10 | Cell 11 | Cell 12 |

The table above doesn't have a [tableHeader](#).

See Also

Other Resources

Block Elements
definitionTable, definedTerm, definition
list, listItem

[v1.1.0.3] Sandcastle MAML Guide

Send comments on this topic to [Eric Woodruff](#)



title

This element describes the name of a part of a document.

Usage

Enter the element in the topic and specify the text to use as the section's title.

Example

[Copy](#)

```
<section>
  <title>Usage</title>
  <content>
    <para>Enter the element in the topic and spec
section's title.</para>
  </content>
</section>
```

Example Output

See the **Usage** section above for an example of the output.

See Also

Other Resources

[Block Elements](#)

Send comments on this topic to [Eric Woodruff](#)



Common Inline Elements

The topics in this section define the common inline elements that can be used in conceptual topics.

▲ Common Inline Elements

Select one of the topics below to learn more about a particular inline element.

- [application](#)
- [codeInline](#)
- [command](#)
- [computerOutputInline](#)
- [database](#)
- [environmentVariable](#)
- [errorInline](#)
- [fictitiousUri](#)
- [foreignPhrase](#)
- [hardware](#)
- [legacyBold](#)
- [legacyItalic](#)
- [legacyUnderline](#)
- [literal](#)

- `localUri`
- `math`
- `newTerm`
- `phrase`
- `placeholder`
- `quoteInline`
- `replaceable`
- `subscript`
- `superscript`
- `system`
- `ui`
- `unmanagedCodeEntityReference`
- `userInput`

[v1.1.0.3] Sandcastle MAML Guide

Send comments on this topic to [Eric Woodruff](#)



application

This element is used to describe the name of an application such as Microsoft Windows or Notepad.

Usage

Enter the element in the topic and specify the application name as the inner text.

Example

[Copy](#)

```
Use the <application>Sandcastle Help File Builder
```

Example Output

Use the **Sandcastle Help File Builder** to create help files.

See Also

Other Resources

[Common Inline Elements](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



codeInline

This element describes a string of code or a single keyword. For multi-line code fragments, use the [code](#) element instead.

Usage

Enter the element in the topic and specify the code as the inner text.

Example

```
Use a <codeInline>foreach</codeInline> loop to it
```

[Copy](#)

Example Output

Use a [foreach](#) loop to iterate over the collection.

See Also

Other Resources

[Common Inline Elements](#)

[code](#)

[codeEntityReference](#)

[codeReference](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



command

This element describes the name of an executable or other software application that can be run.

Usage

Enter the element in the topic and specify the application name as the inner text. The [replaceable](#) element can be nested within it to represent replaceable parameters.

Example

```
<command>COPY <system>/s /e</system> &#160;<replac  
&#160;<replaceable>destFileSpec</replaceable></cc
```

[Copy](#)

Example Output

```
XCOPY /s /e sourceFileSpec destFileSpec
```

See Also

Other Resources

[Common Inline Elements](#)

[replaceable](#)

[system](#)

[v1.1.0.3] [Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



computerOutputInline

This element describes output from either a console prompt or from a UI form.

Usage

Enter the element in the topic and specify the output as the inner text.

Example

```
The result of running <application>HelloWorld.exe  
<computerOutputInline>Hello World!</computerOutput
```

[Copy](#)

Example Output

The result of running **HelloWorld.exe** is: `Hello World!`

See Also

Other Resources
[Common Inline Elements](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



database

This element is used to describe a database.

Usage

Enter the element in the topic and specify the database as the inner text.

Example

[Copy](#)

```
The information is stored in the <database>Conter
```

Example Output

The information is stored in the **Content** database.

See Also

Other Resources

[Common Inline Elements](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



environmentVariable

This element describes an environment variable in an operating system.

Usage

Enter the element in the topic and specify the environment variable as the inner text.

Example

[Copy](#)

```
The Sandcastle tools find their installed locatic  
<environmentVariable>SHFBR00T</environmentVariabl
```

Example Output

The Sandcastle tools find their installed location using the **SHFBR00T** environment variable.

See Also

Other Resources

[Common Inline Elements](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



errorInline

This element describes an error or error message.

Usage

Enter the element in the topic and specify the error as the inner text.

Example

[Copy](#)

```
If a file does not exist, a <errorInline>File Not  
error will occur.
```

Example Output

If a file does not exist, a *File Not Found* error will occur.

See Also

Other Resources

[Common Inline Elements](#)

[v1.1.0.3] [Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



fictitiousUri

This element describes a fictitious internet address such as a web or email address.

Usage

Enter the element in the topic and specify the fictitious URI as the inner text.

Example

[Copy](#)

```
The e-mail address <fictitiousUri>jdoe@contoso.cc  
fake as is the URL <fictitiousUri>http://www.cont
```

Example Output

The e-mail address *jdoe@contoso.com* is fake as is the URL
http://www.contoso.com.

See Also

Other Resources

[Common Inline Elements](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



foreignPhrase

This element is used to describe a phrase that is of a language that is different than the primary language of the document.

Usage

Enter the element in the topic and specify the phrase as the inner text.

Example

```
If it doesn't work out, <foreignPhrase>C'est la vie
```

[Copy](#)

Example Output

If it doesn't work out, *C'est la vie*.

See Also

Other Resources

[Common Inline Elements](#)

[phrase](#)

[quoteInline](#)



hardware

This element describes a physical part of a computer system.

Usage

Enter the element in the topic and specify the hardware as the inner text.

Example

[Copy](#)

```
Plug the <hardware>flash drive</hardware> into the  
<hardware>USB port</hardware>.
```

Example Output

Plug the **flash drive** into the **USB port**.

See Also

Other Resources

[Common Inline Elements](#)

[v1.1.0.3] [Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



legacyBold

This element marks bold text in legacy documents. The schema does not recommend using this element. Instead, use one of the other common inline elements that describe a specific type of item such as [application](#), [ui](#), etc. However, for general items, it is a simple way to bold text.

Usage

Enter the element in the topic and specify the text to bold as the inner text.

Example

```
This text will be rendered in <legacyBold>bold</>
```

Example Output

This text will be rendered in **bold**.

See Also

Other Resources
[Common Inline Elements](#)



legacyItalic

This element marks italicized text in legacy documents. The schema does not recommend using this element. Instead, use one of the other common inline elements that describe a specific type of item such as [application](#), [ui](#), etc. However, for general items, it is a simple way to italicize text.

Usage

Enter the element in the topic and specify the text to italicize as the inner text.

Example

```
This text will be rendered as <legacyItalic>itali
```

Example Output

This text will be rendered as *italicized*.

See Also

Other Resources
[Common Inline Elements](#)



legacyUnderline

This element marks underlined text in legacy documents. The schema does not recommend using this element. Instead, use one of the other common inline elements that describe a specific type of item such as [application](#), [ui](#), etc. However, for general items, it is a simple way to underline text.

Usage

Enter the element in the topic and specify the text to underline as the inner text.

Example

```
This text will be rendered with an <legacyUnderline>
```

Example Output

This text will be rendered with an underline.

See Also

Other Resources
[Common Inline Elements](#)



literal

This element describes a literal value.

Usage

Enter the element in the topic and specify the literal value as the inner text.

Example

[Copy](#)

```
If the property is not used, set it to the string
```



Example Output

If the property is not used, set it to the string **None**.

See Also

Other Resources

[Common Inline Elements](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



localUri

This element describes the path to a local resource such as a file.

▲ Usage

Enter the element in the topic and specify the path as the inner text.

▲ Example

[Copy](#)

```
Sandcastle is installed in the <localUri>C:\Progr  
folder by default.
```

▲ Example Output

Sandcastle is installed in the *C:\Program Files\Sandcastle* folder by default.

▲ See Also

Other Resources

[Common Inline Elements](#)

[v1.1.0.3] [Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



math

This element can be used to mark mathematical expressions.

Usage

Enter the element in the topic and specify the mathematical expression as the inner text.

Example

```
A simple math expression: <math>1 + 1 = 2</math>.
```

Example Output

A simple math expression: $1 + 1 = 2$.

See Also

Other Resources

[Common Inline Elements](#)

[subscript](#)

[superscript](#)



newTerm

This element is used to introduce new terms or technologies.

Usage

Enter the element in the topic and specify the new term as the inner text.

Example

[Copy](#)

```
Topics to cover include <newTerm>LINQ</newTerm>,
<newTerm>WCF</newTerm>.
```

Example Output

Topics to cover include *LINQ*, *WPF*, and *WCF*.

See Also

Other Resources

[Common Inline Elements](#)

[v1.1.0.3] [Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



phrase

This element is a wrapper element for a run of text. It is solely a container and does not specify any semantics on its own.

Usage

Enter the element in the topic and specify the phrase as the inner text.

Example

```
If it doesn't work out, <phrase>that's life</phrase>
```

[Copy](#)

Example Output

If it doesn't work out, *that's life*.

See Also

Other Resources

[Common Inline Elements](#)

[foreignPhrase](#)

[quoteInline](#)



placeholder

This element is used as a placeholder. By default, it renders text in a style determined by the presentation style. It may have specific use in custom build components.

Usage

Enter the element in the topic. It's inner text and the end result is implementation dependent.

Example

```
Something useful could go <placeholder>here</plac  
build component made use of it.
```

[Copy](#)

Example Output

Something useful could go *here* if a build component made use of it.

See Also

Other Resources

[Common Inline Elements](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



quoteInline

This element is used to describe an inline quotation.

Usage

Enter the element in the topic and specify the quotation as the inner text.

Example

[Copy](#)

```
This is an example of an <quoteInline>inline quot
```

Example Output

This is an example of an *inline quotation*.

See Also

Other Resources

[Common Inline Elements](#)

[quote](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



replaceable

This element describes a part of a command or a function that must be replaced by the user.

Usage

Enter the element in the topic and specify the replaceable item as the inner text.

Example

[Copy](#)

```
<command>COPY <system>/s /e</system>#160;<replac  
&#160;<replaceable>destFileSpec</replaceable></cc
```

Example Output

```
XCOPY /s /e sourceFileSpec destFileSpec
```

See Also

Other Resources

[Common Inline Elements](#)

[command](#)

[v1.1.0.3] [Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



subscript

This element can be used to mark a subscript such as those found in mathematical expressions.

Usage

Enter the element in the topic and specify the subscript text as the inner text.

Example

```
The chemical formula for water is <math>H<subscri
```

[Copy](#)

Example Output

The chemical formula for water is H_2O .

See Also

Other Resources

- [Common Inline Elements](#)
- [math](#)



superscript

This element can be used to mark a superscript such as those found in mathematical expressions.

Usage

Enter the element in the topic and specify the superscript text as the inner text.

Example

```
Can you explain <math>E = MC<superscript>2</superscript>
```

[Copy](#)

Example Output

Can you explain $E = MC^2$?

See Also

Other Resources

- [Common Inline Elements](#)
- [math](#)



system

This element describes system commands, command line switches, etc.

Usage

Enter the element in the topic and specify the command or switches as the inner text.

Example

[Copy](#)

```
<command>COPY <system>/s /e</system>&#160;<replac  
&#160;<replaceable>destFileSpec</replaceable></cc
```

Example Output

```
XCOPY /s /e sourceFileSpec destFileSpec
```

See Also

Other Resources

[Common Inline Elements](#)

[command](#)

[replaceable](#)

[v1.1.0.3] [Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



ui

This element describes a user interface element.

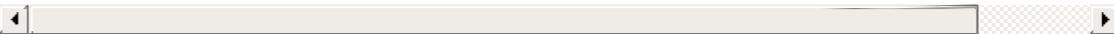
Usage

Enter the element in the topic and specify the user interface item as the inner text.

Example

[Copy](#)

Click the `<ui>OK</ui>` button to close the dialog



Example Output

Click the **OK** button to close the dialog box.

See Also

Other Resources

[Common Inline Elements](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



unmanagedCodeEntityRefere

This element is used to refer to unmanaged code entity references. Unlike managed code entity references, this element does not result in a link to the referenced item.

Usage

Enter the element in the topic and specify the entity as the inner text.

Example

[Copy](#)

```
You can use PInvoke to call  
<unmanagedCodeEntityReference>SetWindowPlacement<
```

Example Output

You can use PInvoke to call **SetWindowPlacement**.

See Also

Other Resources

[Common Inline Elements](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



userInput

This element describes data that is entered by the user.

Usage

Enter the element in the topic and specify the input as the inner text.

Example

```
Enter <userInput>Yes</userInput> or <userInput>No
```



Example Output

Enter **Yes** or **No**.

See Also

Other Resources

[Common Inline Elements](#)

[v1.1.0.3] [Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Media Elements

The topics in this section define the media (image) elements that can be used in conceptual topics.

Media Elements

Select one of the topics below to learn more about a particular media element.

- [mediaLink](#)
- [mediaLinkInline](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



mediaLink

This element is used to insert an image within a conceptual topic. This is a block element and the image appears by itself out of line.

Usage

To use a media link, insert the element within the topic at the location that the image should appear.

The nested `image` element is used to define the image link. The `xlink:href` attribute should refer to the unique ID of an image in the project's media content file. An optional `placement` attribute can be used to specify the placement of the image (`near` (the default), `center`, or `far`).

The nested `caption` element is optional. If specified, a caption will appear with the image. An optional `placement` attribute can be used to specify the placement of the caption (`before` (the default) or `after`). In addition, an optional `lead` attribute can be specified to define lead-in text that will prefix the caption (i.e. "Figure 1").

Example

[Copy](#)

```
<!-- No caption, default image placement -->
<mediaLink>
<image xlink:href="6be7079d-a9d8-4189-9021-0f72d1
</mediaLink>

<!-- Caption before, centered image -->
<mediaLink>
<caption>Caption Before</caption>
<image placement="center" xlink:href="6be7079d-a9
</mediaLink>
```

```
<!-- Caption after with lead-in text, far image &  
<mediaLink>  
<caption placement="after" lead="Figure 1">Captio  
<image placement="far" xlink:href="6be7079d-a9d8-  
</mediaLink>
```



Example Output



Caption Before





Figure 1: Caption after with lead-in

▲ See Also

Other Resources

[Media Elements](#)

[Media Content File](#)

[mediaLinkInline](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



mediaLinkInline

This element is used to insert an image inline within the text of a conceptual topic.

Usage

To use an inline media link, insert the element within the text of the topic at the location that the image should appear. The nested `image` element is used to define the image link. The `xlink:href` attribute should refer to the unique ID of an image in the project's media content file.

Example

[Copy](#)

```
The information icon <mediaLinkInline>  
<image xlink:href="4cfacbe8-d100-471c-9d69-1dbcce  
appears next to an Information message.
```

```
<mediaLinkInline><image xlink:href="b5d069f7-5daf  
An image to the left of the text.
```

Example Output

The information icon  appears next to an Information message.

 An image to the left of the text.

See Also

Other Resources

[Media Elements](#)
[Media Content File](#)
[mediaLink](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Link Elements

The topics in this section define the link elements that can be used in conceptual topics.

▾ Link Elements

Select one of the topics below to learn more about a particular link element.

- [codeEntityReference](#)
- [externalLink](#)
- [link](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



codeEntityReference

This is used to insert a reference to a code entity such as a type, method, property, event, field, etc. It can be a reference to a member in one of your classes or one in the .NET Framework.

Usage

To use a code entity reference, simply insert the fully qualified member name as the inner text of a `codeEntityReference` element anywhere within a topic. For methods, include the data types of any parameters that are passed to the method. The member name should be prefixed with a code indicating the member type:

| Prefix | Description |
|--------|---|
| N: | A namespace |
| T: | A types such as a class, enumeration, interface, or structure |
| F: | A field member |
| E: | An event member |
| P: | A property |
| M: | A method |

There are three optional attributes.

| Name | Description |
|-------------|---|
| qualifyHint | If set to false or omitted, the link text will consist only of the member name without a namespace or class qualifier. If set to true, type names are |

fully qualified with their namespace. Class members are qualified with their class name.

autoUpgrade This only has purpose on method class members. If set to false or omitted, the link will take you directly to the referenced member. If set to true and the method has overloads, the link will take you to the method overloads page instead. If the method has no overloads, it works the same as if set to false and it takes you to the method page.

linkText This can be used to override the default link text based on the member name and provide your own link text.

▲ Example

[Copy](#)

```
<codeEntityReference qualifyHint="true">T:System.  
<codeEntityReference qualifyHint="true" autoUpgra  
M:System.IO.FileStream.#ctor(System.String, System  
</codeEntityReference>  
<codeEntityReference qualifyHint="true" autoUpgra  
M:System.IO.FileStream.Write(System.Byte[], System  
</codeEntityReference>  
<codeEntityReference qualifyHint="false">P:System  
</codeEntityReference>  
<codeEntityReference qualifyHint="false" autoUpgr  
M:System.IO.FileStream.Flush</codeEntityReference  
<codeEntityReference linkText="FileStream.Flush c  
M:System.IO.FileStream.Flush</codeEntityReference
```



▲ Example Output

[System.IOFileStream](#)

[FileStreamFileStream](#)

[FileStreamWrite](#)

[Length](#)

[Flush](#)

[FileStream.Flush on MSDN](#)

▲ See Also

Other Resources

[Link Elements](#)

[code](#)

[codeInline](#)

[codeReference](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



externalLink

This is used to insert a link to an external URL of some sort such as a website.

Usage

An external link consists of the following nested elements.

| Name | Description |
|-------------------|--|
| linkText | This is a required element and is used to specify the text to use for the inner text of the link. |
| linkAlternateText | This is an optional element. It is used to specify alternate text for the link that will be displayed when the mouse hovers over it. |
| linkUri | This is a required element and is used to specify the URL to which the link refers. This can be any sort of link (HTTP, FTP, mail, etc). |
| linkTarget | This is an optional element. It is used to specify the target window to use when the link is clicked. If omitted _blank is used to open the external link in a new window. The value _self can be used to open the link in the same window as the topic thus replacing it. |

Note

If the URL in the `linkUri` element contains XML entities such as the ampersand (&), you must encode them or a build error will occur. See the example below.

Example

Copy

```
Only link text and URI: <externalLink>
  <linkText>Sandcastle Help File Builder</linkText>
  <linkUri>https://GitHub.com/EWSoftware/SHFB</linkUri>
</externalLink>
```

```
Added alternate text: <externalLink>
  <linkText>Sandcastle Help File Builder</linkText>
  <linkAlternateText>Visit SHFB on GitHub</linkAlternateText>
  <linkUri>https://GitHub.com/EWSoftware/SHFB</linkUri>
</externalLink>
```

```
Added link target: <externalLink>
  <linkText>Sandcastle Help File Builder</linkText>
  <linkAlternateText>Visit SHFB on GitHub</linkAlternateText>
  <linkUri>https://GitHub.com/EWSoftware/SHFB</linkUri>
  <linkTarget>_self</linkTarget>
</externalLink>
```

```
If the link contains XML entities, they must be encoded:
<externalLink>
  <linkText>MSDN Documentation Forum</linkText>
  <linkUri>http://forums.microsoft.com/MSDN/ShowThread.aspx?
</externalLink>
```



Example Output

Only link text and URI: [Sandcastle Help File Builder](https://GitHub.com/EWSoftware/SHFB)

Added alternate text: [Sandcastle Help File Builder](https://GitHub.com/EWSoftware/SHFB)

Added link target: [Sandcastle Help File Builder](https://GitHub.com/EWSoftware/SHFB)

If the link contains XML entities, they must be encoded (i.e. the ampersand): [MSDN Documentation Forum](http://forums.microsoft.com/MSDN/ShowThread.aspx?)

▲ See Also

Other Resources

[Link Elements](#)

[link](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



link

This is used to insert a link to another conceptual topic using its ID value. It can also be used to create links to elements within the same page that have an `address` attribute such as the `introduction` and `section` elements.

Usage

For topic links, a link consists of a required `xlink:href` attribute that contains the GUID of the topic to which the link will go and optional inner text. If no inner text is specified, the text will be retrieved from the referenced topic's `companion file`. If the companion file contains a `linkText` element, its value will be used. If not, the value of the `title` element will be used.

You can link to anchors within the target topic by adding a hash character (#) followed by the anchor name after the topic's GUID. For in-page links, simply specify a hash character (#) followed by the name of the anchor from an element's `address` attribute. Inner text is required for in-page links.

Tip

When used in a `relatedTopics` element, an optional `topicType_id` attribute can be added that specifies a GUID value used to classify the link type so that it is grouped with other similar links. See the `relatedTopics` topic for more information.

Example

[Copy](#)

```
Self-closing: <link xlink:href="cfd9dabf-22f3-474
```

```
With inner text: <link xlink:href="cfd9dabf-22f3-  
companion file</link>
```

In-page link to introduction: `<link xlink:href="#"`

Link to other topic anchor:

`<link xlink:href="dc4fcc96-283e-4202-9ecc-08a65e6`



▲ Example Output

Self-closing: [Companion File](#)

With inner text: [companion file](#)

In-page link to introduction: [Back to Intro](#)

Link to other topic anchor: [Links to Resources](#)

▲ See Also

Other Resources

[Link Elements](#)

[Companion File](#)

[externalLink](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Miscellaneous Elements

The topics in this section define miscellaneous elements not fitting one of the other categories that can be used in conceptual topics.

▲ Miscellaneous Elements

Select one of the topics below to learn more about a particular miscellaneous element.

- [autoOutline](#)
- [markup](#)
- [token](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



autoOutline

The `autoOutline` element is a special case used by Sandcastle to insert an auto-generated section outline within a topic's introduction or sub-sections.

Usage

Place the element within a topic's `introduction` element to produce an outline of the topic's section headings or within a `section's content` element to produce an outline of the section's sub-sections. In either case, the outline will be created at the location of the auto-outline element. As such, you can control the placement of the outline within the text.

Each section must have a `title` element as that is the text that will be used for the outline entry. In addition, an `address` attribute must be added to each `section` element to give it a unique ID that can be used as a link. You can also add an optional `address` attribute to the `introduction` element so that you can link back to it from a sub-section.

An optional `lead` attribute can be specified to define the lead-in text displayed before the list of links. If not specified, default lead-in text is used. If set to the value `none`, the lead-in text will be omitted.

An optional `excludeRelatedTopics` attribute can be specified. If set to `True`, the top-level auto-outline will exclude the **Related Topics** link. This can be useful if you simply want to list the main sections of the topic with some alternate lead-in text or no lead-in text at all.

If a numeric inner text value is specified, the section headings down to the indicated depth will be included in the outline. Zero (0), the default, specifies that only top-level section headings should be included in the outline.

Example

```

<?xml version="1.0" encoding="utf-8"?>
<topic id="1a817894-db7a-4c7f-bd67-cca896affa35"
  <developerConceptualDocument
    xmlns="http://ddue.schemas.microsoft.com/aut
    xmlns:xlink="http://www.w3.org/1999/xlink">

  <introduction address="Intro">
    <!-- No depth option so only show top-level
    <autoOutline />
    <para>The introduction.  If <codeInline>&lt
is included, this paragraph can be omitted.
    </para>
    <!-- Note that placing the autoOutline elem
    the section outline below the intro pa
  </introduction>
  <section address="S1">
    <title>Section 1</title>
    <content>
      <para>This is section 1.</para>
      <!-- This will include sub-sections in th
      levels down.  It also uses custom le
      <autoOutline lead="Select a sub-section:'
    </content>
    <sections>
      <section address="SS1">
        <title>Sub-section 1</title>
        <content>
          <para>This is sub-section 1.</para>
        </content>
      </section>
      <section address="SS2">
        <title>Sub-section 2</title>
        <content>
          <!-- No <autoOutline /> in here so no
          <para>This is sub-section 2.</para>
        </content>
      </section>
    </sections>
  </section address="SSS1">

```

```
    <title>Sub-sub-section 1</title>
    <content>
      <para>This is sub-sub-section 1.<
    </content>
  </section>
  <section address="SSS2">
    <title>Sub-sub-section 2</title>
    <content>
      <para>This is sub-sub-section 2.<
      <para>
        <link xlink:href="#Intro">Back
      </para>
    </content>
  </section>
</sections>
</section>
</sections>
</section>
<section address="S2">
  <title>Section 2</title>
  <content>
    <para>This is section 2.</para>
  </content>
</section>
<relatedTopics>
  <link xlink:href="b9a1daac-a2d8-41b7-83b0-k
</relatedTopics>
</developerConceptualDocument>
</topic>
```



▲ Example Output

See [autoOutline Rendered Example](#)

▲ See Also

Other Resources

Miscellaneous Elements
token

[v1.1.0.3] Sandcastle MAML Guide

Send comments on this topic to [Eric Woodruff](#)



autoOutline Rendered Example

This topic contains the following sections:

- [Section 1](#)
- [Section 2](#)
- [See Also](#)

The introduction. If `<autoOutline />` is included, this paragraph can be omitted.

▲ Section 1

This is section 1.

Select a sub-section:

- [Sub-section 1](#)
- [Sub-section 2](#)
 - [Sub-sub-section 1](#)
 - [Sub-sub-section 2](#)

Sub-section 1

This is sub-section 1.

Sub-section 2

This is sub-section 2.

Sub-sub-section 1

This is sub-sub-section 1.

Sub-sub-section 2

This is sub-sub-section 2.

[Back to the Top](#)

▸ Section 2

This is section 2.

▸ See Also

Other Resources

[autoOutline](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



markup

This element can be used to pass a literal piece of HTML markup through to the rendered topic. In general, this element is only used by custom build components to pass HTML through after modifying the conceptual topic in some manner (i.e. to colorize code). Other non-component uses might be to include a video or other such items in the topic that is currently not handled by MAML.

Note

It is recommended that you avoid using this element unless there is no MAML equivalent for what you are trying to do. Usage should be restricted to the output of custom build components when needed.

Usage

Enter the element in the topic and specify the HTML markup as the inner text. It can be used as a block or an inline element.

Example

[Copy](#)

Here is a piece of literal HTML markup:

```
<markup>  
<br/><br/>
```

It is **NOT** recommended that you use the `<en` normal conceptual content when an equivalent `<u>` is available. Its use should be restricted to custom build components.

```
</markup>
```

Example Output

Here is a piece of literal HTML markup:

It is **NOT** recommended that you use the *markup* element for normal conceptual content when an equivalent MAML element already exists. Its use should be restricted to custom build components.

▲ See Also

Other Resources

[Miscellaneous Elements](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



token

A token is used as a replaceable tag within a topic and is represented using a `token` element. The inner text of the element is a token name. The tokens are defined in a separate token file. They are an easy way to represent common items that you use regularly such as a common phrase or external link.

Usage

To use a token, simply insert its name as the inner text of a `token` element anywhere within a topic.

Example

In this example, "Sandcastle" is a token that expands to an external link to a web site.

```
This link (<token>SHFB</token>) is stored in a tc
```

[Copy](#)

Example Output

This link ([Sandcastle Help File Builder](#)) is stored in a token.

See Also

- Other Resources
- [Miscellaneous Elements](#)
- [Token File](#)

Send comments on this topic to [Eric Woodruff](#)



Visual Studio IntelliSense

This topic covers how to integrate MAML schema files into Visual Studio's global schema cache to provide IntelliSense when editing MAML topics within Visual Studio.

Tip

The guided installer used to install Sandcastle and the related tools provides an option to install the MAML schemas in the Visual Studio global schema cache for you.

Note

Because changes are made to the Program Files folder, you will need administrative privileges to perform these updates. In the instructions below, *%ProgramFiles%* is used to refer to the Program Files folder. This will typically be *C:\Program Files* on 32-bit systems and *C:\Program Files (x86)* on 64-bit systems. You may need to alter the paths below if you installed Visual Studio and/or Sandcastle in different locations.

Integrating MAML Schema Files into Visual Studio

1. Create a subfolder named *MAML* under Visual Studio's global schema cache folder which is located at *%ProgramFiles%\Microsoft Visual Studio [x.x]\Xml\Schemas* where "[x.x]" is each version of Visual studio such as "12.0" (VS 2013) and "14.0" (VS 2015). You can repeat the following steps for each version of Visual Studio that you have installed.
2. Copy all of the MAML schemas and the catalog file from the *%ProgramFiles%\Sandcastle\Schemas\Authoring* folder into the new MAML schemas folder that you created in the previous step.
3. Update the root *%ProgramFiles%\Microsoft Visual Studio [x.x]\Xml\Schemas\catalog.xml* file by including the following line:

```
<Catalog href="%InstallRoot%/xml/schemas/MAML/c
```

4. IntelliSense should now work for all MAML topics automatically in Visual Studio. If you had Visual Studio open while making the changes, you may need to restart it in order to see IntelliSense in the MAML topic files.

▲ See Also

Other Resources

[Schema Extensibility](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Schema Extensibility

This topic will explain how you can change the MAML schema so that you can use elements from other namespaces as well. This is useful when extending Sandcastle with build components that provide new elements because it still allows you to validate your topic files.

Tip

The Sandcastle version of the MAML schema files already contain the necessary changes.

XML Schema Extensibility

In general, XML schema provides the [any](#) element for declaring extensibility points in your schema. The [any](#) element has two attributes that control the extensibility contract.

The [namespace](#) attribute allows you to either explicitly specify a closed set of namespaces that are allowed or to specify one of these pseudo-URIs:

##local

This allows only to use elements that do not reside in any namespace. This is not what we want to use because it forces build component authors to declare all elements in the empty namespace. This could easily lead to name clashes between different build components.

##targetNamespace

This only allows elements from the target namespace of the declaring schema. For our purposes this is useless because it would only allow elements from the MAML namespace (<http://ddue.schemas.microsoft.com/authoring/2003/5>).

##any

This allows any namespace, including the target namespace of the declaring schema. We do not want to use this because it

would also allow to use any element from the MAML namespace (<http://ddue.schemas.microsoft.com/authoring/2003/5>) itself and hence undermine the schema validation for MAML.

##other

This allows any namespaces except the target namespace of the declaring schema. For our purposes this is ideal because it allows only to use elements from any namespace except the MAML namespace (<http://ddue.schemas.microsoft.com/authoring/2003/5>).

The `processContents` attribute allows you to control the validation requirements. It can have one of following values:

skip

This is the least restrictive value. It specifies that the schema processor should not try to validate the contents of the new element, even if it is able to find a schema for it.

strict

This is the most restrictive setting. It requires the schema processor to always validate the contents of the new element. If it is not able to find such a schema an error is reported during validation.

lax

This is a compromise between the settings `skip` and `strict`. The schema processor will validate the schema if it is able to find a schema for the new element and will skip it if cannot find such a schema. For our purposes this is ideal because it allows build component authors to provide a schema for validation but does not force them to do so.

Therefore, our extensibility points will all look like this:

```
<any namespace="##other" processContents="lax" />
```

▲ Extending Inline Text

Build components might add elements that you can use inside a

`para` element. Examples for such elements are:

Linking

These are elements that allow you to link to other topics. MAML itself already provides several elements for this purpose, such as the `link` element to link to conceptual topics, the `codeEntityReference` element to link to managed code APIs, and the `externalLink` element to link to some location in the internet.

Semantics

This category includes elements that are used to associate pieces of text with semantics. In many cases these semantics are made visible through a consistent formatting, such as italics or bold text. Amongst others MAML itself provides the `localUri` element for marking text as representing local paths and the `ui` element for marking text as representing elements in your user interface.

To extend the `para` element we will extend the underlying type `inlineType`.

Open the file `inline.xsd`, goto line 60, and add the `any` element to the type:

[Copy](#)

```
<complexType name="inlineType" mixed="true">
  <annotation>
    <documentation>
      The inlineType complexType describes a simple
      content model. It provides both text and element
      similarly simple content models.
    </documentation>
  </annotation>
  <choice minOccurs="0" maxOccurs="unbounded">
    <group ref="maml:inlineGroup"/>
    <element ref="maml:sharedContent"/>
    <element name="conditionalSection">
      <complexType>
        <sequence>
          <element ref="maml:conditions"/>
        </sequence>
      </complexType>
    </element>
  </choice>
</complexType>
```

```

        <element name="conditionalContent" type
        </sequence>
    </complexType>
</element>
<!-- PATCH HERE -->
    <any namespace="##other" processContents="la>
</choice>
    <attributeGroup ref="maml:contentIdentificator
</complexType>

```

▲ Extending Related Topics

For build components that provide elements for linking we also want to be able to use them inside the `relatedTopics` element. To do this, we will extend the type `relatedTopicsType`.

Open the file `hierarchy.xsd`, goto line 123 and add the `any` element to the type:

[Copy](#)

```

<complexType name="relatedTopicsType" mixed="false"
    <choice minOccurs="0" maxOccurs="unbounded">
        <group ref="maml:relatedTopicLinkGroup"/>
        <!-- PATCH HERE -->
        <any namespace="##other" processContents="la>
    </choice>
    <attributeGroup ref="maml:contentIdentificator
</complexType>

```

▲ Extending Section Structure

Last but not least build components might also provide elements that allow you to build a structure. For this purpose MAML itself already provides the `table`, `code`, and `para` elements. To allow new structural elements we have to extend the type `sectionContentType`.

Open the file `structure.xsd`, goto line 67 and add the `any` element to the type:

Copy

```
<complexType name="sectionContentType" mixed="fa]
  <choice minOccurs="0" maxOccurs="unbounded">
    <group ref="maml:structureGroup"/>
    <element name="conditionalSection" type="maml]
      <!-- PATCH HERE -->
    <any namespace="##other" processContents="la>
  </choice>
  <attributeGroup ref="maml:contentIdentificator
</complexType>
```

▲ See Also

Other Resources

[Visual Studio IntelliSense](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Links to Resources

This topics contains links to various resources that you may find useful.

- [MAML Information and Guides](#)
- [Discussion Groups, Blogs, Etc.](#)
- [Build Tools](#)

▲ MAML Information and Guides

- The most recent version of this guide can be download from the [Sandcastle Help File Builder](#) project on GitHub.
- For general background information, see the Wikipedia entry [Microsoft Assistance Markup Language](#) or, for schema information, see the MAML information at help-info.de.
- If you need a tool to convert HTML files to MAML, the [Converting HTML Files to MAML Topics](#) is included with the help file builder along with this guide.

▲ Discussion Groups, Blogs, Etc.

- To discuss the help file builder and tools, suggest new features, report bugs in them, and request help using them, visit the [Sandcastle Help File Builder](#) project site.
- For discussion and questions about help systems in general, visit the [Visual Studio General Questions forum](#).
- The retired [Microsoft Developer Documentation and Help System forum](#) may also provide information on older help systems.

▲ Build Tools

- [Microsoft .NET Frameworks and SDKs.](#)
- The [HTML Help Workshop](#) for building HTML Help 1.x (.CHM) help files.

▲ See Also

Other Resources

[Welcome](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Making a Donation

If you would like to support this project, you can make a donation of any amount you like by clicking on the PayPal donation button below. If you would prefer to donate via check or money order (USD), please send me an e-mail at Eric@EWoodruff.us and I will reply with my mailing address.



The above method of donating is secure. PayPal guarantees your privacy and security. I never receive details of your payment other than the amount, your name, e-mail address, and any optionally entered information.

Thanks to those of you that have made a donation. It is much appreciated!

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Version History

The topics in this section describe the various changes made to the MAML Guide over the life of the project.

▲ Version History

Select a version below to see a description of its changes.

- [Version 1.1.0.3](#)
- [Version 1.1.0.2](#)
- [Version 1.1.0.1](#)
- [Version 1.1.0.0](#)
- [Version 1.0.0.0](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Version 1.1.0.3

Version 1.1.0.3 was released on December 16th, 2012.

▲ Changes in This Release

- Removed references to the Sandcastle Help File Builder components and features that have been moved into the Sandcastle tools and presentation style XSL transformations and reworded the affected sections to reflect their actual use with Sandcastle alone.

▲ See Also

Other Resources

[Version History](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Version 1.1.0.2

Version 1.1.0.2 was released on April 7th, 2012.

▲ Changes in This Release

- Added a solution file for use in editing the project within Visual Studio with the Sandcastle Help File Builder and updated the project to the latest release format.
- Added non-breaking spaces to the examples for the `command`, `replaceable`, and `system` element topics to fix their layout. Added a note about this to the **Known Issues** topic.
- Moved the `markup` element to the **Miscellaneous** group since it can serve as either an inline or block element.
- Fixed the description of the `para` element to note that self-closing and empty paragraphs are ignored rather than creating a line break. The prior implied usage was incorrect and did not conform to the schema.
- Fixed the `code` and `codeReference` examples to include `para` elements in the proper places to conform to the schema.
- Fixed up several other topics so that they conformed to the MAML schema.
- Replaced all `code lang` attributes with `language` attributes to conform to the MAML schema.
- Removed all references to the Sandcastle Styles project. The changes that it contained have been merged into the latest release of Sandcastle itself along with this project.

▲ See Also

Other Resources

[Version History](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Version 1.1.0.1

Version 1.1.0.1 was released on February 21st, 2011.

▲ Changes in This Release

- Updated the `definitionTable`, `list`, and `table` topics to note that text in the `definition`, `listItem`, and `entry` elements should be contained in a `para` element. The topics in the MAML guide have been updated to reflect this schema rule.
- Converted the project to use the new Sandcastle Help File Builder 1.9.0.0 project format.
- Updated references to the latest release of Sandcastle (June 2010) and fixed a few typos.
- Added a topic on integrating the Sandcastle MAML schema files into the Visual Studio global schema cache to provide IntelliSense for MAML topics while editing them in Visual Studio.
- Added a topic on making the Sandcastle MAML schemas extensible for third-party build component elements. This information was supplied by Immo Landwerth.

▲ See Also

Other Resources

[Version History](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Version 1.1.0.0

Version 1.1.0.0 was released on October 25th, 2008.

▲ Changes in This Release

- Corrected the order of the `linkAlternateText` and `linkUri` elements in the `externalLink` topic and all uses of it throughout the guide.
- Fixed a few bugs in the usage examples.
- Updated the conceptual templates to reflect correct usage and to note some elements as being optional.
- Updated the `relatedTopics` topic to include information about the optional `topicType_idlink` attribute.
- Updated the `autoOutline` topic to include information about the optional `lead` and `excludeRelatedTopics` attributes.
- Added topics for:
 - `definitionTable`, `definedTerm`, and `definition`
 - `list` and `listItem`
 - `table`, `tableHeader`, `row`, and `entry`
- Converted the project to use the new Sandcastle Help File Builder 1.8.0.0 project format.

▲ See Also

Other Resources

Version History

[v1.1.0.3] Sandcastle MAML Guide

Send comments on this topic to [Eric Woodruff](#)



Version 1.0.0.0

Version 1.0.0.0 was released on June 4th, 2008.

▲ Changes in This Release

- Initial release.

▲ See Also

Other Resources

[Version History](#)

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)



Glossary

[A](#) | [B](#) | [C](#) | [D](#) | [E](#) | [F](#) | [G](#) | [H](#) | [I](#) | [J](#) | [K](#) | [L](#) | [M](#) | [N](#) | [O](#) | [P](#) | [Q](#) | [R](#) | [S](#) | [T](#) | [U](#) | [V](#) | [W](#) | [X](#) | [Y](#) | [Z](#)

B

BuildAssembler.exe

This is a tool supplied with Sandcastle that is used to build the help topics for conceptual and reference builds. It is passed a configuration file containing a list of build components to use in transforming the topics into HTML and a manifest file that lists the topics to build.

See Also: [Sandcastle](#)

C

Code Block Component

A custom build component that can be used to colorize code, add line numbering and/or collapsible section, and import code from working source code files. It can be used in both conceptual and reference builds. The component is integrated with the [Sandcastle Help File Builder](#) and is also available as a standalone component from the same project site for use in other tools or your own build scripts.

See Also: [Sandcastle](#), [Sandcastle Help File Builder](#)

code snippets file

A file used to store commonly used code snippets. The snippets can be inserted into a conceptual topic using the [codeReference](#) element.

conceptual build

A build that uses Sandcastle to produce help file topics using information extracted from conceptual content topics and their related files.

See Also: [reference build](#), [Sandcastle](#)

conceptual content

A topic file that contains conceptual content. These can be used to add usage notes, walkthroughs, tutorials, etc. to a help file.

G

Globally Unique Identifier, GUID

A unique value that is associated with each conceptual topic and image in a project. When inserting links to topics or images, the ID is used to refer to them. This allows you to alter the names or locations of the topic files without having to change the name or location in each topic that references them.

M

MAML

An acronym that stands for Microsoft Assistance Markup Language. Conceptual content topics are composed of MAML elements.

R

reference build

A build that uses Sandcastle to produce help file topics using information extracted from managed assemblies and their associated XML comments files.

See Also: [conceptual build](#), [Sandcastle](#)

S

Sandcastle

Sandcastle is a set of tools originally produced by Microsoft that can be used to build help files for .NET managed libraries, conceptual content, or a mix of both. Microsoft officially discontinued development in October 2012. The Sandcastle tools have been merged with the [Sandcastle Help File Builder](#) project and are developed and supported there now as part of that project.

See Also: [Sandcastle Help File Builder](#)

Sandcastle Help File Builder, SHFB

The [Sandcastle Help File Builder](#) is a standalone tool used to automate Sandcastle. It consists of a GUI front end that helps you manage and build help file projects. It uses a standard MSBuild format project file which can also be built from the command line using MSBuild or integrated into Visual Studio builds or other build scripts to produce a help file when your application projects are built. In addition, it provides a set of additional features beyond those supplied with Sandcastle that can improve your help file and make it easier to deploy. A Visual Studio integration package is also available for it that integrates the project management and build features into Visual Studio 2013 or later.

See Also: [Sandcastle](#)

T

token, token file

A token is used as a replaceable tag within a topic and is represented using a `token` element. The inner text of the element is a token name. The tokens are defined in a separate [token file](#). They are an easy way to represent common items that you use regularly such as a common phrase or external link.

[\[v1.1.0.3\] Sandcastle MAML Guide](#)

Send comments on this topic to [Eric Woodruff](#)